CUCS-263-87

Finding a Maximum-Genus Graph Imbedding

Merrick L. Furst, Carnegie-Mellon University
Jonathan L. Gross, Columbia University
Lyle A. McGeoch, Carnegie-Mellon University

Abstract. The computational complexity of constructing the imbeddings of a given graph into surfaces of different genus is not well-understood. In this paper, topological methods and a reduction to linear matroid parity are used to develop a polynomial-time algorithm to find a maximum-genus cellular imbedding. This seems to be the first imbedding algorithm for which the running time is not exponential in the genus of the imbedding surface.

Categories and Subject Descriptors: F2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Computations on discrete structures*; G2.2 [**Discrete Mathematics**]: Graph Theory—*Graph Algorithms.*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Graph imbeddings, maximum genus, matroid parity

Research supported in part by NSF grant MCS-8308805, in part by Presidential Young Investigator grant DCR-8352081, in part by ONR Contract N00014-85-0768, and in part by an NSF Graduate Fellowship.

Authors' addresses: M. Furst, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA 15213; J. Gross, Computer Science Department, Columbia University, New York, NY 10027; L. McGeoch, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA 15213.

1 Introduction

Lower-dimensional topology has long been approached combinatorially. For most questions about imbeddings, there exist exhaustive algorithms. Since the number of combinatorial equivalence classes of graph imbeddings is a super-exponential function of the number of vertices, such exhaustive algorithms are computationally infeasible.

Many results have been obtained concerning the computation of minimum-genus imbeddings. Hopcroft and Tarjan [11] discovered a linear-time algorithm to test planarity of graphs. Gross and Rosen [7] solved the same problem for 2-complexes. Filotti [2] found a polynomial-time algorithm to determine if a cubic graph can be imbedded in the torus, and Filotti, Miller, and Reif [3] generalized this to an algorithm that imbeds a graph in a surface of minimum genus G in time $O(v^{O(G)})$. These algorithms produce an imbedding whenever it exists and are based on extending partial imbeddings of graphs. Reif [20] showed that there are limits to this approach, by showing that the problem of deciding whether a partial imbedding in some surface can be extended to a full imbedding in that surface is NP-complete.

Our present concern is the determination of the "maximum genus" of a graph. There is no limit to the number of handles one might add to a surface in which a graph is already imbedded. For the concept of maximum genus to be meaningful, one must stipulate that every region of the imbedding be cellular — that is, the interior of the region must be homeomorphic to an open disk. This is not an artificial restriction. It corresponds to restricting handles to be "essential".

Maximum-genus imbeddings, and the related notion of upper-imbeddable graphs, have received considerable attention in recent years. A graph is called *upper-imbeddable* if it has a maximum-genus imbedding with one or two faces. Nordhaus, Stewart, and White [17], Ringeisen [21],[22], and Zaks [28] showed that various classes of graphs were upper-imbeddable. Nebeský [16] and Jungerman [12] described combinatorial invariants of upper-imbeddable graphs. Xuong [27] showed that all graphs with two disjoint spanning trees, such as 4-edge connected graphs, are upper-imbeddable.

We consider the computational complexity of obtaining a maximum-genus imbedding. Our starting point is the combinatorial characterization by Xuong [26] of the maximum genus of a graph. This involves the consideration of all spanning trees of a graph, of which there can be exponentially many. We improve the obvious exponential-time algorithm to a polynomial-time algorithm.

2 Preliminaries about Topological Graph Theory

In topological graph theory, a "graph" is defined to be a (possibly) non-simplicial 1-complex. In other words, multiple adjacencies and self-loops are permitted. In this paper, we consider only simplicial (simple) graphs. Any graph containing self-loops and multiple adjacencies can be transformed into a simplicial graph by inserting one or more vertices in the interior of these edges. Moreover, the resulting graph is homeomorphic to the original graph, and accordingly, it has the same maximum genus. This enables us to simplify the notation. We use the standard definitions relating to graphs (see, for example, Harary [9]). All the graphs we discuss will be connected and undirected.

2.1 Surfaces

Our terminology is compatible with that of Gross and Tucker [8] and of White [25].

The *topological spaces* of interest here are all homeomorphic to subspaces of E^3. A *homeomorphism* between two topological spaces is a continuous bijective mapping with a continuous inverse. A connected topological space is a *surface* if every point has a neighborhood that is homeomorphic to the closed unit disk. A surface S is *orientable* if it does not contain a Möbius band.

