Electronic Colloquium on Computational Complexity, Report No. 79 (2010)

Space Complexity of Perfect Matching in Bounded Genus Bipartite Graphs

Samir Datta * Raghav Kulkarni † Raghunath Tewari‡
N. V. Vinodchandran§

April 27, 2010

Abstract

We investigate the space complexity of certain perfect matching problems over bipartite graphs embedded on surfaces of constant genus (orientable or non-orientable). We show that the problems of deciding whether such graphs have (1) a perfect matching or not and (2) a unique perfect matching or not, are in the logspace complexity class SPL. Since SPL is contained in the logspace counting classes $\oplus\mathsf{L}$ (in fact in $\mathrm{Mod}_k\mathrm{L}$ for all $k \geq 2$), $\mathrm{C}_=\mathrm{L}$, and PL, our upper bound places the above-mentioned matching problems in these counting classes as well. We also show that the search version, computing a perfect matching, for this class of graphs is in $\mathsf{FL}^{\mathsf{SPL}}$. Our results extend the same upper bounds for these problems over bipartite planar graphs known earlier.

As our main technical result, we design a logspace computable and polynomially bounded weight function which isolates a minimum weight perfect matching in bipartite graphs embedded on surfaces of constant genus. We use results from algebraic topology for proving the correctness of the weight function.

*Chennai Mathematical Institute, India: `email:sdatta@cmi.ac.in`

†University of Chicago: `email:raghav@cs.uchicago.edu`

‡University of Nebraska-Lincoln: `email:rtewari@cse.unl.edu`. Research supported in part by NSF grants CCF-0830730 and CCF-0916525.

§University of Nebraska-Lincoln: `email:vinod@cse.unl.edu`. Research supported in part by NSF grants CCF-0830730 and CCF-0916525.

ISSN 1433-8092

1 Introduction

The *perfect matching* problem and its variations are one of the most well-studied problems in theoretical computer science. Research in understanding the inherent complexity of computational problems related to matching has lead to important results and techniques in complexity theory and elsewhere in theoretical computer science. However, even after decades of research, the exact complexity of many problems related to matching is not yet completely understood.

We investigate the *space complexity* of certain well studied perfect matching problems over bipartite graphs. We prove new uniform space complexity upper bounds on these problems for *graphs embedded on surfaces of constant genus*. We prove our upper bounds by solving the technical problem of 'deterministically isolating' a perfect matching for this class of graphs.

Distinguishing a single solution out of a set of solutions is a basic algorithmic problem with many applications. The *isolating lemma* due to Mulmulay, Vazirani, and Vazirani provides a general randomized solution to this problem. Let $\mathcal{F}$ be a non-empty set system on $U = \{1, \ldots, n\}$. The isolating lemma says, for a random weigh function on U (bounded by $n^{O(1)}$), with high probability there is a *unique* set in $\mathcal{F}$ of minimum weight [MVV87]. This lemma was originally used to give an elegant RNC algorithm for constructing a maximum matching (by isolating a minimum weight perfect matching) in general graphs. Since its discovery, the isolating lemma has found many applications, mostly in discovering new randomized or non-uniform upper bounds, via isolating minimum weight solutions [MVV87, RA00, GW96, ARZ99]. Clearly, derandomizing the isolating lemma in sufficient generality will improve these upper bounds to their deterministic counterparts and hence will be a major result. Unfortunately, recently it is shown that such a derandomization will imply certain circuit lower bounds and hence is a difficult task [AM08].

Can we bypass isolating lemma altogether and deterministically isolate minimum weight solutions in specific situations? Recent results illustrate that one may be able to use the structure of specific computational problem under consideration to achieve non-trivial deterministic isolation. In [BTV09], the authors used the structure of directed paths in planar graphs to prescribe a simple weight function that is computable deterministically in logarithmic space with respect to which the minimum weight directed path between any two vertices is unique. In [DKR08], the authors isolated a perfect matching in planar bipartite graphs. In this paper we extend the deterministic isolation technique of [DKR08] to isolate a minimum weight perfect matching in bipartite graphs embedded on constant genus surfaces.

Our Contribution

Let G be a bipartite graph with weight function w on it edges. For an even cycle $C = e_1 e_2 \cdots e_{2k}$, the circulation of C with respect to w is the sum $\sum_{i=1}^{2k}(-1)^i w(e_i)$. The main technical contribution of the present paper can be stated (semi-formally) as follows.

Main Technical Result. There is a logspace matching preserving reduction f, and a logspace computable and polynomially bounded weight function w, so that given a bipartite graph G with a combinatorial embedding on a surface of constant genus, the circulation of any simple cycle in $f(G)$ with respect to w is non-zero. (This implies that the minimum weight perfect matching in $f(G)$ is unique [DKR08]).

We use this result to establish (using known techniques) the following new upper bounds.

Refer to the next section for definitions.

New Upper Bounds. For bipartite graphs, combinatorially embedded on surfaces of constant genus the problems DECISION-BPM and UNIQUE-BPM are in SPL, and the problem SEARCH-BPM is in $\mathsf{FL}^{\mathsf{SPL}}$.

SPL is a logspace complexity class that was first studied by Allender, Reinhardt, and Zhou [ARZ99]. This is the class of problems reducible to the determinant with the promise that the determinant is either 0 or 1. In [ARZ99], the authors show, using a non-uniform version of isolating lemma, that perfect matching problem for general graphs is in a 'non-uniform' version of SPL. In [DKR08], using the above-mentioned deterministic isolation, the authors show that for planar bipartite graphs, DECISION-BPM is in fact in SPL (uniformly). Recently, Hoang showed that for graphs with polynomially many matchings, perfect matchings and many related matching problems are in SPL [Hoa09]. SPL is contained in logspace counting classes such as Mod_kL for all $k \geq 2$ (in particular in $\oplus$L), PL, and $\text{C}_=\text{L}$, which are in turn contained in NC^2. Thus the upper bound of SPL that we prove implies that the problems DECISION-BPM and UNIQUE-BPM for the class of graphs we study are in these logspace counting classes as well.

