Graph Genera and Minors

Austin Ulrigg

Undergraduate Honors Thesis in Mathematics

Department of Mathematics
University of Washington

December 2025

Abstract

This thesis studies embedding graphs on surfaces, with an emphasis on orientable surfaces, rotation systems, and forbidden minors. We begin by developing the historical background, from the earliest occurrences of the crossing number $\mathrm{cr}(G)$ of a graph, to Heawood's map-coloring conjecture and the eventual invention of rotation systems as a way of encoding cellular embeddings of graphs on surfaces. We then describe and analyze PAGE, a rotation system based genus algorithm developed jointly with Alexander Metzger. For an arbitrary graph G with n vertices and m edges, PAGE determines the orientable genus of G in $\mathcal{O}(n(4^m/n)^{n/t})$ steps where t is the *girth* of G. We illustrate PAGE with examples where it was used to determine the genus of graphs whose genus was previously unknown, most notably the $(3, 12)$ cage which has genus 17. Additionally, we discuss ways in which PAGE can be improved and other strategies that could be used to determine the genus of graphs such as the $(3, 11)$ cage which are so far out of reach for algorithmic approaches. We also give an overview of the problem of determining the full set $\mathfrak{F}(S_1)$ of forbidden toroidal minors, summarize known results, and discuss possible directions for further progress on completing the set.

Contents

Acknowledgments

The author is grateful to Alexander Metzger for collaborating on the design and analysis of the PAGE algorithm and for many helpful conversations along the way, and to Dr. François Clément for his guidance, feedback, and support throughout this project.

1. Introduction to Genus

During World War II, Hungarian mathematician Pál Turán was forced to work in a brick factory [47], pushing carts of bricks from kilns to storage sites. The hardest part of his job, he noted, was when two different rail tracks crossed each other. At each such crossing, he was forced to strenuously lift the heavy cart off of the track and place it back on the other side. Bothered by this, Turán wondered to himself if he could have laid the tracks in a different way, as to minimize the number of crossings. Turán abstracted this problem, imagining the kilns and storage sites as vertices of a graph and the tracks as edges. The problem becomes: given a graph G, what is the minimal number of crossings in any drawing of G, and when can we avoid crossings altogether? This minimal number is called the *crossing number* of G, denoted $\mathrm{cr}(G)$.

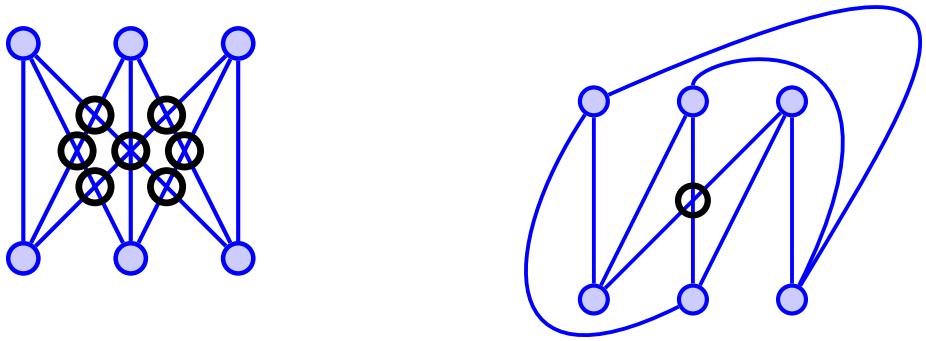
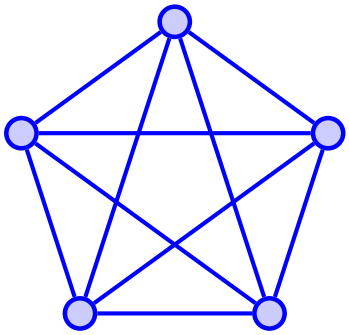


Figure 1. Two drawings of $K_{3,3}$ with a different amount of edge crossings.

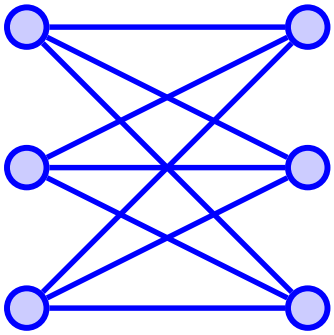
In the real world, one way to avoid crossings in a network like the one we described above is to separate the routes vertically, such as building a bridge or digging a tunnel. Mathematically, this is analogous to letting the graph be on a different surface, such as a surface obtained from the sphere by adding an extra 'bridge' or 'hole' rather than only a flat sheet of paper. To this end, given a graph G one can always find some surface on which G has a crossing-free drawing [24]: for each crossing, modify the surface by adding a 'bridge', allowing one edge to pass above the other eliminating that crossing. A question of interest then is, given a graph G, what is the least number of bridges one needs to add to the plane for G to have a crossing-free drawing. Informally, a surface is a 2D shape, like the sphere or torus, and the number of 'bridges' or 'holes' in a surface is called its genus. We denote the orientable surface of genus g by S_g. Given a graph G, we say *its genus* is the genus of the surface with the least number of holes needed to draw G on it without any edge crossings, which we denote $g(G)$. Later in the thesis we give a formal definition in Section 2. When talking about drawings of graphs in surfaces we will use the word embedding interchangeably with drawing, with a minor clarification to distinguish with the mathematical definition of drawing in Section 2.

Once we begin drawing graphs on surfaces, the simplest case to understand is when the surface is the plane. A graph is called *planar* when it can be drawn in the plane without any

edge crossings. In the 1930s, Polish mathematician Kazimierz Kuratowski gave a complete classification of planar graphs. He proved that a graph is planar if and only if it does not contain a subdivision of two specific graphs: the complete graph on 5 vertices, K_5, or the complete bipartite graph $K_{3,3}$ [16]. So far, we have only considered the plane. Yet, knowing that graphs which contain K_5 or $K_{3,3}$ do not have a crossing-free drawing on the plane, the question remains: on which surfaces do such graphs have a crossing-free drawing? This is a fundamental question that motivates the topics covered in this thesis.



(a) The complete graph K_5.



(b) The complete bipartite graph $K_{3,3}$.

Outside of crossing-free drawings, another way that similar questions about graphs on surfaces began to arise is from a brain-teaser about coloring maps. In the early 1850s, South African mathematician Francis Guthrie first posed what became a famous longstanding problem in graph theory [44]. While trying to color a map of counties of England, he noticed that only four different colors were needed. Translating this problem into one of graphs, one represents regions on the map by vertices, connecting adjacent regions by an edge.

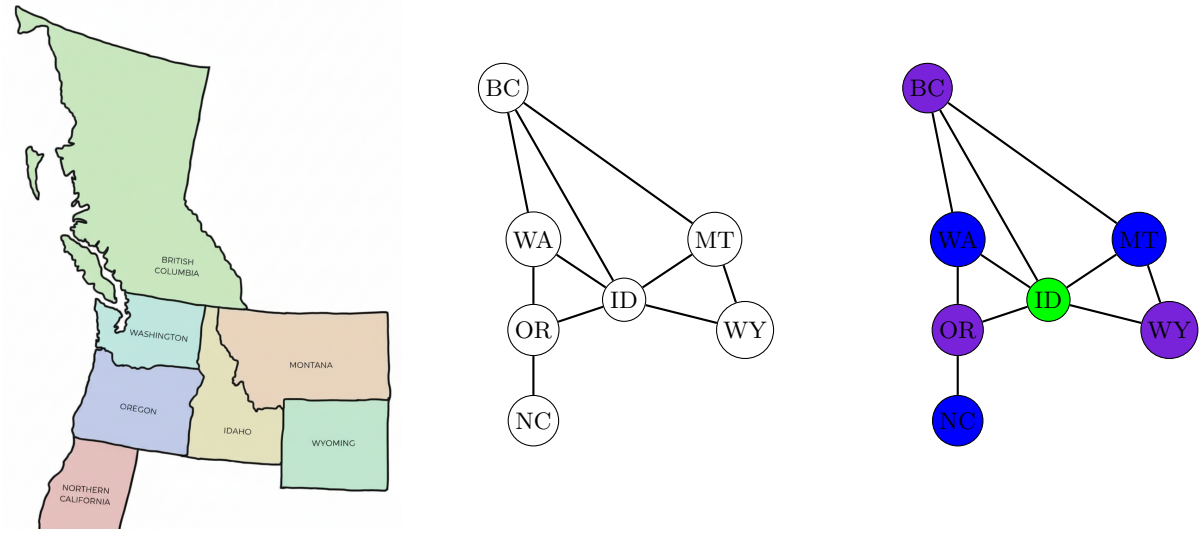


Figure. Example map-coloring [20].

For a graph G we denote its *chromatic number* by $\chi(G)$ as the smallest number of colors needed to color the vertices so that no two adjacent vertices share the same color. Guthrie's observation became the famous *four-color theorem* proven only in 1976, after many false proofs and incorrect counterexamples over the course of the preceding centuries [46]. It states that no more than four colors are needed to color the regions of any map *on a plane*,

so that no two adjacent regions are colored the same [46]. Before the four-color theorem was resolved, in the 1890s, British mathematician Percy John Heawood considered the same question but for maps on different surfaces. He famously conjectured that for an orientable surface with g holes no more than

$$\left\lfloor \frac{7+\sqrt{1+48g}}{2} \right\rfloor$$

colors are needed [45]. Not only is the conjecture related in the sense that it discusses graphs on surfaces, but it has a deep connection to graph genus which sparked the development of the field. The connection becomes clear when we ask which graphs require many colors. If a graph contains the complete graph K_n as a subgraph, then it needs at least n colors, i.e, $\chi(K_n) = n$. Thus, one can observe if there exists a crossing-free drawing of K_n, but not of K_{n+1} on a surface, then coloring any map on that surface requires at most n colors, but no more. Heawood's conjecture can be restated concretely as,

for which integers n does K_n have a crossing-free drawing on S_g?

Heawood's conjecture, in terms of $g(K_n)$, is the same as asking: for which n do we have $g(K_n) \leq g$. The conjecture was resolved in 1968 by mathematicians Gerhard Ringel and W. T. Youngs when they determined the genus of complete graphs, and complete bipartite graphs [33]. Notably,

$$g(K_n) = \left\lceil \frac{(n-3)(n-4)}{12} \right\rceil$$

Solving this quadratic for n recovers exactly Heawood's number in terms of g.

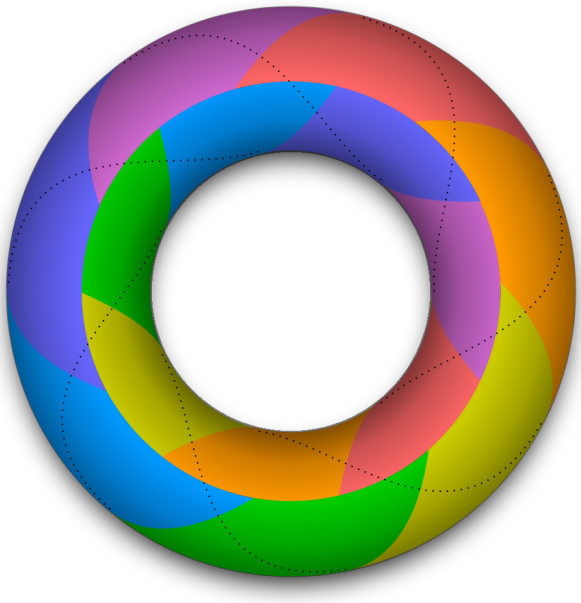


Figure 3. Along with his conjecture in 1890, Heawood also produced an explicit map on the torus that required a 7-coloring. [17].

In proving the above formula, Ringel and Youngs did not rely on constantly redrawing pictures of K_n on different surfaces. Instead, they worked with embeddings of graphs on surfaces

using an entirely combinatorial description. The key idea is attributed to German mathematician Lothar Heffter, who in the 1890s, motivated by Heawood's map-coloring conjecture, constructed explicit embeddings of small complete graphs on orientable surfaces of some genera [13]. However, he described these embeddings via adjacency tables, where for each face, he listed in cyclic order the neighboring faces. This combinatorial view of embeddings was made concrete and systematic in the 1960s when mathematicians Jack Edmonds and W. T. Tutte independently developed combinatorial models for graph embeddings [11, 37]. They showed that, instead of working with an actual drawing of a graph on a surface, one can encode an embedding solely by organizing local permutations of each vertex's incident edges. These ideas served as the foundation of what we now call *rotation systems*. Rotation systems are one of the fundamental objects in this thesis. Later, we will treat rotation systems not only as the basic description of embeddings of graphs in surfaces, but see that they could equivalently be used to *define* embeddings of a certain type.

2. Core Definitions for Graph Embedding

Now that the historical framework and motivation for the topics in this thesis has been laid out, this section begins the formal background and provides the definitions and notation needed for the rest of the thesis. There is a large amount of prerequisite topological knowledge which is assumed of the reader. For a refresher of basic definitions the reader is referred to the Appendix. Section 2.1 gives our conventions for what types of graphs and surfaces we consider. Section 2.2 gives a precise definition for what we mean by drawing a graph on a surface and defines the genus of a graph. Section 2.3 defines a key combinatorial model for embeddings that is used throughout the rest of the thesis. Section 2.4 defines specific substructures of graphs which are directly relevant to the topic of graph genus. Finally, Sections 2.4 and 2.5 explain why the genus of a graph is difficult to compute in general and collect relevant results and theorems which can be used to determine the genus of graphs.

2.1. Graphs and Surfaces

We now introduce a number of definitions we need to define the genus of a graph. Throughout this thesis, we work with finite, simple, undirected graphs. For a graph G, let $V(G)$ and $E(G)$ denote its vertex and edge sets, and let n be the number of vertices and m of edges

$$|V(G)| = n, \qquad |E(G)| = m.$$

A *walk* is a finite sequence

$$v_0, e_1, v_1 \ldots, e_k, v_k$$

in which each e_i is an edge connecting v_{i-1} and v_i. A path is a walk with no repeated vertices. A trail is a walk with no repeated edges. A cycle is a path of length at least 3 where $v_0 = v_k$ for some k. A graph G is *connected* if every pair of vertices is joined by a path. If G is a connected graph, a vertex $v \in V(G)$ is a cut-vertex if the graph $G \setminus v$ is disconnected. If a connected graph G contains a cut-vertex we say it is *1-connected*. Otherwise, at least 2 vertices must be removed from G to disconnect it and we say that G is *2-connected*. A *block* is a maximal 2-connected subgraph. If G is disconnected we work with the connected components of G individually, and in Section 2.6 we see that the genus of a graph is additive over its connected components.

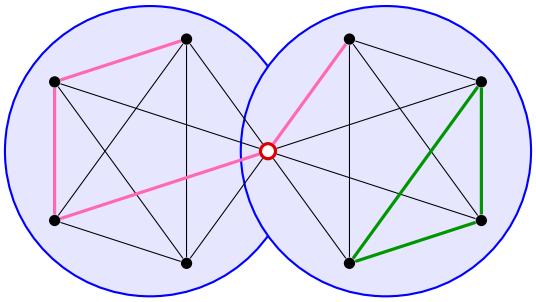
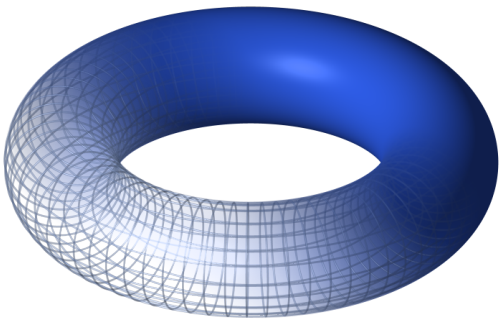


Figure 4. A 1-Connected Graph G with; a Path in pink, a Cut-vertex in red, a Cycle in green, and 2 Blocks in blue.

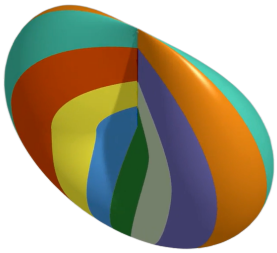
We want to consider each graph as being drawn on some surface, in the same way that we might draw a graph on a piece of paper. We may bend or stretch the piece of paper in space and even glue sides together to make shapes like a cylinder or torus, as long as we do not crease it to a sharp point, tear it, or let it intersect itself. Under these rules, every point on a surface is contained in some locally 'flat' region. Our formal definition for a surface is

Definition 2.1 (Surfaces)**.** A *surface* is a compact, connected, Hausdorff, topological space without boundary that is locally homeomorphic to $\mathbb{R}^2$.