We deal only with closed orientable surfaces. Every such surface S is homeomorphic to a generalized torus. The number of handles is denoted $\gamma(S)$ and is called the *genus* of the surface. A sphere, for example, is a surface of genus 0, a torus is a surface of genus 1, and a 2-handled torus is a surface of genus 2.

2.2 Graph imbeddings and faces

Although a graph is an abstract combinatorial object, there is a topological representation of it: in Euclidean 3-space, we represent each vertex by a distinct point and each edge by a distinct curve between the two endpoints, where a *curve* means a homeomorphic image of the unit interval [0,1]. We require that the interior of an edge intersect no other edge or vertex of the graph. When referring to a graph in a topological setting, we mean such a representation.

An *imbedding* $G \rightarrow S$ of a graph G in the surface S is a continuous one-to-one mapping. The components of $S - G$ are called *regions*. If each region is homeomorphic to an open disk, the imbedding is *cellular*, and the regions are called *faces*. All our imbeddings are cellular. The set of faces of an imbedding is denoted F.

A *maximum-genus imbedding* of a connected graph is a cellular imbedding of the graph in an orientable

surface having maximum genus among all such imbedding surfaces. The Euler polyhedral equation

$$|V| - |E| + |F| = 2 - 2\gamma(S)$$

holds for all cellular imbeddings. Thus, a maximum-genus imbedding is the same thing as a minimum-facecount imbedding.

2.3 Rotation systems

A *rotation* at a vertex is a cyclic permutation of the edges incident on it. A vertex with degree d admits $(d-1)!$ different rotations. A list of rotations, one for each vertex, is call a *rotation system* for the graph. This concept is due to Heffter [10]. Starting with a graph imbedding in an oriented surface, there corresponds an obvious rotation system, namely, the one in which the rotation at each vertex is consistent with the cyclic order of the neighboring vertices in that imbedding.

Edmonds [1] was first to call attention explicitly to a method for inverting that correspondence. To each oriented edge (u, v), one assigns the oriented edge (v, w) such that vertex w is the immediate successor of vertex u in the rotation at vertex v. The result is a permutation on the set of oriented edges, that is, on the set in which each undirected edge appears twice, once with each possible direction. In each edge-orbit under this permutation, the consecutive oriented edges line up head to tail, from which it follows that they form a directed cycle in the graph. We observe that it is possible for both orientations of the same edge to appear twice in the same edge-orbit. If there are n oriented edges in the orbit, then an n-sided polygon can be fitted into it. Fitting a polygon to every such edge-orbit results in a polygon on both sides of each edge, and collectively the polygons form a surface in which the graph is cellularly imbedded.

Sometimes one describes the rotation system of a graph pictorially, as in Figure 1. The graph is drawn in the plane so that the incidence of edges at each vertex is consistent with the rotation system. Obviously, unless the rotation system happens to correspond to a planar imbedding, there will be edge-crossings in the drawing. Such a drawing permits one to trace along the edge-orbits, as illustrated. Since the graph shown has 6 vertices and 9 edges, and since the rotation system has 3 edge-orbits, the imbedding surface has Euler characteristic $6 - 9 + 3$, which equals zero, from which it follows that the imbedding surface has genus one.

The existence of the bijective correspondence between the cellular imbeddings of a graph and the rotation systems enables us to reformulate the problem of finding the maximum genus of the graph as a problem of finding a rotation system with the minimum number of edge-orbits. Since edge-orbits correspond to boundary-walks of faces, this is equivalent to seeking a minimum-facecount imbedding.

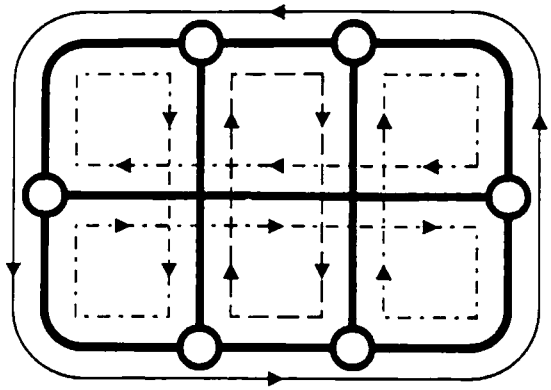

Figure 1: A graph and its edge-orbits.

2.4 Adding and deleting edges

If an edge is added to, or deleted from, an imbedded graph, then all faces in the imbedding are unchanged except those incident on that edge. If the edge is pendant, connecting a vertex of degree one, the facecount is unchanged. Otherwise, either two faces are merged or one face is split into two faces.