The techniques that we use in this paper can also be used to isolate directed paths in graphs on constant genus surfaces. This shows that the reachability problem for this class of graphs can be decided in the unambiguous class UL, extending the results of [BTV09]. But this upper bound is already known since recently Kynčl and Vyskočil show that reachability for bounded genus graphs logspace reduces to reachability in planar graphs [KV09].

Matching problems over graphs of low genus have been of interest to researchers, mainly from a parallel complexity viewpoint. The matching problems that we consider in this paper are known to be in NC. In particular in [KMV08], the authors present an NC^2 algorithm for computing a perfect matching for bipartite graphs on surfaces of $O(\log n)$ genus (readers can also find an account of known parallel complexity upper bounds for matching problems over various classes of graphs in their paper). However, the space complexity of matching problems for graphs of low genus has not been investigated before. The present paper takes a step in this direction.

Proof Outline. We assume that the graph G is presented as a combinatorial embedding on a surface (orientable or non-orientable) of genus g, where g is a constant. This is a standard assumption when dealing with graphs on surfaces, since it is NP-complete to check whether a graph has genus $\leq g$ [Tho89]. We first give a sequence of two reductions to get, from G, a graph G' with an embedding on a genus g 'polygonal schema in normal form'. These two reductions work for both orientable and non-orientable cases. At this point we take care of the non-orientable case by reducing it to the orientable case. Once we have the embedding on an orientable polygonal schema in normal form, we further reduce G' to G'' where G'' is embedded on a constant genus 'grid graph'. These reductions are matching preserving, bipartiteness preserving and computable in logspace. Finally, for G'', we prescribe a set of $4g+1$ weight functions, $\mathcal{W} = \{w_i\}_{1 \leq i \leq 4g+1}$, so that for any cycle C in G'', there is a weight function $w_i \in \mathcal{W}$ with respect to which the circulation of C is non-zero. Since g is constant, we can take a linear combination of the elements in $\mathcal{W}$, for example $\sum_{w_i \in \mathcal{W}} w_i \times (n^c)^i$ (where n is the number of vertices in the grid) for some fixed constant c (say $c = 4$), to get a single weight function with respect which the circulation of any cycle is non-zero.

The intuition behind these weight functions is as follows (for some of the definitions, refer to later sections). The set $\mathcal{W}$ is a disjoint union $\mathcal{W}_1 \cup \mathcal{W}_2 \cup \{w\}$ of the sets of weight functions $\mathcal{W}_1$, $\mathcal{W}_2$, and $\{w\}$. Consider a graph G embedded on a fundamental polygon with $2g$ sides. There are two types cycles in G: *surface separating* and *surface non-separating*. A basic theorem from algebraic topology implies that a surface non-separating cycle will intersect at least one of the sides of the polygon an odd number of times. This leads to $2g$ weight functions in $\mathcal{W}_1$ to take care of all the surface non-separating cycles. There are two types of surface separating cycles: (a) ones which completely lie inside the polygon and (b) the ones which cross some boundary. Type (a) cycles behaves exactly like cycles in plane so the weight function w designed for planar graphs works (from [DKR08]). For dealing with cycles of type (b), we first prove that if such a cycle intersects a boundary, it should alternate between 'coming in' and 'going out'. This leads to $2g$ weight functions in $\mathcal{W}_2$ which handle all type (b) cycles.

Figure 1 gives a pictorial view of the components involved in the proof of our main technical result.

The rest of the paper is organized as follows. In Section 2 we give the necessary definitions and state results from earlier work, that we use in this paper. In Section 3 we state and prove our upper bounds assuming a grid embedding. In Section 4 we reduce the non-orientable case to the orientable one. In Section 5 we give matching preserving, logspace reductions from a combinatorial embedding of the graph on a surface of genus g, to a grid embedding. In Section 6 we add proofs of some necessarylemmas and theorems that we use to prove our results.

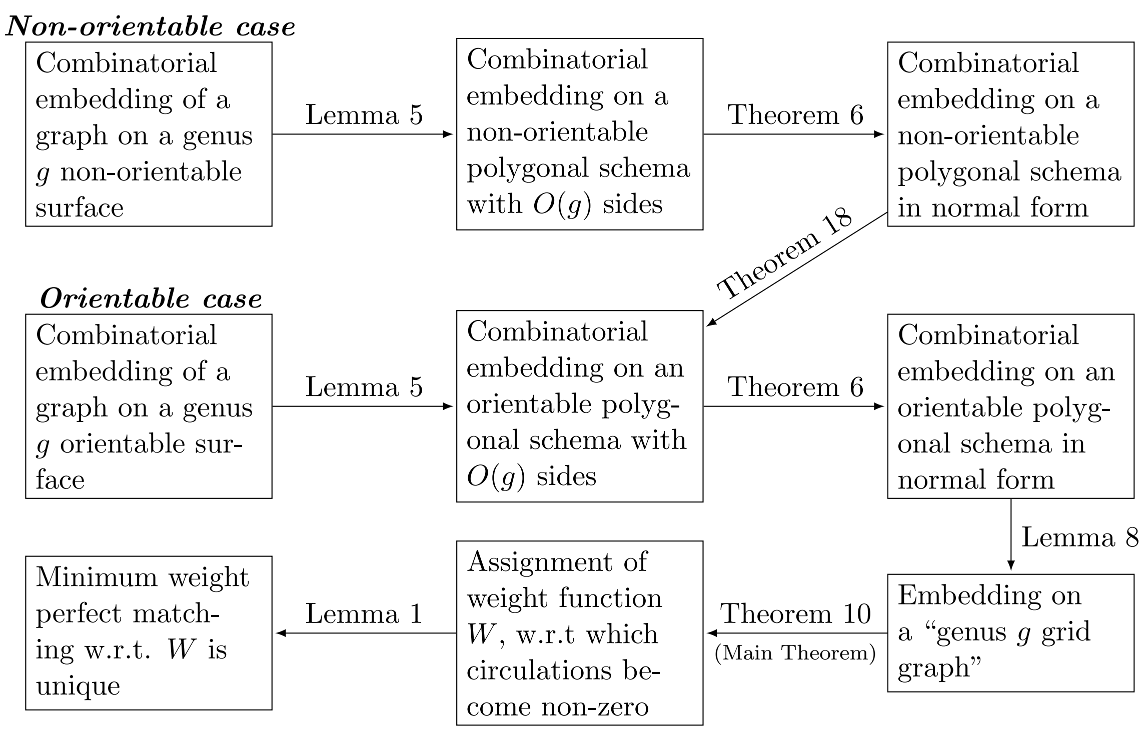


Figure 1: Outline of the steps. Note that all reductions are matching preserving and logspace computable.

