A large set of torus obstructions and how they were discovered

Wendy Myrvold [*]
Department of Computer Science
University of Victoria
Victoria, BC, Canada
wendym@cs.uvic.ca

Jennifer Woodcock
Department of Computer Science
University of Victoria
Victoria, BC, Canada
jwoodcoc@uvic.ca

Submitted: Oct 11, 2013; Accepted: Jan 7, 2018; Published: Jan 25, 2018
Mathematics Subject Classifications: 05C83, 05C10, 05C85

Abstract

We outline the progress made so far on the search for the complete set of torus obstructions and also consider practical algorithms for torus embedding and their implementations. We present the set of obstructions that are known to-date and give a brief history of how these graphs were found. We also describe a nice algorithm for embedding graphs on the torus which we used to verify previous results and add to the set of torus obstructions. Although it is still exponential in the order of the graph, the algorithm presented here is relatively simple to describe and implement and fast-in-practice for small graphs. It parallels the popular quadratic planar embedding algorithm of Demoucron, Malgrange, and Pertuiset.

Keywords: Embedding graphs on surfaces, obstructions for surfaces, torus embedding.

1 Introduction

A *(topological) obstruction* for a surface S is a graph G with minimum degree at least three that is not embeddable on S but for every edge e of G, $G-e$ (G with edge e deleted) is embeddable on S. A *minor-order obstruction* G has the additional property that for every edge e of G, $G \cdot e$ (G with edge e contracted) is embeddable on S.

There is a finite set of obstructions for every surface of fixed genus ([5, 2, 29]), yet to date the only complete sets known are for the plane ([23, 30]) and the projective plane ([17, 1]). Characterizing the complete set of torus obstructions is a natural next step in

[*]Supported by NSERC.

topological graph theory. The progress that has been made to date is summarized later in Section 4.

In this paper, we accomplish three things:

1. describe a conceptually simple and fast-in-practice torus embedding algorithm which has been used to push the exhaustive search boundaries and improve upon previous results, and

2. highlight the major research thus far toward defining the complete set of obstructions to the torus,

3. make our database of torus obstructions available to the research community.

2 Notation

We use n to denote the order, or number of vertices, in a graph, and m to denote the size, or number of edges. When speaking of genus and embeddings we refer implicitly to the orientable variety, and we omit the term torus when referring to embeddings and obstructions where it is implied by context. We also limit our consideration to simple, 2-connected graphs unless otherwise implied or specified.

A graph H is *homeomorphic* to a graph G if H can be obtained from G by a series of edge subdivisions. A *Kuratowski subgraph* is a subgraph that is homeomorphic to either K_5 or $K_{3,3}$.

A *bridge* B with respect to a subgraph H of a graph G is either

Type 1: a connected component C of $G-H$ along with the edges (u,v) such that $u \in V(C)$ and $v \in V(H)$ and the vertices $v \in V(H)$ that are endpoints of these edges, or

Type 2: an edge (u,v) and its endpoints where $(u,v) \in E(G)$ and $u \in V(H)$ and $v \in V(H)$ but $(u,v) \notin E(H)$.

The vertices that are in both B and H are called *attachment vertices* of B. The vertices that are in B but not in H (i.e. the vertices in B that are not attachment vertices) are called *internal vertices* of B. A *bisecting path* in a bridge B is a path P that contains only vertices and edges in B, and where $v \in V(P)$ is an attachment vertex of B if and only if v is an endpoint of P, and also the two ends of the path are distict vertices of B (the path cannot be a cycle).

Given a surface S and an embedding $\Pi(G)$ of a graph G on S, a *face boundary* of Π is a closed walk of G that bounds a maximal contiguous region of $S - G$. A face boundary might not be a cycle, as it may have repeated vertices. We call the faces with repeated vertices *challenging* faces. A face f is *admissible* for a bridge B if all of the attachment vertices of B are on f.

3 An Elegant and Simple Torus Embedding Algorithm

There can be a large gap separating theory and practice when it comes to the study of algorithms (see for example [10]). There are many linear time algorithms for embedding graphs on the plane ([18, 6, 11, 32, 33, 7, 8]). For small enough graphs in small enough numbers, it can be worthwhile to sacrifice asymptotic speed for simplicity and elegance, and thus the quadratic algorithms of Klotz [22] and of Demoucron, Malgrange, and Pertuiset [12] still have relevance. For the torus, Chambers, Myrvold and Neufeld designed an exponential embedding algorithm [28, 27, 9] which, although slow in general, was practical for small graphs. Kocay and Myrvold discuss critical design concerns for designing embedding algorithms for the torus and show that several purported polynomial-time algorithms are incorrect [26]. Mohar gave an outline for a linear time torus embedding algorithm [21] (reference to some of his other papers would be required to fill in enough detail for implementation) and, with Juvan, a simplified $O(n^3)$ variant [20]. The complexity of these algorithms and in some cases, large constant factors in the running time formula are of concern in terms of correctly implementing them and it seems likely that for small graphs, a simpler yet asymptotically slower algorithm might be more appropriate in practice.

Our new Torus Embedding Algorithm is conceptually simpler and faster in practice than that of Myrvold and Neufeld [28, 27]. It is a natural extension to the torus of the quadratic planar embedding algorithm of Demoucron, Malgrange, and Pertuiset [12], hereafter referred to as the *DMP Algorithm*, which we introduce first.