Suppose that an edge $e = (v, w)$ is added to a connected graph and its imbedding, so that its ends are inserted between two corners of one face. If the boundary-walk around the original face was of the form $v\alpha w\beta v$, where α and β are subwalks, then as illustrated by Figure 2, the new edge splits the old boundary-walk into two walks: $v\alpha wev$ and $w\beta vew$. Similarly, if an edge e that is common to two faces is deleted from an imbedding, then two boundary-walks are merged and the new imbedding has one less face.

If an edge is added to a graph and its ends are inserted between corners of two different faces, then both those faces are merged into one larger face. In particular, suppose that new edge e runs from the corner of v in boundary-walk $v\alpha v$ to the corner of w in boundary-walk $w\beta w$. Then a merged face results, with boundary-walk $vew\beta wev\alpha v$. This is depicted in Figure 3; the addition of edge e causes a "short-circuit," merging the two original boundary walks. Likewise, the deletion of an edge e occurring twice on one boundary-walk splits the corresponding face into two smaller faces.

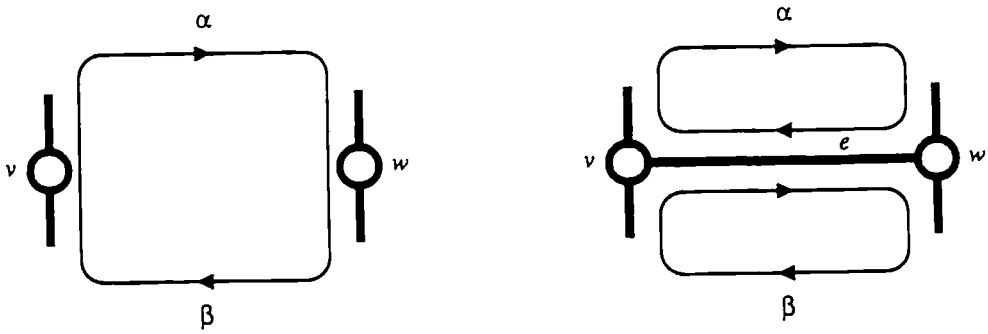

Figure 2: Adding an edge across a face.

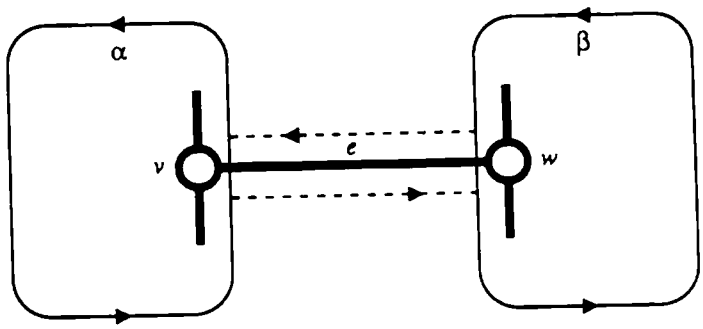

Figure 3: Adding an edge between two faces.

3 Maximum-Genus Imbeddings

We now direct our attention to the problem of constructing a maximum-genus imbedding. Xuong [26] proved that calculating the maximum genus of a graph is reducible to calculating the value of a combinatorial invariant, which he called its *deficiency*.

The *deficiency* $\xi(G,T)$ *of a spanning tree* T in a graph G is defined to be the number of connected components of $G-T$ that contain an odd number of edges. The *deficiency* $\xi(G)$ *of a graph* G is defined to be the minimum tree deficiency over all spanning trees T of G. We call a spanning tree that realizes $\xi(G)$ a *Xuong tree*. Figure 4 shows a graph and one of its Xuong trees. Since the complement of the Xuong tree has two odd components, it follows that the graph has deficiency two.

The edge complement $G-T$ of any tree T is called a *cotree*. Tree T is a spanning tree if and only if $G-T$ is a minimum cotree. The number of edges in any minimum cotree of a connected graph G is equal to $|E|-|V|+1$, and it is called the *cycle rank* (sometimes the *Betti number*) of G and denoted $\beta(G)$.

By an *adjacency matching* in a subgraph of G, we mean a matching such that each edge in the subgraph is paired with an adjacent edge. For example, one maximum adjacency matching in the cotree of Figure 4 contains pairs (a,e) and (b,d), with cotree edges c and f being unpaired.

The following reorganization of Xuong's methods and rederivation of his results is needed for our construction of a maximum-genus algorithm.

