A Practical Algorithmic Approach to Graph Embedding

Alexander Le Metzger

A senior thesis
submitted in partial fulfillment of the
requirements for the degree of

Bachelor's of Science in Mathematics

University of Washington

2025

Reading Committee:

Stephanie Smallwood, Chair

Stefan Steinerberger, Supervisor

Program Authorized to Offer Degree:
UW Department of Mathematics

University of Washington

Abstract

A Practical Algorithmic Approach to Graph Embedding

Alexander Le Metzger

Chair of the Departmental Honors Committee:
Director Stephanie Smallwood
University Honors Program

Minimal-genus graph embedding is about drawing graphs on surfaces with no edges crossing and as few holes as possible. This thesis first covers the necessary background in topological graph theory to understand graph embeddings through rotation systems. It then studies an adaptation of this approach, Practical Algorithm for Graph Embedding (PAGE), that takes advantage of the cycle sequence of a graph to work more efficiently in practice, especially for graphs of high girth or low degree. This enables it to determine the previously intractable genus of the $(3, 12)$-cage as 17.

TABLE OF CONTENTS

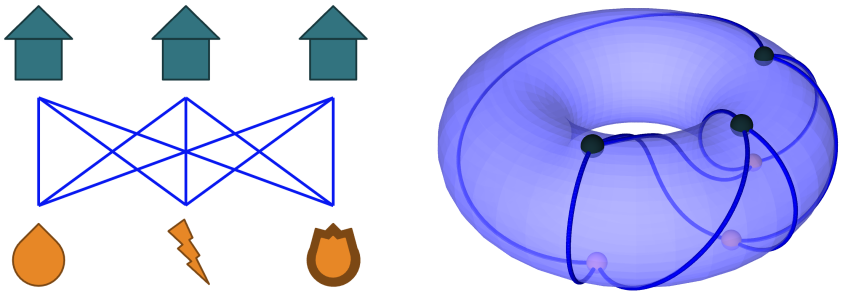
ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to Austin Ulrigg who collaborated on the design and proof of the PAGE algorithm, and to Professor Gunnar Brinkmann for inputs on visualizing results. The thesis is also heavily inspired by *Graphs on Surfaces* by Mohar and Thomassen [13], *Pearls in Graph Theory* by Hartsfield and Ringel [8], *Topological Graph Theory - A Survey* by Archdeacon [1], *Survey of Graph Embeddings into Compact Surfaces* by Potoczak [14], and *Rotation Systems and Cellular Imbeddings* by Waldrop [16].

Chapter 1

INTRODUCTION

Consider the famous Utility Problem. There are 3 houses and 3 utilities—water, electricity, and fire. Each house must be connected to each of the utilities by pipes and no crossing of pipes is allowed. This can be represented by the complete bipartite graph, $K_{3,3}$, which has two sets of 3 vertices as seen in Figure 1.1a. Kuratowski has shown that it is not possible to draw $K_{3,3}$ on a plane without edges crossing [9]. But as seen in Figure 1.1b, it is possible to draw it on a different surface, the torus.



(a) The Utility graph $K_{3,3}$ (b) $K_{3,3}$ embedded on a torus

Figure 1.1: The Utility Problem

The characterizing property of the torus that makes this drawing, called an *embedding*, possible is that it has a hole. Intuitively, more complex graphs will require more holes to embed. Yet any graph has some minimum number of holes, its *genus*, for which it embeds. Finding this minimum genus and its corresponding embedding has many applications in circuit design, data visualization, and more. Chapter 2 expounds the necessary theory and Chapter 3 explores finding these algorithmically.

Chapter 2
BACKGROUND

2.1 Topology

Starting with some definitions [14], a *topological space* X is a set with a collection T of open subsets called a *topology* that satisfies the following properties:

(a) $\emptyset$ and X are open sets

(b) the intersection of a finite number of open sets is an open set

(c) the union of open sets is an open set

A set $S \subseteq X$ is *open* if and only if $S \in T$. A set is *closed* if its complement is open. A *subspace* is a subset $A \subseteq X$ with a topology defined by taking the intersection of A with the open sets of X. An *open cover* of X is a collection of open sets such that X is contained in their union. X is *compact* if every open cover of X has a finite open subcover. X is *disconnected* if there exists a pair of disjoint nonempty open sets whose union is X, and *connected* otherwise. A *component* is a maximal connected subset of X such that adding any other elements would make it disconnected.

A function $f : X \to Y$ mapping between topological spaces X, Y is *continuous* if the preimage of every open set is open. Continuous functions preserve topologicaal properties such as connectedness and compactness. A *homeomorpism* $h : X \to Y$ is a continuous bijection with continuous inverse $h^{-1} : Y \to X$. Topological spaces are *homeomorphic* if there exists a homeomorphism between them. An *embedding* is a function $e : X \to Y$ that homeomorphically maps X to the subspace $f(X) \subseteq Y$. Notably e is not necessarily surjective.