3.1 Quadratic Planar Embedding: the DMP Algorithm

The quadratic ($O(n^2)$) algorithm of Demoucron, Malgrange, and Pertuiset [12], while not the fastest planar embedding algorithm, is elegant, simple, and easy to implement which allows it to retain its viability as an option for planar embedding tasks. The proof of correctness is not trivial and sometimes the details are glossed over or, in the case of [16], are not correct as explained in [26] where a counterexample is given.

The algorithm's foundation relies on the facts that a graph with no cycles is obviously planar and that a cycle has exactly one embedding (up to isomorphism) on the plane. It first finds an embedding $\Pi(C)$ for some cycle C in G and proceeds methodically to add paths from $G - C$ into $\Pi(C)$. The DMP algorithm is given in pseudocode in Algorithms 3.1 and 3.2.

Algorithm 3.1 StartDMP(graph G)

```
1: if $G$ does not contain any cycles then
2:     halt $G$ is trivially planar.
3: end if
4: Choose a cycle $C$ in $G$.
5: Let $\Pi(C)$ be an embedding of $C$ on the plane.
6: DMP($G$, $C$, $\Pi(C)$) //Pseudocode in Algorithm 3.2
```

Algorithm 3.2 DMP(graph G, graph H, embedding $\Pi(H)$)

```
1: if there are no bridges remaining then
2:     halt $\Pi(H)$ is an embedding of $G$.
3: else if there is a bridge with no admissible faces then
4:     halt $G$ is not planar.
5: end if
6: Choose a bridge $B$ of $G$ with a minimum number of admissible faces.
7: Choose a bisecting path $P$ of $B$.
8: Embed $P$ in $\Pi(H)$ and set $H = H \cup P$.
9: DMP($G$, $H$, $\Pi(H)$)
```

A bridge B is *admissable* for a face f if all of its points of attachments lie on f. Demoucron, Malgrange, and Pertuiset proved that for a planar graph, as long as bridges with only one admissible face are chosen first, any face can be chosen in which to embed each bridge, and this algorithm will yield a planar embedding if and only if the input graph is planar ([12], this is also presented in English on pp. 266-267 in this text book [25]).

For simplicity, Algorithm 3.2 does not discuss computing the faces and bridges. It is easy to update both of these in $O(n)$ time each time $\Pi(H)$ is modified - B is no longer a bridge, new bridges arise but only originating from the removal of P from B, the face f that P is embedded into is split into two faces f_1 and f_2 and for the new bridges and the old bridges admissable for f, it is necessary to determine if they are admissable for f_1 and/or f_2.

3.2 Exponential Torus Embedding: DMP-Style

The data structure used to store embeddings in the computer is a rotation system: for each vertex, the neighbours are stored in cyclic order according to their clockwise order in the embedding. There are standard algorithms for walking the faces, and determining the genus from a rotation system. The embeddings represented are 2-cell embeddings.

Our new torus embedding algorithm, initially described in [34], follows a similar structure to the DMP algorithm just described with two caveats which we will explore in the next sections.

1. We assume that our input graph is not planar and begin by choosing a Kuratowski subgraph.
2. After choosing a bridge we must try all possible placements for our chosen bisecting path.

3.3 Finding a Kuratowski Subgraph

The goal is to find a 2-cell embedding of the graph on the torus. By Euler's formula, any simple toroidal graph has at most $3n$ edges. So any graph with more than $3n$ edges can be rejected as non-toroidal. We assume that our input graph is not planar (otherwise, a planarity tester can be used to embed it).

The Demoucron et al. [12] approach on the plane starts with an embedding of a cycle. But embedding a cycle on the torus does not give faces that are homeomorphic to planar disks. Therefore, the algorithm starts by choosing a Kuratowski subgraph, K, which can be found in $O(n^3)$ time (assuming a $O(n^2)$ planarity testing algorithm and taking into account that the graph has at most $3n$ edges) by removing each edge in turn and replacing it only if the resulting graph is planar. More efficient algorithms are given in [22, 33, 8].

Once we have found K, our algorithm must consider all labelled embeddings of K. This step is hidden in the DMP algorithm because a cycle has only one embedding on the plane up to isomorphism.

The *flip* of an embedding is obtained by reversing the sense of clockwise of the neighbours of each vertex. For an embedding and its flip, either both extend to an embedding for the whole graph or neither one does. Hence an embedding and its flip are considered to be equivalent to each other.

Up to isomorphism, there are six different unlabelled embeddings of K_5 and two different unlabelled embeddings of $K_{3,3}$ as pictured in Figure 1. The number of unique labelled embeddings is 231 for K_5 and 20 for $K_{3,3}$.

An easy approach to computing the different labelled embeddings for K (equal to K_5 or $K_{3,3}$) on the computer is to generate all the different rotation systems for K, and then to use a face walking algorithm to count the faces and compute the genus in order to choose the ones that are torus embeddings. When generating the different rotation systems, the only cases considered are those such that the neighbour list for every vertex lists the smallest numbered neighbour first (since the neighbour list represents a cyclic order for the neighbours). To avoid creating both an embedding and its flip, the order of the neighbours of the first vertex also has the additional property that the vertex number for the second neighbour is smaller than that of its last neighbour.

