23 COMPUTATIONAL TOPOLOGY OF GRAPHS ON SURFACES

Éric Colin de Verdière

INTRODUCTION

This chapter surveys computational topology results in the special, low-dimensional case where the ambient space is a surface. Surface topology is very well-understood and comparably simpler than the higher-dimensional counterparts; many computational problems that are undecidable in general (e.g., homotopy questions) can be solved efficiently on surfaces. This leads to a distinct flavor of computational topology and to dedicated techniques for revisiting topological problems on surfaces from a computational viewpoint.

Topological surfaces and graphs drawn on them appear in various fields of mathematics and computer science, and these aspects are not surveyed here:

- in *topology* of three-dimensional manifolds, also in connection to the recent resolution of the Poincaré conjecture, combinatorial and algebraic structures defined on surfaces are often relevant, e.g., via the study of mapping class groups and Teichmüller spaces [FM11];
- in *topological graph theory*, a branch of structural graph theory, graphs on surfaces are studied from a combinatorial point of view, also in relation to the theory of Robertson and Seymour on graph minors; for example, colorability questions of graphs on surfaces, generalizing the four-color theorem for planar graphs, are well-studied [MT01];
- in *enumerative combinatorics*, a natural problem is to count (exactly or asymptotically) maps with given properties in the plane or on surfaces, with the help of generating series; moreover, typical properties of random maps are investigated [Mie09, Bet12, LZ04];
- various *applications* involve surface meshes, in particular in geometry processing and computer graphics, for approximation [CDP04], topological simplification [GW01, WHDS04], compression [AG05], and parameterization [GY03]. Techniques for general surfaces apply also to subsets of the plane, and are thus relevant in VLSI design [LM85] and map simplification [BKS98].

This chapter is organized as follows. We first review the basic concepts and properties of topological surfaces and graphs embedded on them (Sections 23.1 and 23.2). Then we consider three categories of topological problems, mostly from a computational perspective: drawing an abstract input graph on a surface (Section 23.3), homotopy questions and variations (Section 23.4), and optimization of curves and graphs on surfaces, also from a homological point of view (Section 23.5). Then we survey techniques that allow us to solve general graph problems faster in the case where the input graph is embedded on a fixed surface (Section 23.6). Finally, we collect other miscellaneous results (Section 23.7).

23.1 SURFACES

Surfaces are considered from a topological point of view: Two homeomorphic surfaces are regarded as equivalent. Surfaces such as the sphere or the disk are topologically uninteresting; our focus is on surfaces in which some closed curves are non-contractible (they cannot be deformed to a point by a continuous motion on the surface).

GLOSSARY

Homeomorphism: Given two topological spaces X and X', a map $h\colon X \to X'$ is a homeomorphism if h is bijective and both h and its inverse are continuous.

Surface (topological definition): In this chapter, a surface S is a *compact* two-dimensional manifold possibly with boundary. Equivalently, S is a compact topological space that is Hausdorff (any two distinct points have disjoint neighborhoods) and such that every point has a neighborhood homeomorphic to the plane or the closed half-plane. The set of points of a surface S that have no neighborhood homeomorphic to the plane is the ***boundary*** of S.

Surface (combinatorial definition): Equivalently, a surface S is a topological space obtained from finitely many disjoint triangles by identifying some pairs of edges of the triangles (by the quotient topology). The ***boundary*** of S is the union of the edges that are not identified with any other edge.

Path: A path on S is a continuous map $p\colon [0,1] \to S$. Its two ***endpoints*** are $p(0)$ and $p(1)$.

Connectedness: A surface is ***connected*** if any two points of the surface are the endpoints of some path. The inclusionwise maximal connected subsets of a surface form its ***connected components***.

Orientability: A surface is ***non-orientable*** if some subset of it (with the induced topology) is homeomorphic to the Möbius strip (defined in Figure 23.1.1). Otherwise, it is ***orientable***.

PROPERTIES: CLASSIFICATION OF SURFACES

Every connected surface is homeomorphic to exactly one of the following surfaces:

- the orientable surface of ***genus*** $g \geq 0$ with $b \geq 0$ ***boundary components*** (or, more concisely, ***boundaries***), obtained from the sphere by removing g disjoint open disks, attaching a handle (defined in Figure 23.1.1) to each of the resulting g circles, and finally removing b open disks with disjoint closures;

- the non-orientable surface of ***genus*** $g \geq 1$ with $b \geq 0$ ***boundary components*** (or ***boundaries***), obtained from the sphere by removing g disjoint open disks, attaching a Möbius strip (defined in Figure 23.1.1) to each of the resulting g circles, and finally removing b open disks with disjoint closures.

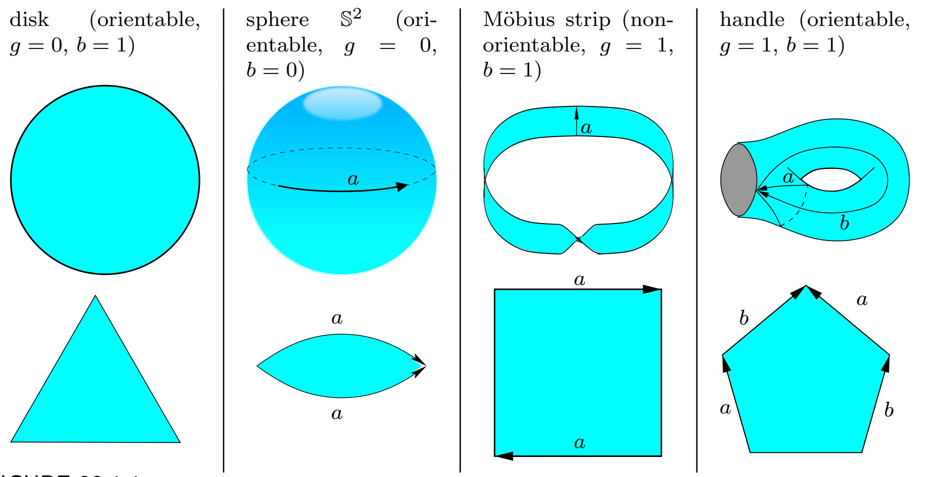
disk (orientable, $g = 0$, $b = 1$)
sphere $\mathbb{S}^2$ (orientable, $g = 0$, $b = 0$)
Möbius strip (non-orientable, $g = 1$, $b = 1$)
handle (orientable, $g = 1$, $b = 1$)
a
b

FIGURE 23.1.1
Examples of surfaces. Each surface (top row) comes with a polygonal schema (bottom row), a polygon with some labeled and directed edges; the surface can be obtained by identifying the pairs of edges with the same labels, respecting their direction. The genus g and number of boundary components b are specified, as well as whether the surface is orientable.

Every surface can be obtained by identifying pairs of edges of disjoint triangles. More concisely, every surface can be defined by a ***polygonal schema***, a polygon with labels and directions on some of the edges specifying how they must be identified. In particular, one can define a ***canonical*** polygonal schema for every connected surface without boundary:

- The canonical polygonal schema of the orientable surface of genus $g \geq 1$ is a $4g$-gon whose successive edges are labeled $a_1, b_1, \bar{a}_1, \bar{b}_1, \ldots, a_g, b_g, \bar{a}_g, \bar{b}_g$, and where edge x is directed clockwise, edge $\bar{x}$ is directed counterclockwise. Identifying edge x with $\bar{x}$, as indicated by their directions, gives the orientable surface of genus g. See Figure 23.1.2.
- Similarly, the canonical polygonal schema of the non-orientable surface of genus $g \geq 1$ is a $2g$-gon whose successive edges are labeled $a_1, a_1, \ldots, a_g, a_g$, and where all edges are directed clockwise.

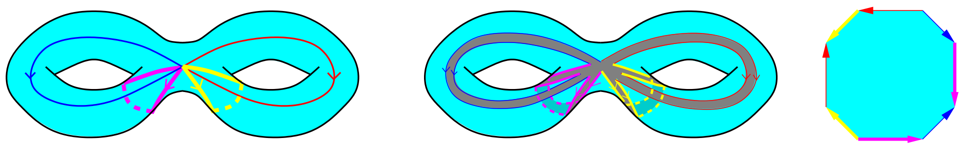

FIGURE 23.1.2
A double torus with a system of loops (left); the surface cut along the loops (middle) is a disk, shown in the form of a (canonical) polygonal schema (right).

EXAMPLES

Table 23.1.1 lists some common connected surfaces. See also Figures 23.1.1 and 23.1.2.

TABLE 23.1.1 Some common surfaces.

Surface	Orientable?	Genus	# of boundary components
Sphere	Yes	0	0
Disk	Yes	0	1
Annulus = cylinder	Yes	0	2
Pair of pants	Yes	0	3
Torus	Yes	1	0
Handle	Yes	1	1
Double torus	Yes	2	0
Projective plane	No	1	0
Möbius strip	No	1	1
Klein bottle	No	2	0

23.2 GRAPHS ON SURFACES

GLOSSARY

Let S be a surface.

Loop: A loop is a path whose two endpoints are equal to a single point, called the ***basepoint*** of the loop.

Closed curve: A closed curve on S is a continuous map from the unit circle $\mathbb{S}^1$ to S. This is almost the same as a loop, except that a closed curve has no distinguished basepoint. A closed curve is sometimes called a ***cycle***, although, contrary to the standard terminology in graph theory, here a cycle may self-intersect.

Curve: A curve is either a path or a closed curve. For most purposes, the parameterization is unimportant; for example, a path p could be regarded as equivalent to $p \circ \varphi$, where $\varphi : [0, 1] \to [0, 1]$ is bijective and increasing.

Simplicity: A path or a closed curve is ***simple*** if it is injective. A loop $\ell\colon [0, 1] \to S$ is ***simple*** if its restriction to $[0, 1)$ is injective.

Graph: In this chapter, unless specified otherwise, graphs are finite, undirected, and may have loops and multiple edges.

Curve (in a graph): A ***curve*** in a graph G (also called ***walk*** in the terminology of graph theory) is a sequence of directed edges $e_1, \ldots, e_k$ of G where the target of e_i equals the source of e_{i+1}. Repetitions of vertices and edges are allowed. The ***endpoints*** of the curve are the source of e_1 and the target of e_k. If they are equal, the curve is ***closed***.

Graph embedding (topological definition): A graph G naturally leads to a topological space $\hat{G}$, defined as follows: One considers a disjoint set of segments, one per edge of G, and identifies the endpoints that correspond to the same vertex of G. This gives a topological space, from which $\hat{G}$ is obtained by adding one isolated point per isolated vertex of G. (As a special case, if G has no loop and no multiple edge, then G is a one-dimensional simplicial complex, and $\hat{G}$

is the associated topological space.) An ***embedding*** of G is a continuous map from $\hat{G}$ into S that is a homeomorphism from $\hat{G}$ onto its image.

Graph embedding (concrete definition): Equivalently, an embedding of G on S is a "crossing-free" drawing of G: It maps the vertices of G to distinct points of S, and its edges to paths of S whose endpoints are the images of their incident vertices; the image of an edge can self-intersect, or intersect the image of another edge or vertex, only at its endpoints. When no confusion arises, we identify G with its embedding on S, or with the image of that embedding.

Face: The faces of an embedded graph G are the connected components of the complement of the image of G.

Degree: The degree of a vertex v is the number of edges incident to v, counted with multiplicity (if an edge is a loop). The degree of a face f is the number of edges incident to f, counted with multiplicity (if an edge has the same face on both sides).

Cellular embedding: A graph embedding is ***cellular*** if its faces are homeomorphic to open disks.

Triangulation: A graph embedding is a triangulation if it is cellular and its faces have degree three. The triangulation may fail to be a simplicial complex: A triangle is not necessarily incident to three distinct vertices, or even to three distinct edges.