Continuous functions allow "gluing" multiple topological spaces into one quotient space. Say a topological space X consists of subsets $A_1, \ldots, A_k$. Let $\sim$ be the equivalence relation such that all elements $a \in A_i$ are in the same equivalence class $[a]$. Then the *quotient space* is $X/\sim = \{[x] : x \in X\}$. Define the *quotient map* $\pi : X \to X/\sim$ by $\pi(x) = [x]$. The *quotient topology* of $X/\sim$ is then defined such that π is continuous. That is, a set $U \subseteq X/\sim$ is open if and only if $\pi^{-1}(U) \subseteq X$ is open in X. This is illustrated in Figure 2.1 which considers the unit interval $[0, 1]$ and takes $0 \sim 1$. The resulting quotient space is a circle obtained by gluing together 0 and 1 because they were placed in the same equivalence class and thus mapped to the same element in the quotient space. In Figure 2.4, various compact surfaces are constructed by "gluing".

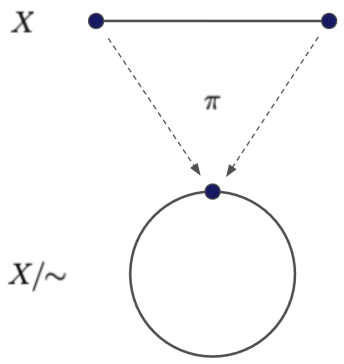


Figure 2.1: $X = [0, 1]$ is split into subsets $\{x\}$ for $x \in X$ and the endpoints are identified by letting the quotient map $\pi(\{0\}) = \pi(\{1\})$ so these subsets end up as the same point in the quotient space $X/\sim$.

2.2 Graph Theory

A *finite simple undirected graph* $G = \{V, E\}$ is a finite set of *vertices* V and *edges* E where each edge $\{u, v\} \in E$ has two distinct endpoints u, v. Henceforth, this is just a *graph*. Combinatorially, each edge and vertex has no other properties than given $\{u, v\} \in E$, u is considered *adjacent* to v. Topologically, a graph G corresponds to a topological space, its *geometric realization*, also denoted G. Here vertices are distinct points and edges are subspaces homeomorphic to $[0, 1]$ joining their endpoints. With

a quotient space, endpoints corresponding to the same vertex are "glued" together. It is this topological interpretation that allows defining the notion of embedding hinted at in the introduction. Namely, an *embedding* of G into some topological space X is a homeomorphism of the geometric realization into a subspace of X.

It is worth expressing some topological properties of the geometric realization using combinatoric definitions of the graph since these are easier to work with algorithmically. In fact, the goal of Section 2.5 will be to define embeddings purely combinatorially. First, the combinatoric definition of a *connected* graph is one such that for every pair of vertices $u, v \in V$, there is a path between them. Since endpoints are glued, this also aligns with the topological definition: a graph is connected if and only if its geometric realization is connected. Likewise, the connected components of the geometric realization directly correspond to the combinatoric connected components—maximal sets of vertices where every pair is connected by a path. A *path* is a sequence of vertices with adjacent elements in the sequence also adjacent in the graph. A *cycle* is a path where the first and last vertex are the same. A *simple cycle* has no repeated vertices. This means it is a *Jordan curve* in the geometric realization, i.e., homeomorphic to a circle. The *girth* of a graph is the number of edges in its shortest cycle. Finally, two graphs are homeomorphic if they can be obtained from the same graph through a sequence of *subdivisions*—replacing edges with paths as seen in Figure 2.2.

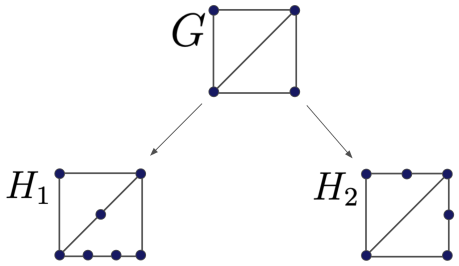


Figure 2.2: H_1 and H_2 are homeomorphic (obtained from G through subdivisions).

2.3 Compact Surfaces

So, graphs can embed into topological spaces through their geometric realization. For the purposes motivated in the introduction, a special type of topological space is of interest, namely a compact surface. This is because, by Theorem 2, all graphs embed into some compact surface. To unpack the definition of a compact surface, first note that a *surface* is a topological space S satisfying the following properties [14]:

(a) every $x \in S$ is contained in an open set that is homeomorphic to the open disk

(b) for $x, y \in S$, there exists disjoint open sets U and V such that $x \in U$, $y \in V$

(c) S has a basis of countably many open sets—a *basis* is a set of open sets such that any open set is a union of basis elements

Property (a) intuitively ensures that, zooming into any point, the surface locally looks like a flat sheet of paper as seen in Figure 2.3.

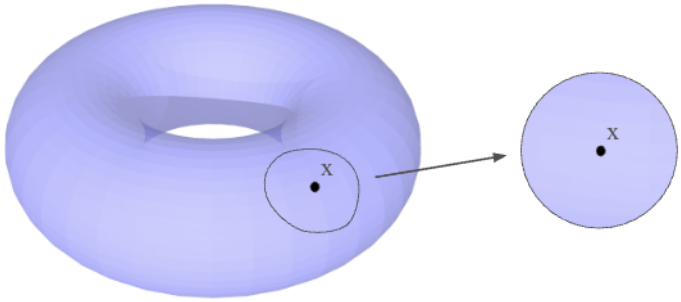


Figure 2.3: Surfaces locally look like discs.