Throughout, we identify surfaces up to homeomorphism. Two surfaces that are homeomorphic as topological spaces are regarded as the same. One particular property of surfaces that is of great importance for embeddings is *orientability*. A surface is called *orientable* if one can choose a continuous unit normal vector field for the surface. Intuitively, this means that it is possible to traverse the surface and have a consistent notion of 'inside' vs. 'outside'. If no such choice exists, the surface is called *non-orientable*.



(a) The torus T [38].



(b) The projective plane $\mathbb{RP}^2$ [19].

Figure 5. Examples of an orientable and a non-orientable surface. (b) The projective plane does not self-intersect, it only appears to when we try to represent it in 3 dimensions as in the image.

Let T denote the standard torus $S_1 \times S_1$ and let $\mathbb{RP}^2$ denote the real projective plane. Two standard families of surfaces we consider are the orientable and non-orientable surfaces of

genus g which we denote S_g and N_g respectively

$$S_g := T\# \dots \#T \qquad g \geq 1^1$$
$$\underset{g \text{ times}}{}$$

$$N_h := \mathbb{RP}^2\# \dots \#\mathbb{RP}^2 \qquad h \geq 1.$$
$$\underset{h \text{ times}}{}$$

These surfaces can be thought of as gluing g tori together in the case of S_g, or h projective planes for N_h.

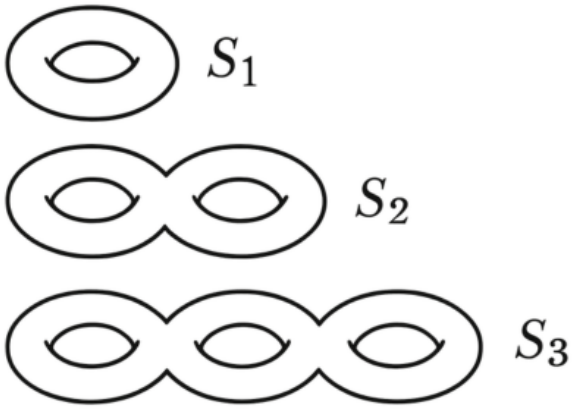


Figure 6. Example of S_1, S_2, S_3 adapted from [22]

.

Because continuously deforming a surface should not change whether or not we can draw a given graph G on it, we only consider the question of embedding G on different surfaces if they are non-homeomorphic. S_g and N_h are the only surfaces to consider up to homeomorphism. The following theorem makes the above observation clear.

Theorem 2.2 (Classification of Surfaces [24])**.** *Every compact, connected surface without boundary is homeomorphic to exactly one orientable surface S_g or one non-orientable surface N_h, for uniquely determined integers $g \geq 0$ or $h \geq 0$.*

Based on this classification we see that the genus of a surface is a fundamental invariant, up to homeomorphism there is only one orientable (resp. non-orientable) surface of each genus.

2.2. Embeddings and Faces

Now that we have formally defined surfaces, we can define explicitly what it means to draw a graph on a surface.

Definition 2.3 (Embeddings)**.** Let G be a graph and S a surface. Regarding G as a one-dimensional CW-Complex, an *embedding* of G in S is a continuous map

$$\varphi : G \to S$$

which is a homeomorphism onto its image.

[1]The symbol # is defined in many topology texts to denote the operation of the connected sum of surfaces. In this thesis we use # only as shorthand: the surfaces S_g and N_h are defined in the Appendix, and we will not need the connected-sum operation.

Importantly, this definition respects the key properties we would want for a drawing of G in a surface S:

- distinct vertices of G are mapped to distinct points in S,
- each edge is mapped to a simple curve, connecting the images of its endpoints,
- two distinct edges meet only if they have a common endpoint.

When discussing an embedding of a specific graph we identify G with its image $\varphi(G) \subseteq S$ and say that G *is embedded in* S. Two embeddings of the same graph in S are equivalent if there is a homeomorphism of S onto itself which maps the image of one embedding onto the other. We now formally define the *genus* of a graph.

Definition 2.4 (Genus of a Graph)**.** Let G be a graph. The *orientable genus* $g(G)$ is the smallest integer g such that G embeds in S_g. The *non-orientable genus* $\tilde{g}(G)$ is the smallest integer h such that G embeds in N_h. An embedding of G in $S_{g(G)}$ is called an *orientable minimum genus embedding* and similarly an embedding of G in $N_{\tilde{g}(G)}$ is called a *non-orientable minimum genus embedding*.

There are graphs which require a non-orientable surface to have more copies of the projective plane to embed in it, than copies of the torus to embed on an orientable surface, so it is not the case that $g(G) = \tilde{g}(G)$ for all graphs G. Returning to embeddings, one can notice that an embedding of G in S partitions the surface of S into regions, which we call *faces*. In an embedding of a graph G in a surface S, a *face* of the embedding is a connected component of $S \setminus G$.

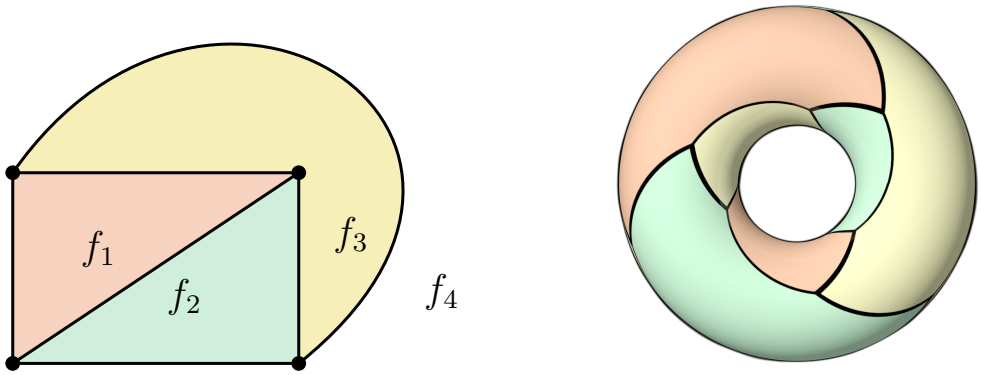


Figure 7. Colored faces of two embeddings: a planar K_4 (left) and the Pappus graph 2-cell embedded on the torus (right) [31].

We let f denote the number of faces in an embedding. Notice that with this definition the boundary of a face in an embedding is a sequence of vertices and edges in G, i.e, a *walk*, which we call a *facial walk*.

Lemma 2.5. *Let G be a graph embedded in a surface S and let F denote the set of faces of this embedding. Then*

$$\sum_{f \in F} \text{length}(f) = 2|E(G)|.$$

Proof. Each $e \in E(G)$ borders either one or two faces. If e borders only one face then it is counted twice in that face's facial walk. Otherwise, e borders two faces and is counted once in both of their facial walks. □

We say that an embedding is *cellular*, or a *2-cell embedding* if every face is homeomorphic to an open disk in $\mathbb{R}^2$. As we have discussed, there are graphs which are too complicated to embed cellularly on a given surface. For example, $K_{3,3}$ does not have a cellular embedding on the sphere S_0, i.e, it is not planar. There is a simple proof which we give after Lemma 2.6. However, $K_{3,3}$ does admit a 2-cell embedding on the torus.

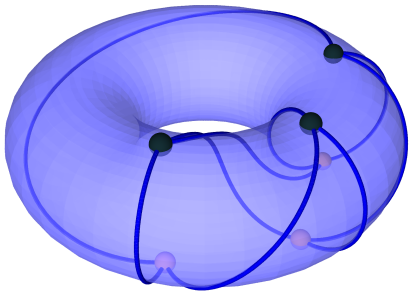


Figure 8. 2-cell Embedding of $K_{3,3}$ on the Torus.

Perhaps surprisingly, graphs can also be too simple to embed cellularly on a given surface. Of course, we can draw a triangle, or the graph K_3 on a torus. Pictured below are three examples. However, none of these drawings are cellular embeddings. In the first two embeddings $T \setminus G$ is homeomorphic to a cylinder, not a disk, and in the third embedding $T \setminus G$ is split into a disk, and a punctured torus which is not homeomorphic to a disk. There is a simple proof (which we provide later) that these three failures are not merely due to our inability to draw, but that it is actually impossible to embed K_3 cellularly on the torus.

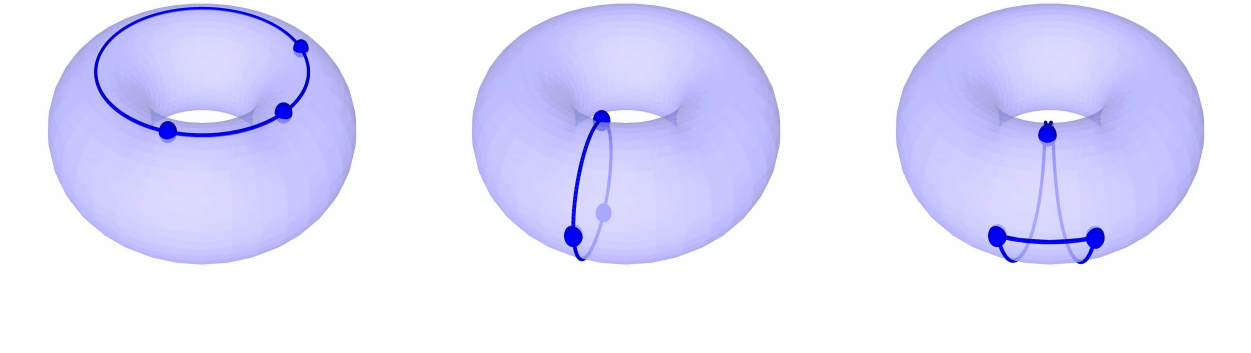


Figure 9. K_3 Embedded (non-cellularly) on the Torus.

These examples show that 2-cell embeddings are restrictive, a graph can embed in a surface without embedding cellularly in the same surface. Despite this, cellular embeddings are the appropriate embeddings to consider for two reasons. Theorem 2.9 shows that every

embedding of a graph in a surface of minimum genus is cellular, and we show that 2-cell embeddings are the embeddings that admit a rotation system, important for a result that classifies minimum genus embeddings. They also satisfy the following fundamental equation:

Lemma 2.6 ([24]). *Let S be a surface and G a graph that is 2-cell embedded in a surface S, with n vertices, m edges, and f faces. Then, S is either homeomorphic to S_g or N_h where g and h are determined by the equations*

$$n - m + f = 2 - 2g = 2 - h.$$

Demonstrating the usefulness of this formula, we can now swiftly prove our earlier claims about K_3 and $K_{3,3}$. If K_3 had a 2-cell embedding on T_1 then by Lemma 2.6 it would require 0 faces which is clearly impossible. Additionally, if $K_{3,3}$ had a 2-cell embedding on S_0 then by Lemma 2.6 it would require 5 faces. However, the length of the shortest closed walk in $K_{3,3}$ is 4, and because each edge is only used twice amongst all facial walks this would require $K_{3,3}$ to have at least 10 edges, a contradiction.
Based on Lemma 2.6 and the fact that g and h are fundamental invariants of a surface, we define another fundamental invariant of surfaces.

Definition 2.7 (Euler Characteristic)**.** Let S be a surface. The Euler characteristic of S is

$$\chi(S) = \begin{cases} 2 - 2g & S \cong S_g \\ 2 - h & S \cong N_h \end{cases}$$

Combining this definition with our previous lemma we have the following theorem.

Theorem 2.8 (Euler's Formula [24])**.** *Let S be a surface and G a graph that is 2-cell embedded in a surface S, with n vertices, m edges, and f faces. Then,*

$$n - m + f = \chi(S).$$

As alluded to earlier, any minimum genus embedding is always cellular.

Theorem 2.9 ([24])**.** *Every minimum genus embedding of a graph G is cellular.*

Thus, when we are talking about minimum genus embeddings, we can assume that they are 2-cell embeddings. From this, we also see in our earlier example that we could conclude $g(K_{3,3}) = 1$. Importantly, we have seen that minimum genus embeddings are cellular. Equally importantly, cellular embeddings have entirely combinatorial descriptions, called rotation systems, that are easy to work with and do not require one to work with the more complicated topology of the surfaces. The next section discusses how this is possible.

2.3. **Rotation Systems**

From here on, we restrict our attention to orientable surfaces and orientable embeddings, although all of the following generalizes naturally to the non-orientable case. So far, the only notion of an embedding that we have (cellular or not) is a map between arbitrary topological spaces. This in no way lends a simple, or intuitive, description of embeddings, and is almost useless for determining the genus of sufficiently complicated graphs. Fortunately, cellular embeddings have combinatorial descriptions that are much more concrete to work with than abstract maps. The point of this subsection is to define and describe these combinatorial descriptions of cellular embeddings.

With that in mind, let G be a graph with at least one edge, cellularly embedded in some orientable surface S. Around each vertex, we can list its incident edges in clockwise (or counter-clockwise) order. This defines a permutation

$$\pi_v : \{\text{edges incident to } v\} \to \{\text{edges incident to } v\}$$

where $\pi_v(e)$ is defined to be the edge that follows e in the clockwise order. This permutation π_v is called a *local rotation* at v.

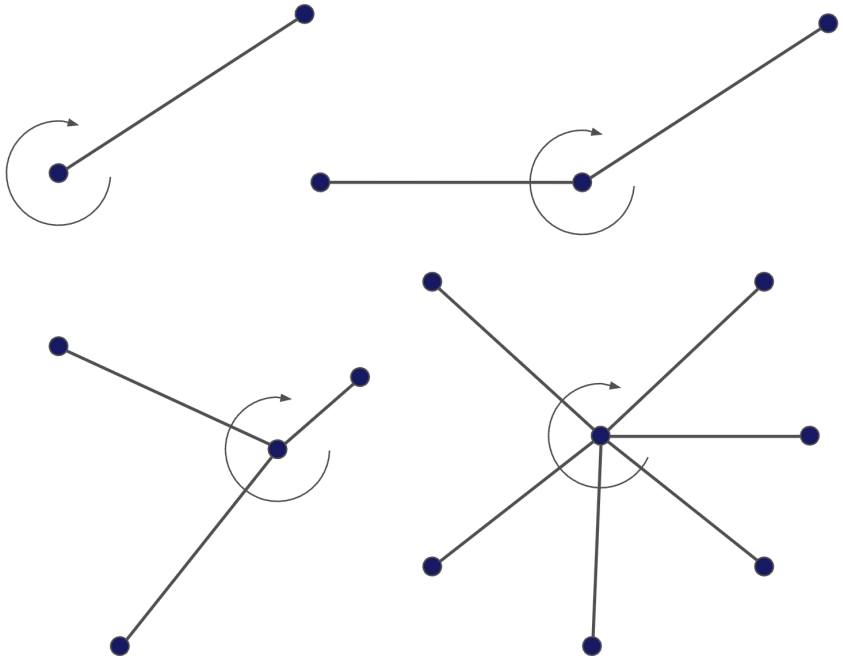


Figure 10. Local Rotations at a Vertex.

Definition 2.10 (Rotation System)**.** Let G be a graph with at least one edge, cellularly embedded in some orientable surface S. The set

$$\pi = \{\pi_v \mid v \in V(G)\}$$

is the *rotation system* of the embedding of G in S.

We will see that the rotation system π completely determines the embedding up to homeomorphism. First, we must describe the faces of the cellular embedding based on π. A *dart* of G is an ordered pair (v, e) where $v \in V(G)$ and e is an edge incident to v. For an edge uv two darts (u, e) and (v, e) are formed. It is useful to think of the dart (v, e) as a directed half edge leaving from v along e.

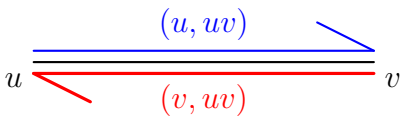


Figure 11. Darts of an edge uv.

Using the darts of G we can obtain from a rotation system both the faces and their facial walks. The process is as follows: choose a dart (v_1, e_1) where e_1 joins v_1 to some vertex v_2.