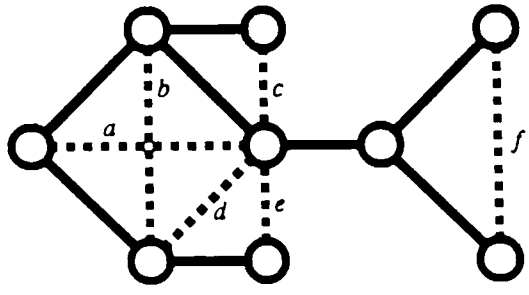

Figure 4: A spanning tree (solid edges) with minimum deficiency.

Lemma 3.1 *If a connected graph G has a completely-paired minimum cotree, then G has a one-face imbedding.*

Proof. By induction on k, the number of edge pairs in the minimum cotree.

Base case: $k = 0$. In this case the graph G is a tree, and every imbedding has exactly one face.

Inductive case: $k > 0$. As an induction hypothesis, assume that a graph with $k - 1$ pairs of edges in a minimum cotree has a one-face imbedding. We now argue that we can add a new pair of adjacent edges $e = (v, w)$ and $f = (w, x)$ to the one-face imbedded graph in the following manner. First insert edge e into the one face in any way between vertices v and w, thereby splitting the single face in two. Note that vertex w now has corners on both faces. Then insert edge f between some corner of x and a corner of w that lies on a different face. This merges the two faces, thereby resulting in a one-face imbedding of $G + e + f$. □

Lemma 3.2 *If a connected graph G has a minimum cotree with k unpaired edges, then G has an imbedding with at most $k + 1$ faces.*

Proof. Obtain a one-face imbedding of the spanning tree edges and paired cotree edges of G by the construction in Lemma 3.1. Add each of the k unpaired edges to that imbedding, creating at most one new face for each edge. □

Lemmas 3.1 and 3.2 are constructive, and given a maximum adjacency matching for a minimum cotree, any reasonable implementation of the construction will run in polynomial-time. A naive upper bound on the running time for a graph with e edges is $O(e^2)$. This can be achieved in the following manner. Imbed the spanning tree in any way, in constant time per edge. Add the first edge of some pair in any way, in constant time. Follow the boundary walks of the resulting imbedding, in $O(e)$ time, to determine a placement of the second edge that merges the two faces. Repeat until all paired edges have been added. Finally add the unpaired edges in any way, in constant time per edge.

Lemma 3.3 *If a connected graph G has a one-face imbedding, then it has a completely-paired minimum cotree.*

Proof. By induction on the number of edges, k, in G.

Base case 1: $k = |V| - 1$. In this case, the graph G is a spanning tree for itself, the cotree is empty, and trivially all edges are paired.

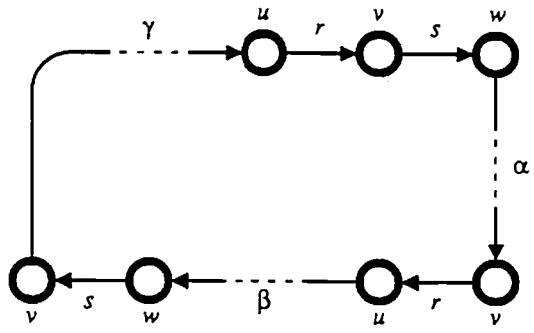

Figure 5: Boundary-walk of G before deleting edges r and s.

Base case II: $k = |V|$. In this case, the graph G is a spanning tree plus one extra edge. A spanning tree can only be imbedded with one face, and the addition of the extra edge to such a one-face imbedding must break the face in two. Thus, the graph G can only be imbedded with two faces, and the lemma holds vacuously.

Inductive case I: $k > |V|$, and G has a vertex v of degree one. Consider the graph G' obtained by deleting v and its incident edge $e = (v, w)$ from G. Since G has a one-face imbedding, we can readily construct a one-face imbedding of G' by starting with the one-face imbedding for G and deleting e and v. By induction hypothesis, the graph G' has a minimum cotree C with all its edges paired. Since the edge e must be in any spanning tree of G, C is a completely-paired minimum cotree of G.