A *compact surface* is then a compact topological space that is also a surface. This is best understood constructively. There are two kinds of compact surfaces. The *orientable* ones, like a sphere or torus, have two distinct sides (an inside and an outside). The *non-orientable* ones, like a crosscap, don't. Formally, non-orientable compact surfaces contain a subspace homeomorphic to the crosscap constructed in Figure 2.4 and orientable ones don't. In fact, any compact surface can be constructed as a quotient space of a polygon with an even number of edges by pairing and gluing certain

edges. In Figure 2.4, the unit square $[0,1] \times [0,1]$ can be partitioned into 4 subsets: $A = \{(x,y) : x,y \in (0,1)\}, B = \{(0,y),(1,y) : y \in (0,1)\}, C = \{(x,0),(x,1) : x \in (0,1)\}, D = \{(0,0),(0,1),(1,0),(1,1)\}$. A sphere, torus, and crosscap can then be formed by taking the quotient space with these as the equivalence classes. Larger polygons can visualize embeddings into more complex surfaces as in Figure 2.7.

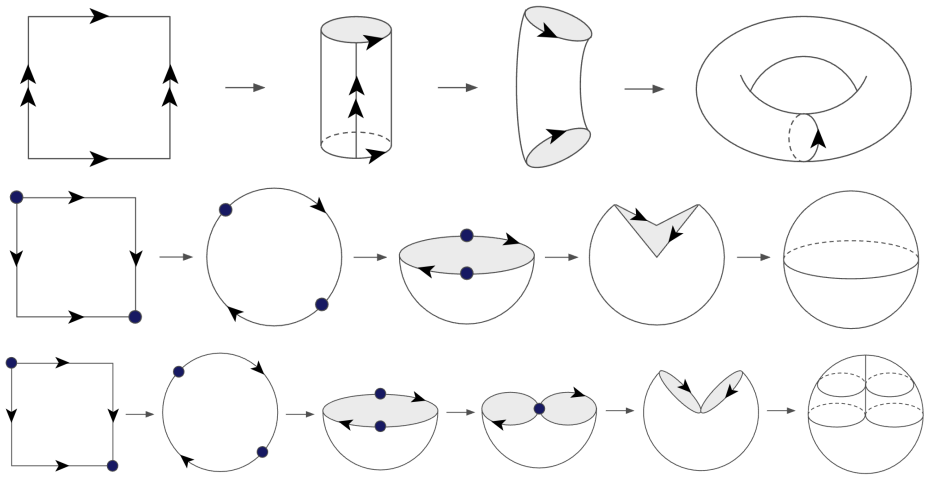


Figure 2.4: Constructing a torus (top), sphere (middle), and crosscap (bottom) by gluing a unit square (4-sided polygon). These constitute S_1, S_0, and $\tilde{S}_1$ respectively.

By Theorem 1, proved by Brahana [2], we can form all orientable compact surfaces by "gluing" together n tori to form an n-hole torus like in Figure 2.5. A compact surface that is homeomorphic to an n-hole torus has *genus* n and is denoted S_n. Similarly, all non-orientable surfaces $\tilde{S}_m$ can be formed by gluing together m crosscaps and have *non-orientable genus* m.

Theorem 1 (Brahana, [2])**.** *For all $n, m \in \mathbb{N}$, S_n and $\tilde{S}_m$ are compact surfaces. All compact orientable surfaces are homeomorphic to S_n. All compact non-orientable surfaces are homeomorphic to $\tilde{S}_m$. The surface obtained by gluing n handles and $m > 1$ crosscaps is homeomorphic to $\tilde{S}_{2n+m}$.*

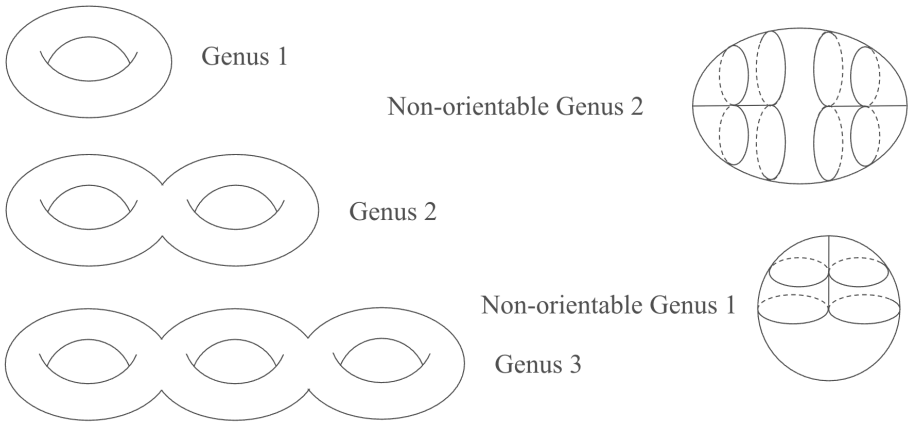


Figure 2.5: One torus is S_1. By identifying a small section of the surface and gluing, we can combine two tori into one compact surface S_2, and so on.