2 Preliminaries

2.1 Topological graph theory

We introduce the necessary terminology from algebraic topology. For a more comprehensive understanding of this topic, refer to any standard algebraic topology book such as [Mas91].

A *2-manifold* is a topological space such that every point has an open neighborhood homeomorphic to $\mathbb{R}^2$ and two distinct points have disjoint neighborhoods. A 2-manifold is often called a *surface*. The *genus* of a surface Γ is the maximum number g, if there are g cycles $C_1, C_2, \ldots, C_g$ on Γ, such that $C_i \cap C_j = \emptyset$ for all i, j and $\Gamma \setminus (C_1 \cup C_2 \cup \ldots \cup C_g)$ is connected. A surface is called *orientable* if it has two distinct sides, else it is called *non-orientable*. A cycle C in Γ is said to be *non-separating* if there exists a path between any two points in $\Gamma \setminus C$, else it is called *separating*.

A *polygonal schema* of a surface Γ, is a polygon with $2g'$ directed sides, such that the sides of the polygon are partitioned into g' classes, each class containing exactly two sides and glueing the two sides of each equivalence class gives the surface Γ (upto homeomorphism). A side in the ith equivalence class is labelled σ_i or $\bar{\sigma}_i$ depending on whether it is directed clockwise or anti-clockwise respectively. The *partner* of a side σ is the other side in its equivalence class. By an abuse of notation, we shall sometimes refer to the symbol of a side's partner, as the partner of the symbol. Frequently we will denote a polygonal schema as a linear ordering of its sides moving in a clockwise direction, denoted by X. For a polygonal schema X, we shall refer to any polygonal schema which is a cyclic permutation, or a reversal of the symbols, or a complementation (σ mapped to $\bar{\sigma}$ and vice versa) of the symbols, as being the same as X. A polygonal schema is called orientable (resp. non-orientable) if the corresponding surface is orientable (resp. non-orientable).

Definition 1. An orientable polygonal schema is said to be in *normal form* if it is in one of the following forms:

$$\sigma_1\tau_1\bar{\sigma_1}\bar{\tau_1}\sigma_2\tau_2\bar{\sigma_2}\bar{\tau_2}\ldots\sigma_m\tau_m\bar{\sigma_m}\bar{\tau_m} \tag{2.1}$$

$$\sigma\bar{\sigma} \tag{2.2}$$

A non-orientable polygonal schema is said to be in normal form if it is of one of the following forms:

$$\sigma\sigma X \tag{2.3}$$

$$\sigma\tau\bar{\sigma}\tau X \tag{2.4}$$

where, X is a string representing an orientable schema in normal form (i.e. like Form 2.1 or 2.2 above).

We denote the polygonal schema in the normal form of a surface Γ as $\Lambda(\Gamma)$. We will refer to two orientable symbols σ, τ which form the following contiguous substring: $\sigma\tau\bar{\sigma}\bar{\tau}$ as being clustered together while a non-orientable symbol σ which occurs like $\sigma\sigma$ as a contiguous subtring is said to form a *pair*. Thus, in the first and third normal forms above all symbols are clustered. The first normal form represents a connected sum of torii and the third of a projective plane and torii. In the fourth normal form all but one of the orientable symbols are clustered while the only non-orientable symbol is sort of clustered with the other orientable symbol. This form represents a connected sum of a Klein Bottle and torii. The second normal form represents a sphere.

We next introduce the concept of $\mathbb{Z}_2$-homology. Given a 2-manifold Γ, a *1-cycle* is a closed curve in Γ. The set of 1-cycles forms an Abelian group, denoted as $\mathcal{C}_1(\Gamma)$, under the *symmetric difference* operation, Δ. Two 1-cycles C_1, C_2 are said to be homologically equivalent if $C_1 \Delta C_2$ forms the boundary of some region in Γ. Observe that this is an equivalence relation. Then the *first homology group* of Γ, $H_1(\Gamma)$, is the set of equivalence classes of 1-cycles. In other words, if $\mathcal{B}_1(\Gamma)$ is defined to be the subset of $\mathcal{C}_1(\Gamma)$ that are homologically equivalent to the empty set, then $H_1(\Gamma) = \mathcal{C}_1(\Gamma)/\mathcal{B}_1(\Gamma)$. If Γ is a genus g surface then $H_1(\Gamma)$ is generated by a system of $2g$ 1-cycles, having only one point in common, and whose complement is homeomorphic to a topological disk. Such a disk is also referred to as the *fundamental polygon* of Γ.

An undirected graph G is said to be embedded on a surface Γ if it can be drawn on Γ so that no two edges cross. We assume that the graph is given with a *combinatorial embedding* on a surface of constant genus. Refer to the book by Mohar and Thomassen [MT01] for details. A graph G is said to have *genus* g if G has a minimal embedding (an embedding where every face of G is homeomorphic to a disc) on a genus g surface. Such an embedding is also called a *2-cell embedding.* A genus g graph is said to be orientable (non-orientable) if the surface is orientable (non-orientable).

Definition 2. The *polygonal schema of a graph G* is a combinatorial embedding given on the polygonal schema of some surface Γ together with the ordered set of vertices on each side of the polygon. Formally it is a tuple $(\phi, \mathcal{S})$, where ϕ is a cyclic ordering of the edges around a vertex and $\mathcal{S} = (S_1, S_2, \ldots, S_{2g})$ is the cyclic ordering of the directed sides of the polygon. Each S_i is an ordered sequence of the vertices, from the tail to the head of the side S_i. Moreover every S_i is paired with some other side, say S_i^{-1} in $\mathcal{S}$, such that the jth vertex of S_i (say from the tail of S_i) is the same as the jth vertex of S_i^{-1} (form the tail of S_i^{-1}).

2.2 Complexity Theory

For a nondeterministic machine M, let $acc_M(x)$ and $rej_M(x)$ denote the number of accepting computations and the number of rejecting computations respectively. Denote $gap_M(x) = acc_M(x) - rej_M(x)$.