Cutting: Given an embedded graph G on S without isolated vertex, the operation of cutting S along G results in a (possibly disconnected) surface with boundary, denoted $S \backslash\backslash G$ (or sometimes S✂G or similar); each connected component of $S \backslash\backslash G$ corresponds to a face of G on S, and by identifying pieces of the boundaries of these components in the obvious way, one recovers the surface S. Similarly, one can cut along a set of disjoint, simple closed curves. (Technically, if S has nonempty boundary, an additional condition is needed: The intersection of an edge with the boundary of S can be either the entire edge, its two endpoints, or one of its endpoints.)

Planarity: A graph is planar if it has an embedding to the plane (or equivalently the sphere).

Dual graph: A dual graph of a cellularly embedded graph G on S (assumed without boundary) is a graph G^* embedded on S with one vertex f^* inside each face f of S, and with an edge e^* for each edge e of G, such that e^* crosses e and no other edge of G. A dual graph is cellularly embedded. Its combinatorial map (see below) is uniquely determined by the combinatorial map of G.

Euler genus: The Euler genus $\bar{g}$ of a connected surface S with genus g equals $2g$ if S is orientable, and g if S is non-orientable.

Euler characteristic: The Euler characteristic of a cellularly embedded graph G equals $\chi(G) := v - e + f$, where v, e, and f are its number of vertices, edges, and faces, respectively.

PROPERTIES: EULER'S FORMULA AND CONSEQUENCES

1. ***Euler's formula:*** If G is *cellularly* embedded on a connected surface S of Euler genus $\bar{g}$ with b boundary components, then $\chi(G) = 2-\bar{g}-b$. In particular,

$\chi(G)$ does not depend on G, only on S, and is consequently called the ***Euler characteristic*** of S.

2. The number of vertices and faces of a graph G cellularly embedded on a connected surface is at most linear in the number of its edges. In particular, the combinatorial complexity of G is linear in its number of edges.
3. Conversely, let G be a (not necessarily cellular) graph embedding on a connected surface with Euler genus $\bar{g}$ and b boundaries. Assume that G has no face of degree one or two that is an open disk. Then the numbers e of edges and v of vertices of G satisfy $e = O(v + \bar{g} + b)$.

DATA STRUCTURES

In all the problems we shall consider, the exact embedding of a graph on a surface is irrelevant; only the actual combinatorial data associated to the embedding is meaningful. If G is a graph cellularly embedded on a surface S without boundary, we only need the information of G together with the *facial walks*, namely, the closed walks in G encountered when walking along the boundary of the faces of G. This information is called the *combinatorial map* of G, and allows us to reconstruct the surface, by attaching disks to every facial walk. (Some conditions on the walks are needed to ensure that the resulting space is indeed a surface.) If S has boundaries, one can specify the corresponding faces of G. If S is orientable and G has no loop edge, instead of the facial walks one could as well specify the cyclic ordering of the edges incident to each vertex.

However, more complicated data structures are needed to perform basic operations efficiently. For example, one should be able to compute the degree of a face in time linear in the degree; to count the number of faces of G in linear time; to determine whether the surface is orientable in linear time; etc. (The last two operations, together with counting the number of vertices and edges, allow us to identify the topology of the surface in linear time using Euler's formula; this can also be done in logarithmic space [BEKT16].)

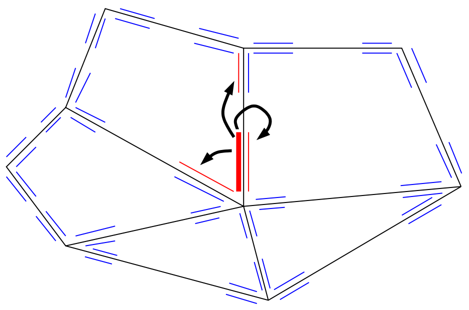

FIGURE 23.2.1
The graph-encoded map data structure. Each edge bears four flags, drawn parallel to it. Three operations allow us to move from a flag to a nearby flag.

One such data structure, the *graph-encoded map* or *gem representation* [Lin82], uses flags (quarter-edges, or, equivalently, incidences between a vertex, an edge, and a face) of G, see Figure 23.2.1; three involutive operations can be applied to a flag to move to an incident flag. Alternative data structures have been designed for more general situations (e.g., to allow surfaces with boundaries) or to take advantage of special situations (e.g., in the case where G is a triangulation, or where S is orientable); see the survey [Ket99]. However, the choice of the data structure is irrelevant for the theoretical design and asymptotic analysis of the algorithms.

CONVENTIONS FOR THIS CHAPTER

Henceforth, **we assume all surfaces to be connected**.

In several works mentioned in the following, only orientable surfaces are considered. In some cases, non-orientable surfaces are just as easy to handle, but sometimes they lead to additional difficulties. **We refer to the original articles to determine whether the results hold on non-orientable surfaces.**

Also, in most problems studied in this chapter, surfaces with boundaries are no harder to handle than surfaces without boundary: Any algorithm for surfaces without boundary immediately implies an algorithm for surfaces with boundary (with the same running time, or by replacing g by $g + b$ in the complexity, where g and b are the genus and the number of boundary components). For this reason, **we mostly focus on computational problems for surfaces without boundary**.

Finally, when we consider cellularly embedded graphs for algorithmic problems, **we implicitly assume that they are specified in the form of a data structure as described above** (e.g., a graph-encoded map).

23.3 EMBEDDING AND DRAWING GRAPHS ON SURFACES

Being able to build embeddings of a graph on a surface with small genus is important; almost all algorithms for graphs embeddable on a fixed surface require an embedding of the input graph (there are a few exceptions [ES14, Kel06, MS12]). We discuss algorithmic results related to the problem of embedding a graph on a surface, and then consider more general drawings where crossings are allowed.

EMBEDDING GRAPHS ON SURFACES

Let G be an abstract graph (not embedded on any surface), given, e.g., by the (unordered) list of the edges incident to every vertex. We assume that G is connected. Let n denote the combinatorial complexity of G, that is, the total number of vertices and edges of G.

1. *General facts:* An embedding on an orientable surface with minimum possible genus is cellular. If G is embeddable on an orientable (resp., non-orientable) surface of genus g, then it is embeddable on an orientable (resp., non-orientable) surface of genus g', for every $g' \geq g$.
2. *General bound:* G can be cellularly embedded on some orientable surface with genus $O(n)$.
3. *Planar case:* There is an $O(n)$-time algorithm for deciding embeddability in the sphere (equivalently, in the plane) [HT74]; also in $O(n)$ time, the graph can

be embedded with straight-line segments in the plane [Sch90a] (see also [NR04, Ch. 4]), if it has no loop or multiple edge. See Chapter 55 for more results on graph drawing.

4. *Time complexity:* Given a graph G and a surface S, specified by its Euler genus $\bar{g}$ and by whether it is orientable, determining whether G embeds on S is NP-hard [Tho89], but can be done in $2^{\text{poly}(\bar{g})} \cdot n$ time [KMR08, Moh99] (where poly($\bar{g}$) is a polynomial in $\bar{g}$), which is linear if $\bar{g}$ is fixed. Such an embedding can be computed in the same amount of time if it exists.
5. *Space complexity:* For every *fixed* $\bar{g}$, determining whether an input graph G embeds on some surface (orientable or not) of Euler genus at most $\bar{g}$ can be done in space logarithmic in the input size [EK14].
6. *Approximation:* Given as input a graph G and an integer $\bar{g}$, one can in polynomial time either correctly report that G embeds on no surface of Euler genus $\bar{g}$, or compute an embedding on some surface of Euler genus $\bar{g}^{O(1)}$ [KS15].

Except for the planar case, these algorithms are rather complicated, and implementing them is a real challenge. For example, there seems to be no available implementation of a polynomial-time algorithm for testing embeddability in the torus, and no publicly available implementation of any algorithm to decide whether a graph embeds on the double torus; attempts of implementing some known embedding algorithms, even in the simplest cases, have unveiled some difficulties [MK11]. On the other hand, a recent approach is promising in practice for graphs of moderate size, using integer linear programming or Boolean satisfiability reformulations [BCHK16].

In contrast, determining the *maximum* genus of an orientable surface without boundary on which a graph can be *cellularly* embedded can be done in polynomial time [FGM88]. There are also results on the embeddability of two-dimensional simplicial complexes on surfaces [Moh97].

On a less algorithmic side, in the field of topological graph theory, a lot more is known about the embeddability of some classes of graphs on some surfaces; see, e.g., [Arc96, Sect. 4.2] and references therein.

GLOSSARY ON DRAWINGS

Let G be a graph and S be a surface.

Drawing: Drawings are more general than embeddings in that they allow a finite set of crossing points, where exactly two pieces of edges intersect and actually cross. Formally, recall that G has an associated topological space $\hat{G}$. A (topological) ***drawing*** of G on S is a continuous map from $\hat{G}$ into S such that the preimage of every point in S has cardinality zero or one, except for a finite set of points ("crossings"), whose preimages have cardinality two; moreover, each such crossing point has a disk neighborhood that contains exactly the images of two pieces of edges of $\hat{G}$, which form, up to homeomorphism, two crossing straight lines.

Arrangement: Let D be a drawing of G on S. The ***arrangement*** of D on S is the graph G' embedded on S that has the same image as D and is obtained from D by inserting a vertex of degree four at each crossing in D and subdividing the edges of G accordingly. Similarly, one can consider the arrangement of a set of curves drawn on S.

Crossing number: The crossing number of G with respect to S is the minimum number of crossings that G has in any drawing of G on S.

Pair crossing number: The pair crossing number of G with respect to S is the minimum number of pairs of edges of G that cross, over all drawings of G on S.

Odd crossing number: The odd crossing number of G with respect to S is the minimum number of pairs of edges of G that cross an odd number of times, over all drawings of G on S.

DRAWING GRAPHS ON SURFACES WITH FEW CROSSINGS

1. *Crossing numbers:* Computing the planar crossing number of a graph is NP-hard (even in very special cases, such as that of a planar graph with a single additional edge [CM13]), and there exists no polynomial-time algorithm with approximation guarantee better than a certain constant [Cab13]. However, for every fixed k, one can, in linear time, determine whether an input graph has planar crossing number at most k [KR07], although the problem admits no polynomial kernel [HD16]. Some approximation algorithms for the planar crossing number are known in restricted cases, such as bounded maximum degree [HC10, Chu11].
2. *Variations on crossing numbers:* The relations between the various notions of crossing numbers are not fully understood. Let c, p, and o denote the planar crossing number, planar pair crossing number, and planar odd crossing number, respectively, of some graph G. It is clear that $o \leq p \leq c$, and it is known that the left inequality can be strict [PSŠ08]. It is widely believed that $p = c$, but the best bound known so far is $c = O(p^{3/2} \log^2 p)$ (this follows essentially from [Tót12]). See, e.g., [Mat14] for more details, and [Sch13a] for a wide survey on the various notions of crossing numbers.
3. *Hanani–Tutte theorem:* The (weak) Hanani–Tutte theorem [Han34, Tut70], however, states that if $o = 0$ then $c = 0$. Furthermore it holds not only for the plane, but for arbitrary surfaces [CN00, PSŠ09b]: If a graph G can be drawn on a surface S in a way that every pair of edges crosses an even number of times, then G can be embedded on S. In the planar case, it actually suffices to assume that every pair of *independent* edges (which do not share any endpoints) crosses an even number of times, but whether this generalizes to arbitrary surfaces is open, except for the projective plane [PSS09a, CVK$^+$16]. We refer to surveys [Sch13b, Sch14] for more details.