The nice consequence of Theorem 1 is that every graph embeds into some compact surface as promised in Theorem 2 based on the similar argument by Potoczak [14].

Theorem 2 (Potoczak, [14], 2.2.6)**.** *All graphs can be embedded into a compact surface. Moreover, all graphs can be embedded in S_n for some n and $\tilde{S}_m$ for some m.*

Proof. Let G be an arbitrary graph. Draw it in the plane with edge crossings. Add a torus at each edge crossing to make one of the edges go over the other as the other passes through the hole. This is S_n which is a compact surface by Theorem 1. Likewise by Theorem 1, G can be embedded in $\tilde{S}_{2n}$. □

2.4 Graph Embedding

The topological notion of embedding a graph is now well defined. What remains is a combinatoric definition that enables easy enumeration of possible embeddings of a graph into different surfaces so the minimum genus one can be found. First consider a graph G embedded into a surface S by $e : G \to S$. Looking at the space $S \setminus e(G)$, the connected components are called the *faces* of the embedding. The *boundary* can be obtained by walking around the subgraph consisting of the edges and vertices

touching a face. The *length* of a face is the number of edges in its boundary. If G is connected, the boundary of each face is a cycle as seen in Figure 2.6a.

(a) In connected graphs, each boundary is a cycle (possibly non-simple like f).

(b) In disconnected graphs, some faces have boundaries made of multiple cycles (like g).

Figure 2.6: Faces of Graphs.

It is well established [13] that minimum genus is additive over the connected components. Thus, without loss of generality, just the connected graph case has to be handled. Since each face is a cycle and there is a finite number of cycles, one way of enumerating embeddings would be to enumerate all cycle combinations. Unfortunately, this is an astronomical number of combinations to check and not all cycle combinations correspond to an embedding. For one, Proposition 1 restricts the total length of the faces in an embedding and Theorem 3, proved by Simon L'huiler [10], restricts the number of faces. Remark 5 discusses a number of other restrictions.

Proposition 1. *Embedded graph* $G = \{V, E\}$ *with faces* F *has* $\sum_{f \in F} \texttt{length}(f) = 2|E|$.

Proof. Each edge borders either one or two faces. If it borders two faces $f, g \in F$, it is counted once in $\texttt{length}(f)$ and once in $\texttt{length}(g)$. Otherwise it borders only one face $f \in F$, and is still counted twice in $\texttt{length}(f)$. □

Theorem 3 (Euler's Formula, [10])**.** *A connected genus* g *graph* $G = \{V, E\}$ *with faces* F *satisfies* $|V| - |E| + |F| = 2 - 2g$.

Luckily, thanks to Heffter, Edmonds, and Youngs, we have a more elegant combinatoric definition known as the rotation system.

2.5 Rotation Systems

Consider an embedding of a graph G into an orientable surface S_n. At each vertex, order the d_v incident edges counter-clockwise to form a cyclic permutation $\pi_v : e_1 \to \cdots \to e_{d_v} \to e_1$ called a *local rotation* at v. The *rotation system* is the collection of all local roations $\Pi_G = \{\pi_v : v \in V\}$. Typically, the notation for each local rotation is simplified to only list e_1 once and, since each edge has v as one of its endpoints, only the other endpoint is listed: $\pi_v : v_1 \to \cdots \to v_{d_v}$ where $e_i = \{v_i, v\}$. In some readings, such as the SageMath source code [5, 6], this is often referred to as the *darts*. This notation is exemplified in Figure 2.7 showcasing the rotation system for the embedding of $K_{3,3}$ from Figure 1.1b. Importantly, not only can every embedding be expressed as a rotation system, but every rotation system also corresponds to an embedding by Theorem 4. Consult [7] for a full proof.

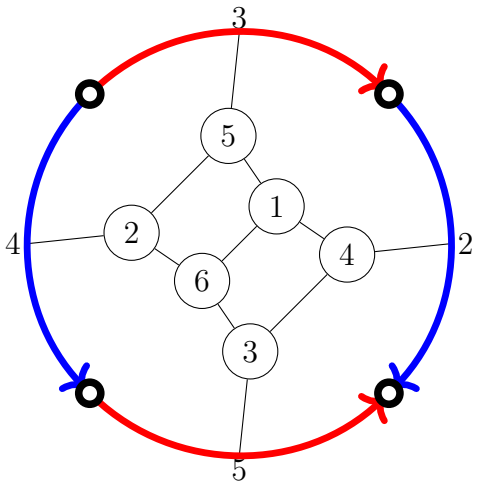


Figure 2.7: The 4 sided polygon representing the torus when corresponding colors are glued. Inside is $K_{3,3}$. On the right is the rotation system for the embedding.