Inductive case II: $k > |V|$, and G has no vertex of degree one. Consider the boundary-walk around the single face. There must be an edge $r = (u, v)$ whose two appearances in the walk occur as closely together as the two appearances of any other edge. Give the two appearances of r the labels $\overrightarrow{r}$ and $\overleftarrow{r}$, so as to minimize the length of subwalk α from $\overrightarrow{r}$ to $\overleftarrow{r}$. Subwalk α must contain of at least one edge other than r, or else G would have a vertex of degree one, a contradiction. Similarly, if $\overrightarrow{s}$ is the edge following $\overrightarrow{r}$, then α can not also contain $\overleftarrow{s}$, since the two appearances of edge s would then be closer together than those of edge r. Therefore, the boundary-walk around G's face must be of the form $u\overrightarrow{r}v\overrightarrow{s}w\alpha v\overleftarrow{r}u\beta w\overleftarrow{s}v\gamma u$, where $s = (v, w)$ is an edge adjacent to r in G, and α, β, and γ are subwalks. See Figure 5.

Delete edges r and s from G to obtain the graph G'. Vertices u and v are connected in G' by edges that appeared in subwalk γ, and vertices v and w are connected by edges that appeared in subwalk α. Every other vertex in G' appeared in α, β or γ and is thus connected to u, v, or w by edges in G'. Since

those three vertices are all connected, it follows that G' is connected.

By the induction hypothesis G' has a cotree C that is completely paired. Clearly the tree $G' - C$ is also a spanning tree of G. Edges r and s can be paired and added to C to form a completely-paired minimum cotree of G. □

Lemma 3.4 *If a connected graph G has a $(k+1)$-face imbedding, then it has a minimum cotree with at most k unpaired edges in its maximum adjacency matching.*

Proof. By induction on the number k.

Base case: $k = 0$. This follows from the previous lemma.

Inductive case: $k > 1$. Let e be an edge in G that lies on two different faces in some $(k+1)$-face imbedding. The graph $G - e$ is connected, for otherwise e would lie on only one face, and it has a k-face imbedding when edge e is deleted from the $(k+1)$-face imbedding of G. By the induction hypothesis, the graph $G - e$ has a minimum cotree C with at most $k - 1$ unpaired edges. Thus $C + e$ is a minimum cotree of G with at most k unpaired edges. □

A *Xuong cotree* of graph G is any minimum cotree of G that admits an adjacency matching with number of paired edges maximized (over all minimum cotrees). The number of unpaired edges in such a cotree is denoted $U(G)$.

Although Xuong did not emphasize algorithms, Theorem 3.5 is essentially contained in [26]. Theorem 3.6, which relates maximum genus to deficiency, is generally regarded as Xuong's main result.

Theorem 3.5 *A connected graph G has maximum genus*

$$\gamma_M(G) = \frac{\beta(G) - U(G)}{2}.$$

Furthermore, given a Xuong cotree C and a maximum adjacency matching of C, an imbedding of G that minimizes facecount (and thereby maximizes genus) can be found in polynomial-time.

Proof. Follow the construction in Lemma 3.2 to obtain, from a maximum adjacency matching of a Xuong cotree of G, an imbedding with $U(G) + 1$ faces. Lemma 3.4 shows that such an imbedding minimizes the number of faces. Therefore, this is a maximum-genus imbedding in which, by Euler's polyhedral equation, $\gamma_M(G) = (\beta(G) - U(G))/2$. □

Theorem 3.6 *[26] Let G be a connected graph. The maximum genus of G is given by the formula*

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

Proof. It suffices to show that $\xi(G) = U(G)$. We do this by proving that the deficiency of a spanning tree in a graph equals the minimum number of unpaired edges in the corresponding minimum cotree.

A maximum pairing of a connected non-tree component can be found in the following manner. Do a depth-first search of the component. On the post-visit to a vertex (i.e. while moving back up the search tree) pair all the unpaired incident edges. If the number of edges is odd, leave the edge to the parent unpaired. The unpaired edges never become disconnected, so eventually there will be no unpaired edges (if the component had even size) or one unpaired edge (if the component had odd size).

It follows that the minimum deficiency of G, $\xi(G)$, equals the minimum number of unpaired edges in a minimum cotree, $U(G)$. Moreover, we see that Xuong trees and Xuong cotrees, as defined here, are indeed complementary objects. □

4 Reduction of Maximum Genus to Linear Matroid Parity

In order to determine the maximum genus and find a maximum imbedding for an arbitrary graph G in polynomial-time, we have shown that it suffices to show how to find a Xuong cotree and a maximum adjacency matching of its edges in polynomial time. This problem resembles what is known as the matroid parity problem for cographic matroids. We use the definitions relating to matroid parity that are found in Stallman and Gabow's paper on linear matroid parity [24].