23.4 HOMOTOPY AND ISOTOPY

Most works in computational topology for surfaces do not take as input a given abstract graph, as in the previous section; instead, they consider an already embedded graph, given by its combinatorial map.

GLOSSARY

Let S be a surface.

Reversal: The reversal of a path $p\colon [0,1] \to S$ is the path $p^{-1}\colon [0,1] \to S$ defined by $p^{-1}(t) = p(1-t)$.

Concatenation: The concatenation of two paths $p, q\colon [0,1] \to S$ with $p(1) = q(0)$ is the path $p \cdot q$ defined by $(p \cdot q)(t) = p(2t)$ if $t \leq 1/2$ and $(p \cdot q)(t) = q(2t-1)$ if $t \geq 1/2$.

Homotopy for paths: Given two paths $p, q\colon [0,1] \to S$, a ***homotopy*** between p and q is a continuous deformation between p and q that keeps the endpoints fixed. More formally, it is a continuous map $h\colon [0,1] \times [0,1] \to S$ such that $h(0,\cdot) = p$, $h(1,\cdot) = q$, and both $h(\cdot,0)$ and $h(\cdot,1)$ are constant maps (equal, respectively, to $p(0) = q(0)$ and to $p(1) = q(1)$). The paths p and q are ***homotopic***. Being homotopic is an equivalence relation, partitioning the paths with given endpoints into ***homotopy classes***.

Fundamental group: The homotopy classes of loops with a given basepoint form a group, where concatenation of loops accounts for the multiplication and reversal accounts for the inverse operation: if $[p]$ denotes the homotopy class of path p, then we have $[p \cdot q] = [p] \cdot [q]$ and $[p^{-1}] = [p]^{-1}$.

Homotopy for closed curves (also called ***free homotopy***): Given two closed curves $\gamma, \delta\colon \mathbb{S}^1 \to S$, a ***homotopy*** between γ and δ is a continuous deformation between them, namely, a continuous map $h\colon [0,1] \times \mathbb{S}^1 \to S$ such that $h(0,\cdot) = \gamma$ and $h(1,\cdot) = \delta$.

Contractibility: A loop or closed curve is ***contractible*** if it is homotopic to a constant loop or closed curve.

Isotopy: An isotopy between two *simple* paths, loops, or closed curves is a homotopy h that does not create self-intersections: for each t, $h(t,\cdot)$ is a simple path, loop, or closed curve. An isotopy of a graph G is a continuous family of embeddings of G (the vertices and edges move continuously).

Ambient isotopy: An ambient isotopy of a surface S is a continuous map $i : [0,1] \times S \to S$ such that for each $t \in [0,1]$, $i(t,\cdot)$ is a homeomorphism.

Minimally crossing: A family of closed curves $\Gamma = (\gamma_1, \ldots, \gamma_k)$ is minimally crossing if for every family of closed curves $\Gamma' = (\gamma'_1, \ldots, \gamma'_k)$ with γ_i and γ'_i homotopic for each i, the number of intersections and self-intersections in Γ is no larger than in Γ'.

Covering space: Let $\tilde{S}$ be a possibly non-compact connected surface. A continuous map $\pi\colon \tilde{S} \to S$ is a ***covering map*** if every point $x \in S$ has a connected neighborhood U such that $\pi^{-1}(U)$ is a disjoint union of open sets $(U_i)_{i \in I}$ and $\pi|_{U_i}\colon U_i \to U$ is a homeomorphism for each i. We say that $(\tilde{S}, \pi)$ is a ***covering space*** of S. A ***lift*** of a path p is a path $\tilde{p}$ on $\tilde{S}$ such that $\pi \circ \tilde{p} = p$. Finally, if each loop in $\tilde{S}$ is contractible, then $(\tilde{S}, \pi)$ is a ***universal covering space*** of S, which is essentially unique (precisely: if $(\tilde{S}, \pi)$ and $(\tilde{S}', \pi')$ are universal covering spaces, then there is a homeomorphism $\tau : \tilde{S} \to \tilde{S}'$ such that $\pi = \pi' \circ \tau$).

BASIC PROPERTIES

1. Two paths p and q are homotopic if and only if $p \cdot q^{-1}$ is a (well-defined and) contractible loop.
2. Two loops p and q with the same basepoint are freely homotopic (viewed as closed curves without basepoint) if the homotopy classes of the loops p and q are conjugates in the fundamental group.
3. The fundamental group of a surface S without boundary of genus g is best understood by looking at a canonical polygonal schema of the surface: If S is orientable, it is the group generated by $2g$ generators $a_1, b_1, \ldots, a_g, b_g$ and with a single relation, $a_1 b_1 a_1^{-1} b_1^{-1} \ldots a_g b_g a_g^{-1} b_g^{-1}$, corresponding to the boundary of the polygonal schema. Similarly, if S is non-orientable, it is the group generated by g generators $a_1, \ldots, a_g$ and with a single relation, $a_1 a_1 \ldots a_g a_g$.
4. The fundamental group of a surface with at least one boundary component is a free group (because such a surface has the homotopy type of a graph).
5. Let $(\tilde{S}, \pi)$ be a covering space of S. Every path p on S admits lifts on $\tilde{S}$; moreover, if $\tilde{x}$ is a lift of $p(0)$, then p has a unique lift $\tilde{p}$ such that $\tilde{p}(0) = \tilde{x}$. Two paths are homotopic on S if and only if they have homotopic lifts on $\tilde{S}$. In particular, two paths are homotopic if they admit lifts with the same endpoints in the universal covering space.

DECIDING HOMOTOPY AND ISOTOPY

1. *Homotopy:* One of the first and most studied problems regarding curves on surfaces is concerned with homotopy tests: (1) The *contractibility problem*: Is a given closed curve (or, equivalently here, loop) contractible? (2) The *free homotopy problem*: Are two given closed curves (freely) homotopic? These problems translate to central problems from group theory, in the special case of fundamental groups of surfaces: Given a finitely generated group, presented in the form of generators and relations, (1) does a given word in the generators represent the trivial element of the group (the *word problem*)? Do two given words in the generators represent conjugate elements in the group (the *conjugacy problem*)?

   In computational geometry, these problems are studied in the following context: The input is a cellularly embedded graph G and one or two closed curves in G, represented as closed walks in G. There exist linear-time (and thus optimal) algorithms for both the contractibility and the free homotopy problems [LR12, EW13]. (An earlier article [DG99] claims the same results, but it is reported [LR12] that the algorithm for free homotopy in that article has a subtle flaw.) The approaches rely on the construction of a part of the universal covering space, or on results from small cancellation theory in group theory [GS90]. We remark that Dehn's algorithm [Deh12] can be implemented in linear time, but assuming that the surface is fixed and that the graph has a single face, which the other algorithms mentioned above do not require.
2. *Isotopy:* Deciding whether two simple closed curves are isotopic can also be done in linear time, because this equivalence relation is a simple refinement of

homotopy for simple closed curves [Eps66]. Deciding isotopy of graph embeddings is more complicated, but can also be done efficiently, since it essentially reduces to homotopy tests for closed curves [CVM14].

3. *Minimum-cost homotopies:* Often, when it is known that two curves are homotopic, one would like to compute a "reasonable" homotopy. Relevant questions include finding a homotopy that sweeps the minimum possible area (in a discretized sense) [CW13], or has the minimum possible number of "steps"; a homotopy in which the maximum length of the intermediate curves is minimal ("height" of the homotopy) [CL09]; a homotopy in which the maximum distance traveled by a point from the first to the second curve is minimal ("width" of the homotopy—this is related to the *homotopic Fréchet distance*) [HPN$^+$16]; etc. Several of these questions have been studied only in the case of the plane, and extensions to surfaces are still open.

ELEMENTARY MOVES AND UNCROSSING

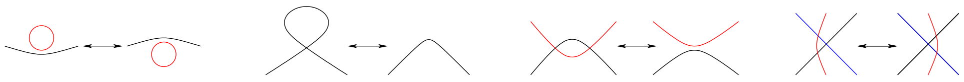
FIGURE 23.4.1
The four Reidemeister moves, up to ambient isotopy. The pictures represent the intersection of the union of the curves with a small disk on S; in particular, in these pictures, the regions bounded by the curves are homeomorphic to disks, and no other parts of curves intersect the parts of the curves shown.

1. *Elementary moves:* Every family of closed curves in general position can be made minimally crossing by a finite sequence of Reidemeister moves, described in Figure 23.4.1. If a closed curve has k self-crossings, $\Omega(k^2)$ Reidemeister moves can be needed; this is tight if the curve is homotopic to a simple curve, but in general no subexponential upper bound seems to be known [CE16]. Actually, one can deform a family of curves continuously to make it minimally crossing without increasing the total number of crossings at any step, and moreover, in a minimally crossing family, each curve is itself minimally self-crossing, and each pair of curves is minimally crossing [GS97] (see also [HS94a]). There are other characterizations of curves not in minimally crossing position [HS85].
2. *Making curves simple:* Let G be a graph cellularly embedded on a surface S. One can decide whether an input curve, represented by a closed walk in G, is homotopic to a simple closed curve in S in near-linear time. More generally, one can compute the minimum number of self-intersections of a curve in S homotopic to an input closed walk in G, and the minimum number of intersections between two curves in S respectively homotopic to two input closed walks in G, in quadratic time [DL17].
3. *Untangling curves by a homeomorphism:* Given two families of disjoint, simple curves, one can try to minimize the number of crossings between them by changing one of them by a homeomorphism of the surface; some bounds are known on the number of crossings that one can achieve [MSTW16].

4. *Simultaneous graph drawing:* This also relates to the problem of embedding two input graphs on the same surface in a way that the embeddings cross each other few times. Here also some results are known [Neg01, RS05, HKMT16]; one can also require both combinatorial maps to be fixed.
5. *Number of homotopy classes:* How many simple closed curves in different homotopy classes can one draw such that they pairwise cross at most k times, for a given integer k? On orientable surfaces of genus $g \geq 2$ without boundary and $k = 0$, the answer is $3g - 2$ (a pants decomposition, see below, together with a contractible closed curve). The problem is more interesting for larger values of k; it was recently proved that, for fixed k, the number of curves one can draw is polynomial in the genus [Prz15].

23.5 OPTIMIZATION: SHORTEST CURVES AND GRAPHS

The problem of computing shortest curves and graphs satisfying certain topological properties on surfaces has been widely considered. This leads to problems with a flavor of combinatorial optimization.

For these problems to be meaningful, a metric must be provided. In computational geometry, one could naturally consider piecewise linear surfaces in some Euclidean space (perhaps $\mathbb{R}^3$); however, efficient algorithms for computing shortest paths in such surfaces [MMP87, CH96] need additional assumptions because distances involve square roots, which leads to deep and unrelated questions on the complexity of comparing sums of square roots [Blö91]. Furthermore, in the context of graph problems in the specific case of surface-embedded graphs (Section 23.7 below), that model would be insufficient. The notions of combinatorial and cross-metric surfaces, defined below, have been developed to avoid these technical distractions, and are suitable in various settings. On the other hand, with an oracle for shortest path computations, several of the results in this section extend to more geometric settings, for example piecewise linear surfaces in some Euclidean space (see, e.g., [EW05, Sect. 3.6]).

GLOSSARY

Discrete metrics on surfaces

Combinatorial surface: A combinatorial surface is the data of a cellular graph embedding G, with positive weights on the edges. The only allowed curves are walks in G; the length of a curve is the sum of the weights of the edges of G traversed by the curve, counted with multiplicity. Algorithmically, curves are stored as closed walks in G. The complexity of the combinatorial surface is the complexity of the embedding G (asymptotically, its number of edges).