Theorem 4 (Heffter-Edmonds Equivalence Theorem, [7])**.** *Every rotation system on a graph G defines a unique (up to equivalence) embedding of G onto an oriented surface. Conversely, every such embedding determines a rotation system for G.*

Proof. The converse follows from the definition, so consider a rotation system Π_G. It can be used to trace out the faces of an embedding as follows: choose a starting edge $e_0 = \{v_0, v_1\}$; walk along it from v_0 to v_1; π_{v_1} contains $\cdots \to v_0 \to v_2 \to \cdots$ so the next edge to walk along is $e_1 = \{v_1, v_2\}$; keep going until the walk returns to e_0; a cycle $e_0 \to \cdots \to e_0$ that uses each edge at most once in each direction has been traced out; starting over for each possible starting edge and direction obtains a collection of cycles that cover all the edges; from this collection of cycles, select a subset that traverses each edge exactly once in each direction; these are the faces. Each face forms a convex polygon with number of sides equal to the length of the face and each side labeled by the corresponding edge. The sides corresponding to the same edge can now be glued together. The result is a surface because each edge lies in two faces. Around each vertex v, the polygon faces are aligned with the cyclic ording π_v. It follows that the surface corresponds to the rotation system, is orientable, and thus is homeomorphic to S_n for some n. Therefore, this is an embedding. □

Remark 5. The rotation system definition highlights a number of necessary conditions for a collection of cycles to correspond to an embedding. In the proof of Theorem 4, it is clear that each directed edge must be traversed exactly once. This is not sufficient since, if a cycle contains vertices $a \to b \to a$ with b of degree greater than 1, then we'd have to glue an edge to itself which does not yield a convex polygon. Even removing all such *backtracking* cycles is not sufficient since, if one cycle contains $a \to b \to c$, another contains $c \to b \to a$, and a third contains b, then gluing the first two together as faces results in a flat crease at b that the third cycle cannot be glued to. Further, removing all such cycle pairs for vertices of degree greater than 2 is also not sufficient. For vertices of degree greater than 5, two groups of three cycles can form two "pyramids" touching at a point when glued together which violates property (a) of the definition of a surface. This pattern of edge cases continues for higher degree vertices and is illustrated in Figure 2.8.

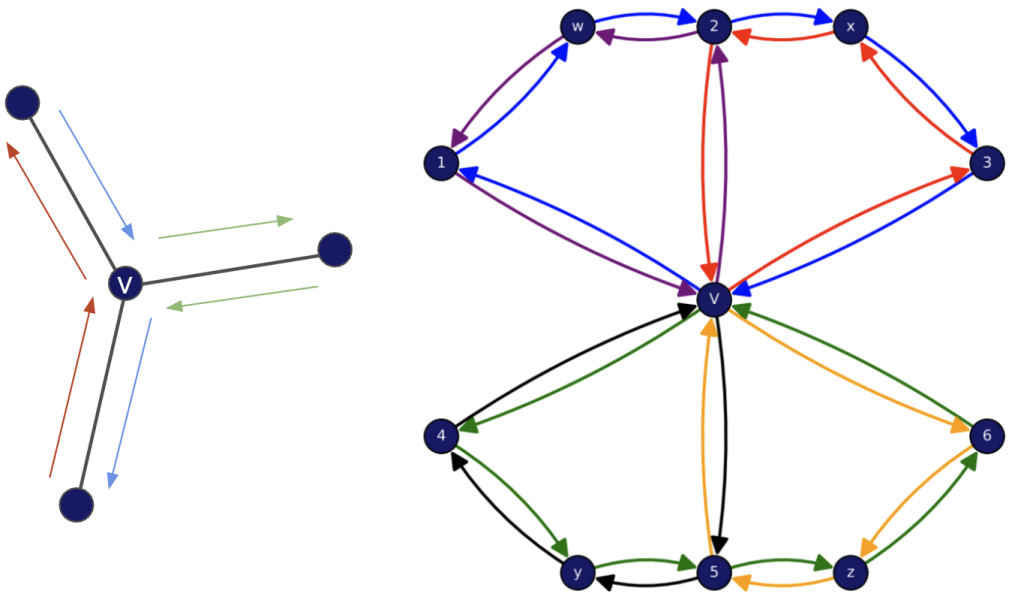


Figure 2.8: Vertex v on the left has degree $3 > 2$ and the blue sequence $a \to v \to c$ mirrors the red sequence $c \to v \to a$ which forces the green sequence to backtrack. Vertex v on the right has degree $6 > 5$ and the local rotation π_v is not cyclic because it involves a permutation with two cycles: 1, 2, 3 and 4, 5, 6.

Chapter 3

ALGORITHM

3.1 Motivation

Rotation systems as defined in Section 2.5 and combined with the procedure to convert a rotation system into an embedding from the proof of Theorem 4, yield a straightforward algorithm for determining the minimum genus and corresponding embedding by iterating through all the rotation systems. This is the algorithm adopted by many popular computational software packages such as SageMath [5, 6]. However it is quite inefficient both theoretically and in practice. Theoretically, it is $\mathcal{O}(|V| \prod_{v \in V}(d_v - 1)!)$ which is $\mathcal{O}(2^{|V|})$ for 3-regular graphs and $\mathcal{O}(n(n-1)!^n)$ for the complete graph on n vertices K_n. Practically, it takes days to compute small graphs like K_7 or $K_{5,5}$. K_8 and above is completely out of the picture.