A *matroid* $M = (E, I)$ consists of a finite ground set E and a family I of "independent" subsets of E satisfying the following axioms:

1. If $A \in I$ and $B \subseteq A$, then $B \in I$.
2. If $A, B \in I$ and $|A| = |B| + 1$, then there exists $a \in A$ such that $B + a \in I$.

The *matroid parity problem* [14] is the following. Given a matroid $M = (E, I)$ and a perfect pairing of the elements of the ground set E, find an optimum subset of E such that an element is in the subset if and only if its paired edge is in the subset. Optimum means either a largest subset (the *cardinality parity problem*) or a maximum weighted subset (the *weighted parity problem*). Both can be solved in polynomial time for a large class of matroids known as *linear (or matric)* matroids [15], [18], [23], [24]. The most efficient known algorithm for the cardinality parity problem on general linear matroids runs in $O(nm^3)$ time, where $m = |E|$ and n is the size of the optimum subset. Matroid parity is a generalization of two well-known problems: graph matching and matroid intersection.

For any graph $G = (V, E)$, there is a linear matroid $M = (E, I)$, called the *cographic matroid*, in which the ground set is the edge set of the graph and $C \subseteq E$ is an independent set if and only if $G - C$ is connected. Maximum independent sets in cographic matroids are minimum cotrees of the corresponding

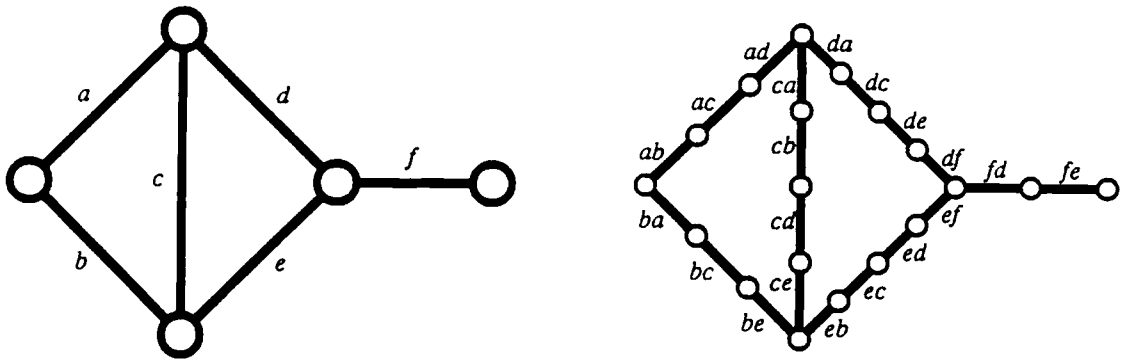

Figure 6: A graph G and a corresponding auxiliary graph G'.

graph. For any perfect matching of the edges of the graph, we have an instance of the matroid parity problem on cographic matroids, which we call the *cotree parity problem*. The cardinality parity problems for both graphic (spanning tree) and cographic matroids are easier than general linear matroid parity, and can be solved in $O(nm^2)$ time [13], [24]. Stallman and Gabow have reduced this time bound to $O(mn \log^6 n)$ [5].

If each edge of a graph G were adjacent to exactly one other edge, then we could directly apply an algorithm for cotree parity to G and obtain a maximum adjacency matching. However, adjacency is not an unambiguous pairing rule for most graphs, so direct application of a cotree parity algorithm is impossible.[1] Therefore, in this section, we shall transform G into an auxiliary graph G' with unambiguous pairs. The auxiliary graph G' is a subdivision of the graph G itself. Precisely, each edge of G is subdivided into as many edges as its number of edge-neighbors in G. Figure 6 illustrates such a subdivision.

As illustrated in Figure 6, we label each edge of the subdivided graph G' by a label of the form xy, where x is the name of the edge in G of which it is a segment and where y is the name of some distinct neighbor of edge x in the original graph G. The choice of which segment of G is to be labeled xy, for any particular adjacent edge y, is completely arbitrary, provided there is exactly one segment of x labeled xy.

We now consider edge xy to be paired with edge yx. Since this matching is unambiguous, we can apply a cotree parity algorithm to G' and construct—in polynomial time—a minimum cotree C' with a

[1]Orlin and Vande Vate [19] recently obtained a polynomial-time algorithm for a variation on the matroid parity problem for linear matroids, the *non-simple parity problem*, that is general enough to include the maximum adjacency matching problem.

maximum number of paired edges.