Cross-metric surface: A cross-metric surface [CVE10] is also the data of a cellular graph embedding G on some surface S, with positive weights on the edges. However, in contrast to the combinatorial surface model, here the curves are drawn on the surface S in *general position* with respect to G; the length of a curve is the sum of the weights of the edges of G crossed by the curve, counted with multiplicity. Algorithmically, a family of curves (or a graph) on a cross-metric surface is stored by the combinatorial map of the arrangement of that family of curves (or graph) together with G. The complexity of the cross-metric surface is the complexity of the embedding G (asymptotically, its number of edges).

Without loss of generality, one could draw the curves in a neighborhood of the dual graph G^* of G. Pushing them completely onto G^* would transform them into curves on the combinatorial surface defined by G^*. However, the cross-metric surface defined by G retains more information than the combinatorial surface defined by G^*: In the latter case, when curves share edges of G^*, they automatically overlap; the cross-metric model allows us to make them disjoint except at some well-defined crossing points. (We should point out that it is still possible to define the notion of crossing between two curves in a combinatorial surface, but this is still insufficient for some of the algorithms described below.)

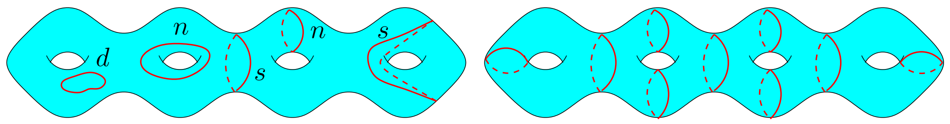

FIGURE 23.5.1
Left: Some closed curves on surfaces, (d) disk-bounding, (n) non-separating, (s) splitting. Right: A pants decomposition of a surface.

Types of simple closed curves

Let γ be a simple closed curve in the interior of a surface S. See Figure 23.5.1.

Disk-bounding curve: γ is disk-bounding if the surface S cut along γ (denoted by $S\backslash\!\backslash\gamma$) has two connected components, one of which is homeomorphic to the disk.

Separating curve: γ is separating if $S\backslash\!\backslash\gamma$ has two connected components.

Splitting curve: γ is splitting if γ is separating but not disk-bounding.

Essential curve: γ is essential if no component of $S\backslash\!\backslash\gamma$ is a disk or an annulus.

Topological decompositions

Cut graph: A cut graph is a graph G embedded on a surface S such that $S\backslash\!\backslash G$ is homeomorphic to a closed disk.

System of loops: A system of loops on a surface without boundary is a cut graph with a single vertex. See Figure 23.1.2.

Canonical system of loops: A system of loops G on a surface without boundary S is canonical if the edges of the polygon $S\backslash\!\backslash G$ appear in the same order as in a canonical polygonal schema (see Section 23.1)

Pants decomposition: A pants decomposition of an orientable surface S is a family Γ of simple, disjoint closed curves on S such that $S\backslash\!\backslash\Gamma$ is a disjoint union of pairs of pants. See Figure 23.5.1.

Octagonal decomposition: An octagonal decomposition of an orientable surface S without boundary is a family Γ of closed curves on S such that each (self-)intersection point in Γ is a crossing between exactly two closed curves, and each face of the arrangement of Γ on S is an octagon (a disk with eight sides).

Homology

In the context of graphs on surfaces, *one-dimensional homology on surfaces over the field* $\mathbb{Z}/2\mathbb{Z}$ is used; it can be described somewhat more concisely than more general homology theories. Let S be a surface. Here we assume graph embeddings to be piecewise linear (with respect to a fixed triangulation of S).

Homological sum: By the previous assumption, the closure of the symmetric difference of the images of two graph embeddings G and G' is the image of some graph embedding G'', called the homological sum of G and G'. (G'' is defined up to subdivision of edges with degree-two vertices, insertion of isolated vertices, and the reverse operations; here, graph embeddings are considered up to such operations.)

Homology cycle: A graph G embedded on S is a homology cycle if every vertex of G has even degree. The set of homology cycles forms a vector space over the field $\mathbb{Z}/2\mathbb{Z}$: The empty graph is the trivial element and addition is the homological sum.

Homology boundary: A graph G embedded on S is a homology boundary if the faces of G can be colored in two colors, say black and white, such that G is the "boundary" between the two colors: Exactly one side of each edge of G is incident to a black face. The set of homology boundaries forms a vector space over $\mathbb{Z}/2\mathbb{Z}$. Every homology boundary is a homology cycle.

Homology group: It is the $\mathbb{Z}/2\mathbb{Z}$-vector space, denoted by $H_1(S)$, that is the quotient of the homology cycles by the homology boundaries. A graph embedding is ***homologically trivial*** if it is a homology boundary.

The homology of sets of loops or closed curves can be defined similarly, because these loops and closed curves are the images of some graph embedding. Using the more advanced theory of *singular homology* one can remove the restriction of dealing with piecewise-linear graph embeddings.

BASIC PROPERTIES

1. A simple closed curve is disk-bounding if and only if it is contractible.
2. A simple closed curve is separating if and only if it is homologically trivial.
3. The homology group of a surface S without boundary has dimension $\bar{g}$, the Euler genus of S, and is generated by the loops appearing on the boundary of a canonical polygonal schema.

SHORTEST CURVES

Deciding whether a simple closed curve in a cross-metric (or combinatorial) surface is separating or disk-bounding can be done in time linear in the size of the data structure used to store the cellular graph and the curve; this boils down to determining whether some graph is connected, or whether some surface is a disk (which is easy using Euler's formula). Here we consider the optimization version, by looking for shortest curves with a given topological type in a combinatorial or cross-metric surface. Non-disk-bounding or non-separating curves are of particular interest, because cutting along such a curve simplifies the topology of a surface. Below we use *non-trivial* as a shorthand for either non-disk-bounding or non-separating.

TABLE 23.5.1 Algorithms for shortest non-trivial closed curves on surfaces without boundary, depending on whether the graph is weighted and whether it is directed. "Non-sep" and "non-db" mean non-separating and non-disk-bounding, respectively; k is the size of the output. The best complexities known to date are in bold (there can be several of them in each category due to the tradeoff between g, n, and k). Of course, the undirected case reduces to the directed case, and the unweighted case reduces to the weighted case; in each cell, we do not repeat the algorithms that are available for more general scenarios.

	UNDIRECTED	DIRECTED
WEIGHTED	$\boldsymbol{O(n^2 \log n)}$ [EHP04] $\left.\begin{array}{ll} O(g^{3/2}n^{3/2}\log n) & \text{non-sep} \\ g^{O(g)}n^{3/2} & \text{non-db} \end{array}\right\}$ [CM07] $g^{O(g)}n\log n$ [Kut06] $O(g^3 n\log n)$ [CC07] $\boldsymbol{O(g^2 n\log n)}$ [CCE13] $g^{O(g)}n\log\log n$ [INS+11] $\boldsymbol{2^{O(g)}n\log\log n}$ [Fox13] $\boldsymbol{O(gn\log n)}$ for 2-approx. [EHP04]	$O(n^2\log n)$ [CCVL16] $\boldsymbol{O(g^{1/2}n^{3/2}\log n)}$ [CCVL16] $2^{O(g)}n\log n$ non-sep [EN11b] $\left.\begin{array}{ll} \boldsymbol{O(g^2 n\log n)} & \text{non-sep} \\ g^{O(g)}n\log n & \text{non-db} \end{array}\right\}$ [Eri11] $\boldsymbol{O(g^3 n\log n)}$ non-db [Fox13]
UNWEIGHTED	$O(n^3)$ [Tho90] (see [MT01]) $\boldsymbol{O(n^2)}$ [CCVL12] $\boldsymbol{O(gnk)}$ [CCVL12] $\boldsymbol{O(gn/\varepsilon)}$ for $(1+\varepsilon)$-approx. [CCVL12]	$\boldsymbol{O(n^2)}$ [CCVL16] $\boldsymbol{O(gnk)}$ [CCVL16]

1. *Structural properties:* In a combinatorial surface, a shortest noncontractible or non-null-homologous loop based at a vertex x is made of two shortest paths from x and of a single edge (this is the so-called 3-path condition [Tho90]). It follows that the globally shortest non-contractible and non-null-homologous closed curves do not repeat vertices and edges, and are also shortest non-disk-bounding and non-separating closed curves. More generally, in the algorithms mentioned below, a typical tool is to prove a bound on the number of crossings between the (unknown) shortest curve and any shortest path.

2. *Different scenarios for shortest non-trivial curves:* Table 23.5.1 summarizes the running times of the known algorithms. In such problems, it is relevant to look for more efficient algorithms in the case where the genus g is smaller compared to the complexity n of the graph defining the surface. The standard scenario, which is the only one considered elsewhere in this chapter, is that of a combinatorial (or equivalently, cross-metric) surface (the undirected,

weighted case, in the upper left corner in Table 23.5.1). One can also aim for faster algorithms in the *unweighted* case (unit weights). Finally, one can extend the techniques to the case of *directed* graphs, where the edges of the combinatorial surface are directed and can only be used in a specified direction (equivalently, the edges of the cross-metric surface can only be crossed in a specific direction).

3. *Other topological types:* Shortest simple closed curves of other topological types have been investigated as well (in the following, n denotes the complexity of the cross-metric surface): shortest splitting curves [CCV$^+$08] (NP-hard, but computable in $O(n \log n)$ time for fixed genus); shortest essential curves [EW10] ($O(n^2 \log n)$ time, or $O(n \log n)$ for fixed genus and number of boundaries—in this case, surfaces with boundary require more sophisticated techniques); and non-separating curves which are shortest in their (unspecified) homotopy class [CDEM10] ($O(n \log n)$).

4. *Shortest homotopic curves:* A slightly different problem is that of computing a shortest curve homotopic to a given curve (either a path or a closed curve); this is also doable in small polynomial time, using octagonal decompositions to build a part of the universal covering space [CVE10] (earlier algorithms dealt with simple curves only, with an iterated shortening process that leads to a global optimum [CVL05, CVL07]).

5. *Shortest paths:* All these algorithms rely on shortest path computations on combinatorial (or cross-metric) surfaces, which can be done in $O(n \log n)$ time using Dijkstra's algorithm [Dij59] classically speeded up with Fibonacci heaps [FT87] in the primal (or dual) graph. This actually computes the shortest paths from a single source to all other vertices of the combinatorial surface. Other algorithms are available for computing multiple shortest paths quickly under some conditions on the locations of the endpoints [CCE13].

SHORTEST DECOMPOSITIONS

Decompositions of surfaces are central in topology; for example, the standard proof of the classification theorem transforms an arbitrary cut graph into a canonical system of loops. Many algorithms described in the previous subsection rely on topological decompositions and their properties.

1. *Shortest cut graph:* The problem of computing a shortest cut graph on a cross-metric surface has been extensively studied. Computing the shortest cut graph is NP-hard, but there is an $O(\log^2 g)$-approximation algorithm that runs in $O(g^2 n \log n)$ time [EHP04]. Moreover, for every $\varepsilon > 0$ one can compute a $(1+\varepsilon)$-approximation in $f(\varepsilon, g) \cdot n^3$ time, for some function f [CAM15]. If one is looking for a shortest cut graph with a specified vertex set P (for example, a shortest system of loops with given basepoint [EW05]), then there is an algorithm with running time $O(n \log n + gn + |P|)$ [CV10]. At the root of several of these articles lies the *tree-cotree property* [Epp03]: If G is a cellular graph embedding, there exists a partition (T, C, X) of the edges of G such that T is a spanning tree of G and the edges dual to C form a spanning tree of the dual graph G^*. Contracting T and deleting C transforms G into a system of loops, each loop corresponding to an element of X.