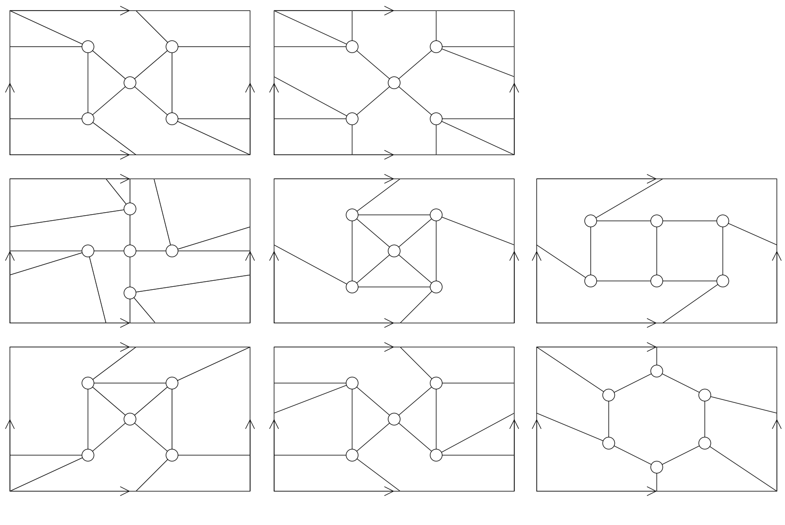


Figure 1: Unlabelled embeddings of K_5 and $K_{3,3}$ on the torus.

Note that all of these embeddings divide the torus into faces that are homeomorphic to a planar disk. Unfortunately, it is not possible to proceed as with Demoucron's algorithm and find out whether the graph is embeddable on the torus in $O(n^3)$ time. There are two reasons why this does not work on the torus, which we discuss next.

3.4 Where to Embed a Bisecting Path

Recall that in the DMP algorithm, if there are bridges with only one admissible face, we embed a bisecting path from these first. Otherwise, we arbitrarily choose a bridge and one of its admissible faces in which to embed a bisecting path. For embedding a graph on the plane, we do not need to try all of the admissible faces because, as Demoucron, Malgrange, and Pertuiset proved, if a graph is planar every choice of admissible face for a bridge leads to an embedding ([12], p. 266 of [31]). Their theorem does not extend to the torus and thus we must try to embed a bisecting path in each admissible face in turn.

Further, some embeddings of K_5 and $K_{3,3}$ have challenging faces with repeated vertices. Four of the unlabelled K_5 embeddings and one of the unlabelled $K_{3,3}$ embeddings yield labelled embeddings with challenging faces. Figure 2 illustrates the $K_{3,3}$ case which has a ten-face which has four repeated vertices (1, 3, 4, and 6) and two repeated edges ($(1, 6)$ and $(3, 4)$).

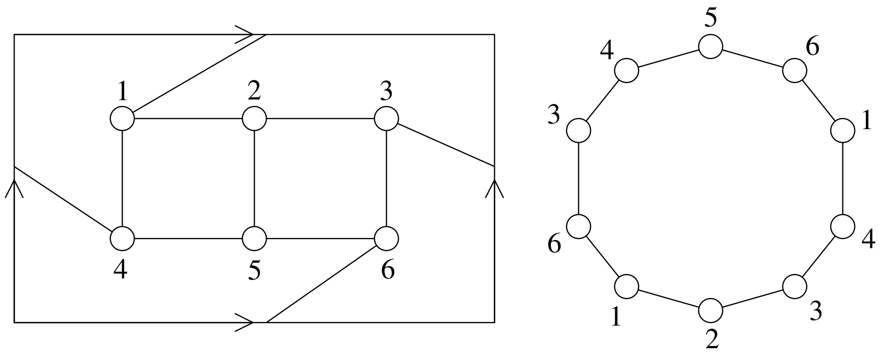


Figure 2: Challenging face of one of the $K_{3,3}$ embeddings.

Although K_5 and $K_{3,3}$ have a maximum of four repeated vertices on a challenging face, we may need to consider more repeated vertices than this due to our initially chosen subgraph being homeomorphic and not necessarily equal to K_5 or $K_{3,3}$. It is clear, however, that a vertex cannot appear more than twice on a challenging face. This is because no embedding of K_5 or $K_{3,3}$ on the torus has the same edge appearing more than twice on a face. It is also obvious that after embedding a path across a face, the number of times a vertex repeats on a face cannot increase. Thus there can be at most four possible ways to embed a bisecting path across a face.

3.5 Bridge Penalties

The caveats described in Section 3.4 lend our algorithm nicely to the use of recursion to explore all of the faces and all placements within each face of the bisecting path from our chosen bridge. To make a sensible choice of bridge and bisecting path and thus minimize the number of recursive calls, we define a penalty $P(B)$ for each bridge B as follows. For each admissible face f for B:

- let x_i be the number of times attachment vertex i of B appears on f, and
- choose two different attachment vertices u_f and v_f of B such that $x_{u_f} \cdot x_{v_f}$ is minimized.

Now,

$$P(B) = \sum_{f \textit{ is admissible for } B} x_{u_f} \cdot x_{v_f}.$$

If there is a bridge B with $P(B) = 0$, our algorithm must backtrack as there is a bridge that has no admissible faces. Otherwise, it chooses a bridge B with minimum penalty and, for each admissible face f for B, a bisecting path between u_f and v_f.