The problem is that embedding a graph into an arbitrary surface is NP-hard [15] and thus very unlikely to have an efficient polynomial time solution. Thus the natural approach is to develop specialized algorithms that are efficient for particular graph families. A promising constraint on the genus is given by the girth in Proposition 2.

Proposition 2. *A genus g graph satisfies $|E| \leq (|V| - g)t/(t-2)$ where t is the girth.*

Proof. Observe that a connected graph has face lengths bounded below by the girth t. By Proposition 1, $t|F| \leq 2|E|$. Combine with Euler's formula for the result. □

This not only suggests that an algorithm could take advantage of the girth property of a graph. It also suggests that the entire distribution of cycle lengths might be useful. In fact, this is useful and is what the PAGE algorithm [12] explained in the next section takes advantage of to compute previously intractable high girth graphs.

3.2 Walkthrough

Euler's formula shows that finding the minimum genus means finding the largest collection of faces. As noted in Remark 5, this means finding the largest collection of cycles that satisfy a rotation system for which a necessary constraint is to exclude backtracking cycles. All such cycles can be enumerated efficiently using an algorithm in [12]. $K_{3,3}$ has many cycles, including 39 simple ones shown in Figure 3.1.

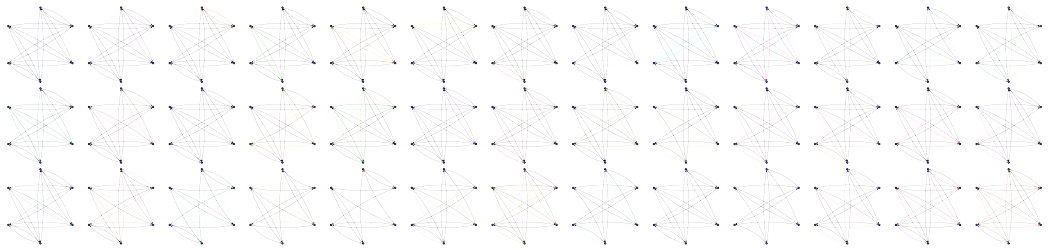


Figure 3.1: The 9 length 2, 18 length 4, and 12 length 6 simple cycles of $K_{3,3}$.

The length 2 ones are backtracking and can be excluded. Even then, there is an astronomical number of cycle combinations to try. Luckily, since $\sum_{f \in F} \texttt{length}(f) = 2|E|$, shorter cycles enable fitting more cycles into F. Starting with length 4 cycles, there are 36 non-backtracking ones as shown in Figure 3.2.

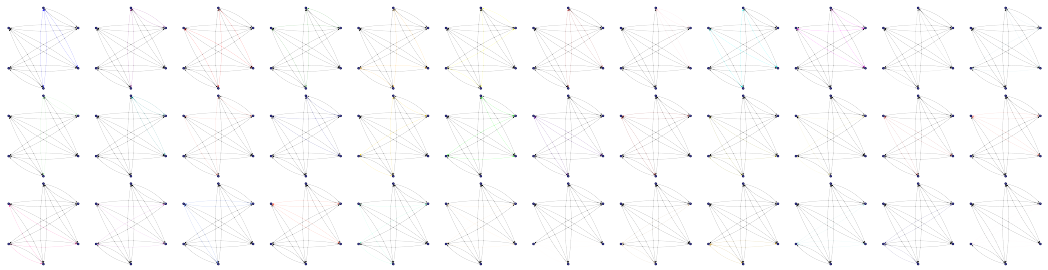


Figure 3.2: The 36 length 4 non-backtracking cycles of $K_{3,3}$.

Moreover, Theorem 4 shows that each directed edge must be used exactly once. Thus one can be choosen arbitrarily and then determines that exactly one of the cycles that include it must be in the set of faces F. Choosing, say edge $6 \to 2$, leaves just 8 cycles in Figure 3.3. A cycle can be chosen arbitrarily, and the algorithm

can continue choosing more cycles in this manner until either a complete set of faces with $2|E|$ edges is achieved or a rotation system becomes impossible since the added cycle introduced multiple cycles in a local rotation (see Figure 2.8). If at any point a rotation system becomes impossible, the algorithm simply backtracks to one of the arbitrary cycle choices and chooses another one. The small branching factor, in this case 8, makes it tractable.