Traverse the dart (v_1, e_1) to arrive at v_2. Once, we have arrived at v_i along the dart (v_{i-1}, e_{i-1}) define the next dart by $(v_i, \pi_{v_i}(e_{i-1}))$ and then traverse this dart to v_{i+1}. Repeating this, because there are only finitely many darts the process eventually returns to the original dart (v_1, e_1) creating a closed walk

$$v_1, e_1, v_2, \ldots, v_k, e_k, v_1$$

called a π-*walk*. We do not distinguish between a π-walk and its cyclic shifts. Using this construction, and keeping track to ensure we use each dart exactly once, we arrive at a finite collection of π-walks. From these π-walks, we associate each with a π-*polygon* in the plane, defined by labeling its sides in cyclic order as the e_k in the order they occur in the π-walk. Now we may identify the π-polygons in pairs along sides which share an edge, identifying the same vertices $v \in V(G)$ with themselves in other π-polygons. After performing all identifications, we obtain a compact surface S' containing a cellular embedding of G, whose edges are the images of the π-polygons' sides. By construction, the interiors of the π-polygons are the faces of this embedding, and the π-walks are the facial walks of these faces. For details and a proof of this construction, see the book *Graphs on Surfaces* [24].
Before moving on to the next section, at this point, it would be helpful to work out a concrete example. In the figure below we represent the torus as a square in the plane, with opposite sides glued together.

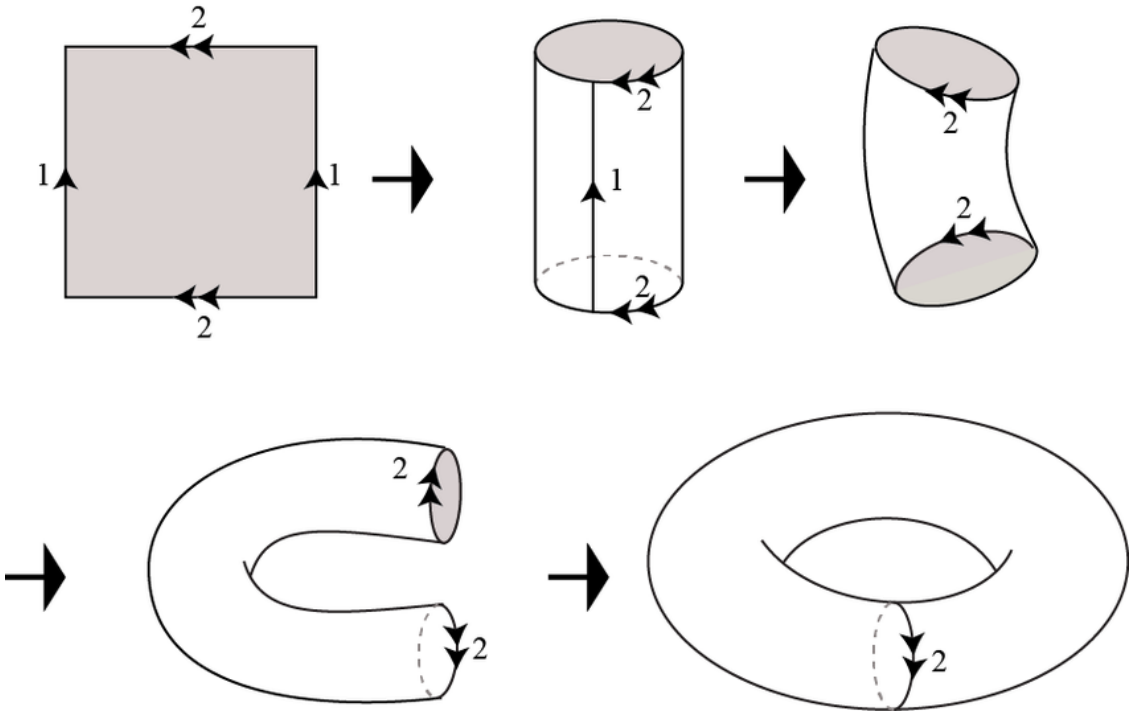


Figure 12. Square Representation of the Torus [30].

Throughout the thesis we continue to represent the torus in this way. We also find it useful to use the following row notation to display rotation systems. For the graph $K_{3,3}$ let its bipartition classes be $\{1, 3, 5\}$ and $\{2, 4, 6\}$. We display its rotation system π and the resulting facial walks side by side.

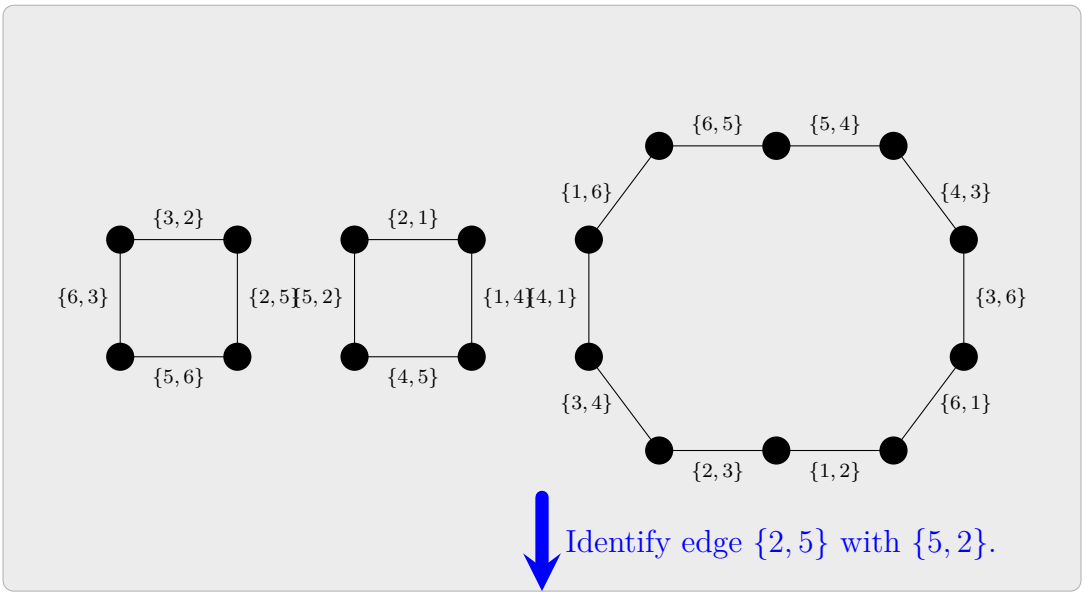
π
$1 : 624$
$2 : 135$
$3 : 462$
$4 : 315$
$5 : 426$
$6 : 153$

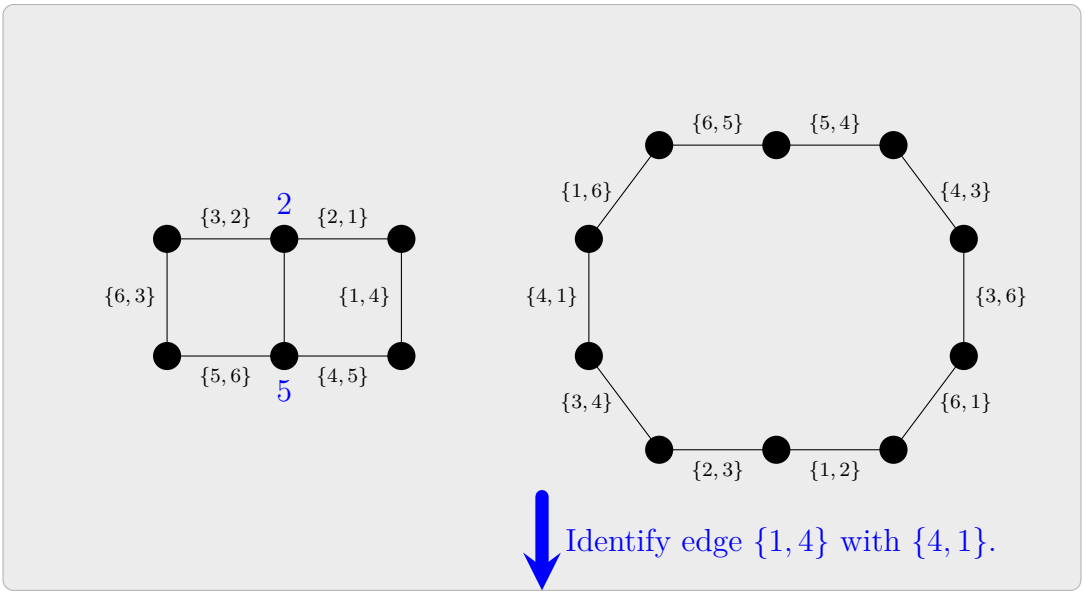
$W_1 : 1 \underset{14}{\longrightarrow} 4 \underset{45}{\longrightarrow} 5 \underset{52}{\longrightarrow} 2 \underset{21}{\longrightarrow} 1,$

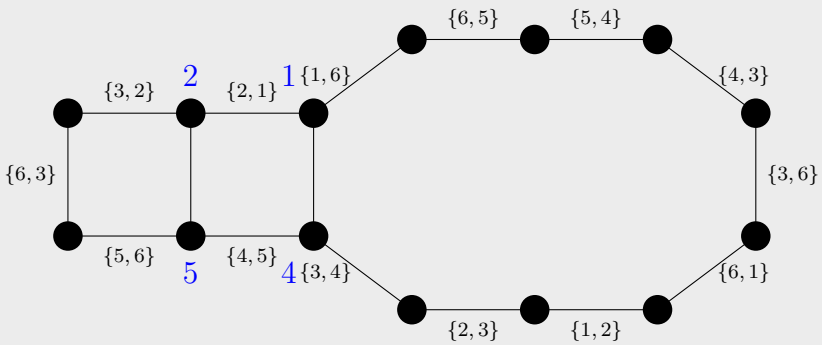
$W_2 : 4 \underset{41}{\longrightarrow} 1 \underset{16}{\longrightarrow} 6 \underset{65}{\longrightarrow} 5 \underset{54}{\longrightarrow} 4 \underset{43}{\longrightarrow} 3 \underset{36}{\longrightarrow} 6 \underset{61}{\longrightarrow} 1 \underset{12}{\longrightarrow} 2 \underset{23}{\longrightarrow} 3 \underset{34}{\longrightarrow} 4,$

$W_3 : 6 \underset{63}{\longrightarrow} 3 \underset{32}{\longrightarrow} 2 \underset{25}{\longrightarrow} 5 \underset{56}{\longrightarrow} 6.$

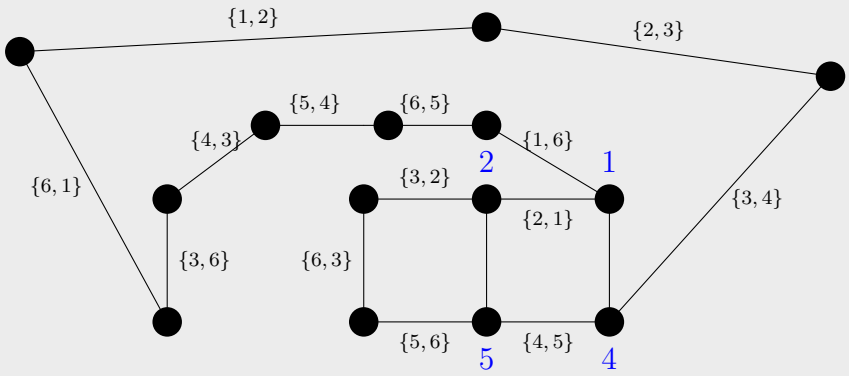
To obtain the facial walks W_1, W_2, W_3 on the right, we apply the face-tracing procedure to the rotation system π on the left, starting from the dart $(1, \{1, 4\})$ and continuing until all darts have been used.



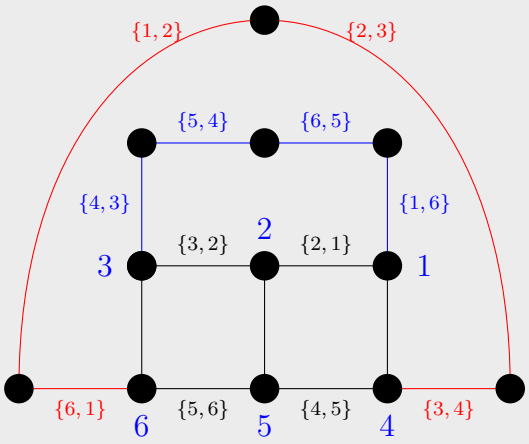




Reposition the polygon for clarity.



Identify edge $\{3, 6\}$ with $\{6, 3\}$.



Reposition the polygon for clarity, identify opposite sides to form a torus.

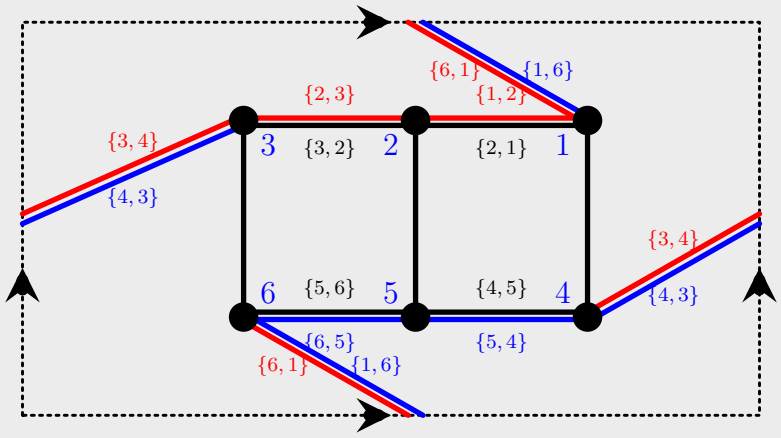


Figure 10. Constructing a torus embedding of $K_{3,3}$ from its rotation system.

We now have a square representation of the torus containing a cellular embedding of $K_{3,3}$. Any cellular embedding of $K_{3,3}$ on the torus must satisfy Euler's formula

$$6 - 9 + f = 0$$

and thus have 3 faces. Figure 13 below shows the colored faces of the embedding, found from the face-tracing procedure we performed. Identifying sides as in Figure 12 completes the construction.

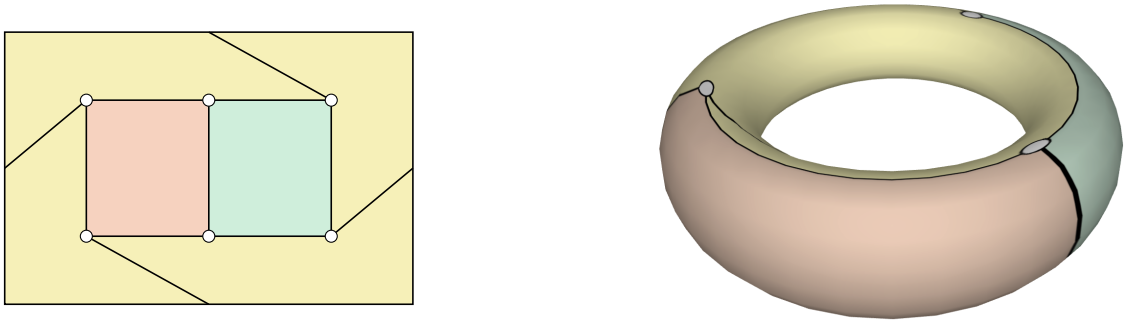


Figure 13. Colored faces of the embedding of $K_{3,3}$ from Figure 10.

We have now seen that: (1) given a 2-cell embedding of G in S we can define its rotation system, and (2) given a rotation system of G we can define a compact surface with G cellularly embedded in it (see Figure 10). This motivates the following theorem.

Theorem 2.11 (Heffter-Edmonds-Ringel Rotation Principle [24])**.** *Let G be a graph cellularly embedded in an orientable surface S. Let π be the rotation system of this embedding of G, and let S' be the surface of the corresponding 2-cell embedding of G from the above construction. Then, there exists a homeomorphism of S onto S' taking G in S onto G in S' (in such a way that we induce the identity map from G onto its copy in S'). In particular,*

every cellular embedding of a graph G in an orientable surface is uniquely determined up to homeomorphism by its rotation system.

Rotation systems π and π' are said to be *equivalent* if they are equal as sets or for every $v \in V(G)$ we have $\pi_v = \pi_v'^{-1}$. Combining this definition with the previous theorem we obtain the following corollary.