3.6 Pseudocode

Our torus embedding algorithm, is given in pseudocode in Algorithms 3.3 and 3.4. It proceeds by choosing a Kuratowski subgraph and, for each embedding of that subgraph, taking a backtracking approach to embedding bisecting paths from the bridges in all possible ways, and using the penalty just described to choose which bridge to embed first. As with the DMP Algorithm, for simplicity, we make no mention of finding or updating the faces and bridges at each step, or of calculating the penalty for the bridges. Again, these can be found initially in $O(n^2)$ time and maintained in $O(n)$ time per recursive call.

Algorithm 3.3 StartTorusEmbed(graph G)

```
1: if $G$ is planar then
2:     halt a planar embedding of $G$ is also a torus embedding of $G$.
3: else
4:     Choose a subgraph $H$ of $G$ that is homeomorphic to either $K_5$ or $K_{3,3}$.
5:     for all non-isomorphic, nonequivalent labelled embeddings $\tau(H)$, of $H$ do
6:         TorusEmbed($G$, $H$, $\tau(H)$) //Pseudocode given in Algorithm 3.4
7:     end for
8: end if
```

3.7 Improving the Running Time

Initial timing studies of our algorithm led us to discover graphs which caused our algorithm to be very slow. Analysis of the structure of such graphs revealed that the slowness was a result of having multiple ways to embed some bridges when the graph had one or more 2-vertex cuts. It is possible, however, because of Theorem 3.1 below (which generalizes Lemma 2.4 of [13]) to preprocess these graphs to avoid the slow running time when they are given as input to our algorithm.

Algorithm 3.4 TorusEmbed(graph G, graph G', embedding $\tau(G')$)

```
1: if there are no bridges remaining then
2:     halt  $\tau(G')$ is an embedding of $G$.
3: end if
4: if there is a bridge $B$ with $P(B) = 0$ then
5:     return  $\tau(G')$ cannot lead to an embedding of $G$ (backtrack).
6: end if
7: Choose a bridge $B$ with minimum $P(B)$.
8: for all admissible faces, $f$, for $B$ do
9:     Choose a bisecting path $P$ from $B$ with endpoints $u_f$ and $v_f$ (see Section 3.5).
10:    Let $U$ and $V$ be the sets of all copies of $u_f$ and $v_f$ on f, respectively.
11:    for all pairs $(u, v)$ in $U \times V$ do
12:        Set $\tau(G') = \tau(G' \cup P)$ with $P$ embedded in $\tau(G')$ using endpoints $u$ and $v$.
13:        Set $G' = G' \cup P$.
14:        TorusEmbed($G$, $G'$, $\tau(G')$)
15:        Remove $P$ from $\tau(G')$ and from $G'$.
16:    end for
17: end for
```

Theorem 3.1. Let B be a bridge of a graph G with respect to the subgraph consisting of the two vertices a and b (and no edges) of a 2-vertex cut $\{a, b\}$ in G. If $B + (a, b)$ is planar, then G embeds on S if and only if

$$G - \{v \mid v \textit{ is an internal vertex of } B\} + (a, b)$$

embeds on S.

Proof. Given an embedding of

$$G - \{v \mid v \textit{ is an internal vertex of } B\} + (a, b),$$

we can replace (a, b) with a planar embedding of B or, if (a, b) is an edge in G, replace (a, b) with a planar embedding of $B + (a, b)$. □

We created a preprocessor to reduce to a single edge (a, b) any bridge B with respect to some 2-vertex cut $\{a, b\}$ such that $B + (a, b)$ is planar. This significantly reduced the running time on input graphs containing such bridges.

Another improvement can come from the fact that $K_{3,3}$ has fewer labelled embeddings on the torus than K_5 does by a factor of almost twelve. If our algorithm finds a subgraph homeomorphic to K_5, then either there is a subgraph homeomorphic to $K_{3,3}$ and it can be found in linear time [14, 3], or if the graph does not contain a $K_{3,3}$ subgraph, the algorithm of Gagarin and Kocay [14] could be used to test for toroidality in linear time.

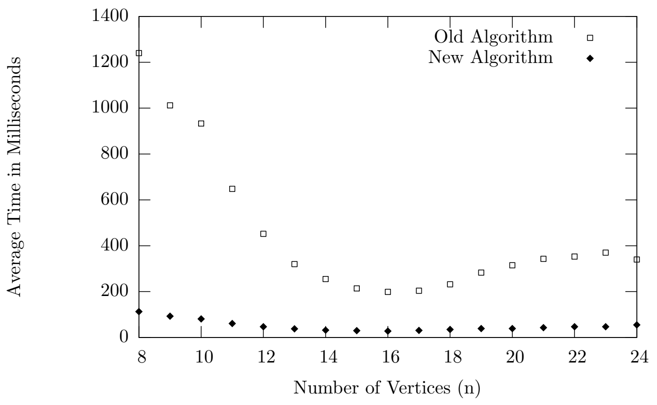


Figure 3: Plots of average running time for checking obstructions.