Let T' be the edge-complement of the cotree C' in the auxiliary graph G'. Since T' is a spanning tree for the auxiliary graph G', it contains either all the segments or all but one of the segments of every edge of the original graph G. We now associate with spanning tree T' in graph G' a subgraph T in G, according to the rule that an edge x of G appears in T if and only if every segment of x in G' occurs in T'. It is a consequence of the construction of G', T' and T that T is a spanning tree for G: T is acyclic and connected because T' is acyclic and connected.

Let the edge-complement of spanning tree T in the original graph G be called C. Then C is a minimum cotree. Two edges of C are matched if and only if they have matched segments in the cotree C' of the auxiliary graph G'.

This adjacency matching of the edges of cotree C in G is a maximum matching among all possible minimum cotrees of G, because there is a bijection between adjacency matchings in minimum cotrees of G and matchings in minimum cotrees of G' that preserves the size of the matching.

Thus, we have constructed a Xuong cotree for G and a maximum adjacency pairing of its edges in polynomial time.

5 The Algorithm

We now summarize and analyze the algorithm for obtaining a maximum-genus imbedding. It includes a method for solving the cotree parity problem, described by Gabow [4], that takes advantage of the special structure of our auxiliary graphs.

Suppose graph G has v vertices, e edges, and maximum degree d. The following steps are used.

1. Create auxiliary graph G' by subdividing edges in G. The new graph has $e' = O(ed)$ edges and $v' = v + e' - e$ vertices. This step runs in time $O(ed)$.

2. Find a maximum adjacency matching M in the original graph G. A matching can be found in $O(e)$-time by depth-first search. (On the post-visit to any vertex v, pair up all unpaired edges incident on it. If the number is odd, leave the edge to v's parent unpaired.) If G has an odd number of edges, some edge remains unpaired. In this case, add to M an extra pair consisting of this edge and some adjacent edge, even though this means that some edge is paired twice. Matching M consists of at most $\lceil e/2 \rceil$ pairs.

3. Find a matching M' in G' consisting of all edge pairs except those corresponding to pairs in M. The

edges in this matching form a cycle-free subset of G', because every edge in G is in some pair in M. At least $e' - e - 1$ edges are paired in G'. This step requires $O(e') = O(ed)$ time.

4. Extend M' to a maximum matching in a spanning tree of G', by applying the graphic matroid parity algorithm of Gabow and Stallman [5]. Each step of this algorithm, which either adds another pair to the matching or determines that no larger matching is possible, runs in $O(e' \log^6 v')$ time. Because a complete spanning tree of G' has $v' - 1 = v + e' - e - 1$ edges, matching M' can be augmented at most v times, giving a total time of $O(evd \log^6 v)$ for this step.

5. Extend M' to a complete spanning tree of G' by greedily adding edges, in $O(e') = O(ed)$ time. This spanning tree has a minimum number of unpaired edges, hence the corresponding minimum cotree has minimum number of unpaired edges.

6. For each cotree edge in G', label the corresponding edge in G as a cotree edge. Pair the edges in G that correspond to paired edges in G'. This requires $O(e') = O(ed)$ time.

7. Find a one-face imbedding of the spanning tree edges of G. This requires $O(e)$ time.

8. Add the paired cotree edges to the imbedding. The first edge of each pair can be added in constant time, but $O(e)$ time is required to find the two resulting faces and determine the placement of the second edge relative to the first. This step requires a total of $O(e^2) = O(evd)$ time.

9. Add unpaired cotree edges to the imbedding. This takes $O(v)$ time, since there is at most one unpaired edge per vertex.

The entire algorithm takes $O(evd \log^6 v)$ time.

6 Open Problems

1. The fact that maximum genus is reducible to linear matroid parity, which is a generalization of maximum matching, suggests that the corresponding counting problem may be provably difficult. Is it possible that counting the number of ways a graph may be imbedded in a surface of maximum genus is #P-complete?

2. Our algorithm for computing a maximum genus imbedding runs in time polynomial in the size of the graph. This is the only algorithm we know of for constructing any kind of imbedding that runs

in time independent of the genus. Is it possible to extend the algorithm to return imbeddings in which the genus is a fixed constant less than the maximum?

3. Suppose graphs G and H are non-isomorphic. One might ask how the non-isomorphism shows up in the way the graphs may be imbedded in different surfaces. Knowing all the "counting information" about how a graph imbeds in all surfaces is not a complete invariant for isomorphism. It clearly isn't a complete invariant for trees, which only have planar imbeddings, and we have examples of non-isomorphic, highly-connected graphs such that counting the number of imbeddings in all surfaces does not distinguish them [6]. However, randomly sampling imbeddings and making estimates of the number of ways different graphs imbed in different surfaces may prove to be an interesting new isomorphism heuristic.