2. *Other topological decompositions:* Some canonical system of loops (for orientable surfaces without boundary) can be computed in $O(gn)$ time [LPVV01]. An octagonal decomposition or a pants decomposition made of closed curves which are as short as possible in their respective homotopy classes can be computed in $O(gn \log n)$ time [CVE10]. But in general the complexity of computing shortest such decompositions is open. On the other hand, there are bounds on the maximum length of some decompositions, assuming that the combinatorial surface is an unweighted triangulation, or, dually, that the cross-metric surface is unweighted and each vertex has degree three [CVHM15].
3. *Stretch:* Let S be a cross-metric surface, and let G be the associated embedded graph. The stretch of S is the minimum of the *product* of the lengths of γ and δ, over all closed curves γ and δ crossing exactly once. This quantity is related to the planar crossing number and the size of a largest toroidal grid minor of G^* [HC10], and can be computed in small polynomial time [CCH14].

HOMOLOGY AND ITS RELATION TO CUTS AND FLOWS

As hinted above, homology is useful because a simple closed curve is separating if and only if it is null-homologous; the algorithms for computing shortest non-separating closed curves actually compute shortest non-null-homologous closed curves, which turn out to be simple.

Homology is a natural concept; in particular, it is interesting to look for a family of closed curves, of minimum total length, the homology classes of which generate the homology group. Some efficient algorithms have been given for this purpose [EW05], also in connection with an algorithm to compute a minimum cycle basis of a surface-embedded graph [BCFN16].

Another reason for the importance of homology is its relation to cuts: Given a graph G cellularly embedded on a surface S without boundary, the (s,t)-cuts in G are dual to the subgraphs of G^* in some fixed homology class on the surface obtained from S by removing the faces of G^* containing s and t. Thus, computing minimum cuts amounts to computing shortest homologous subgraphs. This property has been exploited to study general graph problems, where better algorithms can be designed in the specific case of graphs embedded on a fixed surface, to:

1. compute minimum (s,t)-cuts in near-linear time [CEN09, EN11b]. The best algorithm runs in $2^{O(g)} n \log n$ time, where g is the genus [EN11b], and relies on the *homology cover*, a particular type of covering space;
2. compute maximum (s,t)-flows faster, by exploiting further the duality between flows and cuts [CEN12, BEN$^+$16];
3. count and sample minimum (s,t)-cuts efficiently [CFN14];
4. compute global minimum cuts efficiently (without fixing s and t) [EFN12];
5. deal with other problems, e.g., to compute the edge expansion and other connectivity measures [Pat13] or to bound the space complexity of bipartite matching [DGKT12].

23.6 ALGORITHMS FOR GRAPHS EMBEDDED ON A FIXED SURFACE

Some general graph problems can be solved faster in the special case of graphs embedded on a fixed surface. Examples include cut and flow problems (see previous section), multicommodity problems, domination and independence problems, connectivity problems (Steiner tree, traveling salesman problem, etc.), disjoint paths problems, shortest paths problems, subgraph problems, and more.

Sometimes the problems are solvable in polynomial-time on arbitrary graphs, and the goal is to obtain faster algorithms for surface-embedded graphs. But in many cases, the problems considered are NP-hard on arbitrary graphs, and polynomial-time algorithms are obtained for graphs embeddable on a fixed surface (occasionally by fixing some other parameters of the problem). Typically, optimization problems are considered, in which case it is relevant to look for approximation algorithms.

The methods involved usually combine topological aspects (as described above) with techniques from structural and algorithmic graph theory.

GLOSSARY

Minor: A graph H is a minor of another graph G if H can be obtained from G by removing edges and isolated vertices, and contracting edges.

Minor-closed family: A family $\mathscr{F}$ of graphs is minor-closed if every minor of a graph in $\mathscr{F}$ is also in $\mathscr{F}$.

Tree decomposition: A tree decomposition of a graph $G = (V, E)$ is a tree T in which each node is labeled by a subset of V, such that:

- for each $v \in V$, the set of nodes in T whose labels contain v induces a non-empty connected subtree of T, and
- if G has an edge connecting vertices u and v, then the label of at least one node of T contains both u and v.

Width: The width of a tree decomposition is the maximum cardinality of the labels minus one.

Treewidth: The treewidth of a graph G is the minimum width of a tree decomposition of G.

SURVEY OF TECHNIQUES

Central to algorithmic and structural graph theory is the study of minor-closed families of graphs; by a deep result of Robertson and Seymour [RS04], for each such family $\mathscr{F}$, there is a *finite* set $X_{\mathscr{F}}$ of graphs such that $G \in \mathscr{F}$ if and only if no graph in $X_{\mathscr{F}}$ is a minor of G. We refer to [KM07] for a survey on these structural aspects.

The graphs embeddable on a fixed surface form a minor-closed family, and have the benefit that they can be studied using topological techniques. Robertson and Seymour provide a decomposition theorem for minor-closed families of graphs involving graphs embeddable on a fixed surface [RS03]; efficient algorithms for surface-embedded graphs are sometimes extended to minor-closed families of graphs (different from the family of all graphs).

It is impossible to list all results in algorithms for surface-embedded graphs here, so we focus on general methods. Several algorithms are based on topological techniques described in the previous sections (in particular, shortest non-trivial curves or shortest decompositions), in several cases with advanced algorithmic techniques [EN11a, KKS11, ES14, PPSL14]. Sometimes the same techniques have led to new results for planar graphs [Eri10, EN11c, CV17b]. Methods applicable to several algorithmic problems have also emerged, in many cases extending previous ones invented for planar graphs:

1. *Graph separators and treewidth:* Let G be a graph with n vertices embedded on a surface with genus g. In linear time, one can compute a balanced separator of size $O(\sqrt{gn})$, namely, a set of $O(\sqrt{gn})$ vertices whose removal leaves a graph without connected component of more than $2n/3$ vertices [GHT84, Epp03]. Also, the treewidth of G is $O(\sqrt{gn})$.
2. *Dynamic programming:* Small treewidth implies efficient algorithms using dynamic programming in arbitrary graphs. When the graph is embedded, one can exploit this fact to obtain algorithms with smaller dependence on the treewidth for some problems [Bon12, RST13, RST14].
3. *Irrelevant vertex technique:* Several graph problems enjoy the following property [Thi12]: If the input graph has large treewidth, there exists an irrelevant vertex, whose removal creates an equivalent instance of the problem (e.g., a vertex at the center of a large grid minor). This property is widely used in structural graph theory and has been exploited several times in the context of algorithms for surface-embedded graphs [KR07, KT12, RS12].
4. *Polynomial-time approximation schemes (PTASs):* Baker [Bak94] has introduced a technique for designing approximation schemes for some optimization problems with local constraints in planar graphs: She has showed that one can delete a small part of the input graph without changing too much the value of the solution and such that the resulting graph has small treewidth. The technique has been extended to graphs embeddable on a fixed surface [Epp00], to graphs that can be drawn on a fixed surface with a bounded number of crossings per edge [GB07], and to more general *contraction-closed* problems where contraction instead of deletion must be used [Kle05, DHM10]. A crucial step in making the latter technique effective is the construction of a *spanner*: In the case of a minimization problem, this is a subgraph of the input graph containing a near-optimal solution and whose weight is linear in that of the optimal solution. *Brick decomposition* is a technique that builds spanners for some problems, originally in planar graphs, but also sometimes in graphs on surfaces [BDT14].
5. *Bidimensionality:* This theory [Thi15, DFHT05, DHT06, DH08] applies to minimization problems on unweighted graphs where contracting an edge of the graph does not increase the value of the solution, and where the value of the solution in grid graphs (and generalizations) is large. It leads to output-sensitive algorithms for graphs embeddable on a fixed surface with running time of the form $2^{O(\sqrt{k})} \cdot n^{O(1)}$, where k is the value of the solution and n is the input

size. This also provides PTASs in some cases [DH05]. For the problems where bidimensionality applies, PTASs can sometimes also be obtained in weighted graphs using a different framework [CACV+16].

6. *Stochastic embeddings:* Let $G = (V, E)$ and $G' = (V', E')$ be positively edge-weighted graphs. A *non-contracting metric embedding* f from G to G' is a mapping from V to V' such that $d'(f(x), f(y)) \geq d(x, y)$ for each $x, y \in V$, where d and d' represent shortest path distances in G and G', respectively. The *distortion* of f is the maximum of $d'(f(x), f(y))/d(x, y)$ over all $x \neq y \in V$ (see Chapter 8). Every graph G embeddable on an orientable surface S of genus g admits a probability distribution of non-contracting metric embeddings into planar graphs such that for each $x, y \in V$, one has $\mathbb{E}[d'(f(x), f(y))] \leq O(\log g) \cdot d(x, y)$, where the expectation is over all f in the distribution [Sid10]. This reduces several optimization problems on graphs on S to the same problem in planar graphs, up to the loss of an $O(\log g)$ factor. Actually, such a distribution can be computed in polynomial time even if no embedding of G on S is known [MS12].

23.7 OTHER MODELS

A rather large number of results relate to the concepts described in this chapter, and it would be impossible to cover them all. Below, we provide a selection of miscellaneous results that consider other models for representing graphs on surfaces.

COMPUTATIONAL TOPOLOGY IN THE PLANE WITH OBSTACLES

The plane minus finitely many points or polygons ("obstacles") forms a (non-compact) surface S. Taking any cellular graph embedding on S makes S a combinatorial (or cross-metric) surface, so most of the topological algorithms above apply. However, it is much more natural to consider arbitrary piecewise-linear curves in S, whose length is defined by the Euclidean metric. In this model, S is defined by the obstacles (a finite set of disjoint simple polygons, for simplicity of exposition); curves are arbitrary polygonal lines avoiding the interior of the obstacles. Some of the problems defined in the previous sections and related problems have been studied in this model:

1. *Homotopy and isotopy tests:* There are efficient algorithms to test whether two curves are (freely) homotopic [CLMS04], or whether two graphs are isotopic [CVM14].
2. *Shortest homotopic paths* can be computed efficiently as well [HS94b, Bes03, EKL06]; see also Section 31.2. A variant where several simple and disjoint paths must be shortened while preserving their homotopy class and keeping their neighborhoods simple and disjoint (i.e., the paths are "thick") has also been investigated [GJK+88].
3. *Shortest disjoint paths:* Here the goal is to compute disjoint paths with minimum total length (or, more precisely, non-crossing paths, since in the limit case, the solution may consist of overlapping paths). If the endpoints lie on the boundary of a bounded number of obstacles, the problem is solvable in polynomial time [EN11c].

4. *Other results* include a constant-factor approximation algorithm for the shortest pants decomposition in the case where the obstacles are points [Epp09] and an algorithm for computing the homotopic Fréchet distance, a measure of similarity between curves that takes the obstacles into account topologically [CCV$^+$10, HPN$^+$16].

SIMPLE AND DISJOINT CURVES IN GRAPHS

In the cross-metric model, defined by a cellularly embedded graph G, one can think of curves as being drawn in a neighborhood of G^*. So, intuitively, curves are drawn in G^*, but they can share vertices and edges of G^* while being simple and pairwise disjoint.