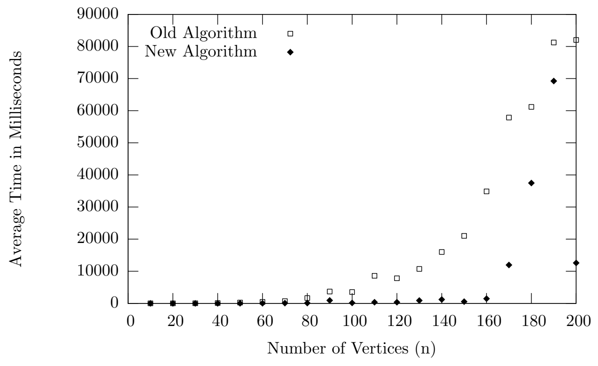
3.8 Results

Timing comparisons of an implementation in C of our DMP-style torus embedding algorithm as compared to the exponential algorithm of Myrvold and Neufeld ([28]) show significant improvements on all tested inputs. The first chart in Figure 3 shows the comparison of the time it took to confirm that the 239,451 torus obstructions found by Chamber [9] are in fact obstructions using the two algorithms and to determine if they are minor order obstructions or not. This involves checking if the original graph is toroidal, and then removing and contracting each edge in turn to check if the resulting graph is toroidal.

We also generated random toroidal graphs by randomly choosing an embedding of K_5 or $K_{3,3}$ on the torus and then randomly subdividing and/or adding edges to the embedding until the desired order is reached (See [34] for more details). The times for these are shown in Figure 4. To generate random non-toroidal graphs, we used the same process, and then added random edges until the resulting graph was not toroidal. Figure 5 shows the results of timing the two algorithms on these randomly generated non-toroidal graphs. The last chart shows quite impressive timing improvements for large graphs.

4 The Known Torus Obstructions

It should be noted that some of the work on finding obstructions [9, 19, 20, 27, 34] has been presented in theses or preprints and as a result, was not subject to the usual refereeing standards of journals. Myrvold and Neufeld found all torus obstructions on up to ten vertices by exhaustive search. They also found some larger ones bringing the total number of torus obstructions to 3884 topological obstructions, 2249 of which are also minor-order obstructions [28, 27]. Almost ten years later, in 2002, Chambers found all torus obstructions on up to eleven vertices and all (206) 3-regular torus obstructions on up to 24 vertices by exhaustive search [9] using an improved version of the same program as Myrvold and Neufeld. Chambers also used a “split-delete” approach and brought the



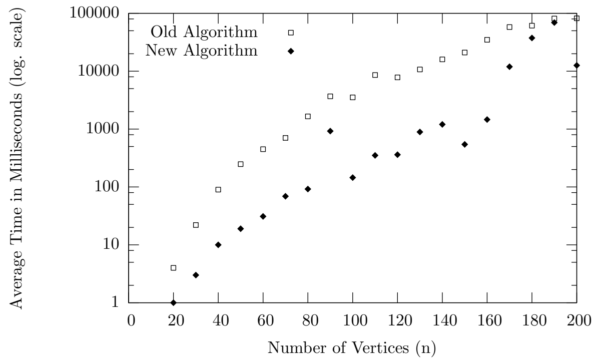


Figure 4: Plots of average running time for random toroidal graphs.

total to 239,451 obstructions, 16,682 of which are minor-order.

In 2006, an implementation of the algorithm described in this paper was used by Woodcock to confirm the previous exhaustive search results [34]. Shortly after this, we used it in an effort to confirm Chambers' "split-delete" results [9], and discovered that many obstructions had been missed due to an oversight when examining the output. Chambers was running his software on many different machines without automated software to collate results together and to ensure that all cases completed correctly. Further, we used it to augment the exhaustive search from 11-vertex graphs to 12-vertex graphs and from 24-vertex 3-regular graphs to 26-vertex 3-regular graphs (none of the latter are obstructions).

At the moment, our database contains 250,815 torus obstructions, 17,523 of which are minor-order. See Tables 1 and 2, respectively, for a breakdown of the obstructions by order and size. An exhaustive search was possible for $n \leqslant 12$ and hence the counts in the tables include all of these small obstructions. The counts in these cases were verified computationally by at least two different computer implementations of obstruction checking algorithms. The larger ones were obtained either by taking the smaller ones and applying split/delete as per Chambers [9], by non-exhaustive search of the larger graphs, or by random search in triangulations of the 2-handled torus.

Juvan described the 270 projective planar torus obstructions [19]. These are all obtainable from a 4 by 4 projective planar grid using $\Delta - Y$ and $Y - \Delta$ transformations.

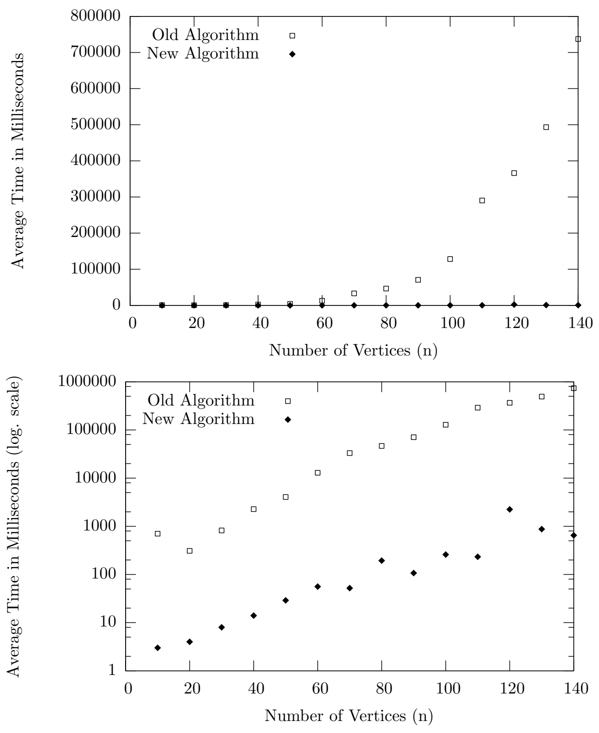


Figure 5: Plots of average running time for random non-toroidal graphs.