Corollary 2.12 ([24])**.** Suppose that we have cellular embeddings of a graph G in orientable surfaces S and S' with rotation systems π and π' respectively. Then, there is a homeomorphism $S \to S'$ whose restriction to G induces the identity if and only if π and π' are equivalent.

Proof. Suppose there is a homeomorphism $h : S \to S'$ whose restriction to G induces the identity. If h is orientation-preserving, it preserves the cyclic orders of the edges around each vertex and $\pi_v = \pi_v'$ for all v, otherwise, if h is not orientation-preserving, it reverses the cyclic orders of the edges around each vertex and $\pi_v = \pi_v'^{-1}$ for all v. In either case, π and π' are equivalent. Suppose instead that π and π' are equivalent and $\pi = \pi'$ then by Theorem 2.11 there exists such a homeomorphism h. Otherwise, if $\pi_v = \pi_v'^{-1}$ for all v then reversing the orientation of S' produces a rotation system of G on S' with $\pi_v = \pi_v'$ for all v and another application of Theorem 2.11 completes the proof. □

Furthermore, we also want a notion of equivalence of cellular embeddings and rotation systems of a graph G up to re-labeling of G's vertices, i.e., up to graph automorphism. The following definition allows us to capture when two embeddings are the same up to what we name the vertices.

Definition 2.13 (Isomorphic Embeddings)**.** Let φ_1, φ_2 be two embeddings of a graph G into the orientable surfaces S_1, S_2. We say that the embeddings are *isomorphic* if there exists

$$\phi : S_1 \xrightarrow{\sim} S_2 \qquad \psi : G \to G, \quad \psi \in \mathrm{Aut}(G)$$

such that

$$\phi \circ \varphi_1 = \varphi_2 \circ \psi.$$

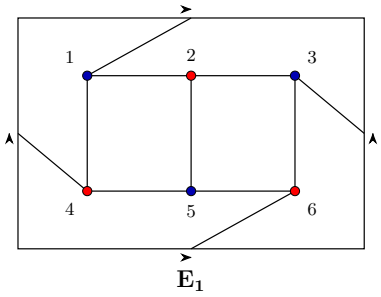
Definition 2.14 (Isomorphic Rotation Systems)**.** Rotation systems π and π' are *isomorphic* if there exists $\varphi \in \mathrm{Aut}(G)$ such that

$$\pi' = \varphi(\pi) \quad \text{or} \quad \pi' = \varphi(\pi)^{-1}.$$

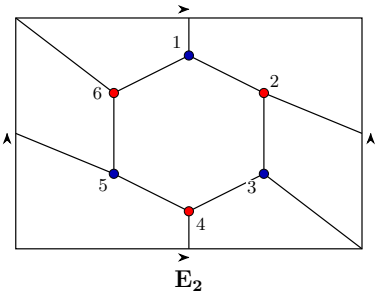
In the same way as before, these definitions combined with the Heffter-Edmonds-Ringel rotation principle yield a useful corollary.

Corollary 2.15. Suppose that we have cellular embeddings of a graph G in orientable surfaces S and S' with rotation systems π and π' respectively. Then, the two embeddings are isomorphic if and only if π and π' are isomorphic.

Thus, up to homeomorphism of the surface and graph automorphism, 2-cell embeddings of a graph G correspond exactly to the isomorphism classes of rotation systems of G.
Before considering any equivalences, a graph G has $\prod_{v \in V(G)} (\deg(v) - 1)!$ rotation systems. In our example of $K_{3,3}$, it has $2^6 = 64$ rotation systems. Previously, we saw that $g(K_{3,3}) = 1$, so a natural follow-up question is: how many non-isomorphic 2-cell embeddings does $K_{3,3}$ have on the torus? It turns out there are only two non-isomorphic embeddings, both pictured below.

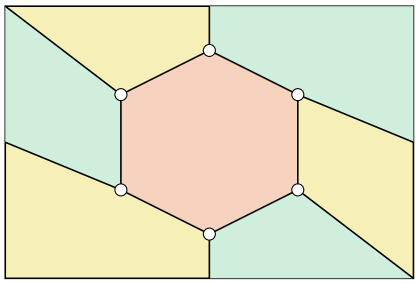


Rotation for E_1:
1: 426
2: 153
3: 264
4: 153
5: 456
6: 531



Rotation for E_2:
1: 624
2: 135
3: 462
4: 315
5: 426
6: 153

Figure 14. Two Cellular Embeddings of $K_{3,3}$ on the Torus and their Rotations.



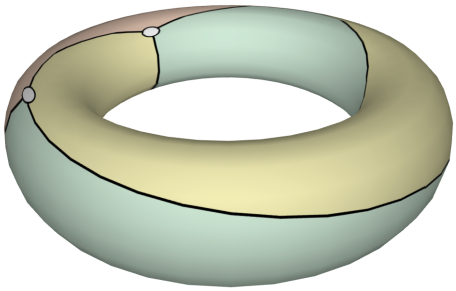


Figure 15. Colored Faces of $\mathbf{E_2}$.

We only sketch the proof that these are the only 2 embeddings.

Sketch of Proof 2.1. *By Euler's formula, any 2-cell embedding of $K_{3,3}$ in T_1 must have 3 faces. Additionally, any closed walk in $K_{3,3}$ has an even length of at least 4. Therefore, the only possible distribution of the lengths of the facial walks in a minimum orientable genus embedding of $K_{3,3}$ are $(6,6,6),(4,4,10),(4,6,8)$. One can rule out the third possibility by noticing that fixing a facial walk of length 4 forces the local rotations of 4 vertices in $K_{3,3}$ and forces a second facial walk of length 4.*

So far we have been fixing a graph G and discussing how it can be drawn on a surface, and for which surfaces it is possible. We now point out a way in which this problem can greatly simplify. In our running example $g(K_{3,3}) = 1$, so it is not far-fetched to assume that if a graph G has $K_{3,3}$ as a subgraph then $g(G) \geq 1$. This is indeed the case, but it does not capture the full picture. If we instead take $K_{3,3}$ and sub-divide one of the edges into two, the resulting graph does not contain $K_{3,3}$ as a subgraph, however, of course we expect it to also have genus one. The next section introduces further definitions which address this point and discusses their implications on problems of graph genus.

2.4. Minors and Subdivisions

Definition 2.16 (Subdivision and Topological Minor)**.** Let H be a graph. A *subdivision* of H is a graph obtained by repeatedly replacing an edge uv of H by a path $u - w - v$, where w is a new vertex of degree 2. A graph G is called a *subdivision of H* if it can be obtained from H by subdividing its edges (possibly several times). We say that G contains H as a *topological minor* if G has a subgraph that is a subdivision of H.

However, as illustrated by Figure 16 this still does not capture the full situation.

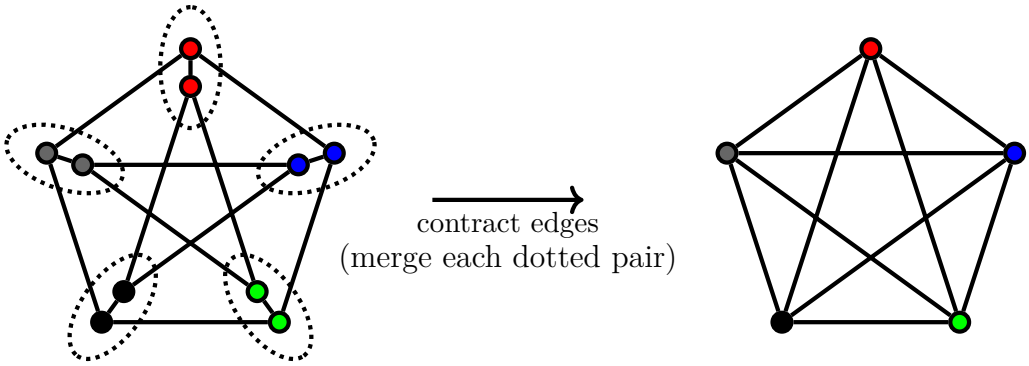


Figure 16. The Peterson Graph Containing a K_5 Minor.

Contracting the 5 circled edges of the Peterson graph in the picture we obtain K_5. Thus, on any surface where we can not embed K_5 we also expect to not be able to embed the Peterson graph, as any embedding could contract to one of K_5. However, the Peterson graph does not contain K_5 as a topological minor, as this would require the Peterson graph to contain 5 vertices of degree 4 which it does not. This motivates the following definition of a more general graph minor.

Definition 2.17 (Graph Minor)**.** Let G and H be graphs. We say that H is a graph minor of G if an isomorphic copy of H can be obtained from of G by a sequence of the following operations:

- deleting a vertex,
- deleting an edge,
- contracting an edge uv to a single vertex (identifying u and v).

As expected, we get the following theorems about the genus of a graph.

Theorem 2.18 (Subdivisions Preserve Genus [24])**.** *Let G be a graph and TG be a subdivision of G. Then,*

$$g(G) = g(TG) \quad \textit{and} \quad \tilde{g}(G) = \tilde{g}(TG).$$

Theorem 2.19 (Genus is Monotone Under Graph Minors [24])**.** *Let G be a graph and H be a minor of G. Then,*

$$g(H) \leq g(G) \quad \textit{and} \quad \tilde{g}(H) \leq \tilde{g}(G).$$

Previously, we've seen $g(K_{3,3}) = 1$. Similarly it is an easy exercise to show $g(K_5) = 1$. As a first application of Theorem 2.18 we can prove one direction of one of the foundational results in the topic of graph minors

Theorem 2.20 (Kuratowski's Theorem [16])**.** *A finite graph G is planar if and only if it does not contain a topological minor of K_5 or $K_{3,3}$.*

Sketch of Proof. *We prove the 'only if' direction. Suppose G is planar. If G contained a topological minor H of K_5, or $K_{3,3}$. Then, by Theorem 2.18 $g(H) = 1$. Yet, H is a minor of G and by Theorem 2.19 we have*

$$1 = g(H) \leq g(G)$$

which is a contradiction. To see a proof of the other direction, see [16].

If a graph G contains another graph H as a topological minor, it also contains H as a graph minor. Kuratowski's theorem can then be rephrased, in other words, the class of planar graphs is completely determined by a finite list of forbidden minors. Because genus is monotone under taking graph minors, the property $g(G) \leq 1$ is closed under the operation of taking graph minors on G. Once a graph embeds on the torus, so does every minor of it. This is also true for every other fixed genus value. A natural question that arises is: for a fixed genus g and surface S_g, is there a finite list of graphs that are minimal obstructions for embedding in S_g? Remarkably the answer is yes, and it comes from the following even stronger result.

Theorem 2.21 (The Graph Minor Theorem (Robertson-Seymour) [34])**.**

The proof of this theorem is highly non-trivial, and was completed in a series of 20 papers written from 1983 to 2004 spanning over 500 pages. Unfortunately, the proof is also non-constructive. For the cases of graph families of fixed genus the full list of forbidden minors is only known for planar and projective planar graphs [1]. Later in Section 4 we return to discussing consequences of the Robertson-Seymour Theorem. For now, we continue discussing the topic of computing the genus of a fixed graph.

2.5. **Why Computing Graph Genus is Hard**

Conceptually the question of finding the genus of a graph is simple. Yet, when confronted with a complex graph such as the Tutte (3,12) cage in Figure 17 the difficulty becomes clear. This graph has genus 17 [23], but to prove this one must (1) prove that it cannot embed on S_{16} and (2) construct an explicit 2-cell embedding of it on S_{17}, both of which are highly non-trivial, and almost hopeless by hand.

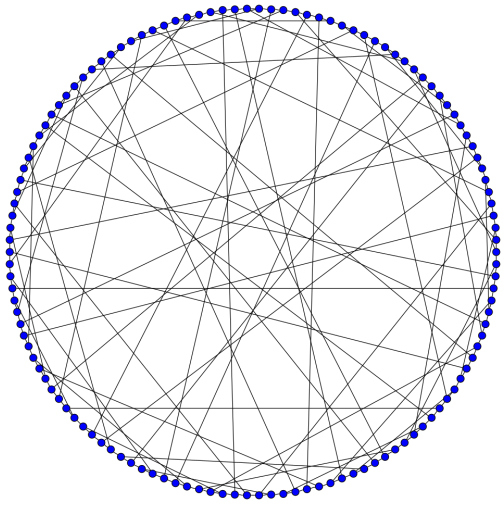


Figure 17. Tutte (3,12) Cage Graph [15]

Realizing that it is likely hopeless, it is reasonable to instead attempt to find the genus using the combinatorial description of 2-cell embeddings using rotation systems. As we saw earlier in Theorem 2.11, every cellular embedding of G is uniquely determined, up to homeomorphism, by its rotation system, and conversely every rotation system defines a cellular embedding of G on some orientable surface. A naive algorithmic approach for computing $g(G)$ could then be:

(1) enumerate all rotation systems of G,
(2) for each rotation system, follow the construction of Figure 10 and compute the genus of the resulting embedding using Euler's formula,
(3) return the smallest genus over all such embeddings.

However, the issues with this approach are two-fold: the number of rotation systems of a large graph is huge, and most rotation systems produce a non-optimal embedding. As we noted earlier, before considering any equivalences a graph G has

$$\prod_{v \in V(G)} (\deg(v) - 1)!$$

rotation systems. On top of this, for each rotation system any algorithm following the approach above must construct all of the π-walks and apply Euler's formula. This quickly becomes infeasible for even moderately sized graphs, for example, it is well known that determining the genus of a graph is NP-hard [36]. Currently, the best implemented genus algorithms are exponential time, and the rest are too complex for implementation and have intractable constant factors [6, 27, 28].
Because the genus of a graph is so difficult to compute, it helps to compute combinatorial bounds which can more quickly restrict the genus of a graph before moving to more algorithmic methods. The next subsection derives and explains several such bounds.

2.6. **Easy Genus Bounds**

We begin this subsection with bounds coming from Euler's formula.

Lemma 2.22. *Let G be a graph with $n \geq 3$ vertices and m edges that is 2-cell embedded on S_g. Then,*

$$\left\lceil \frac{m - 3n + 6}{6} \right\rceil \leq g(G).$$

Proof. Let f be the number of faces in the embedding. Because each edge is incident with exactly 2 faces, summing the lengths of the facial walks we get

$$2m = \sum_F \text{length}(F).$$

Since every facial walk has length at least 3, we have

$$\sum_F \text{length}(F) \geq 3f$$

which implies $2m \geq 3f$ and $f \leq 2m/3$. Substituting this into Euler's formula for a 2-cell embedding and re-arranging yields the lemma. □

Lemma 2.23. *Let G be a graph with $n \geq 3$ vertices and m edges, and suppose that G has girth $t \geq 3$ (i.e, every cycle in G has length at least t). If G is 2-cell embedded on S_g, then*

$$1 + \frac{t-2}{2t}m - \frac{n}{2} \leq g(G).$$

However, lower bounds do not tell the full story for what surfaces graphs can cellularly embed on, as we saw earlier in Figure 9 there are also upper limits. Having computed these lower bounds on $g(G)$, we now compute some upper bounds.

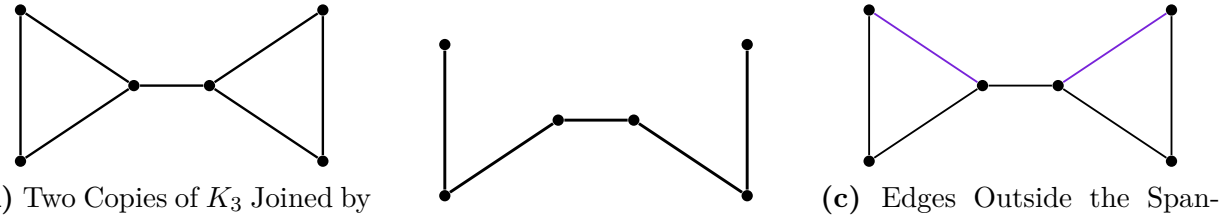
Definition 2.24 (Betti Number)**.** Let G be a graph with n vertices, m edges, and c connected components. The Betti number of G is

$$\beta(G) = m - n + c$$

which simplifies for a connected graph to

$$\beta(G) = n - m + 1.$$

The Betti number of a connected graph is equivalent to the number of excess edges G has compared to a spanning tree of G.