Definition 3. A language L is in SPL if there exists a logspace bounded nondeterministic machine M so that for all inputs x, $gap_M(x) \in \{0, 1\}$ and $x \in L$ if and only if $gap_M(x) = 1$. $\mathsf{FL}^{\mathsf{SPL}}$ is the class of functions computed by a logspace machine with an SPL oracle. UL is the class of languages L, decided by a nondeterministic logspace machine (say M), such that for every string in L, M has exactly one accepting path and for a string not in L, M has no accepting path.

Alternatively, we can define SPL as the class of problems logspace reducible to the problem of checking whether the determinant of a matrix is 0 or not under the promise that the determinant is either 0 or 1. For definitions of other complexity classes refer to any standard textbooks such as [AB09, Vol99]. All reductions discussed in this paper are logspace reductions.

Given an undirected graph $G = (V, E)$, a *matching* M is a subset of E such that no two edges in M have a vertex in common. A *maximum matching* is a matching of maximum cardinality. M is said to be a perfect matching if every vertex is an endpoint of some edge in M.

Definition 4. We define the following computational problems related to matching:

- Decision-BPM : Given a bipartite graph G, checking if G has a perfect matching.
- Search-BPM: Given a bipartite graph G, constructing a perfect matching, if one exists.
- Unique-BPM: Given a bipartite graph G, checking if G has a unique perfect matching.

2.3 Necessary Prior Results

Lemma 1 ([DKR08]). *For any bipartite graph G and a weight function w, if all circulations of G are non-zero, then G has a unique minimum weight perfect matching.*

Lemma 2 ([ARZ99]). *For any weighted graph G assume that the minimum weight perfect matching in G is unique and also for any subset of edges $E' \subseteq E$, the minimum weight perfect matching in $G \setminus E'$ is also unique. Then deciding if G has a perfect matching is in SPL. Moreover, computing the perfect matching (in case it exists) is in $\mathsf{FL}^{\mathsf{SPL}}$.*

Sketch of proof. Let w_{max} and w_{min} be the maximum and minimum possible weights respectively, that an edge in G can get. Then any perfect matching in G will have a weight from the set $W = \{k : k \in \mathbb{Z}, n \cdot w_{min} \leq k \leq n \cdot w_{max}\}$. Similar to [ARZ99], there exists a GapL function f, such that for some value of $k \in W$, $|f(G,k)| = 1$ if G has a perfect matching of weight k, else $f(G,k) = 0$ for all values of k. Note that in [ARZ99] the authors actually give a $\mathsf{GapL/poly}$ function since the weight function for the graphs (which are unweighted to begin with) are required as an advice in their GapL machine. Here we consider weighted graphs, thus eliminating the need for any advice. Now consider the function

$$g(G) = 1 - \prod_k \left(1 - (f(G,k))^2\right).$$

By definition, $g(G) = 1$ if G has a perfect matching, else it is 0.

To compute a perfect matching in G, we will construct a logspace transducer that makes several queries to the function f defined above. For a graph G' having a unique minimum weight perfect matching (say M'), the weight of M' can be computed by iteratively querying the function $f(G',k)$ for values of $k \in W$ in an increasing order, starting from $n \cdot w_{min}$. The value k, for which the function outputs a non-zero value for the first time, is the weight of M'. We denote this weight by $w_{G'}$. First compute w_G. For an e in G, define the graph $G^{-e} = G \setminus \{e\}$. Now compute $w_{G^{-e}}$ for every edge e in G. Output the edges e for which $w_{G^{-e}} > w_G$. The set of outputted edges comprise a perfect matching (in fact the minimum weight perfect matching) because deleting an edge in this set had increased the weight of the minimum weight perfect matching in the resulting graph. □

3 Embedding on a Grid

Theorem 3. *Given a 2-cell combinatorial embedding of a graph G of constant genus, there is a logspace transducer that constructs a graph $G' \in$ k-ori-GG, such that, there is a perfect matching in G iff there is a perfect matching in G'. Moreover, given a perfect matching M' in G', in logspace one can construct a perfect matching M in G.*

Proof. Using Corollary 7 reduce G to a graph G_1 that has an embedding on the polygonal schema in the normal form. If the schema is non-orientable, then by applying Theorem 18 we get a graph G_2 along with its embedding on an orientable polygonal schema (need not be in the normal form). Again by Corollary 7, we reduce it to a graph on a polygonal schema in the normal form. Finally we apply Lemma 8, we get the desired graph. $\square$

3.0.1 Combinatorial Embedding to a Polygonal Schema

Lemma 4 ([ABC$^+$09])**.** *Let G be a graph embedded on a surface, and let T be a spanning tree of G. Then there is an edge $e \in E(G)$ such that $T \cup \{e\}$ contains a non-separating cycle.*

Notice that in [ABC$^+$09] the graph was required to be embedded on an orientable surface but the proof did not use this requirement.

Definition 5. Given a cycle (or path) C in an embedded graph G, define by G✂C the graph constructed by "cutting" the edges incident on the cycle from the right. In other words, the neighbors of $u \in C$ (which are not on the cycle) can be partitioned into two sets, arbitrarily called left and right. For every neighbbor v of u which lies to the right of C, cut the edge (u, v) into two pieces (u, x_{uv}) and (y_{uv}, v) where x_{uv}, y_{uv} are (new) spurious vertices. We add spurious edges between consecutive spurious vertices along the cut and label all the newly formed spurious edges with the label L_C along the left set and L_C^{-1} along the right set. (see Figure 2).