These 270 obstructions correspond to five torus obstructions that are split-delete minimal (one with eleven vertices, one with 12 vertices and three with 13 vertices).

Chambers, Gagarin and Myrvold characterized the torus obstructions which do not contain a subgraph homeomorphic to $K_{3,3}$ [15] (there are eleven) and gave a proof that this set is complete. We know that there are exactly three disconnected obstructions and exactly ten obstructions which have a cut-vertex [4] (see Figures 6 and 7).

We conjecture that our database contains the complete set of 3-regular obstructions (based on the evidence that exhaustive search was done up to 26 vertices and no 26-vertex obstructions were found). The obstructions we have with a 2-vertex cut match the theoretical characterization of the complete set of these given by Mohar and Skoda [24].

Our suspicion is that the set of obstructions that we have is not complete. The reason for this suspicion is because it was not difficult to find some new obstructions using a randomized search.

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. All of the 250,815 obstructions than we have correspond to a set of 402 split-delete minimal obstructions. Table 3 gives a breakdown of the split-delete minimal obstructions by order and size. Up to $n = 12$, the table includes all split-delete minimal obstructions. One strategy that seems plausible for getting a complete set of torus obstructions is to characterize the obstructions that are split-delete minimal. Since the total number of these is probably going to be reasonably small (it appears as though the maximum number for a value for n occurs when $n = 10$ and that for each subsequent value of $n > 10$ the number is decreasing). A characterization done by hand might be feasible.

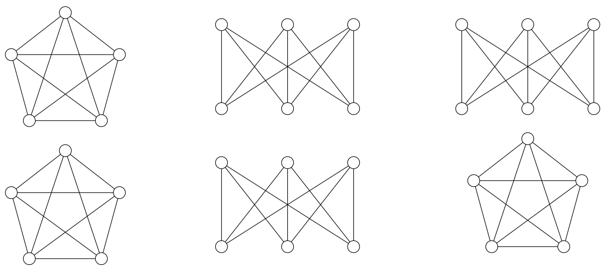


Figure 6: 3 Disconnected Obstructions.

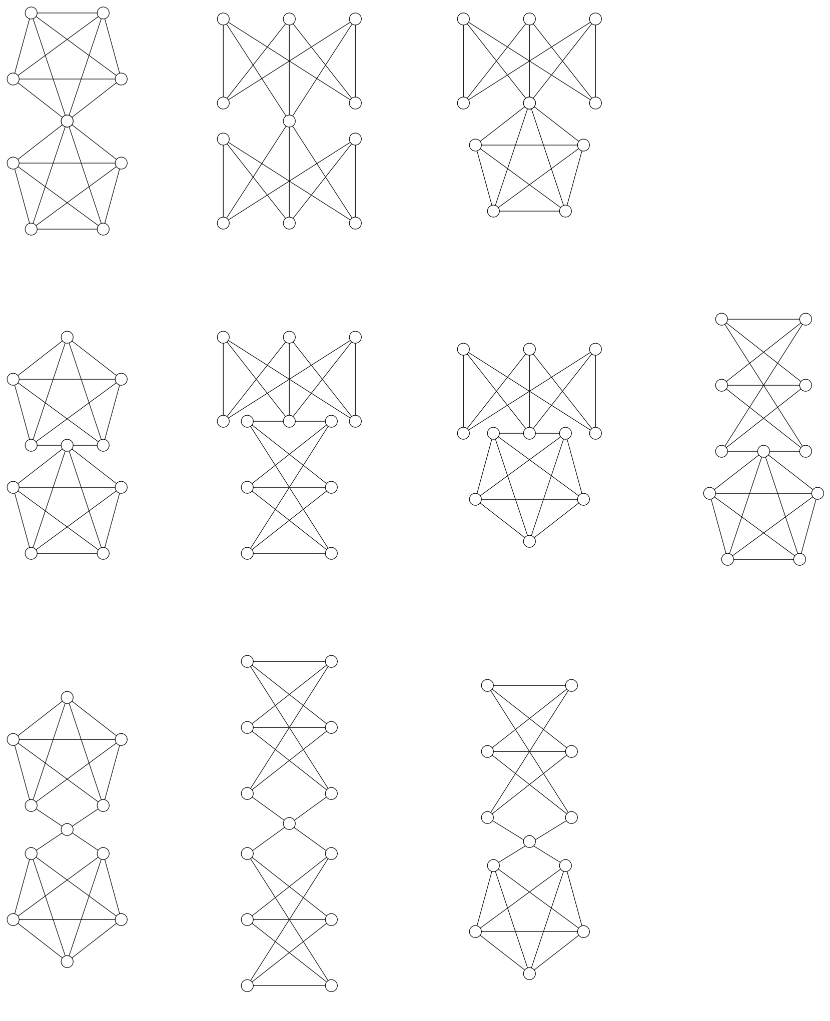


Figure 7: 10 Obstructions with a cut-vertex.