(a) Two Copies of K_3 Joined by an Edge

(b) A Spanning Tree

(c) Edges Outside the Spanning Tree

Figure 18. The Betti Number.

Rewriting Euler's formula in terms of the Betti number of G and using that every embedding has at least one face yields the following lemma.

Lemma 2.25. *Let G be a connected graph that is 2-cell embedded on S_g. Then*

$$g \leq \frac{\beta(G)}{2}.$$

In particular,

$$\max\{g \mid G \text{ embeds cellularly on } S_g\} := g_{\max}(G) \leq \left\lfloor \frac{\beta(G)}{2} \right\rfloor.$$

Unfortunately, this upper-bound is not always attained. The graph in Figure 18 has $\beta(G) = 2$ and thus $g_{\max}(G) \leq 1$. Yet, as we see in Theorem 2.32 the genus is additive over blocks, and hence so is $g_{\max}(G)$, and the graph of Figure 18 contains 2 blocks of K_3 which as we saw in Figure 9 does not have a 2-cell embedding on the Torus. This is a simple example of the upper-bound given by the Betti number not being attained, but to this end, there is a strong theorem by Xuong which describes exactly how much the upper-bound can fail and exactly when it does.

Remark 2.26 (Xuong's Theorem [48])**.** Xuong defines an integer $\xi(G)$, called the *deficiency* of G and proves that

$$g_{\max}(G) = \frac{\beta(G) - \xi(G)}{2}.$$

Because we do not use this theorem later in the thesis, we refer the reader to the relevant definitions and proof provided in [48].

Now we have an explicit formula for $g_{\max}(G)$, and we know from Theorem 2.9 that G embeds cellularly on $g(G)$. A natural question is then, for every integer h satisfying $g(G) \leq h \leq g_{\max}(G)$ does G embed cellularly on S_h? The following theorem presents the answer.

Theorem 2.27 (Interpolation of Genus [2])**.** *Let G be a graph that cellularly embeds on S_n and S_m with $n \leq m$. Then G has a 2-cell embedding on S_i for all $n \leq i \leq m$.*

We now finally connect the genus of a graph back to the crossing number as alluded to in the story about Turán. Earlier, our definition of embedding did not allow for any edge crossings. Now, we allow for edge crossings but restrict ourselves to embedding only on the sphere S_0.

Definition 2.28 (Drawing of a Graph)**.** A drawing of a graph G in the plane is a continuous map

$$f : G \to S_0$$

such that

- distinct vertices of G are mapped to distinct points in S,
- each edge is mapped to a simple arc, connecting the images of its endpoints
- edges meet only in finitely many points, and every such point is either a common endpoint of two edges or a point where exactly two edges cross.

Any intersection of edges that is not at a common endpoint is called a *crossing*.

Definition 2.29 (Crossing Number of a Graph)**.** The crossing number $\mathrm{cr}(G)$ of a graph is the minimum number of *crossings* over all drawings of G.

The first relationship between the genus and crossing number is an well-known upper-bound.

Theorem 2.30. *For every graph G,*

$$g(G) \leq \mathrm{cr(G)}\,.$$

Intuitively, it is clear why we expect this to be true. For every crossing in a drawing of G, we may add one copy of T, i.e, add one hole, to obtain an embedding of G on $S_{\mathrm{cr}(G)}$. However, it is less clear if $g(G) = \mathrm{cr}(G)$ for all graphs G. It turns out that this is actually far from the case. For example, it was shown by Ringeisen and Beineke [32] that $\mathrm{cr}(\mathrm{C_3} \times \mathrm{C_n}) = \mathrm{n}$ (where C_n is the cycle graph on n vertices) for all $n \geq 3$, yet one can prove that $g(C_3 \times C_n) = 1$ for $n \geq 3$.
Finally, we get a similar bound to Lemma 2.22 on the crossing number.

Lemma 2.31. *Clearly $0 \leq \mathrm{cr}(G)$. Let G be a graph with $n \geq 3$ vertices and m edges. Then,*

$$\max\{0, m - 3n + 6\} \leq \mathrm{cr}(G).$$

Proof. Let $k = \mathrm{cr}(G)$ and consider a drawing of G in the plane with exactly k crossings. At each crossing, replace the intersection of the edges with a new vertex of degree 4. This forms a planar graph with $n' = n + k$ vertices and $m' = m + 2k$ edges. Since G' is planar by Lemma 2.22 we have $m' \leq 3n' - 6$. Substituting,

$$m + 2k \leq 3(n + k) - 6 = 3n + 3k - 6,$$

thus,

$$m - 3n + 6 \leq k = \mathrm{cr}(G).$$

□

We conclude this section by mentioning that the orientable genus is additive over blocks.

Theorem 2.32 (Additivity Over Blocks [3])**.** *Let G be a graph with blocks $B_1, \ldots, B_n$. Then,*

$$g(G) = g(B_1) + \cdots + g(B_n).$$

Remarkably, the non-orientable genus of a graph is *not* additive over its blocks [35].
The results and theorems of this section show that the genus of a graph depends on its edge density, cycle structure, and its blocks. In the next sections we use these results as both quick checks in bounding the genus and as constraints in determining the genus of a specific graph. As we have seen the question of determining the genus of the graph is a difficult and time-consuming task. The following section turns to the algorithmic side of this problem and presents PAGE, a fast algorithm for computing the genus of a graph, and discusses the insights that went into its development.

3. A Practical Algorithm for Graph Embedding PAGE

We now present a simple algorithm to determine the genus of a graph: PAGE. PAGE was developed jointly by Alexander Metzger[2] and the author in our preprint *An Efficient Genus Algorithm Based on Graph Rotations* [23]. Given a finite connected simple graph G with n vertices, m edges, and girth t, PAGE computes the orientable genus $g(G)$ and produces an explicit 2-cell embedding of G on $S_{g(G)}$. The algorithm runs faster than previously implemented algorithms, including those presented in [4, 7, 10]. PAGE also provides upper and lower bounds, which iteratively narrow as it runs, enabling practical use. In Section 3.2 we construct the algorithm PAGE and illustrate its correctness. Then, in Section 3.3 we discuss the main results and discoveries of PAGE. Before introducing PAGE, the following subsection provides an overview of existing algorithms for computing the genus of a graph and their limitations.

3.1. **Existing Algorithms and Limitations**

The algorithm MULTI_GENUS created by Gunnar Brinkmann is a fast method for computing the genus of graphs with relatively low genus compared to their average vertex degree [5]. Roughly speaking, MULTI_GENUS is a backtracking algorithm on rotation systems: it incrementally builds an embedding starting from a planar subgraph and inserts edges one by one, informed by simple genus bounds from short closed walks. A formal runtime analysis of MULTI_GENUS has not been presented, but experimental results suggest it runs on graphs with high vertex degrees efficiently. However, PAGE seems to be advantageous for graphs whose maximum degree is at most 5. Currently, MULTI_GENUS seems to be the fastest known approach for high-degree graphs, and low relative genus, whereas PAGE provides an effective alternative, and is advantageous for graphs with bounded vertex degree or high girth. Several other genus algorithms other than MULTI_GENUS have been developed, using existing optimized solvers such as those for integer linear programming [4, 7, 10]. They can compute the genus of graphs as small as K_6 in under a second. However, adding just one vertex, as in K_7, could take hundreds of hours [4]. K_8 and larger graphs are almost entirely out of reach. These algorithms also omit formal runtime bounds, which are likely super-exponential for general graphs. For example, the best available open-source implementation in SageMath exhibits a worst-case complexity of $\mathcal{O}(n(n-1)!^n)$ [10]. In the next subsection, we discuss the results and discoveries of PAGE. These will be presented without proof, we refer the reader to the preprint *An Efficient Genus Algorithm Based on Graph Rotations* for a complete presentation [23].

[2]See also Metzger's honors thesis [22]

3.2. The PAGE Algorithm

We now formally construct the algorithm PAGE. To begin, we re-arrange the fundamental Euler's formula from earlier. In any orientable 2-cell embedding of G with f faces on S_g,

$$n - m + f = 2 - 2g \iff g = 1 + \frac{m - n - f}{2}.$$

We note that for a fixed graph G this implies that minimizing the orientable genus is equivalent to maximizing the number of faces f of a cellular embedding. Of course, the boundary of each face is as we noted earlier a closed walk. Additionally, each such boundary can be viewed as a closed, non-backtracking trail of the directed edges of G. Throughout this thesis, a *closed trail* always refers to a closed directed trail that does not immediately follow any directed edge by its reverse. Additionally we use the following definition.

Definition 3.1 (Realizability)**.** We say that a collection $\mathcal{F}$ of *closed trails* is *realizable* as a rotation system of G if there exists a rotation system π of G whose facial walks exactly match the closed trails in $\mathcal{F}$.

PAGE is motivated by the observation that every 2-cell embedding of G on an orientable surface S can be recovered from a realizable collection of closed trails $\mathcal{F}$, and that, conversely, collections of closed-trails must follow a number of easily verifiable constraints to be realizable. Thus, instead of searching over all rotation systems, PAGE searches over collections of closed trails and uses these constraints to quickly prune the search space.

3.2.1. *Pre-Processing.* We limit PAGE to only analyzing connected graphs with all vertices having degree 3 or more, as all graphs simplify to this case per the following lemma and Theorem 2.32.

Lemma 3.2. *Any graph G can be simplified to a graph G' in which every vertex has degree at least 3 without changing its orientable genus.*

Proof. Isolated degree 0 vertices can be placed anywhere on S without affecting crossings. A degree 1 vertex can always be attached to its neighbor via a short edge without affecting crossings. A degree 2 vertex v with neighbors x and y can be removed by contracting one of the edges xv or yv. This is the inverse of subdividing the resulting edge xy and as we saw in Theorem 2.18 subdividing an edge does not affect the genus. □

3.2.2. *Necessary Conditions for Realizability.* We now collect basic necessary conditions a collection of closed trails $\mathcal{F}$ must satisfy in order to be realizable.

Lemma 3.3 ([23])**.** *Given a graph G,*

- *any set of facial walks induced by a rotation system must use each directed edge exactly once,*
- *the set of closed trails C of G is finite and any set of facial walks induced by a single rotation system of G is a subset of C,*
- *any vertex $v \in V$ of degree d occurs exactly d times as part of a closed trail in any set of facial walks induced by a rotation system of G,*
- *any set of facial walks induced by a rotation system must be a set of closed trails whose lengths add up to the number of directed edges $2m$.*

The first condition follows from the face-tracing procedure in Section 2.3. In particular, an edge is either on the boundary of two distinct faces (each dart used once, one in each face) or appears twice on the boundary of the same face (each dart used once, for the two occurrences of the edge on that face). The second condition is simply stating that facial walks from a rotation system are closed trails, so any set of facial walks is a set of closed trails. For the third condition, if a vertex v has degree d, there are d darts incident to v which by the first condition must be used exactly once each so that v occurs exactly d times in the entire collection of facial walks. Finally, the fourth condition is immediate from the first as the length of the facial walks uses each directed edge exactly once, they use a total of $2m$ directed edges.

Theorem 3.4 ([23])**.** *Let G be a graph. Then any collection of facial walks $\mathcal{F}$ induced by a rotation system of G is a set of closed trails that satisfies the following:*

(1) Every directed edge appears in exactly one closed trail.
(2) There do not exist two distinct closed trails c_1 and c_2 such that c_1 contains the sequence of directed edges $(a, b), (b, c)$ and c_2 contains the reversed sequence $(c, b), (b, a)$ for any vertex b with $\deg(b) > 2$.

Condition (2) is necessary, as if two such closed trails existed the local rotation at vertex b would be impossible to define as a single cyclical permutation. We refer the reader to the proofs in the preprint *An Efficient Genus Algorithm Based on Graph Rotations* [23]. In the following, we discuss why the above necessary conditions are not also sufficient conditions for ensuring the realizability of a collection of closed trails $\mathcal{F}$.

3.2.3. *Sufficiency and Limitations.* The following is a partial converse of Theorem 3.4.

Theorem 3.5 ([23])**.** *Let G be a graph and $\mathcal{F}$ be a collection of closed trails satisfying the conditions of Theorem 3.4. If every vertex v of G has $\deg(v) \leq 5$ then $\mathcal{F}$ is realizable as a set of facial walks of some rotation system of G. Additionally, if some vertex $v \in V(G)$ has $\deg(v) > 5$ the conditions in Theorem 3.4 are not sufficient.*

Proof. Fix a vertex v of degree d and label its incident edges $e_1, \ldots, e_d$. Traverse the closed trails in $\mathcal{F}$. Each time a closed trail enters v along a directed edge $e_i = (u, v)$ and leaves along $e_j = (v, w)$, we say that the ordered pair (e_i, e_j) *occurs* at v. This defines a map

$$\pi_v : \{e_1, \ldots, e_d\} \to \{e_1, \ldots, e_d\}, \qquad \pi_v(e_i) = e_j \quad \text{if } (e_i, e_j) \text{ occurs at } v.$$

We now show that π_v is a cyclic permutation of the incident edges of v. Let $e_j = \{v, w\}$. Since each directed edge (v, w) occurs in exactly one closed trail by Theorem 3.4, there exists some trail entering v along (u, v) and exiting along (v, w). Let $e_i = \{u, v\}$. Then (e_i, e_j) occurs at v, and so $\pi_v(e_i) = e_j$. Thus, π_v is surjective. As the set $\{e_1, \ldots, e_d\}$ is finite π_v is also injective, so it is a bijective map from $\{e_1, \ldots, e_d\}$ to itself and hence is a permutation of $\{e_1, \ldots, e_d\}$. To show that π_v has no fixed points, suppose for contradiction that $\pi_v(e_i) = e_i$. Then (e_i, e_i) occurs at v, meaning some trail passes through v along (u, v) and immediately backtracks along (v, u), contradicting the non-backtracking condition. To show that π_v has no 2-cycles, suppose $\pi_v(e_i) = e_j$ and $\pi_v(e_j) = e_i$ for $e_i \neq e_j$. Let $e_i = \{u, v\}$ and $e_j = \{w, v\}$. Then there exist two trails c_1 and c_2 such that c_1 contains $(u, v), (v, w)$ and c_2 contains $(w, v), (v, u)$, violating condition (3) of Theorem 3.4. Suppose that π_v decomposes into $k > 1$ disjoint cycles $\pi_v = C_1 \circ C_2 \circ \cdots \circ C_k$. Since there are no fixed points or transpositions,

each C_i has length at least 3, so

$$d \geq \sum_{i=1}^{k} \text{Length}(C_i) \geq 3k \quad \Rightarrow \quad k \leq \left\lfloor \frac{d}{3} \right\rfloor.$$

But $d \leq 5$, so $k = 1$. Therefore, π_v is a cyclic permutation. Defining π_v at each vertex $v \in V$ gives a rotation system for G, and by construction, the facial walks of this rotation system are precisely the closed trails in $\mathcal{F}$. In Figure 19 the conditions of Theorem 3.4 are met, yet the closed trails cannot be realized by a rotation system. □

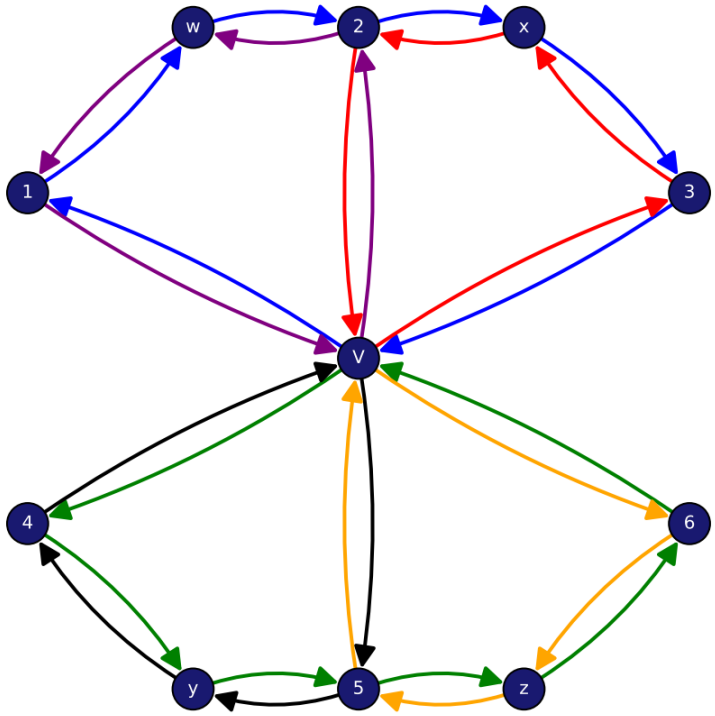


Figure 19. Degree 6 example showing Theorem 3.4 is not sufficient.