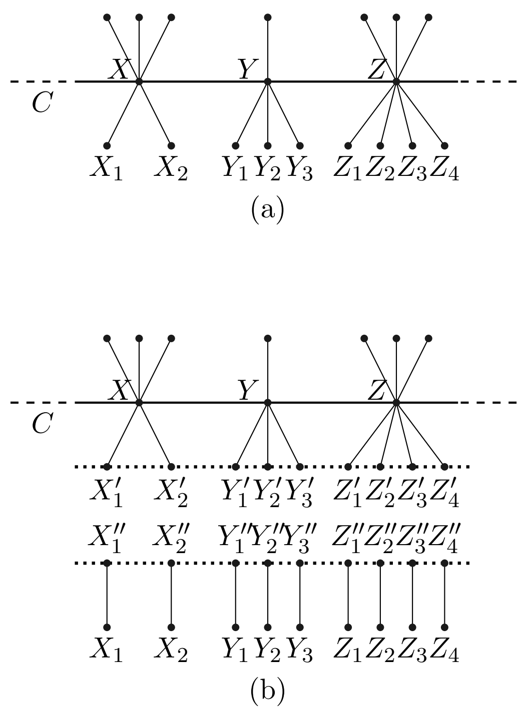


Figure 2: An example of the cut operation ✂, cutting graph G along cycle (or path) C. (a) Part of graph G and cycle C. (b) Part of the resulting graph G✂C, with the dotted lines representing the spurious edges.

Also, if C is a path, its endpoints will lie on two paths. Consider the first path - if the two edges on either side of C on this path have the same label L_1. This can be broken into two cases - firstly, if the left and right side of this endpoint are the same (in other words, the path is a cycle). In this case, we just keep the same label L_1. When the left and right side of this endpoint are distinct, we will need to split the label into two or three new labels as detailed below and similarly for the other path and common label L_2. We will only describe the case when L_1, L_2 are both defined - the other cases are similar and simpler.

First assume that $L_1 \neq L_2$ and $L_1 \neq L_2^{-1}$. Then we will split remove labels L_1, L_2 and replace them by four new labels say $L'_{1,C}, L''_{1,C}$ and $L'_{2,C}, L''_{2,C}$, respectively for the two sides of the intersection. If, on the other hand, L_1 is the same as L_2 or its inverse - then there are two subcases. Firstly, if the path C is between two copies of the same vertex then we replace L_1 by two new labels $L'_{1,C}, L''_{1,C}$ one for either side of the cut. L_2 being a copy or an inverse copy of L_1 splits automatically. The second case is if C is between two distinct points on two copies or inverse copies. Then we split L_1 into three parts according to the two points. The rotation system is modified appropriately. We illustrate this in Figure 3.

Notice that in the process of cutting, for every new label L_C we are adding at most 4 new labels.

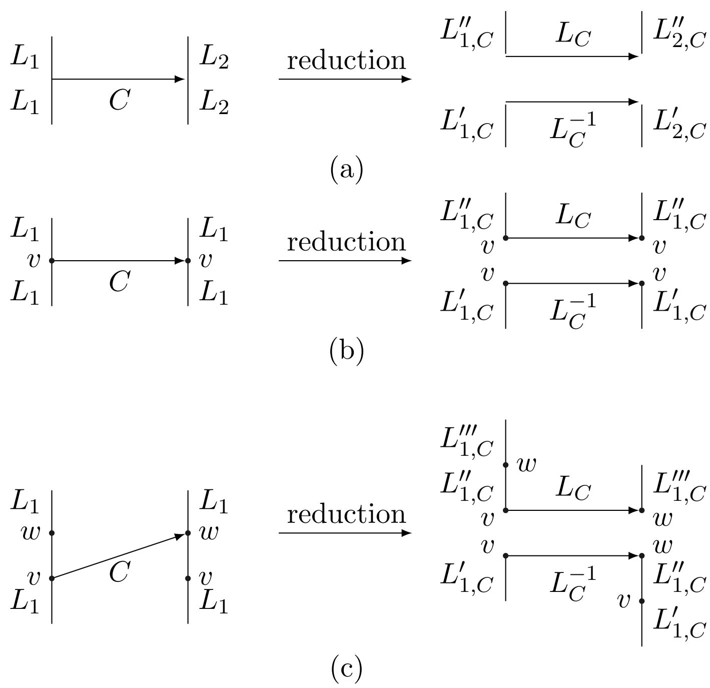


Figure 3: Cutting along a path C when (a) $L_1 \neq L_2$ and $L_1 \neq L_2^{-1}$, (b) $L_1 = L_2$ or $L_1 = L_2^{-1}$ and C is between copies of the same vertex v, and (c) $L_1 = L_2$ or $L_1 = L_2^{-1}$ and C is between distinct vertices v and w.

Given a graph G_i embedded on a surface, potentially with spurious edges, we can find C_{i+1}, a non-separating cycle (which does not use a spurious edge) by invoking Lemma 4. Define G_{i+1} to be G_i✂C_{i+1}.

Starting with $G_0 = G$ of genus g and repeating the above operation at most g times, we get a planar graph H with at most $2g$ spurious faces (which consist of spurious vertices and edges).

Now find a spanning tree of this graph which does not use a spurious edge - that such a tree exists follows from noticing that the graph without spurious edges is still connected.

Find a tree path connecting any two spurious faces. Cut along this path to combine the two spurious faces into one larger spurious face. Repeat the operation till all the spurious faces are merged into one spurious face and re-embed the planar graph so that it forms the external face.

It is easy to see that the procedure above can be performed in logspace, provided that g is constant. Thus we have sketched the proof of the following:

Lemma 5. *Given the combinatorial embedding of a constant genus graph we can find a polygonal schema for the graph in logspace.*

3.0.2 Normalizing a Polygonal Schema

We adapt the algorithmic proof of Brahana-Dehn-Heegaard (BDH) [Bra21, DH07] classification theorem as described in Vegter-Yap [VY90] so that it runs in logspace for constant genus graphs. The algorithm starts with a polygonal schema and uses the following five transforms $O(m)$ times to yield a normalized polygonal schema, where the original polygonal schema has $2m$ sides.