Edges	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	total
Vertices																				
8	-	-	-	-	1	-	1	1	-	-	-	-	-	-	-	-	-	-	-	3
9	-	2	5	2	9	17	6	2	5	-	-	-	-	-	-	-	-	-	-	48
10	-	15	9	35	40	190	170	102	76	21	1	-	1	-	-	-	-	-	-	660
11	5	2	49	87	270	892	1878	1092	501	125	22	4	1	-	-	-	-	-	-	4928
12	1	12	6	201	808	2698	6690	6387	1971	499	98	10	3	1	-	-	-	-	-	19385
13	-	-	12	19	820	4967	12781	16727	7287	1548	301	63	7	-	-	-	-	-	-	44532
14	-	-	-	9	38	2476	15219	24355	16397	4053	930	259	32	1	-	-	-	-	-	63769
15	-	-	-	-	-	33	3646	22402	20957	8474	1948	812	197	10	-	-	-	-	-	58479
16	-	-	-	-	-	-	20	2689	17469	10582	3150	1317	777	68	2	-	-	-	-	36074
17	-	-	-	-	-	-	-	-	837	8099	4154	1131	1073	434	11	-	-	-	-	15739
18	-	-	-	-	-	-	-	-	-	133	2332	1472	526	642	156	1	-	-	-	5262
19	-	-	-	-	-	-	-	-	-	-	-	393	435	294	256	48	-	-	-	1426
20	-	-	-	-	-	-	-	-	-	-	-	-	39	100	164	63	17	-	-	383
21	-	-	-	-	-	-	-	-	-	-	-	-	-	-	16	63	12	4	-	95
22	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	22	1	1	26
23	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	4	-	4
24	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	2
total	6	31	81	353	1986	11273	40411	73757	65500	33534	12936	5461	3091	1550	605	177	51	9	3	250815

Table 1: The 250,815 known topological torus obstructions.

Edges	18	19	20	21	22	23	24	25	26	27	28	29	30	31	total
Vertices															
8	-	-	-	-	1	-	1	1	-	-	-	-	-	-	3
9	-	2	5	2	9	13	6	2	4	-	-	-	-	-	43
10	-	15	3	18	31	117	90	92	72	17	1	-	1	-	457
11	5	2	-	46	131	569	998	745	287	45	8	3	1	-	2840
12	1	-	-	52	238	1218	2519	1841	505	91	23	2	1	1	6492
13	-	-	-	5	98	836	1985	1924	512	84	46	2	1	-	5493
14	-	-	-	-	9	68	463	943	251	43	86	1	-	-	1864
15	-	-	-	-	-	-	21	118	45	12	85	-	-	-	281
16	-	-	-	-	-	-	-	4	3	5	41	-	-	-	53
17	-	-	-	-	-	-	-	-	-	-	8	-	-	-	8
18	-	-	-	-	-	-	-	-	-	-	1	-	-	-	1
total	6	19	8	123	517	2821	6083	5670	1679	297	299	8	4	1	17535

Table 2: The 17,535 known minor order torus obstructions.

Edges	20	21	22	23	24	25	26	27	28	29	30	31	total
Vertices													
8	-	-	1	-	1	1	-	-	-	-	-	-	3
9	2	-	2	3	3	1	4	-	-	-	-	-	15
10	-	1	-	1	16	40	54	17	1	-	1	-	131
11	-	-	-	-	4	44	36	15	7	3	1	-	110
12	-	-	-	-	2	31	41	17	3	1	1	1	97
13	-	-	-	-	2	6	24	5	3	-	-	-	40
14	-	-	-	-	-	2	4	-	-	-	-	-	6
total	2	1	3	4	28	125	163	54	14	4	3	1	402

Table 3: The 402 known split-delete minimal torus obstructions.

5 Conclusions and Future Work

As evidenced by the number of authors mentioned in Section 4 as well as the span of years over which the papers and theses have been published, this is a large research problem, which is being explored in a piecemeal fashion. We are hopeful that it will soon be completed so that new chapters in the field of graph embedding can be explored.

To this end, we are investigating the completeness of obstruction sets for several subclasses of graphs. Having such a large database of obstructions proves incredibly helpful in finding and proving theorems about the structure of these graphs. Eventually, we hope that these theorems will provide a comprehensive categorization of the complete set of torus obstructions.

From an algorithmic perspective, one approach would be to implement the $O(n)$ torus embedding algorithm of Juvan, Marincek and Mohar [21] or Juvan and Mohar's simplified $O(n^3)$ variant [20]. Alternatively, perhaps some of their techniques could be applied to make the DMP-style torus embedding algorithm more efficient. For small graphs such as the obstructions for the torus, however, we believe it is likely that the DMP-style algorithm presented here will be faster than one promising a better Big-Oh time complexity. Regardless, our algorithm has value because of its simplicity and readability.

6 The Torus Obstructions

The torus obstruction described in the paper are available from:

http://www.combinatorics.org/ojs/index.php/eljc/article/view/v25i1p16/html

and

http://webhome.cs.uvic.ca/~wendym/torus/torus_obstructions.html

References

[1] Dan Archdeacon. A Kuratowski theorem for the projective plane. *J. Graph Theory*, 5(3):243–246, 1981.

[2] Dan Archdeacon and Phil Huneke. A Kuratowski theorem for nonorientable surfaces. *J. Combin. Theory Ser. B*, 46(2):173–231, 1989.

[3] Takao Asano. An approach to the subgraph homeomorphism problem. *Theoret. Comput. Sci.*, 38(2-3):249–267, 1985.

[4] Joseph Battle, Frank Harary, Yukihiro Kodama, and J. W. T. Youngs. Additivity of the genus of a graph. *Bull. Amer. Math. Soc.*, 68:565–568, 1962.