7 Acknowledgements

The authors thank Hal Gabow for his careful review of an early draft of this paper. His ideas on solving the cotree parity problem in the special case of "auxiliary graphs" greatly improved the imbedding algorithm.

References

[1] Edmonds, J. A combinatorial representation for polyhedral surfaces. *Notices Amer. Math. Soc.*, 7:646, 1960.

[2] Filotti, I. S. An algorithm for imbedding cubic graph in the torus. *J. Comp. Sys.*, 20:255–276, 1980.

[3] Filotti, I. S., Miller, G., and Reif, J. On determining the genus of a graph in $O(v^{O(G)})$ steps. In *Proceedings of the 11th ACM Symposium on Theory of Computing*, pages 27–37, Atlanta, 1979.

[4] Gabow, H. N. Personal communication, 1986.

[5] Gabow, H. N. and Stallman, M. Efficient algorithms for graphic matroid intersection and parity. In *Automata, Languages and Programming: 12th Colloquium*, Volume 194 of *Lecture Notes in Computer Science*, pages 210–220, Springer-Verlag, 1985.

[6] Gross, J. L. and Furst, M. L. Hierarchy for imbedding-distribution invariants of a graph. 1986. Manuscript.

[7] Gross, J. L. and Rosen, R. A linear-time planarity algorithm for 2-complexes. *JACM*, 20:611–617, 1979.

[8] Gross, J. L. and Tucker, T. *Topological Graph Theory*. Wiley-Interscience, New York, to appear.

[9] Harary, F. *Graph Theory*. Addison-Wesley, Reading, MA, 1969.

[10] Heffter, L. Über das problem der nachbargebiete. *Math. Ann.*, 38:477–508, 1891.

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

[12] Jungerman, M. A characterization of upper embeddable graphs. *Trans. Amer. Math. Soc.*, 241:401–406, 1978.

[13] Lawler, E. *Combinatorial Optimization, Networks, and Matroids*. Holt, Rinehart and Winston, New York, 1976.

[14] Lawler, E. *Matroids with Parity Conditions: A New Class of Combinatorial Optimization Problems*. Memorandum ERL-M334, Electronics Research Laboratory, Berkeley, 1971.

[15] Lovász, L. The matroid matching problem. In *Algebraic Methods in Graph Theory*, Colloquia Mathematica Societatis Janos Bolyai, Szegad, Hungary, 1978.

[16] Nebeský, L. Every connected, locally connected graph is upper embeddable. *J. Graph Theory*, 5:197–199, 1981.

[17] Nordhaus, E., Stewart, B., and White, A. T. On the maximum genus of a graph. *J. Comb Th. B*, 11:258–267, 1971.

[18] Orlin, J. B. and Vande Vate, J. H. *An Algorithm for the Linear Matroid Parity Problem*. ISyE Report J-86-3, Georgia Institute of Technology, Atlanta, 1986.

[19] Orlin, J. B. and Vande Vate, J. H. On the non-simple parity problem. 1986. Manuscript.

[20] Reif, J. The complexity of extending a graph imbedding. 1979. Unpublished manuscript.

[21] Ringeisen, R. $K_{m,n}$ has 2-cell imbeddings. *J. Comb. Th. B*, 12:101–104, 1972.

[22] Ringeisen, R. *The Maximum Genus of a Graph*. PhD thesis, Michigan State University, 1970.

[23] Stallman, M. Weighted matroid parity. 1984. Unpublished manuscript.

[24] Stallman, M. and Gabow, H. N. An augmenting path algorithm for the parity problem on linear matroids. In *Proceedings of the 25th IEEE Symposium on the Foundations of Computer Science*, pages 217–227, Singer Island, FL, October 1984.

[25] White, A. T. *Graphs, Groups, and Surfaces.* North-Holland, Amsterdam, 1984. Second edition.

[26] Xuong, N. H. How to determine the maximum genus of a graph. *J. Comb. Th. B*, 26:216–225. 1979.

[27] Xuong, N. H. Upper-embeddable graphs and related topics. *J. Comb. Th. B*, 26:226–232. 1979.

[28] Zaks, J. The maximum genus of cartesian products of graphs. *Canad. J. Math*, 26:1025–1035. 1974.