A. Replace $X\sigma\bar{\sigma}$ by X (Example given in Figure 4).

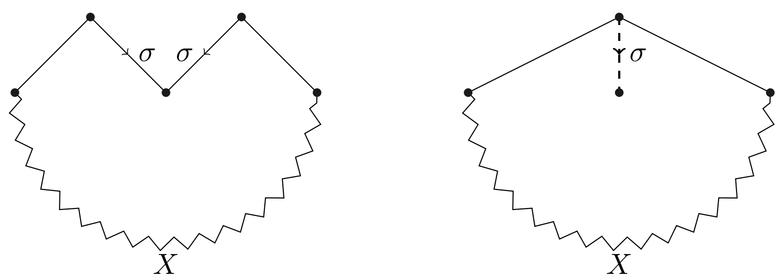


Figure 4: Reduction A (pasting along σ)

B. Replace $\sigma\tau X\bar{\tau}Y$ by $\rho X\bar{\rho}\sigma Y$ (Example given in Figure 5).

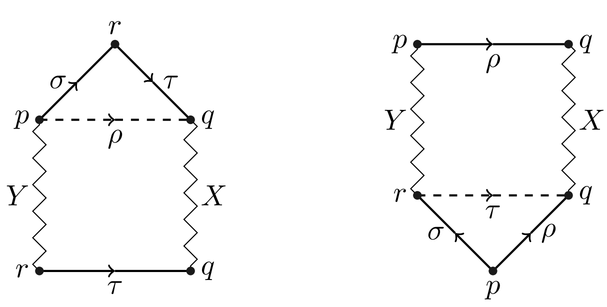


Figure 5: Reduction B (Cutting along ρ followed by pasting along τ). Note that the number of vertices in the equivalence class of r, reduces by 1.

C. Replace $\sigma X\sigma Y$ by $\tau\tau Y^*X$, where Y^* is reverse complement of Y (Example given in Figure 6).

D. Replace $\sigma X\tau Y\bar{\sigma}U\bar{\tau}V$ by $\rho\pi\bar{\rho}\bar{\pi}UYXV$ (Example given in Figure 7).

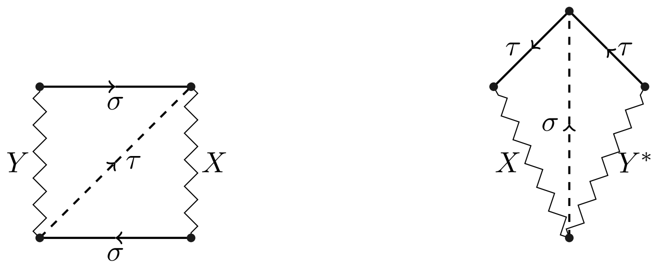
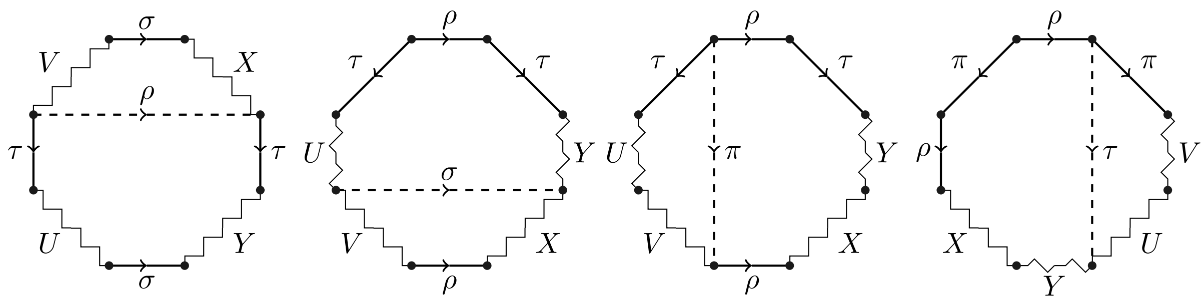


Figure 6: Reduction C (Cutting along τ followed by pasting along ρ)



(a) (b) (c) (d)

Figure 7: Reduction D (a) Cutting along ρ. (b) Pasting along σ. (c) Cutting along π. (d) Pasting along τ.

E. Replace $\sigma_1\sigma_1X\sigma_2\sigma_3\bar{\sigma}_2\bar{\sigma}_3Y$ by $\tau_1\tau_1\tau_2\tau_2\tau_3\tau_3XY$ (Example given in Figure 8).

F. Replace $\sigma\sigma\tau\tau X$ by $\sigma\rho\bar{\sigma}\rho X$ (Example given in Figure 9).

The procedure is to

1. Use reductions A,B,C several times to ensure that all the sides of the polygonal schema have a common endpoint.

2. (a) Orientable case: Use transform D repeatedly to bring the polygon in normal form.

   (b) Non-orientable case:

   - Use reductions C,D to convert the schema into a form where the orientable symbols are clustered and non-orientable symbols are paired
   - Use reduction E repeatedly (in the forward direction) to eliminate all orientable symbols.
   - Use Reduction E in the *reverse* direction repeatedly to eliminate all but at most one non-orientable symbol.
   - Use Reduction F, if necessary, to ensure that there is at most one non-orientable symbol.

Possibly, the only step requiring any explanation is the last one. We apply Reduction E in reverse with X as the empty string to replace three non-orientable symbols by two orientable ones forming a cluster of 4 and a single non-orientable one which forms a pair. The way

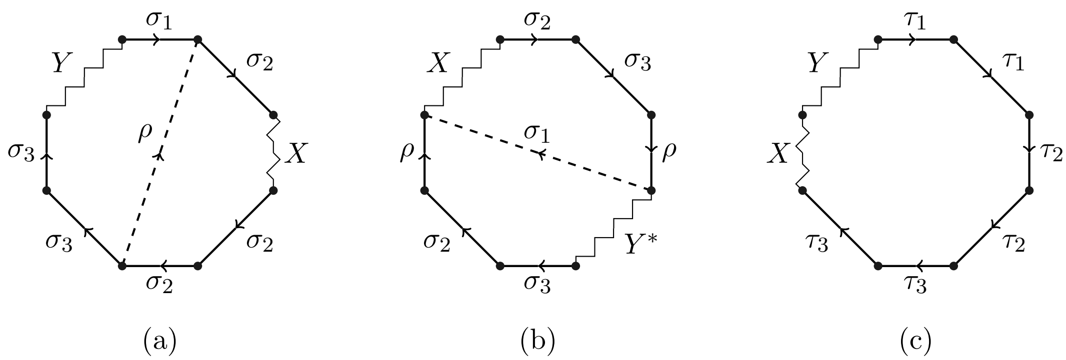


Figure 8: Reduction E (a) Cutting along ρ. (b) Pasting along σ_1. (c) Obtained from Figure 8(b) by applying Reduction C thrice.