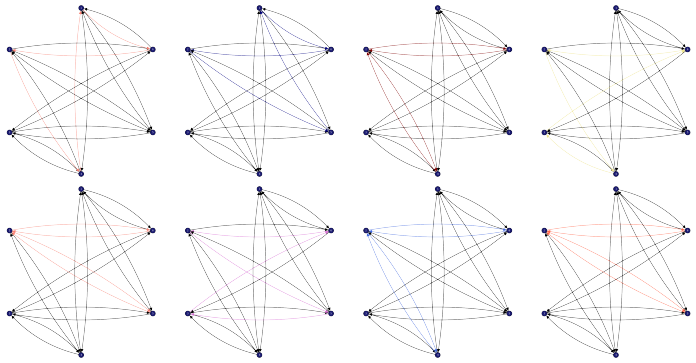
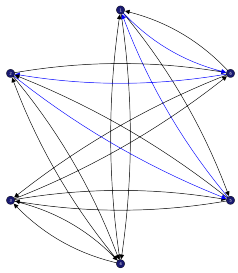
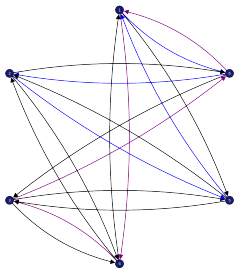


Figure 3.3: The 8 length 4 non-backtracking cycles of $K_{3,3}$ with $6 \to 2$.

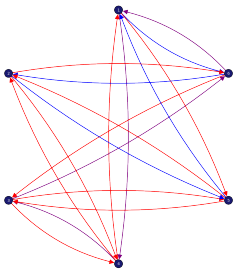
To complete the $K_{3,3}$ example, say the dark blue cycle from Figure 3.3 is chosen. Then Figure 3.4 concludes the computation and since there are 3 cycles in the set of faces, by Euler's formula, the genus is correctly determined as 1.



(a) This cycle has an edge $1 \to 6$ so another cycle in F must have $6 \to 1$.



(b) A cycle containing $6 \to 1$ is the purple cycle. It has an edge $4 \to 3$.



(c) All directed edges are used exactly once so the algorithm terminates.

Figure 3.4: PAGE applied to $K_{3,3}$.

3.3 Results

An important high girth graph family is the cage graphs. An (r, t)-cage is the smallest (fewest vertex) graph of degree r with girth t. The full PAGE algorithm can be found in pseudo-code in [12] and has been implemented in C here. Experimental results are compared against the SageMath algorithm [5, 6] and one of the fastest general purpose algorithms, multi_genus [3] using a single CPU core on an M1 Macbook Pro for fair comparison. It is worth noting that one of the advantageous properties of PAGE is that it is easily parallelized [12]. This is not reflected in Table 3.1. Even then, multi_genus struggles to keep up around $t \geq 9$ and PAGE is the only tractable algorithm for $t \geq 12$. Accordingly, PAGE yields a new result [12]:

Theorem 6. *The genus of the* $(3, 12)$*-cage is* 17*.*

Unfortunately, the structure of the cage graphs is not known for $t > 12$. Likely, PAGE will be useful in determining their genus once these larger cages are discovered.

r	t	$\|V\|$	$\|E\|$	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 3.1: Practical runtime comparison for $(3, t)$-cages in seconds

In addition to performing well on high girth graphs in practice, PAGE also has good theoretical runtime on high girth graphs as seen in Theorem 7 and proved in [12].

Theorem 7. PAGE *is* $\mathcal{O}(n(4^m/n)^{n/t})$ *for graphs of girth* t*,* n *vertices, and* m *edges.*

Another useful property of PAGE is that it outputs not only the minimal genus but also the corresponding rotation system. This allows for easy verification of the results both programatically and visually as seen in Figure 3.5 and Theorem 8.

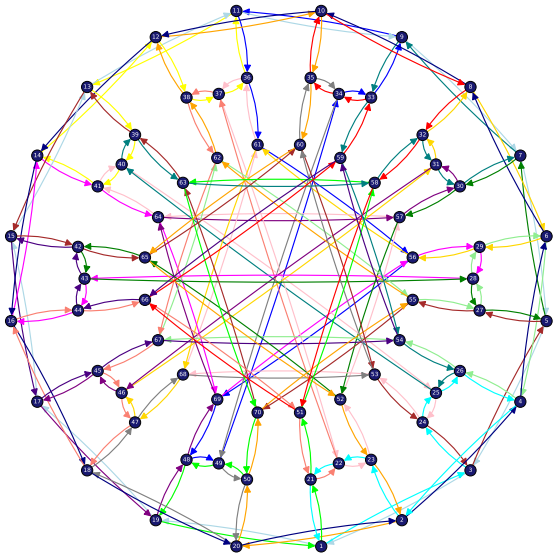


Figure 3.5: $(3, 10)$ cage with minimal genus 9 and the faces color coded.

Theorem 8. PAGE *takes any connected graph G, calculates its genus g, and produces the rotation system for an embedding of G on a minimal genus surface S_g.*

Proof. The algorithm finds the facial cycles to deduce the minimum genus. These faces can be glued together as specified in Theorem 4 to construct S_g which yields the embedding and consequently the rotation system by Section 2.5. □

A final useful property of PAGE proved in [12] is that it outputs progressively narrowing bounds following Theorem 9. This is a natural consequence of the backtracking nature. By slightly relaxing the order that the cycles are explored to no longer be entirely in increasing order of genus, the algorithm will find more collections of cycles that correspond to faces in an embedding at the cost of not being able to stop at the first one. The algorithm can then keep searching to progressively lower

the upper bound on the genus. By choosing the order appropriately, the lower bound on the genus can also be made to increase at a reasonable rate. Consult [12] for a rigorous explanation.