It is very natural, especially in topological graph theory, to forbid such overlaps: A set of disjoint simple curves cannot repeat any vertex or edge of G^*. Many of the problems mentioned in the previous sections make sense in this setup, which turns out to be generally more difficult to handle than the cross-metric model. In this model, the following results are known (here by *circuit* we mean a closed curve in the graph without repeated vertex, and containing at least one edge):

1. Determining whether there exists a separating (resp., splitting) circuit is NP-complete [CCVL11].
2. Determining some contractible (resp., non-contractible, resp., non-separating) circuit, if such a circuit exists, is possible in linear time, even if one requires the circuit to pass through a given vertex [CCVL11].
3. Computing a *shortest* contractible circuit is possible in polynomial time, but if one requires the circuit to pass through a given vertex, the problem becomes NP-hard [Cab10].
4. Computing a *shortest* separating circuit is NP-hard [Cab10].
5. There is a combinatorial characterization on whether curves can be made simple and disjoint in the graph by a homotopy on the surface [Sch91]. In the case of a planar surface with boundaries, this leads to a polynomial-time algorithm [Sch90b, Th. 31], which in turn has some algorithmic consequences on the problem of computing vertex-disjoint paths in planar graphs [Sch90b, Th. 34]. See also [Sch03, Ch. 76].

NORMAL CURVES ON SURFACES

Let Γ be a family of disjoint simple closed curves on a surface S in general position with respect to a triangulation T of S. A natural way to represent Γ, as described in the previous sections, is by its arrangement with T. *Normal curves* are a more economical representation, at the price of a mild condition: For every triangle t of T, the intersection of the image of Γ with t must be a set of (disjoint simple) paths, called *normal arcs*, connecting *different* sides of t. For such a Γ, and for each triangle t of T, one stores three integers recording the number of normal arcs connecting each of the three pairs of sides of t. Overall, Γ is described by $3n$ non-negative integers, where n is the number of triangles in T. Conversely, given a vector of $3n$ non-negative integers, one can unambiguously reconstruct Γ

up to *normal isotopy*, that is, up to an ambient isotopy that leaves the edges of T globally unchanged.

To store the vector of normal coordinates, $O(n \log(X/n))$ bits are needed, where X is the number of crossing points of Γ with T. In contrast, representing these curves on a cross-metric surface requires at least to store a constant amount of information per vertex of the arrangement, which is $\Theta(n+X)$ in total. So the normal curve representation can be exponentially compressed compared to the "naïve" one. Despite this, in time polynomial in the input size one can:

1. count the number of connected components of a normal curve (note that a "normal curve" does not have to be connected), and partition these components according to their (normal or not) isotopy classes, given by their multiplicities and the normal coordinates of a representative [SSŠ02, EN13];
2. decide whether two normal curves are isotopic [SSŠ02, EN13];
3. compute the algebraic [SSŠ02, EN13] or the geometric [SSŠ08] intersection number of two normal curves. (The algebraic intersection number of γ and δ is the sum, over all crossings between γ and δ, of the sign of the crossing, which is $+1$ if γ crosses δ from left to right at that crossing point and -1 otherwise; this is well-defined if the surface is orientable, since it is invariant by isotopy. The geometric intersection number of γ and δ is the minimum number of crossings between curves γ' and δ' isotopic to γ and δ.)

These problems have been initially studied using *straight-line programs*, a concise encoding of words over a finite alphabet; many algorithms on words can be solved efficiently using the straight-line program representation, in particular because straight-line programs can represent exponentially long words; this leads to efficient algorithms for normal curves [SSŠ02, SSŠ08]. The same and other problems have been revisited using more topological techniques [EN13]. Normal curves are the lower-dimensional analog of *normal surfaces*, widely used in three-dimensional topology.

23.8 OTHER RESOURCES

Books. Graphs on surfaces from a combinatorial viewpoint are treated in detail in [MT01]; see also [GT87]. For basic surface topology, we recommend [Arm83, Sti93, Hen94].

Survey. [Eri12] surveys optimization problems for surface-embedded graphs, providing more details on a large fraction of Section 23.5.

Course notes and unpublished material. [Eri13] provides some notes in computational topology with a strong emphasis on graphs on surfaces. [CV12, CV17a] survey some algorithms for optimization of graphs and curves on surfaces. [DMST11] emphasizes graph algorithms for surface-embedded graphs.

RELATED CHAPTERS

Chapter 20: Polyhedral maps
Chapter 31: Shortest paths and networks
Chapter 55: Graph drawing

REFERENCES

[AG05] P. Alliez and C. Gotsman. Recent advances in compression of 3D meshes. In N.A. Dodgson, M.S. Floater, and M.A. Sabin, editors, *Advances in Multiresolution for Geometric Modelling*, pages 3–26, Springer-Verlag, Berlin, 2005.

[Arc96] D. Archdeacon. Topological graph theory. A survey. *Congr. Numer.*, 115:5–54, 1996.

[Arm83] M.A. Armstrong. *Basic Topology.* Undergraduate Texts in Mathematics, Springer-Verlag, Berlin, 1983.

[Bak94] B.S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. ACM*, 41:153–180, 1994.

[BCFN16] G. Borradaile, E.W. Chambers, K. Fox, and A. Nayyeri. Minimum cycle and homology bases of surface embedded graphs. In *Proc. 32nd Sympos. Comput. Geom.*, vol 51 of *LIPIcs*, article 23, Schloss Dagstuhl, 2016.

[BCHK16] S. Beyer, M. Chimani, I. Hedtke, and M. Kotrbčík. A practical method for the minimum genus of a graph: Models and experiments. In *Proc. 15th Sympos. Experimental Algorithms*, vol. 9685 of *LNCS*, pages 75–88, Springer, Cham, 2016.

[BDT14] G. Borradaile, E.D. Demaine, and S. Tazari. Polynomial-time approximation schemes for subset-connectivity problems in bounded-genus graphs. *Algorithmica*, 68:287–311, 2014.

[BEKT16] B.A. Burton, M. Elder, A. Kalka, and S. Tillmann. 2-manifold recognition is in logspace. *J. Comput. Geom.*, 7:70–85, 2016.

[BEN$^+$16] G. Borradaile, D. Eppstein, A. Nayyeri, and C. Wulff-Nilsen. All-pairs minimum cuts in near-linear time for surface-embedded graphs. In *Proc. 32nd Sympos. Comput. Geom.*, vol. 51 of *LIPIcs*, article 22, Schloss Dagstuhl, 2016.

[Bes03] S. Bespamyatnikh. Computing homotopic shortest paths in the plane. *J. Algorithms*, 49:284–303, 2003.

[Bet12] J. Bettinelli. The topology of scaling limits of positive genus random quadrangulations. *Ann. Probab.*, 40:1897–1944, 2012.

[BKS98] M. de Berg, M. van Kreveld, and S. Schirra. Topologically correct subdivision simplification using the bandwidth criterion. *Cartography and GIS*, 25:243–257, 1998.

[Blö91] J. Blömer. Computing sums of radicals in polynomial time. In *Proc. 32nd IEEE Sympos. Found. Comp. Sci.*, pages 670–677, 1991.

[Bon12] P. Bonsma. Surface split decompositions and subgraph isomorphism in graphs on surfaces. In *Proc. 29th Sympos. Theoret. Aspects Comp. Sci.*, vol. 14 of *LIPIcs*, pages 531–542, Schloss Dagstuhl, 2012.

[Cab10] S. Cabello. Finding shortest contractible and shortest separating cycles in embedded graphs. *ACM Trans. Algorithms*, 6:24, 2010.

[Cab13] S. Cabello. Hardness of approximation for crossing number. *Discrete Comput. Geom.*, 49:348–358, 2013.

[CACV$^+$16] V. Cohen-Addad, É. Colin de Verdière, P.N. Klein, C. Mathieu, and D. Meierfrankenfeld. Approximating connectivity domination in weighted bounded-genus graphs. In *Proc. 48th ACM Sympos. Theory Comput.*, pages 584–597, 2016.

[CAM15] V. Cohen-Addad and A. de Mesmay. A fixed parameter tractable approximation scheme for the optimal cut graph of a surface. In *Proc. 23rd European Sympos. Algorithms*, vol. 9294 of *LNCS*, pages 386–398, Springer, Berlin, 2015.

[CC07] S. Cabello and E.W. Chambers. Multiple source shortest paths in a genus g graph. In *Proc. 18th ACM-SIAM Sympos. Discrete Algorithms*, pages 89–97, 2007.

[CCE13] S. Cabello, E.W. Chambers, and J. Erickson. Multiple-source shortest paths in embedded graphs. *SIAM J. Comput.*, 42:1542–1571, 2013.

[CCH14] S. Cabello, M. Chimani, and P. Hliněný. Computing the stretch of an embedded graph. *SIAM J. Discrete Math.*, 28:1391–1401, 2014.

[CCV$^+$08] E.W. Chambers, É. Colin de Verdière, J. Erickson, F. Lazarus, and K. Whittlesey. Splitting (complicated) surfaces is hard. *Comput. Geom.*, 41:94–110, 2008.

[CCV$^+$10] E.W. Chambers, É. Colin de Verdière, J. Erickson, S. Lazard, F. Lazarus, and S. Thite. Homotopic Fréchet distance between curves — or, walking your dog in the woods in polynomial time. *Comput. Geom.*, 43:295–311, 2010.

[CCVL11] S. Cabello, É. Colin de Verdière, and F. Lazarus. Finding cycles with topological properties in embedded graphs. *SIAM J. Discete Math.*, 25:1600–1614, 2011.

[CCVL12] S. Cabello, É. Colin de Verdière, and F. Lazarus. Algorithms for the edge-width of an embedded graph. *Comput. Geom.*, 45:215–224, 2012.

[CCVL16] S. Cabello, É. Colin de Verdière, and F. Lazarus. Finding shortest non-trivial cycles in directed graphs on surfaces. *J. Comput. Geom.*, 7:123–148, 2016.

[CDEM10] S. Cabello, M. DeVos, J. Erickson, and B. Mohar. Finding one tight cycle. *ACM Trans. Algorithms*, 6:61, 2010.

[CDP04] S.-W. Cheng, T.K. Dey, and S.-H. Poon. Hierarchy of surface models and irreducible triangulations. *Comput. Geom.*, 27:135–150, 2004.

[CE16] H.-C. Chang and J. Erickson. Untangling planar curves. In *Proc. 32nd Sympos. Comput. Geom.*, vol. 51 of *LIPIcs*, article 29, Schloss Dagstuhl, 2016.

[CEN09] E.W. Chambers, J. Erickson, and A. Nayyeri. Minimum cuts and shortest homologous cycles. In *Proc. 25th Sympos. Comput. Geom.*, pages 377–385, ACM Press, 2009.

[CEN12] E.W. Chambers, J. Erickson, and A. Nayyeri. Homology flows, cohomology cuts. *SIAM J. Comput.*, 41:1605–1634, 2012.

[CFN14] E.W. Chambers, K. Fox, and A. Nayyeri. Counting and sampling minimum cuts in genus g graphs. *Discrete Comput. Geom.*, 52:450–475, 2014.

[CH96] J. Chen and Y. Han. Shortest paths on a polyhedron. *Internat. J. Comput. Geom. Appl.*, 6:127–144, 1996.

[Chu11] J. Chuzhoy. An algorithm for the graph crossing number problem. In *Proc. 43rd ACM Sympos. Theory Comput.*, pages 303–312, 2011.

[CL09] E.W. Chambers and D. Letscher. On the height of a homotopy. In *Proc. 21st Canad. Conf. Comput. Geom.*, pages 103–106, 2009. See also the erratum at `mathcs.slu.edu/~chambers/papers/hherratum.pdf`.

[CLMS04] S. Cabello, Y. Liu, A. Mantler, and J. Snoeyink. Testing homotopy for paths in the plane. *Discrete Comput. Geom.*, 31:61–81, 2004.

[CM07] S. Cabello and B. Mohar. Finding shortest non-separating and non-contractible cycles for topologically embedded graphs. *Discrete Comput. Geom.*, 37:213–235, 2007.

[CM13] S. Cabello and B. Mohar. Adding one edge to planar graphs makes crossing number hard. *SIAM J. Comput.*, 42:1803–1829, 2013.