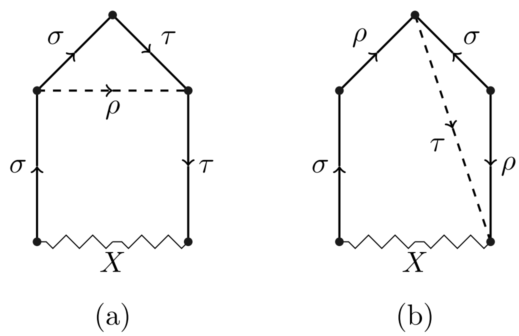


Figure 9: Reduction F (a) Cutting along ρ. (b) Pasting along τ.

we apply the reduction, ensures that both the orientable and the non-orientable parts are contiguous.

Fianlly we will be left with a string in one of the first two normal forms or a string of the form $\sigma\sigma\tau\tau X$ (where X is an orientable schema in normal form) in which case Reduction F is applicable.

To see that the above procedure can be carried out in L it suffices to prove that each of the above reductions can be carried out in L, the number of reductions is bounded by a constant and we can decide in L when to carry out a reduction.

The Vegter-Yap paper does careful book-keeping in order to ensure that the number of operations in Step 1 is linear in the original genus. We can alternatively, follow the brute force approach and keep on applying Reductions A,B,C while the sides of the polygon do not have a common end-point. This will require at most linear number of applications of the first two reductions.

Observe that for the orientable case, each application of reduction D reduces the number of unclustered symbols by two. Thus we are done in $O(m)$ applications of this reduction. Similarly, each application of reduction C reduces the number of unpaired non-orientable symbols by one and as before every application of reduction D reduces the number of unclustered orientable symbols by two. So in $O(m)$ steps all the orientable symbols are clustered and the non-orientable symbols are paired. Now every application of reduction E in the forward direction gets rid of two orientable symbols so in $O(m)$ steps all the orientable symbols are removed. Finally $O(m)$ applications of reduction E in reverse lead to removal of all but one non-orientable symbols.

To see that each of the steps is in L observe that each of the steps involves one or more of the following operations:

- find a path through the interior of the polygon between two points on its boundary
- cut along a path
- paste two paired sides of (a cut) polygon together

We know how to do the second operation in L while the third, being the reverse of the second one is even easier, since we just have to identify corresponding spurious vertices and then excise them out of the corresponding edge. The first operation is just an undirected reachability question in the graph (minus its boundary) hence is in L by Reingold's Theorem.

Finally, a determination of when to apply a particular reduction is easily seen to be in L for all but, possibly, reduction D. In this case, for an orientable symbol σ separated from its mate $\bar{\sigma}$ on both sides, sequentially test for each other symbol τ if it lies in one of the two stretches that σ and its mate divide the schema into, while its mate $\bar{\tau}$ lies in the other. Having found the first such τ suffices to enable a use of the reduction.

Thus, using the above argument and Lemma 5 we have sketched the proof of the following theorem:

Theorem 6. *Given a combinatorial embedding of constant genus, say g (which is positive or otherwise), for a graph G, in logspace we can find a polygonal schema for the graph in normal form. of genus $O(|g|)$ in magnitude, and also the corresponding combinatorial embedding.*

Let K-GON-BI be the class of constant genus, bipartite graphs along with an embedding given on the polygonal schema in normal form of the surface in which the graph has an embedding. Moreover, for every graph in this class, no edge has both its end points on the boundary of the polygon.

At this point, there are no vertices lying on the boundary of the polygonal schema, only edges crossing it. It is easy to see that for each such edge $e = (u, v)$ which has two halves lying on segments of the polygon, if we introduce internal vertices $u' = v', v''$ on the edge (converting it to a path $u, u' = v', v'', v$) so that u', v' lie on the boundary of the polygon on the sides nearer to u, v respectively, then, because the path has odd length the number of perfect matchings in the modified graph is preserved.

Thus we have proved that:

Corollary 7. *Given the combinatorial embedding of a graph of constant genus, there is an logspace reduction, which preserves perfect matchings, to a graph in the class* K-GON-BI.

3.0.3 From Polygonal Schema in normal form to a Grid

Lemma 8. *If G is an orientable graph in* K-GON-BI*, then one can get a logspace, matching-preserving reduction form G to a graph $H \in$* K-ORI-GG

Proof. We start with a graph $G \in$ K-GON-BI and construct a graph $H \in$ K-ORI-GG such that the number of perfect matchings in G and H are the same.

We can assume that the maximum degree of G is 3 and there exists a vertex s of degree 2 [KMV08]. Think of G as a planar graph. Reduce G to a grid graph G' using [ABC$^+$09]. It follows from the reduction that faces are preserved (modulo subdivision of edges). Let T be the spanning tree of G constructed by the algorithm that would be embedded on the course grid and let T' be the tree corresponding to T in G'. Every vertex (say v) on boundary of the polygon in G is a leaf node since every edge has at most one of its end points on the boundary of the polygon (by definition of K-GON-BI). Therefore v is also a leaf in T. Let s be the root of T and $h(u)$ be the height of a vertex u in T. It follows from the reduction that $h(u)$ is the value of its y-coordinate in G'.

For the rest of this proof we will use the notation u' and v' to denote the respective copies of some two vertices u and v in G. Now subdivide every horizontal edge in G' into 2 edges to get the grid graph G''. This ensures that the horizontal distance between the copies of any two vertices in G' is even. First claim is that the number of matchings in G and G'' are the same. To see this it is enough to show that: $e = (u, v)$ is an edge in G iff any simple path from u' to v' has odd length. If e is a tree edge then the vertical distance between u' and v' is 1 and the horizontal distance is even. Thus the distance between them on the grid is odd and therefore any path between them on the grid has odd length. Similarly, if e is a non-tree edge, then $h(u)$ and $h(v)$ have different parity and therefore the vertical distance between them is odd.