3.2.4. *Construction of PAGE.* The above lemmas and theorems describe what any realizable collection of closed trails must satisfy. We now explain how PAGE is constructed to search for such a collection.

Definition 3.6 (Candidate Set of Closed Trails)**.** Let $C(G)$ denote the finite set of closed trails of G. A subset $\mathcal{F} \subseteq C(G)$ is called a candidate set of closed trails if

- no directed edge appears in more than one trail of $\mathcal{F}$, and
- the total length of all trails in $\mathcal{F}$ is at most $2m$.

It is called *full* if the total length is exactly $2m$, otherwise it is a *partial* candidate set. Every rotation system of G produces a full candidate set via its set of facial walks, and conversely, any realizable collection of facial walks must be a full candidate set.
The subroutine `PotentialMaxFit` tests whether a partial candidate set $\mathcal{F}$ can be extended to a full candidate set that is also realizable. As we grow $\mathcal{F}$ by adding one closed trail at a time, `PotentialMaxFit` maintains:

- for each directed edge of G, whether it occurs in some trail of $\mathcal{F}$,
- for each vertex $v \in V(G)$ how many times v occurs in all trails of $\mathcal{F}$,
- the total number of darts used so far, and
- for each vertex $v \in V(G)$, a partial permutation π_v of the edges incident with v where: whenever a trail in $\mathcal{F}$ contains $e_i \to v \to e_j$ we record $\pi_v(e_i) = e_j$.

When attempting to add a closed trail $\mathcal{T}$ to $\mathcal{F}$, `PotentialMaxFit` immediately rejects the new set if any of the following occur:

(1) some directed edge of T already lies in $\mathcal{F}$,
(2) for some vertex $v \in T$, the updated amount of total occurrences of v in $\mathcal{F}$ would exceed $\deg(v)$,
(3) the updated total number of darts exceeds $2m$,
(4) two trails violate condition (2) of Theorem 3.4, or
(5) for some vertex $v \in V(G)$, the permutation π_v decomposes into more than one cycle.

Conditions 1-3 ensure that $\mathcal{F}$ can be extended to a full candidate set, condition 4 ensures that requirement (2) of Theorem 3.4 is satisfied, and condition 5 ensures realizability in case there are vertices of degree greater than 5 where, as we saw in Theorem 3.5, satisfying all of the above conditions may not be sufficient for realizability. In particular, once $\mathcal{F}$ is full, `PotentialMaxFit` accepts $\mathcal{F}$ if and only if $\mathcal{F}$ is realizable as a rotation system.
At this point, we could naively implement PAGE by enumerating all subsets of $C(G)$, keeping the subsets that form candidate sets, using `PotentialMaxFit` to filter out those which cannot be extended to a full realizable collection, and whenever we reach a full candidate set that `PotentialMaxFit` accepts, we have a 2-cell embedding and can record its number of faces to compute its genus. However, there are still optimizations that greatly improve the runtime of `PAGE`.

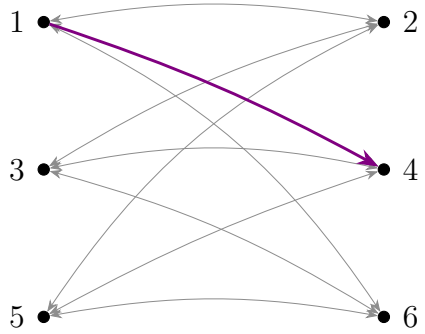
3.2.5. *Optimization Via Backtracking.* First, we enumerate all closed trails in $C(G)$ the k-trail finding algorithm. Next, we filter with `PotentialMaxFit` to only keep the subsets $\mathcal{F}$ which form candidate sets. During the backtracking search, we keep a candidate set $\mathcal{F}$ and the information maintained by `PotentialMaxFit`. At each recursive call we choose a directed edge e by:

- If $\mathcal{F}$ is empty, every dart is unused. We choose a dart that is contained in as few trails $\mathcal{T}$ of $C(G)$ as possible.
- If $\mathcal{F}$ is nonempty, we choose from directed edges (b, a) that are not included in any trail $\mathcal{T}$ of $\mathcal{F}$ but where (a, b) is included in some trail $\mathcal{T}$ of $\mathcal{F}$. Among all such edges, we choose the one that lies in the least amount of trails of $C(G)$ that have not already been ruled out.

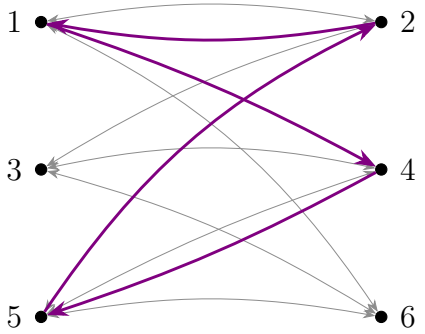
We iterate over trail combinations in order of decreasing number of trails, in order to stop searching when we first find such a realizable candidate set. Since genus is inversely related to the number of faces in an embedding, and each face corresponds to a closed trail, ordering by decreasing number of trails prioritizes embeddings on surfaces of lower genus. This allows the algorithm to halt as soon as a minimal-genus realization is found, while guaranteeing the search finds the minimum genus over all valid rotation systems without having to search them all. We define methods `RequiredEdge` and `AllClosedTrailsWithEdge` to identify the edge that must be included given the current candidate set and enumerate all the closed trails containing that edge as described above. All this being said, we can finally formalize these optimizations in The PAGE Algorithm which completes PAGE.

The PAGE Algorithm

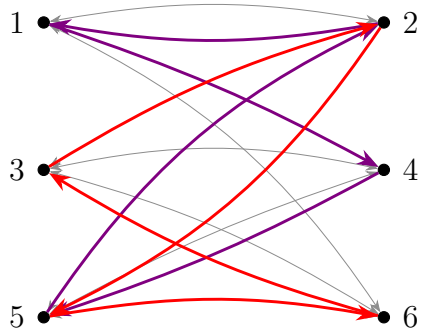
```
1: procedure SEARCH(G, candidate_set)
2:     required_edge ← REQUIREDEDGE(G, candidate_set)
3:     trails_to_check ← ALLCLOSEDTRAILSWITHEDGE(G, required_edge)
4:     for each trail in trails_to_check do
5:         candidate_set ← candidate_set ∪ trail
6:         if PotentialMaxFit(candidate_set) rejects then
7:             candidate_set ← candidate_set \ trail
8:             continue
9:         end if
10:        if candidate_set contains 2m directed edges then
11:            return (V − E + |candidate_set| − 2)/(−2)
12:        end if
13:        recurse ← SEARCH(G, ∅)
14:        if recurse ≠ −1 then
15:            return recurse
16:        else
17:            candidate_set ← candidate_set \ trail
18:            continue
19:        end if
20:    end for
21:    return -1
22: end procedure
23: SEARCH(G, ∅)
```



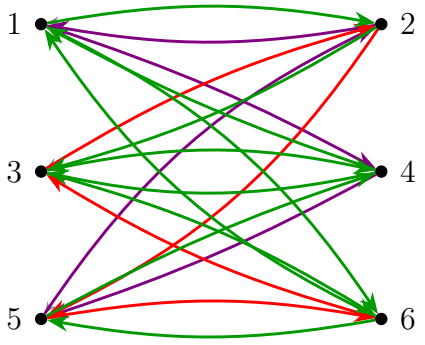
(a) PAGE starts with the initial graph and chooses a dart, such as $1 \to 4$.



(b) PAGE then iterates over cycles containing $1 \to 4$ such as this purple cycle. This cycle contains the dart $5 \to 2$, knowing that another facial walk must contain the dart $2 \to 5$ PAGE iterates again.



(c) Such cycle as the red cycle, it contains the dart $6 \to 3$, again, some facial walk must contain the dart $3 \to 6$ PAGE begins iterating over all such cycles.



(d) PAGE finds the green cycle which completes a full realizable collection of trails. PAGE applies Euler's formula and determines the genus to be 1.

Figure 20. PAGE applied to $K_{3,3}$, incrementally building a collection of closed trails. Adapted from [22].

k-trail Finding Algorithm

```
Input: Graph G = (V, E) and length k
Output: Sequence of length-k closed trails of G
closed_trails ← ∅
queue ← FIFO queue with each single vertex path
while queue is not empty do
    path ← dequeue(queue)
    if len(path) = k then
        if first vertex of path is a neighbor of last vertex then
            if last vertex is not the same as second vertex then
                if directed edge from last to first vertex is not a repeat then
                    closed_trails ← closed_trails ∪ path
                end if
            end if
        end if
    else
        for each neighbor v of last vertex in path do
            if len(path) > 2  & v is the second to last vertex of path then
                continue
            end if
            if directed edge from last vertex to v is a repeat then
                continue
            end if
            if v ≥ first vertex of path then
                enqueue(queue, path ∪ {v})
            end if
        end for
    end if
end while
```

For a more detailed description of the algorithms, or to access PAGE's open-source implementation[3] the reader is referred to the preprint [23].

3.3. PAGE Results and Discoveries

Theorem 3.7 (Main Result)**.** *Let G be a finite connected simple graph with n vertices, m edges, and girth t. PAGE as described in Section 3.3 determines the genus of G with runtime in*

$$\mathcal{O}(n(4^m/n)^{n/t}).$$

We emphasize that PAGE is relatively easy to implement. We now describe a number of examples where PAGE determined the genus of graphs with previously unknown genus.
A graph family of special interest is the (r, t)-cage graphs. They are the smallest r-regular graphs with girth t. The genus of the $(3, t)$ cage graphs was known up to $t = 10$, and PAGE extended these results by determining the genus of the $(3, 12)$ cage in just 254.45 seconds.

Theorem 3.8 ([23])**.** *The genus of the unique $(3, 12)$ cage graph is 17.*

Although the $(3, t)$ cage graphs are not known for $t > 12$, PAGE is likely applicable to higher values of t as these graphs are discovered. PAGE outperforms all existing algorithms, including MULTI_GENUS, for $t > 8$, and is the only tractable algorithm for $t \geq 12$. A related example is the (6, 12) cage graph. It has 7812 vertices, 23436 edges, and an automorphism group of nearly 6 billion elements. Nonetheless, PAGE established the bounds in Figure 21 for the genus of the (6,12) cage graph before encountering memory limitations.

[3]The open-source implementation of PAGE is available on Metzger's GitHub.

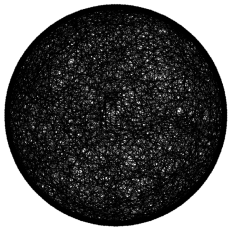


Figure 21. The (6,12) Cage Graph $5860 \leq g \leq 7810$ [23].

Additionally, another graph family that PAGE performed well on is the complete n-partite graph $K_{2,2,\ldots,2}$, also known as the cocktail party graph of order n. It is conjectured to have genus $\lceil (n-1)(n-3)/3 \rceil$ for all n, proven for all n not a multiple of 3 [14]. The complete n-partite graph is meant to represent the problem of how many handshakes are possible in a room of n couples if no one shakes their own partner's hand, and has many applications in combinatorics. It is known that $K_{2,2,\ldots,2}$ is isomorphic to the circulant graph $Ci_{2n}(1, 2, \ldots, n-1)$, which is defined as the graph with vertices labeled $0, 1, \ldots, 2n-1$ where each vertex i is adjacent to vertices $(i+j) \bmod 2n$ and $(i-j) \bmod 2n$ for every $j \in \{1, 2, \ldots, n-1\}$. All circulant graphs with genus 1 and 2 have already been discovered [8]. However, not all circulant graphs are complete n-partite graphs, or of genus less than 3. In the vast majority of cases, the genus of arbitrary circulant graphs is unknown. PAGE was able to determine the genus for several circulant graphs with unknown genus in less than a second:

Theorem 3.9 (Circulants [23])**.** *The genus of $Ci_7(1, 2, 3, 6)$ is 4. The genus of $Ci_9(1, 3, 9)$ is 4. The genus of $Ci_{10}(1, 3, 5)$ is 6. The genus of $Ci_{10}(1, 6, 9)$ is 6.*

Furthermore, significant interest surrounded the genus of the Gray graph (it happens to be 7), which was addressed in a dedicated study [21]. Most existing genus algorithms, (except MULTI_GENUS, which computed its genus in 28.3 seconds), required over 42 hours to compute the genus of similar graphs, whereas PAGE achieves the same result in just a few minutes.

For much larger graphs, computing their exact genus is often infeasible and unnecessary in practice. Previous algorithms such as MULTI_GENUS either find a minimal-genus embedding or certify that no embedding exists at a given genus, thus only raising the lower bound and never lowering the upper bound. However, PAGE can improve both bounds simultaneously. When it suffices to have an embedding within some error tolerance of the fewest holes, PAGE outputs bounds on the genus that narrow with increasing iterations.

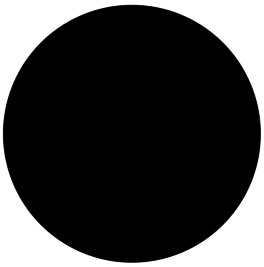
Theorem 3.10 ([23])**.** *For any graph G with genus g, PAGE computes two monotone sequences of integers*

$$g_0 \leq g_1 \leq \cdots \leq g_r = g, \qquad G_0 \geq G_1 \geq \cdots \geq G_s = g,$$

that converge to g and,

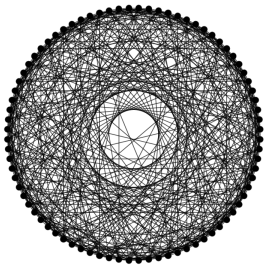
(1) PAGE halts when $g_i = G_j$, at which point $g_i = g = G_j$.
(2) The length of both sequences is finite, so PAGE always terminates.

To demonstrate the usefulness of Theorem 3.10, PAGE was applied to the below graphs and established the following genus bounds within 15 minutes.



(a) Bipartite Kneser Graph (12, 3) [41].

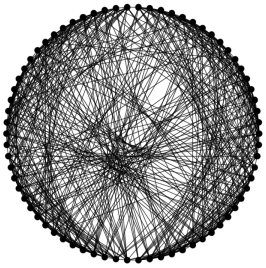
$4401 \leq g \leq 8979$



(b) DifferenceSetIncidence(40, 13, 4).

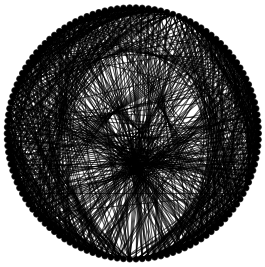
$91 \leq g \leq 214$

Figure 22. Bipartite Kneser and Difference Set Incidence Bounds.



(a) Johnson Graph (8,4) [42].

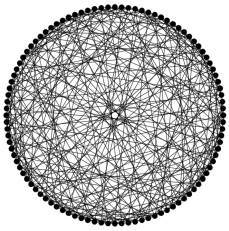
$60 \leq g \leq 238$



(b) Johnson Graph (9, 4) [42].

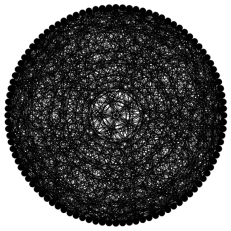
$148 \leq g \leq 558$

Figure 23. Bounds for Johnson Graphs.



(a) HoffmanSingletonBipartDoubleGraph.

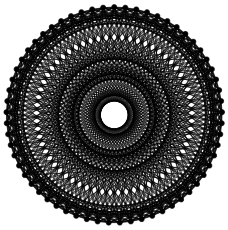
$68 \leq g \leq 100$



(b) Higman Sims Graph [39].

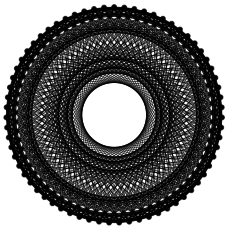
$226 \leq g \leq 490$

Figure 24. Bounds for Hoffman Singleton and Higman Sims.