[CN00] G. Cairns and Y. Nicolayevsky. Bounds for generalized thrackles. *Discrete Comput. Geom.*, 23:191–206, 2000.

[CV10] É. Colin de Verdière. Shortest cut graph of a surface with prescribed vertex set. In *Proc. 18th European Sympos. Algorithms, part 2*, vol. 6347 of *LNCS*, pages 100–111, Springer, Berlin, 2010.

[CV12] É. Colin de Verdière. *Topological algorithms for graphs on surfaces.* Habilitation thesis, École normale supérieure, Paris, 2012. Available at `http://monge.univ-mlv.fr/~colinde/pub/12hdr.pdf`.

[CV17a] É. Colin de Verdière. *Algorithms for embedded graphs.* Course notes, `http://monge.univ-mlv.fr/~colinde/cours/all-algo-embedded-graphs.pdf`, 2017.

[CV17b] É. Colin de Verdière. Multicuts in planar and bounded-genus graphs with bounded number of terminals. *Algorithmica*, 78:1206–1224, 2017.

[CVE10] É. Colin de Verdière and J. Erickson. Tightening nonsimple paths and cycles on surfaces. *SIAM J. Comput.*, 39:3784–3813, 2010.

[CVHM15] É. Colin de Verdière, A. Hubard, and A. de Mesmay. Discrete systolic inequalities and decompositions of triangulated surfaces. *Discrete Comput. Geom.*, 53:587–620, 2015.

[CVK$^+$16] É. Colin de Verdière, V. Kaluža, P. Paták, Z. Patáková, and M. Tancer. A direct proof of the strong Hanani–Tutte theorem on the projective plane. In *Proc. 24th Sympos. Graph Drawing Network Visualization*, pages 454–467, vol. 9801 of *LNCS*, Springer, Cham, 2016.

[CVL05] É. Colin de Verdière and F. Lazarus. Optimal system of loops on an orientable surface. *Discrete Comput. Geom.*, 33:507–534, 2005.

[CVL07] É. Colin de Verdière and F. Lazarus. Optimal pants decompositions and shortest homotopic cycles on an orientable surface. *J. ACM*, 54:18, 2007.

[CVM14] É. Colin de Verdière and A. de Mesmay. Testing graph isotopy on surfaces. *Discrete Comput. Geom.*, 51:171–206, 2014.

[CW13] E.W. Chambers and Y. Wang. Measuring similarity between curves on 2-manifolds via homotopy area. In *Proc. 29th Sympos. Comput. Geom.*, pages 425–434, ACM Press, 2013.

[Deh12] M. Dehn. Transformation der Kurven auf zweiseitigen Flächen. *Math. Ann.*, 72:413–421, 1912.

[DFHT05] E.D. Demaine, F.V. Fomin, M.T. Hajiaghayi, and D.M. Thilikos. Subexponential parameterized algorithms on bounded-genus graphs and H-minor-free graphs. *J. ACM*, 52:866–893, 2005.

[DG99] T.K. Dey and S. Guha. Transforming curves on surfaces. *J. Comput. System Sci.*, 58:297–325, 1999.

[DGKT12] S. Datta, A. Gopalan, R. Kulkarni, and R. Tewari. Improved bounds for bipartite matching on surfaces. In *Proc. Sympos. Theoret. Aspects Comp. Sci.*, vol. 14 of *LIPIcs*, pages 254–265, Schloss Dagstuhl, 2012.

[DH05] E.D. Demaine and M. Hajiaghayi. Bidimensionality: new connections between FPT algorithms and PTASs. In *Proc. 16th ACM-SIAM Sympos. Discrete Algorithms*, pages 590–601, 2005.

[DH08] E.D. Demaine and M. Hajiaghayi. The bidimensionality theory and its algorithmic applications. *The Computer Journal*, 51:292–302, 2008.

[DHM10] E.D. Demaine, M. Hajiaghayi, and B. Mohar. Approximation algorithms via contraction decomposition. *Combinatorica*, 30:533–552, 2010.

[DHT06] E.D. Demaine, M. Hajiaghayi, and D.M. Thilikos. The bidimensional theory of bounded-genus graphs. *SIAM J. Discrete Math.*, 20:357–371, 2006.

[Dij59] E.W. Dijkstra. A note on two problems in connexion with graphs. *NumerṀath.*, 1:269–271, 1959.

[DL17] V. Despré and F. Lazarus. Computing the geometric intersection number of curves. In *Proc. 33rd Sympos. Comput. Geom.*, vol. 77 of *LIPIcs*, article 35, Schloss Dagstuhl, 2017.

[DMST11] E. Demaine, S. Mozes, C. Sommer, and S. Tazari. *Algorithms for Planar Graphs and Beyond*. Course notes, `http://courses.csail.mit.edu/6.889/fall11/lectures/`, 2011.

[EFN12] J. Erickson, K. Fox, and A. Nayyeri. Global minimum cuts in surface embedded graphs. In *Proc. 23rd ACM-SIAM Sympos. Discrete Algorithms*, pages 1309–1318, 2012.

[EHP04] J. Erickson and S. Har-Peled. Optimally cutting a surface into a disk. *Discrete Comput. Geom.*, 31:37–59, 2004.

[EK14] M. Elberfeld and K. Kawarabayashi. Embedding and canonizing graphs of bounded genus in logspace. In *Proc. 46th ACM Sympos. Theory of Computing*, pages 383–392, 2014.

[EKL06] A. Efrat, S.G. Kobourov, and A. Lubiw. Computing homotopic shortest paths efficiently. *Comput. Geom.*, 35:162–172, 2006.

[EN11a] J. Erickson and A. Nayyeri. Computing replacement paths in surface-embedded graphs. In *Proc. 22nd ACM-SIAM Sympos. Discrete Algorithms*, pages 1347–1354, 2011.

[EN11b] J. Erickson and A. Nayyeri. Minimum cuts and shortest non-separating cycles via homology covers. In *Proc. 22nd ACM-SIAM Sympos. Discrete Algorithms*, pages 1166–1176, 2011.

[EN11c] J. Erickson and A. Nayyeri. Shortest non-crossing walks in the plane. In *Proc. 22nd ACM-SIAM Sympos. Discrete Algorithms*, pages 297–308, 2011.

[EN13] J. Erickson and A. Nayyeri. Tracing compressed curves in triangulated surfaces. *Discrete Comput. Geom.*, 49:823–863, 2013.

[Epp00] D. Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27:275–291, 2000.

[Epp03] D. Eppstein. Dynamic generators of topologically embedded graphs. In *Proc. 14th ACM-SIAM Sympos. Discrete Algorithms*, pages 599–608, 2003.

[Epp09] D. Eppstein. Squarepants in a tree: sum of subtree clustering and hyperbolic pants decomposition. *ACM Trans. Algorithms*, 5, 2009.

[Eps66] D.B.A. Epstein. Curves on 2-manifolds and isotopies. *Acta Math.*, 115:83–107, 1966.

[Eri10] J. Erickson. Maximum flows and parametric shortest paths in planar graphs. In *Proc. 21st ACM-SIAM Sympos. Discrete Algorithms*, pages 794–804, 2010.

[Eri11] J. Erickson. Shortest non-trivial cycles in directed surface graphs. In *Proc. 27th Sympos. Comput. Geom.*, pages 236–243, ACM Press, 2011.

[Eri12] J. Erickson. Combinatorial optimization of cycles and bases. In A. Zomorodian, editor, *Advances in Applied and Computational Topology*, vol. 70 of *Proc. Sympos. Appl. Math.*, pages 195–228, AMS, Providence, 2012.

[Eri13] J. Erickson. *Computational Topology.* Course notes, `http://compgeom.cs.uiuc.edu/~jeffe/teaching/comptop/`, 2013.

[ES14] J. Erickson and A. Sidiropoulos. A near-optimal approximation algorithm for asymmetric TSP on embedded graphs. In *Proc. 13th Sympos. Comput. Geom.*, pages 130–135, ACM Press, 2014.

[EW05] J. Erickson and K. Whittlesey. Greedy optimal homotopy and homology generators. In *Proc. 16th ACM-SIAM Sympos. Discrete Algorithms*, pages 1038–1046, 2005.

[EW10] J. Erickson and P. Worah. Computing the shortest essential cycle. *Discrete Comput. Geom.*, 44:912–930, 2010.

[EW13] J. Erickson and K. Whittlesey. Transforming curves on surfaces redux. In *Proc. 24th ACM-SIAM Sympos. Discrete Algorithms*, pages 1646–1655, 2013.

[FGM88] M.L. Furst, J.L. Gross, and L.A. McGeoch. Finding a maximum-genus graph imbedding. *J. ACM*, 35:523–534, 1988.

[FM11] B. Farb and D. Margalit. *A Primer on Mapping Class Groups.* Princeton University Press, 2011.

[Fox13] K. Fox. Shortest non-trivial cycles in directed and undirected surface graphs. In *Proc. 24th ACM-SIAM Sympos. Discrete Algorithms*, pages 352–364, 2013.

[FT87] M.L. Fredman and R.E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM*, 34:596–615, 1987.

[GB07] A. Grigoriev and H.L. Bodlaender. Algorithms for graphs embeddable with few crossings per edge. *Algorithmica*, 49:1–11, 2007.

[GHT84] J.R. Gilbert, J.P. Hutchinson, and R.E. Tarjan. A separator theorem for graphs of bounded genus. *J. Algorithms*, 5:391–407, 1984.

[GJK$^+$88] S. Gao, M. Jerrum, M. Kaufmann, K. Mehlhorn, W. Rülling, and C. Storb. On continuous homotopic one layer routing. In *Proc. 4th Sympos. Comput. Geom.*, pages 392–402, ACM Press, 1988.

[GS90] S.M. Gersten and H.B. Short. Small cancellation theory and automatic groups. *Invent. Math.*, 102:305–334, 1990.

[GS97] M. de Graaf and A. Schrijver. Making curves minimally crossing by Reidemeister moves. *J. Combin. Theory Ser. B*, 70:134–156, 1997.

[GT87] J.L. Gross and T.W. Tucker. *Topological Graph Theory.* Wiley, New York, 1987.

[GW01] I. Guskov and Z.J. Wood. Topological noise removal. In *Proc. Graphics Interface*, pages 19–26, Canad. Inf. Process. Soc., Toronto, 2001.

[GY03] X. Gu and S.-T. Yau. Global conformal surface parameterization. In *Proc. Eurographics/ACM Sympos. Geom. Processing*, pages 127–137, 2003.

[Han34] C. Chojnacki (H. Hanani). Über wesentlich unplättbare Kurven im dreidimensionalen Raume. *Fund. Math.*, 23:135–142, 1934.

[HC10] P. Hliněný and M. Chimani. Approximating the crossing number of graphs embeddable in any orientable surface. In *Proc. ACM-SIAM Sympos. Discrete Algorithms*, pages 918–927, 2010.

[HD16] P. Hliněný and M. Derňár. Crossing number is hard for kernelization. In *Proc. 32nd Sympos. Comput. Geom.*, vol. 51 of *LIPIcs*, article 42, Schloss Dagstuhl, 2016.

[Hen94] M. Henle. *A Combinatorial Introduction to Topology.* Dover Publications, Mineola, 1994.

[HKMT16] A. Hubard, V. Kaluža, A. de Mesmay, and M. Tancer. Shortest path embeddings of graphs on surfaces. In *Proc. 32nd Sympos. Comput. Geom.*, vol. 51 of *LIPIcs*, article 43, Schloss Dagstuhl, 2016.

[HPN$^+$16] S. Har-Peled, A. Nayyeri, M. Salavatipour, and A. Sidiropoulos. How to walk your dog in the mountains with no magic leash. *Discrete Comput. Geom.*, 55:39–73, 2016.