Theorem 9. *For any connected graph with m edges, n vertices, and variable genus g,* PAGE *computes two monotone sequences g_k and G_k satisfying*

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

that converge to g in a finite number of steps and time.

Chapter 4

CONCLUSION

Minimal-genus graph embedding has many important applications. It is relevant in infrastructure optimization where having fewer bridges (torus handles) is less expensive, easier to navigate, and reduces maintenance burdens. It is also important in circuit design and Very Large Scale Integration (VLSI) where components can be modeled as vertices, wires as edges, and edge crossings are undesirable because they can cause short-circuits or interference. Yet, despite its importance to practical applications, there are no known efficient algorithms to embed graphs into arbitrary surfaces. In fact this problem is NP-hard and many popular mathematical software packages such as Mathematica, completely forego a standard implementation of a genus calculation algorithm.

This thesis started by rigorously defining the problem with topological graph theory, and then introduced important work by Heffter, Edmonds, and Youngs to express the problem of graph embedding purely combinatorially. Combinatoric representations are often a crucial step in general when trying to approach an optimization problem algorithmically. Building on this work, the thesis then studied an adaptation of rotation systems, Practical Algorithm for Graph Embedding (PAGE), that takes advantage of the cycle length distribution of a graph to work particularly well for graphs of high girth, or low degree, since these reduce the number of cycles to explore. This yields a practical algorithm for graph families such as the 3-regular graphs and cage graphs. In particular, PAGE determined the previously intractable genus of the (3, 12)-cage as 17.

BIBLIOGRAPHY

[1] Dan Archdeacon. Topological graph theory - a survey. 1996. URL: http://www.math.u-szeged.hu/~hajnal/courses/PhD_Specialis/Archdeacon.pdf.

[2] H.R. Brahana. Systems of circuits on two-dimensional manifolds. *Annals of Mathematics*, 23(2):144–168, 1921. doi:10.2307/1968030.

[3] Gunnar Brinkmann. A practical algorithm for the computation of the genus. *Ars Mathematica Contemporanea*, 22, July 2022. doi:10.26493/1855-3974.2320.c2d.

[4] Andries E. Brouwer. Cages. Collection of adjacency lists by a professor at Techn. Univ. Eindhoven. URL: https://www.win.tue.nl/~aeb/graphs/cages/cages.html.

[5] The Sage Developers. Genus computation module. Accessed: 2025-05-12. URL: https://github.com/sagemath/sage/blob/develop/src/sage/graphs/genus.pyx.

[6] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 10.3)*, 2025. URL: https://www.sagemath.org.

[7] Jonathan L. Gross and Thomas W. Tucker. *Topological Graph Theory*. Dover books on mathematics. Dover Publications, 2012. ISBN: 9780486417417. URL: https://books.google.com/books?id=6HmA_x0dL9oC.

[8] Nora Hartsfield and Gerhard Ringel. *Pearls in Graph Theory: A Comprehensive Introduction*. Dover Books on Mathematics. Dover Publications, 2013. ISBN: 9780486315522. URL: https://store.doverpublications.com/products/9780486315522.

[9] Casimir Kuratowski. Sur le problème des courbes gauches en topologie. *Fundamenta Mathematicae*, 15(1):271–283, 1930. URL: https://eudml.org/doc/212352.

[10] Simon L'Huilier and Joseph Diez Gergonne. Géométrie. mémoire sur la polyédrométrie; contenant une démonstration directe du théorème d'euler sur

les polyèdres, et un examen des diverses exceptions auxquelles ce théorème est assujetti. *Annales de mathématiques pures et appliquées*, 3:169–189, 1812-1813. URL: http://www.numdam.org/item/AMPA_1812-1813__3__169_0/.

[11] Gaku Liu. Math 461 and 462: Combinatorial theory i and ii, Winter and Spring Quarters 2024-2025. Course content is only available to students.

[12] Alexander Metzger and Austin Ulrigg. An efficient genus algorithm based on graph rotations, 2025. arXiv:2411.07347.

[13] Bojan Mohar and Carsten Thomassen. *Graphs on Surfaces*. Johns Hopkins University Press, July 2001. ISBN: 9780801866890. URL: https://www.sfu.ca/~mohar/Book.html.

[14] Sophia N. Potoczak. *Survey of Graph Embeddings into Compact Surfaces*. PhD thesis, University of Maine, 2014. URL: https://digitalcommons.library.umaine.edu/cgi/viewcontent.cgi?article=3192&context=etd.

[15] Carsten Thomassen. The graph genus problem is np-complete. *Journal of Algorithms*, 10(4):568–576, 1989. doi:10.1016/0196-6774(89)90006-0.

[16] Alex Waldrop. Rotation systems and cellular imbeddings. University of Washington REU 2011. URL: https://sites.math.washington.edu/~reu/papers/2011/alex/RotationSystemsImbeddings_AWaldrop.pdf.