(a) Cyclotomic Graph 61 [40].

$73 \leq g \leq 265$



(b) Cyclotomic Graph 67 [40].

$91 \leq g \leq 325$

Figure 25. Bounds for Cyclotomic Graphs.

The graph names shown in figures 22-25 use the built-in naming in *Mathematica*.
Finally, Table 1 gives a comparison of the runtime of PAGE with SAGEMATH [10] and MULTI_GENUS when computing the genus of the 3-regular cage graphs.

k	g	v	e	genus	PAGE (s)	SAGEMATH (s)	MULTI_GENUS (s)
3	3	4	6	0	0.008	0.004	0.006
3	4	6	9	1	0.008	0.039	0.006
3	5	10	15	1	0.008	0.027	0.006
3	6	14	21	1	0.008	0.010	0.006
3	7	24	36	2	0.010	1.737	0.006
3	8	30	45	4	0.032	118.958	0.012
3	9	58	87	7	1.625	DNF	45.099
3	10	70	105	9	39.211	DNF	9863.72
3	12	126	189	17	254.45	DNF	DNF

Table 1. Genus and time measurements (in seconds).

While PAGE and other genus algorithms are useful for determining the genus of graphs and producing embeddings, they do not structurally explain their results in the sense that Kuratowski's theorem does for planar and non-planar graphs. On the sphere, the forbidden minors $K_{3,3}$ and K_5 tell us exactly why a graph fails to be planar, and on other surfaces we would like a similar description. As we have seen throughout the thesis, the next surface of interest where the behavior of graph embeddings becomes more complicated than in the plane is the torus. The next section shifts the focus from computing the genus of arbitrary graphs to the problem of determining the forbidden toroidal minors.

4. The Forbidden Toroidal Minors

So far in this thesis, we have discussed *how* to compute the genus of a graph and the difficulties that come with it. Now, we will discuss the question of, given a surface, which graphs do not embed in it? When the surface is the sphere, we've seen that Kuratowski's Theorem gives a definitive answer: a graph fails to embed in the sphere if and only if it contains a K_5 or $K_{3,3}$ subdivision. Additionally, we know from the Robertson-Seymour Theorem that for any fixed surface, there is such a classification by forbidden minors. In this section, we will discuss the problem of determining the forbidden minors for the torus, S_1. First, we lay out the necessary terminology, then, we describe some partial classifications, and finally, we discuss the difficulty that remains in finishing the classification and potential paths forward.

4.1. Obstructions and Terminology

We say that a graph G is called *toroidal* if it embeds (possibly non-cellularly) on the torus. A non-toroidal graph G is called a *topological obstruction* for the torus if for every edge $e \in E(G)$ the graph $G \setminus e$ *is* toroidal. Further we define

Definition 4.1 (Minor-Order Obstruction)**.** A non-toroidal graph G is a minor-order obstruction for the torus if every proper graph minor of G is toroidal.

The distinction between *topological* and *minor-order* obstructions for the torus is essentially the same as the difference between topological minors and graph minors. As minor-order obstructions are minimal with respect to the operation of taking graph minors, in this thesis when we say *toroidal obstructions*, we are referring specifically to *minor-order* toroidal

obstructions. Therefore, as discussed earlier, by the Robertson-Seymour Theorem there is a finite set

$$\mathfrak{F}(S_1) = \{\mathcal{H}_1, \mathcal{H}_2 \ldots, \mathcal{H}_n\}$$

of toroidal obstructions $\mathcal{H}_i$, such that a graph is toroidal if and only if it does not contain any such $\mathcal{H}_i$ as a graph minor.

4.2. The Known Forbidden Toroidal Minors

The full set $\mathfrak{F}(S_1)$ is not known. However, there has been repeated computational progress towards building up a collection of toroidal obstructions. In 1997, Neufeld and Myrvold [29] developed an algorithm to perform an exhaustive search for small toroidal obstructions. Despite the algorithm being exponential at worst, it completed an exhaustive search on graphs up to 10 vertices, along with some larger examples, altogether finding 3884 topological obstructions (2249 of which were minor-order obstructions). Despite this list being far from complete, it was the first indication that $\mathfrak{F}(S_1)$ was much larger than the known planar or projective-planar obstruction sets. Following this, Chambers, using an improved implementation of the same algorithm, brought the exhaustive search up to 11 vertices and all 3-regular graphs on up to 24 vertices [6]. Initially, his work brought the total to 239,451 obstructions, 16,682 of which were minor-order. Chambers also introduced the following definition for classifying obstructions

Definition 4.2 (Split-Delete Minimal Obstructions)**.** A torus obstruction G is split-delete minimal if G cannot be obtained from an obstruction that has one less vertex by splitting at some vertex and then deleting some subset of the edges.

This definition motivates a classification of the full set, $\mathfrak{F}(S_1)$, by enumerating all split-delete minimal obstructions and building the rest from them. Motivated by this, in 2005 Myrvold and Woodcock [28] implemented a toroidality testing algorithm, inspired by the planarity testing algorithm of Demoucron, Malgrange, and Pertuiset [9] and used it to verify Chamber's work and to extend the exhaustive search from 11-vertex graphs to 12-vertex graphs, and from 24-vertex 3-regular graphs to 26-vertex 3-regular graphs. This enlarged the size of the database to contain, 250,815 toroidal obstructions, 17,523 of which are minor-order, all of which arise from only 402 split-delete minimal obstructions. Despite this large collection, Myrvold and Woodcock point out that their collection is likely not complete, noting that it is not difficult to find new obstructions using random search [28]. Despite this large and growing collection, there is complete classifications of some subcategories of toroidal obstructions. For example, it easy to classify disconnected and 1-connected toroidal obstructions using the earlier theorem that the genus of a graph is additive over its blocks.

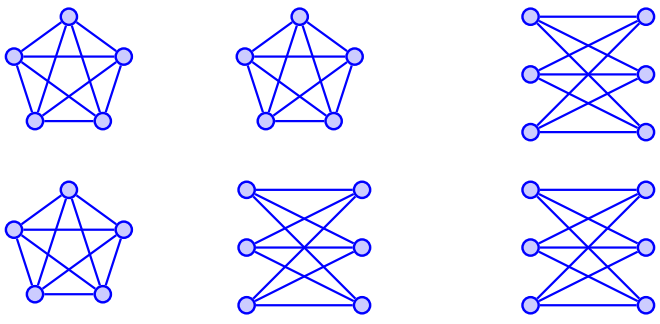


Figure 26. Disconnected toroidal obstructions.

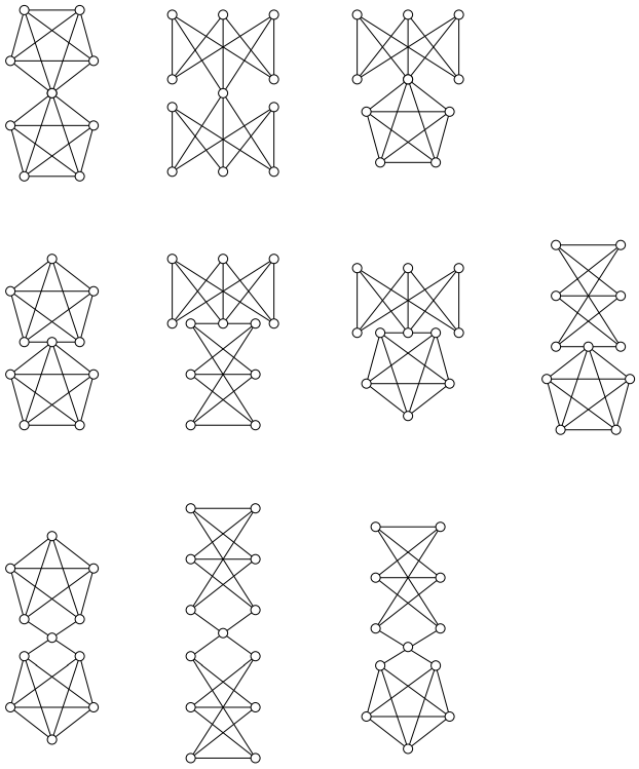


Figure 27. 1-Connected Toroidal Obstructions [28].

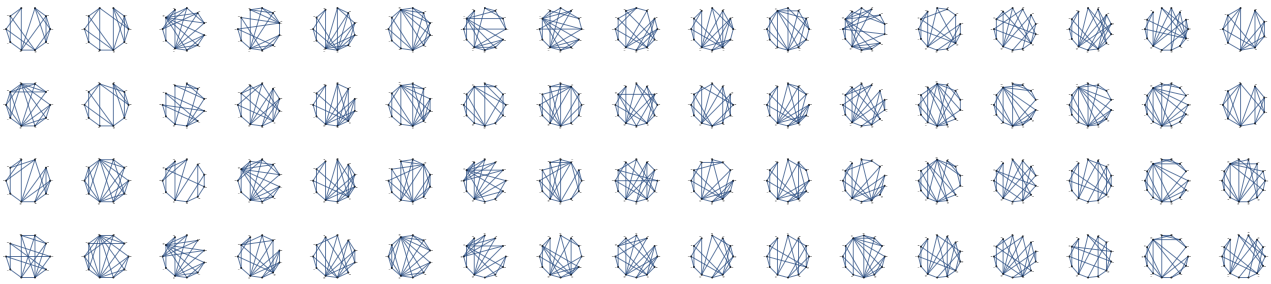


Figure 28. 68 non-isomorphic 2-connected minor-order obstructions.

Additionally, Mohar and Škoda classified all obstructions of connectivity 2 [25]. Recall that by Kuratowski's theorem, any toroidal obstruction must contain a graph minor of either K_5 or $K_{3,3}$. So, outside of categorizing obstructions purely by connectivity, in 2007 Gagarin, Myrvold, and Chambers classified all toroidal obstructions that do not contain a $K_{3,3}$ graph minor [12]. However, there are only 4 such minor-order obstructions and 11 such topological obstructions. It seems plausible, that in determining the full set $\mathfrak{F}(S_1)$ the approach should be a piece-meal strategy of forbidding specific substructures and categorizing subcategories of obstructions based on that, as in Mohar and Škoda's work [25], along with Gagarin, Myrvold, and Chamber's [12]. However, it would seem that partitioning the set of obstructions into 'those that do not contain a $K_{3,3}$', 'those that do not contain a K_5', and 'the rest', is not the right partition. Of Myrvold's current database of 17,523 minor-order toroidal obstructions,

it seems that at most 74 of them do not contain K_5 minor. Along with the 4 minor-order obstructions found by Gagarin, Myrvold, and Chambers [12], these make up a minuscule portion of the total set $\mathfrak{F}(S_1)$ of forbidden toroidal minors.
In the next section, we identify and discuss other open-problems and potential improvements related to the topics covered in this thesis.

5. Open Problems and Future Work

In this final section, we briefly outline some open problems and natural directions that follow from the topics discussed in this thesis. First, we discuss avenues for progress in finding the full set $\mathfrak{F}(S_1)$ of toroidal obstructions. Then, we illustrate the shortcomings of the discussed genus computing algorithms from earlier and methods that could be used to push them further.

5.1. Partitioning $\mathfrak{F}(S_1)$ by Connectivity

As we noted in the previous section, a hopeful strategy for completing the set $\mathfrak{F}(S_1)$ of forbidden toroidal minors is to split it into manageable subcategories. A natural way to partition obstructions is by their connectivity. It is easy to see that there are no toroidal obstructions with connectivity greater than 6.

Lemma 5.1. *If G is a toroidal obstruction, then $\kappa(G) \leq 6$.*

Proof. If G is a toroidal obstruction, it is trivial by Euler's formula that, removing any edge e,

$$\begin{aligned} g(G) = 2 \qquad &\implies m \leq 3v + 6, \\ g(G \setminus e) = 1 \qquad &\implies m \leq 3v. \end{aligned}$$

Suppose that $\kappa(G) \geq 7$ then $\delta(G) \geq 7$. The handshake lemma implies,

$$m \geq \frac{v}{2}\delta(G) \geq \frac{7v}{2}.$$

Then,

$$|E(G \setminus e)| \geq \frac{7v}{2} - 1 = 3v + \left(\frac{v}{2} - 1\right) > 3v,$$

for all $v > 2$. This contradicts that $\gamma(G \setminus e) = 1$. Thus, $\kappa(G) \leq 6$. □

We can see that the result is tight, as there are obstructions of connectivity 6, one is found to be split-delete minimal in [28]. The following simple results could be used to determine the structure that toroidal obstructions of connectivity 6 must have.

Theorem 5.2. *Toroidal obstructions with connectivity $4 \leq \kappa(G) \leq 6$ must contain both $K_{3,3}$ and K_5 as a minor.*

Proof. Let G be a toroidal obstruction with $4 \leq \kappa(G) \leq 6$. It is known that every non-planar 4-connected graph contains K_5 as a minor [49], so it only remains to show that G has $K_{3,3}$ as a minor. Also, [12] classifies all toroidal obstructions that do not have $K_{3,3}$ as a minor, and they all have connectivity less than or equal to 2. □

Lemma 5.3. *If G is a 6-connected toroidal obstruction, then $m = 3n$ or $m = 3n + 1$.*

Proof. By the handshake lemma,

$$2m = \sum_v \deg(v) \geq 6n \implies m \geq 3n$$

Since G is a toroidal obstruction for any edge e, $G \setminus e$ has genus 1. Thus, an embedding of $G \setminus e$ satisfies,

$$n' - m' + f = 0 \implies f = m' - n' = (m-1) - n$$

Because every edge is used in exactly 2 faces and each face has at least 3 sides, summing over the face lengths yields,

$$2(m-1) = \sum_f |f| \geq 3f = 3((m-1) - n) \tag{1}$$

This implies $m \leq 3n + 1$, completing the proof. □

Notice from (1) that $m = 3n + 1 \iff G \setminus e$ triangulates the torus. Also, if $m = 3n$ then G is necessarily 6-regular, and if $m = 3n + 1$ then G has either two degree 7 vertices or 1 degree 8 vertex.

5.2. **The Genus of the Balaban (3,11)-Cage**

A second direction for further work is to improve the current algorithmic methods for determining the genus. As discussed earlier in the thesis, the $(3, g)$-cage graphs are the smallest 3-regular graphs of girth g. The $(3, g)$-cages are known for $g \leq 12$, and as far as we know, the unique Balaban $(3, 11)$-cage is the only $(3, g)$-cage with unknown orientable genus. For instance, MULTI_GENUS reports a computation timeout, and PAGE is only able to narrow the bounds to

$$14 \leq g(G) \leq 19$$

after a few days of run-time[4]. However, we now illustrate a few ways these bounds can be easily improved. It is known that the (3,11)-cage can be obtained from the Tutte (3,12)-cage via excision[5] [43]. From this observation, we start with a rotation system computed by PAGE of the (3,12)-cage on S_{17} and modify it directly. In the rotation system, we delete the internal vertices of the excised path, and at each endpoint we replace the old neighbor on the path for the other endpoint in the cyclic list, leaving all other local rotations unchanged. This produces a rotation system of the (3,11)-cage with $f = 24$ and hence improves the upper bound to

$$g(G) \leq 17.$$

Additionally, we use the following heuristic to improve the lower-bound on the genus of the (3,11)-cage. Suppose for the sake of contradiction that there were a 2-cell embedding of G on S_{14}. By Euler's Formula,

$$n - m + f = 2 - 2g \iff f = 2 + m - n - 2g = 2 + 168 - 112 - 2(14) = 30.$$

[4]Amending PAGE to run with the improved bounds of $15 \leq g(G) \leq 17$ it still does not determine the genus in reasonable time. After a long search not improving the lower-bound of 15, it is tempting to think that the bottleneck of the search is that the genus is *likely not* 15, and that PAGE simply takes too long to rule out the vast possibilities of a genus-15 rotation system. But, keep in mind that the genus is already fixed, the randomness is entirely in our search. A long runtime without improving bounds is equally compatible with 'there are few genus-15 rotations in a huge search space' and 'no genus-15 rotation exists'. In that informal sense, it is fun to think that $g(G)$ is both 15 and not 15 at the same time until we either find a genus-15 rotation or prove one does not exist. Like Schrödinger's-genus.