[HS85] J. Hass and P. Scott. Intersections of curves on surfaces. *Israel J. Math.*, 51:90–120, 1985.

[HS94a] J. Hass and P. Scott. Shortening curves on surfaces. *Topology*, 33:25–43, 1994.

[HS94b] J. Hershberger and J. Snoeyink. Computing minimum length paths of a given homotopy class. *Comput. Geom.*, 4:63–98, 1994.

[HT74] J. Hopcroft and R. Tarjan. Efficient planarity testing. *J. ACM*, 21:549–568, 1974.

[INS$^+$11] G.F. Italiano, Y. Nussbaum, P. Sankowski, and C. Wulff-Nilsen. Improved algorithms for min cut and max flow in undirected planar graphs. In *Proc. 43rd ACM Sympos. Theory of Computing*, pages 313–322, 2011.

[Kel06] J.A. Kelner. Spectral partitioning, eigenvalue bounds, and circle packings for graphs of bounded genus. *SIAM J. Comput.*, 35:882–902, 2006.

[Ket99] L. Kettner. Using generic programming for designing a data structure for polyhedral surfaces. *Comput. Geom.*, 13:65–90, 1999.

[KKS11] K. Kawarabayashi, P.N. Klein, and C. Sommer. Linear-space approximate distance oracles for planar, bounded-genus and minor-free graphs. In *Proc. Int. Coll. Automata, Languages and Progr., part 1*, vol. 6755 of *LNCS*, pages 135–146, Springer, Berlin, 2011.

[Kle05] P.N. Klein. A linear-time approximation scheme for planar weighted TSP. In *Proc. 46th IEEE Sympos. Found. Comp. Sci.*, pages 647–657, 2005.

[KM07] K. Kawarabayashi and B. Mohar. Some recent progress and applications in graph minor theory. *Graphs Combin.*, 23:1–46, 2007.

[KMR08] K. Kawarabayashi, B. Mohar, and B. Reed. A simpler linear time algorithm for embedding graphs into an arbitrary surface and the genus of graphs of bounded tree-width. In *Proc. 49th IEEE Sympos. Found. Comp. Sci.*, pages 771–780, 2008.

[KR07] K. Kawarabayashi and B. Reed. Computing crossing number in linear time. In *Proc. 39th ACM Sympos. Theory of Computing*, pages 382–390, 2007.

[KS15] K. Kawarabayashi and A. Sidiropoulos. Beyond the Euler characteristic: approximating the genus of general graphs. In *Proc. 47th ACM Sympos. Theory of Computing*, pages 675–682, 2015.

[KT12] M. Kamiński and D.M. Thilikos. Contraction checking in graphs on surfaces. In *Proc. 29th Sympos. Theoret. Aspects Comp. Sci.*, vol. 14 of *LIPIcs*, pages 182–193, Schloss Dagstuhl, 2012.

[Kut06] M. Kutz. Computing shortest non-trivial cycles on orientable surfaces of bounded genus in almost linear time. In *Proc. 22nd Sympos. Comput. Geom.*, pages 430–438, ACM Press, 2006.

[Lin82] S. Lins. Graph-encoded maps. *J. Combin. Theory Ser. B*, 32:171–181, 1982.

[LM85] C.E. Leiserson and F.M. Maley. Algorithms for routing and testing routability of planar VLSI layouts. In *Proc. 17th ACM Sympos. Theory of Computing*, pages 69–78, 1985.

[LPVV01] F. Lazarus, M. Pocchiola, G. Vegter, and A. Verroust. Computing a canonical polygonal schema of an orientable triangulated surface. In *Proc. 17th Sympos. Comput. Geom.*, pages 80–89, ACM Press, 2001.

[LR12] F. Lazarus and J. Rivaud. On the homotopy test on surfaces. In *Proc. 53rd IEEE Sympos. Found. Comp. Sci.*, pages 440–449, 2012.

[LZ04] S.K. Lando and A.K. Zvonkin. *Graphs on Surfaces and Their Applications.* Springer-Verlag, Berlin, 2004.

[Mat14] J. Matoušek. String graphs and separators. In J. Nešetřil and M. Pellegrini, *Geometry, Structure and Randomness in Combinatorics*, pages 61–97, Scuola Normale Superiore, Pisa, 2014.

[Mie09] G. Miermont. Tessellations of random maps of arbitrary genus. *Annales Scientifiques de l'École normale supérieure, Quatrième série*, 42:725–781, 2009.

[MK11] W. Myrvold and W. Kocay. Errors in graph embedding algorithms. *J. Comput. System Sci.*, 77:430–438, 2011.

[MMP87] J.S.B. Mitchell, D.M. Mount, and C.H. Papadimitriou. The discrete geodesic problem. *SIAM J. Comput.*, 16:647–668, 1987.

[Moh97] B. Mohar. On the minimal genus of 2-complexes. *J. Graph Theory*, 24:281–290, 1997.

[Moh99] B. Mohar. A linear time algorithm for embedding graphs in an arbitrary surface. *SIAM J. Discete Math.*, 12:6–26, 1999.

[MS12] Y. Makarychev and A. Sidiropoulos. Planarizing an unknown surface. In *Proc. 15th Workshop on Approximation*, vol. 7408 of *LNCS*, pages 266–275, Springer, Berlin, 2012.

[MSTW16] J. Matoušek, E. Sedgwick, M. Tancer, and U. Wagner. Untangling two systems of noncrossing curves. *Israel J. Math.*, 212:37–79, 2016.

[MT01] B. Mohar and C. Thomassen. *Graphs on Surfaces.* Johns Hopkins University Press, 2001.

[Neg01] S. Negami. Crossing numbers of graph embedding pairs on closed surfaces. *J. Graph Theory*, 36:8–23, 2001.

[NR04] T. Nishizeki and M.S. Rahman. *Planar Graph Drawing.* World Scientific, Singapore, 2004.

[Pat13] V. Patel. Determining edge expansion and other connectivity measures of graphs of bounded genus. *SIAM J. Comput.*, 42:1113–1131, 2013.

[PPSL14] Ma. Pilipczuk, Mi. Pilipczuk, P. Sankowski, and E.J. van Leeuwen. Network sparsification for Steiner problems on planar and bounded-genus graphs. In *Proc. 55th IEEE Sympos. Found. Comp. Sci.*, pages 186–195, 2014.

[Prz15] P. Przytycki. Arcs intersecting at most once. *Geom. Funct. Anal.*, 25:658–670, 2015.

[PSŠ08] M.J. Pelsmajer, M. Schaefer, and D. Štefankovič. Odd crossing number and crossing number are not the same. *Discrete Comput. Geom.*, 39:442–454, 2008.

[PSS09a] M.J. Pelsmajer, M. Schaefer, and D. Stasi. Strong Hanani–Tutte on the projective plane. *SIAM J. Discrete Math.*, 23:1317–1323, 2009.

[PSŠ09b] M.J. Pelsmajer, M. Schaefer, and D. Štefankovič. Removing even crossings on surfaces. *European J. Combin.*, 30:1704–1717, 2009.

[RS03] N. Robertson and P.D. Seymour. Graph minors. XVI. Excluding a non-planar graph. *J. Combin. Theory Ser. B*, 89(1):43–76, 2003.

[RS04] N. Robertson and P.D. Seymour. Graph minors. XX. Wagner's conjecture. *J. Combin. Theory Ser. B*, 92:325–357, 2004.

[RS05] R.B. Richter and G. Salazar. Two maps with large representativity on one surface. *J. Graph Theory*, 50:234–245, 2005.

[RS12] N. Robertson and P.D. Seymour. Graph minors. XXII. Irrelevant vertices in linkage problems. *J. Combin. Theory Ser. B*, 102:530–563, 2012.

[RST13] J. Rué, I. Sau, and D.M. Thilikos. Asymptotic enumeration of non-crossing partitions on surfaces. *Discrete Math.*, 313:635–649, 2013.

[RST14] J. Rué, I. Sau, and D.M. Thilikos. Dynamic programming for graphs on surfaces. *ACM Trans. Algorithms*, 10:8, 2014.

[Sch90a] W. Schnyder. Embedding planar graphs on the grid. In *Proc. 1st ACM-SIAM Sympos. Discrete Algorithms*, pages 138–148, 1990.

[Sch90b] A. Schrijver. Homotopic routing methods. In B. Korte, L. Lovász, H.J. Prömel, and A. Schrijver, editors, *Paths, Flows, and VLSI-layout*, pages 329–371. Springer-Verlag, Berlin, 1990.

[Sch91] A. Schrijver. Disjoint circuits of prescribed homotopies in a graph on a compact surface. *J. Combin. Theory Ser. B*, 51:127–159, 1991.

[Sch03] A. Schrijver. *Combinatorial optimization: Polyhedra and efficiency.* Vol. 24 of *Algorithms and Combinatorics*, Springer-Verlag, Berlin, 2003.

[Sch13a] M. Schaefer. The graph crossing number and its variants: A survey. *Electron. J. Combin.*, Dynamic Surveys, article 21, 2013. Updated in 2014.

[Sch13b] M. Schaefer. Toward a theory of planarity: Hanani–Tutte and planarity variants. *J. Graph. Alg. Appl.*, 17:367–440, 2013.

[Sch14] M. Schaefer. Hanani–Tutte and related results. In I. Bárány, K.J. Böröczky, and L. Fejes Tóth, editors, *Geometry—Intuitive, Discrete, and Convex: A tribute to László Fejes Tóth*, vol. 24 of *Bolyai Society Math. Studies*, pages 259–299, Springer, Berlin, 2014.

[Sid10] A. Sidiropoulos. Optimal stochastic planarization. In *Proc. 51st IEEE Sympos. Found. Comp. Sci.*, pages 163–170, 2010.

[SSŠ02] M. Schaefer, E. Sedgwick, and D. Štefankovič. Algorithms for normal curves and surfaces. In *Proc. 8th Conf. Computing and Combinatorics*, vol. 2387 of *LNCS*, pages 370–380, Springer, Berlin, 2002.

[SSŠ08] M. Schaefer, E. Sedgwick, and D. Štefankovič. Computing Dehn twists and geometric intersection numbers in polynomial time. In *Proc. 20th Canad. Conf. Comput. Geom.*, pages 111–114, 2008.

[Sti93] J. Stillwell. *Classical Topology and Combinatorial Group Theory*, 2nd edition. Springer-Verlag, New York, 1993.

[Thi12] D.M. Thilikos. Graph minors and parameterized algorithm design. In H.L. Bodlaender, R.G. Downey, F.V. Fomin, and D. Marx, editors, *The Multivariate Algorithmic Revolution and Beyond*, vol. 7370 of *LNCS*, pages 228–256, Springer, Berlin, 2012.

[Thi15] D.M. Thilikos. Bidimensionality and parameterized algorithms. In *Proc. 10th Sympos. Parameterized and Exact Computation*, vol. 43 of *LIPIcs*, pages 1–16, Schloss Dagstuhl, 2015.

[Tho89] C. Thomassen. The graph genus problem is NP-complete. *J. Algorithms*, 10:568–576, 1989.

[Tho90] C. Thomassen. Embeddings of graphs with no short noncontractible cycles. *J. Combin. Theory Ser. B*, 48:155–177, 1990.

[Tót12] G. Tóth. A better bound for pair-crossing number. In J. Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 563–567, Springer, New York, 2012.

[Tut70] W.T. Tutte. Toward a theory of crossing numbers. *J. Combin. Theory*, 8:45–53, 1970.

[WHDS04] Z.J. Wood, H. Hoppe, M. Desbrun, and P. Schröder. Removing excess topology from isosurfaces. *ACM Trans. Graph.*, 23:190–208, 2004.