[5] R. Bodendiek and K. Wagner. Solution to König's graph embedding problem. *Math. Nachr.*, 140:251–272, 1989.

[6] Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *J. Comput. System Sci.*, 13(3):335–379, 1976. Working Papers presented at the ACM-SIGACT Symposium on the Theory of Computing (Albuquerque, N. M., 1975).

[7] John Boyer and Wendy Myrvold. Stop minding your P's and Q's: a simplified $O(n)$ planar embedding algorithm. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (Baltimore, MD, 1999)*, pages 140–146. ACM, New York, 1999.

[8] John M. Boyer and Wendy J. Myrvold. On the cutting edge: simplified $O(n)$ planarity by edge addition. *J. Graph Algorithms Appl.*, 8(3):241–273, 2004.

[9] J. Chambers. Hunting for torus obstructions. Master's thesis, Department of Computer Science, University of Victoria, Victoria, B.C., 2002.

[10] M. Chimani and K. Klein. Algorithm engineering: Concepts and practice. In T. Bartz-Beielstein, M. Chiarandini, L. Paquete, and M. Preuss, editors, *Experimental Methods for the Analysis of Optimization Algorithms*, pages 131–158. Springer-Verlag, 2010.

[11] H. de Fraysseix and P. Rosentiehl. A depth-first search characterization of planarity. *Ann. Discrete Math.*, 13:75–80, 1982.

[12] G. Demoucron, Y. Malgrange, and R. Pertuiset. Graphes planaires. *Revue Francaise Recherche Operationnelle*, 8:33–47, 1964.

[13] Andrei Gagarin and William Kocay. Embedding graphs containing K_5-subdivisions. *Ars Combin.*, 64:33–49, 2002.

[14] Andrei Gagarin and William Kocay. Embedding graphs containing K_5-subdivisions on the torus. *J. Combin. Math. Combin. Comput.*, 80:207–223, 2012.

[15] Andrei Gagarin, Wendy Myrvold, and John Chambers. The obstructions for toroidal graphs with no $K_{3,3}$'s. *Discrete Math.*, 309(11):3625–3631, 2009.

[16] Alan Gibbons. *Algorithmic graph theory.* Cambridge University Press, Cambridge, 1985.

[17] Henry H. Glover, John P. Huneke, and Chin San Wang. 103 graphs that are irreducible for the projective plane. *J. Combin. Theory Ser. B*, 27(3):332–370, 1979.

[18] John Hopcroft and Robert Tarjan. Efficient planarity testing. *J. Assoc. Comput. Mach.*, 21:549–568, 1974.

[19] M. Juvan. *Algorithms and obstructions for embedding graphs in the torus.* PhD thesis, University of Ljubljana, 1995. (in Slovene).

[20] M. Juvan and B. Mohar. A simplified algorithm for embedding graphs in the torus. 2001 preprint. 10 pages.

[21] Martin Juvan, Jože Marinček, and Bojan Mohar. Embedding graphs in the torus in linear time. In *Integer programming and combinatorial optimization (Copenhagen, 1995)*, volume 920 of *Lecture Notes in Comput. Sci.*, pages 360–363. Springer, Berlin, 1995.

[22] W. Klotz. A constructive proof of Kuratowski's theorem. *Ars Combin.*, 28:51–54, 1989.

[23] K. Kuratowski. Sur le problème des courbes gauches en topologie. *Fundamenta Mathematicae*, 15:271–283, 1930.

[24] Bojan Mohar and Petr Škoda. Obstructions of connectivity two for embedding graphs into the torus. *Canad. J. Math.*, 66(6):1327–1357, 2014.

[25] Bojan Mohar and Carsten Thomassen. *Graphs on surfaces*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, 2001.

[26] Wendy Myrvold and William Kocay. Errors in graph embedding algorithms. *J. Comput. System Sci.*, 77(2):430–438, 2011.

[27] E. Neufeld. Practical toroidality testing. Master's thesis, Department of Computer Science, University of Victoria, Victoria, B.C., 1993.

[28] Eugene Neufeld and Wendy Myrvold. Practical toroidality testing. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (New Orleans, LA, 1997)*, pages 574–580. ACM, New York, 1997.

[29] Neil Robertson and P. D. Seymour. Graph minors. VIII. A Kuratowski theorem for general surfaces. *J. Combin. Theory Ser. B*, 48(2):255–288, 1990.

[30] K. Wagner. Über eine Eigenschaft der ebenen Komplexe. *Mathematische Annalen*, 114:570–590, 1937.

[31] Douglas B. West. *Introduction to graph theory*. Prentice Hall, Inc., Upper Saddle River, NJ, 1996.

[32] S. G. Williamson. Embedding graphs in the plane—algorithmic aspects. *Ann. Discrete Math.*, 6:349–384, 1980. Combinatorial mathematics, optimal designs and their applications (Proc. Sympos. Combin. Math. and Optimal Design, Colorado State Univ., Fort Collins, Colo., 1978).

[33] S. G. Williamson. Depth-first search and Kuratowski subgraphs. *J. Assoc. Comput. Mach.*, 31(4):681–693, 1984.

[34] J. Woodcock. A faster algorithm for torus embedding. Master's thesis, Department of Computer Science, University of Victoria, Victoria, B.C., 2004. Available from http://dspace.library.uvic.ca:8080/handle/1828/130.