[5]https://mathworld.wolfram.com/GraphExcision.html

Since the girth of the (3,11)-cage is 11, every facial-walk has length at least 11. If we partition the $2m = 336$ darts into 30 facial-walks of length at least 11, a simple counting argument shows that at least 24 facial-walks must have a length of exactly 11. Since the (3,11)-cage has girth 11, any such facial walk must be a simple chordless cycle of length 11, which are easy to enumerate. After enumerating all such cycles, we can set up an ILP problem which attempts to maximize the number of 11-cycles subject to the constraint that each edge $e \in E(G)$ can be used at most twice in all the chosen cycles. Solving this shows that the maximum amount of cycles subject to that condition is 16, not the required 24 and thus,

$$15 \leq g(G).$$

Combining this with the improved upper-bound we have

$$15 \leq g(G) \leq 17.$$

This shows that general purpose genus algorithms still lend themselves to further optimizations, especially when altered to compute the genus of a very specific graph.

Throughout this thesis, we examined the topic of graph embeddings from both an algorithmic and structural perspective. The discussion of PAGE, toroidal obstructions, and the specific examples such as the $(3, 11)$-cage show how both algorithmic and structural tools can be used together, culminating in the discussion of the open problems at the end, which give natural directions where these topics can be developed further.

Appendix A. Topological Prerequisites

Definition A.1. [Topological Space [26]] A *topology* on a set X is a collection τ of subsets of X having the following properties:

(1) $\emptyset$ and X are in τ.
(2) The union of the elements of any subcollection of τ is in τ.
(3) The intersection of the elements of any finite subcollection of τ is in τ.

A set X for which a topology τ has been specified is called a *topological space*.

Definition A.2. [Homeomorphism [26]] Let X and Y be topological spaces and $f : X \to Y$ a bijection. If both the function f and the inverse function

$$f^{-1} : Y \to X$$

are continuous then f is a *homeomorphism*.

Definition A.3 (Covers [26])**.** A collection $\mathcal{A}$ of subsets of a space (X, τ) is said to cover X, or to be a covering of X, if the union of elements of $\mathcal{A}$ is equal to X. It is called an open covering of X if its elements are open subsets of X.

Definition A.4. [Compact Spaces [26]] A topological space (X, τ) is said to be *compact* if every open covering of X contains a finite subcollection that also covers X.

Definition A.5. [Connected Spaces [26]] Let X be a topological space. A separation of X is a pair U, V of disjoint nonempty open subsets of X whose union is X. The space X is said to be connected if there does not exist a separation of X.

Definition A.6. [Hausdorff Spaces [26]] A topological space X is called a Hausdorff space if for each pair x_1, x_2 of distinct points of X, there exist neighborhoods U_1, and U_2 of x_1 and x_2, respectively, that are disjoint.

Definition A.7. [Surface with Boundary [26]] Let H^2 be the subspace of $\mathbb{R}^2$ consisting of all points (x_1, x_2) with $x_2 \geq 0$. A surface with boundary is a Hausdorff space X with a countable basis such that each point x of X has a neighborhood homeomorphic with an open set of $\mathbb{R}^2$ or H^2.

Definition A.8 (Polygonal Regions [26])**.** Given a point $c \in \mathbb{R}^2$ and given $a > 0$, consider the circle of radius a in $\mathbb{R}^2$ with center at c. Given a finite sequence $\theta_0 < \theta_1 < \cdots < \theta_n$ of real numbers where $n \geq 3$ and $\theta_n = \theta_0 + 2\pi$, consider the points $p_i = c + a(\cos(\theta_i), \sin(\theta_i))$, which lie on this circle. They are numbered in counterclockwise order around the circle, and $p_n = p_0$. The line through p_{i-1} and p_i splits the plane into two closed half-planes; let H_i be the one that contains all the points p_k. Then the space

$$P = H_1 \cap \cdots \cap H_n$$

is called the *polygonal region* determined by the points p_i. The points p_i are called the *vertices* of P; the line segment joining p_{i-1} to p_i is called an *edge* of P; the union of the edges is denoted $\operatorname{Bd} P$; and $P \setminus \operatorname{Bd} P$ is denoted $\operatorname{Int} P$. Given a line segment L in $\mathbb{R}^2$, an orientation of L is simply an ordering of its end points; the first, say a, is called the initial point, and the second, say b, is called the final point, of the oriented line segment. We often say that L is oriented from a to b; and we picture the orientation by drawing an arrow on L that points from a towards b. If L' is another line segment, oriented from c to d, then the positive linear map of L onto L' is the homeomorphism h that carries the point $x = (1 - s)a + sb$ of L to the point $h(x) = (1 - s)c + sd$ of L'.

Definition A.9 (Labelings [26])**.** Let P be a polygonal region in the plane. A labeling of the edges of P is a map from the set of edges of P to a set S called the set of labels. Given an orientation of each edge of P, and given a labeling of the edges of P, we define an equivalence relation on the points of P as follows: Each point of $\operatorname{int} P$ is equivalent only to itself. Given any two edges of P that have the same label, let h be the positive linear map of one onto the other, and define each point x of the first edge to be equivalent to the point $h(x)$ of the second edge. This relation generates an equivalence relation on P. The quotient space X obtained from this equivalence relation is said to have been obtained by pasting the edges of P together according to the given orientations and labeling.

Definition A.10 (Labeling Schemes [26])**.** Let P be a polygonal region with successive vertices $p_0, \ldots, p_n$, where $p_0 = p_n$. Given orientations and a labeling of the edges of P, let $a_1, \ldots, a_m$ the distinct labels that are assigned to the edges of P. For each k, let a_{i_k} be the label assigned to the edge $p_{k-1}p_k$ and let $\epsilon_k = +1$ or -1 according as the orientation assigned to this edge goes from p_{k-1} to p_k or the reverse. Then the number of edges of P, the orientations of the edges, and the labeling are completely specified by the symbol

$$w = (a_{i_1})^{\epsilon_1}(a_{i_2})^{\epsilon_2} \ldots (a_{i_n})^{\epsilon_n}$$

We call this symbol a labeling scheme of length n for the edges of P; it is simply a sequence of labels with exponents $+1$ or -1.

Definition A.11. [n-Fold Connected Sum of Tori [26]] Consider the space obtained from a $4n$-sided polygonal region P by means of the labeling scheme

$$(a_1 b_1 a_1^{-1} b_1^{-1})(a_2 b_2 a_2^{-1} b_2^{-1}) \ldots (a_n b_n a_n^{-1} b_n^{-1}).$$

This space is called the n-fold connected sum of tori, or simply the n-fold torus, and denoted $T\# \ldots \# T$.

Definition A.12. [h-Fold Connected Sum of Projective Planes [26]] Let $h > 1$. Consider the space obtained from a $2h$-sided polygonal region P by means of the labeling scheme

$$(a_1 a_1)(a_2 a_2) \ldots (a_h a_h).$$

This space is called the h-fold connected sum of projective planes, or simply the h-fold projective plane, and denoted

$$\mathbb{RP}^2 \# \ldots \# \mathbb{RP}^2$$

Definition A.13 (Cell Decomposition [18])**.** If X is a nonempty topological space, a *cell decomposition* of X is a partition $\mathcal{E}$ of X into subspaces that are open cells of various dimensions, such that the following condition is satisfied: for each cell $e \in \mathcal{E}$ of dimension $n \geq 1$, there exists a continuous map Φ from some closed n-cell D into X (called a *characteristic map* for e) that restricts to a homeomorphism from $\operatorname{Int} D$ onto e and maps ∂D into the union of all cells of $\mathcal{E}$ of dimensions strictly less than n.

Definition A.14. [Cell Complex [18]] A *cell complex* is a Hausdorff space X together with a specific cell decomposition of X.

Definition A.15 (Coherence [18])**.** Suppose X is a topological space, and $\mathcal{B}$ is any family of subspaces of X whose union is X. To say that the topology of X *is coherent with* $\mathcal{B}$ means that a subset $U \subseteq X$ is open in X if and only if its intersection with each $B \in \mathcal{B}$ is open in B.

Definition A.16. [CW Complexes [18]] A *CW complex* is a cell complex $(X, \mathcal{E})$ satisfying the following additional conditions:

(C) The closure of each cell is contained in a union of finitely many cells.

(W) The topology of X is coherent with the family of closed subspaces $\{\overline{e} : e \in \mathcal{E}\}$.

References

1. Dan Archdeacon, *A kuratowski theorem for the projective plane*, Journal of Graph Theory **5** (1981), no. 3, 243–246.
2. Dan Archdeacon, *Topological graph theory - a survey*, Topological Graph Theory - A Survey, vol. 115, 1996, p. 18.
3. Joseph Battle, Frank Harary, Yukihiro Kodama, and J.W.T Youngs, *Additivity of the genus of a graph*, Bulletin of the American Mathematical Society **68** (1962), 565–568.
4. Stephan Beyer, Markus Chimani, Ivo Hedtke, and Michal Kotrbčík, *A practical method for the minimum genus of a graph: Models and experiments*, Experimental Algorithms (Cham) (Andrew V. Goldberg and Alexander S. Kulikov, eds.), Springer International Publishing, 2016, pp. 75–88.
5. Gunnar Brinkmann, *A practical algorithm for the computation of the genus*, Ars Mathematica Contemporanea **22** (2022), no. 4.
6. John Chambers, *Hunting for torus obstructions*, Master's thesis, University of Victoria, Victoria, British Columbia, 2002, Department of Computer Science.
7. Markus Chimani and Tilo Wiedera, *Stronger ILPs for the Graph Genus Problem*, 27th Annual European Symposium on Algorithms (ESA 2019) (Dagstuhl, Germany), vol. 144, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019, pp. 30:1–30:15.
8. Marston Conder and Ricardo Grande, *On embeddings of circulant graphs*, Electronic Journal of Combinatorics **22** (2015), no. 2.
9. Georges Demoucron, Yves Malgrange, and Robert Pertuiset, *Graphes planaires*, Revue Française de Recherche Opérationnelle **8** (1964), 33–47, In French.
10. SageMath Developers, *Genus calculation in sagemath*, 2024.
11. Jack Edmonds, *A combinatorial representation for polyhedral surfaces*, Notices of the American Mathematical Society **7** (1960), 646.
12. Andrei Gagarin, Wendy Myrvold, and John Chambers, *The obstructions for toroidal graphs with no $k_{3,3}$'s*, Discrete Mathematics **309** (2009), no. 11, 3625–3631.
13. Lothar Heffter, *Ueber das problem der nachbargebiete*, Mathematische Annalen **38** (1891), 477–508.
14. Mark Jungerman and Gerhard Ringel, *The genus of the n-octahedron: Regular cases*, Journal of Graph Theory **2** (1978), no. 1, 69–75.
15. Koko90, *The tutte 12-cage*, Wikimedia Commons, 2010, Image "Tutte 12-cage.svg".
16. Kazimierz Kuratowski, *Sur le problème des courbes gauches en topologie*, Fundamenta Mathematicae **15** (1930), 271–283 (fra).
17. CMG Lee, *7 colour torus*, *Wikimedia Commons*, 2021, Accessed 27 November 2025.
18. John M. Lee, *Introduction to topological manifolds*, 2 ed., Graduate Texts in Mathematics, vol. 202, Springer, New York, 2011.
19. Jos Leys, *The cross-cap*, YouTube video, 2016, The construction of the cross-cap.
20. Berty Mandagie and Emily Mandagie, *Pacific northwest map*, https://www.themandagies.com/wp-content/uploads/2020/02/Pacific-Northwest-Map-The-Mandagies.jpg, 2020, Accessed: 2025-12-02.
21. Dragan Marušič, Tomaž Pisanski, and Steve Wilson, *The genus of the gray graph is 7*, European Journal of Combinatorics **26** (2005), no. 3, 377–385.
22. Alexander Metzger, *A practical algorithmic approach to graph embedding*, Bachelor's thesis, University of Washington, September 2025.
23. Alexander Metzger and Austin Ulrigg, *An efficient genus algorithm based on graph rotations*, arXiv preprint (2025), arXiv:2411.07347.
24. Bojan Mohar and Carsten Thomassen, *Graphs on surfaces*, Johns Hopkins University Press, 2001.
25. Bojan Mohar and Petr Škoda, *Obstructions of connectivity two for embedding graphs into the torus*, Canadian Journal of Mathematics **66** (2014), no. 6, 1327–1357.
26. James R. Munkres, *Topology*, 2 ed., Prentice Hall, Upper Saddle River, NJ, 2000.
27. Wendy Myrvold and William Kocay, *Errors in graph embedding algorithms*, Journal of Computer and System Sciences **77** (2011), no. 2, 430–438.
28. Wendy Myrvold and Jennifer Woodcock, *A large set of torus obstructions and how they were discovered*, Electronic Journal of Combinatorics **25** (2018), no. 1, P1.16.

29. Eugene Neufeld and Wendy Myrvold, *Practical toroidality testing*, Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (USA), SODA '97, Society for Industrial and Applied Mathematics, 1997, p. 574–580.
30. Sophia Potoczak Bragdon, *Survey of graph embeddings into compact surfaces*, Master's thesis, 2014, Figure 1.11, "The construction of a torus from the unit square".
31. PSL27, *Pappus graph and associated map embedded in the torus*, Wikimedia Commons, 2021, Image "Pappus-graph-on-torus.png".
32. Richard D. Ringeisen and Lowell W. Beineke, *The crossing number of* $c_3 \times c_n$, Journal of Combinatorial Theory, Series B **24** (1978), no. 2, 134–136.
33. Gerhard Ringel and J.W.T Youngs, *Solution of the heawood map-coloring problem**, Proceedings of the National Academy of Sciences **60** (1968), no. 2, 438–445.
34. Neil Robertson and Paul Seymour, *Graph minors .xiii. the disjoint paths problem*, Journal of Combinatorial Theory, Series B **63** (1995), no. 1, 65–110.
35. Saul Stahl and Lowell W. Beineke, *Blocks and the nonorientable genus of graphs*, Journal of Graph Theory **1** (1977), no. 1, 75–78.
36. Carsten Thomassen, *The graph genus problem is np-complete*, Journal of Algorithms **10** (1989), no. 4, 568–576.
37. William T. Tutte, *A census of planar maps*, Canadian Journal of Mathematics **15** (1963), 249–271.
38. Lucas Vieira, *Torus*, Wikimedia Commons, 2006, Image, originally uploaded as *File:Torus.png*.
39. Eric Weisstein, *Higman–sims graph*, https://mathworld.wolfram.com/Higman-SimsGraph.html, 2024a, From MathWorld – A Wolfram Web Resource.
40. Eric Weisstein, *Cyclotomic graph*, https://mathworld.wolfram.com/CyclotomicGraph.html, 2024b, From MathWorld – A Wolfram Web Resource.
41. Eric Weisstein, *Bipartite kneser graph*, https://mathworld.wolfram.com/BipartiteKneserGraph.html, 2024c, From MathWorld – A Wolfram Web Resource.
42. Eric Weisstein, *Johnson graph*, https://mathworld.wolfram.com/JohnsonGraph.html, 2024d, From MathWorld – A Wolfram Web Resource.
43. Wikipedia contributors, *Balaban 11-cage*, *Wikipedia, The Free Encyclopedia*, 2024, Accessed: 2025-12-04.
44. Wikipedia contributors, *Francis guthrie*, *Wikipedia, The Free Encyclopedia*, 2024, Accessed: 2025-11-27.
45. Wikipedia contributors, *Heawood conjecture*, *Wikipedia, The Free Encyclopedia*, 2024, Accessed: 2025-11-27.
46. Wikipedia contributors, *Four color theorem*, *Wikipedia, The Free Encyclopedia*, 2025, Accessed: 2025-11-27.
47. Wikipedia contributors, *Turán's brick factory problem*, *Wikipedia, The Free Encyclopedia*, 2025, https://en.wikipedia.org/wiki/Tur%C3%A1n%27s_brick_factory_problem. Accessed: 2025-12-04.
48. Xuong, Nguyen H., *How to determine the maximum genus of a graph*, Journal of Combinatorial Theory, Series B **26** (1979), no. 2, 217–225.
49. H. Peyton Young, *A quick proof of wagner's equivalence theorem*, Journal of the London Mathematical Society **s2-3** (1971), no. 4, 661–664.