Now we will see how to construct the grid graph H as required by the Lemma. Let G'' be a $m_1 \times m_2$ grid. Construct an empty grid H, of size $(m_1 + 2) \times (m_2 + 2)$. Place the grid G'' on H so that G'' lies properly inside the grid (that is no edge of G'' has an end point on any of the boundary vertices of H). Suppose two vertices u and v in G get identified when G is thought of as a genus g graph. Then from our earlier observation we have that both u' and v' must be leaf nodes and lie on the outer face of G''. Also $h(u)$ and $h(v)$ must have the same parity, since otherwise we can construct an odd cycle in G by traversing from s to u (which is the same as v) and back to s via v. This implies that the y-coordinate of both u' and v' in G'' has the same parity. Drop a path from u (and similarly a path from v) by going down all the way to the south border of H. Observe that the sum of the lengths of these two paths is even. This is because, the difference in their y-coordinates is even. This ensures that matching is preserved by adding these paths.

The ordering of the segments in the outer face that get glued, is same in both H and G since faces are preserved by the reduction in [ABC$^+$09]. Also the by our construction the length of each segment is even since the horizontal distance between two vertices is a multiple of 2. Additionally from there are no edges along the boundary of the grid as required. □

3.1 Any graph in a"genus g grid" is bipartite

Lemma 9. *Any graph $G \in$ K-ORI-GG is bipartite.*

Proof. Let C be a cycle in G. First we consider the case when C is a simple cycle. Partition C into paths $P_1 = (p_1, \ldots, p_2), P_2 = (p_2, \ldots, p_3), \ldots, P_k = (p_k, \ldots, p_1)$, such that each P_i lies entirely in the grid with its two end points p_i and p_{i+1} lying on some two segments. An example of this partition is shown in Figure 10(a) for the respective cycle. For each path P_i construct a path P_i' by moving along the border of the grid from p_i to p_{i+1} along a fixed direction (say in clockwise direction).

Fix an $i \in [k]$. Consider the partition of P_i', induced by the segments along which it passes. Denote the first and the last partition by P_{i_1}' and P_{i_2}' respectively. Note that any of the intermediate partitions of P'_i has even length since the length of an intermediate partition equals the length of the corresponding segment and hence is even. Therefore we have,

$$|P'_i| \mod 2 = (|P'_{i_1}| + |P'_{i_2}|) \mod 2. \tag{3.1}$$

Also,

$$|P_i| \mod 2 = |P'_i| \mod 2, \tag{3.2}$$

because the path P_i and P'_i together form a simple cycle on the grid and any cycle that lies entirely on the grid has even length. Consider the sum,

$$\mathcal{S} = \sum_{i=1}^{k}(|P'_{i_1}| + |P'_{i_2}|). \tag{3.3}$$

Rearranging we get,

$$\mathcal{S} = |P'_{1_1}| + |P'_{k_2}| + \sum_{i=1}^{k-1}(|P'_{i_2}| + |P'_{(i+1)_1}|). \tag{3.4}$$

Since $|P'_{i_2}|$ and $|P'_{((i+1) \mod k)_1}|$ are equal, we have,

$$\mathcal{S} = 2|P'_{1_1}| + \sum_{i=1}^{k-1} 2|P'_{i_2}|. \tag{3.5}$$

Now combining Equations (3.1), (3.2) and (3.5), we have $\sum_{i=1}^{k} |P_i|$ is even and thus C is of even length.

If C is non-simple, then C can be decomposed into a collection of simple cycles $\{C_j\}$ such that $|C| = \sum_j C_j$. Now using the previous part we get that C has even length. □

4 New Upper Bounds

In this section we establish new upper bounds on the space complexity of certain matching problems on bipartite constant genus graphs, embedded on a 'genus g grid'.

We define K-ORI-GG to be the class of genus g graphs such that: for every $G \in$ K-ORI-GG, G is a grid graph embedded on a grid of size $2m \times 2m$. We assume that the distance between adjacent horizontal (and similarly vertical) vertices is of unit length. The entire boundary of the grid is divided into $4g$ segments, and each segment has even length, for some constant g. The $4g$ segments are labelled as $(S_1, S_2, S'_1, S'_2, \ldots S_{2i-1}, S_{2i}, S'_{2i-1}, S'_{2i}, \ldots, S_{2g-1}, S_{2g}, S'_{2g-1}, S'_{2g})$, together with a direction, namely, S_i is directed from left to right and S'_i is directed from right to left for each $i \in [2g]$. The jth vertex on a segment S_i is the jth vertex on the border of the grid, starting from the head of the segment S_i and going along the direction of the segment. Finally the segments S_i and S'_i are glued to each other for each $i \in [2g]$ in the same direction. In other words, the jth vertex on segment S_i is the same as the jth vertex on segment S'_i. Also there are no edges along the boundary of the grid.

Definition 6. If C is a cycle in G, we denote the circulation of C with respect to a weight function w as $circ_w(C)$. For any subset $E' \subseteq C$, $circ_w(E')$ is the value of the circulation restricted to the edges of E'.

Theorem 10 (Main Theorem). *There exists a logspace computable and polynomially bounded weight function W, such that for any graph $G \in$ K-ORI-GG and any cycle $C \in G$, $circ_W(C) \neq 0$.*

Theorem 11. *For a graph embedded on a constant genus surface,*

(a) DECISION-BPM *is in* SPL,

(b) SEARCH-BPM *is in* $\mathsf{FL}^{\mathsf{SPL}}$ *and*

(c) UNIQUE-BPM *is in* SPL.

Proof. As a result of Theorem 3, we can assume that our input graph $G \in$ K-ORI-GG. Using Theorem 10 and Lemma 1 we get a logspace computable weight function W, such that the minimum weight perfect matching in G with respect to W is unique. Moreover, for any subset $E' \subseteq E$, Theorem 10 is valid for the subgraph $G \setminus E'$ also, with respect to the same weight function W. Now (a) and (b) follows from Lemma 2. Checking for uniqueness can be done by first computing a perfect matching, then deleting an edge from the matching and rechecking to see if a perfect matching exists in the new graph. If it does, then G did not have a unique perfect matching, else it did. Note that Theorem 10 is valid for any graph formed by deletion of edges of G. $\square$

Theorem 10 also gives an alternative proof of directed graph reachability for constant genus graphs.

Theorem 12 ([BTV09, KV09]). *Directed graph reachability for constant genus graphs is in* UL.

The proof of Theorem 12 follows from Lemma 13 and [BTV09]. We adapt Lemma 13 from the journal version of [DKR08] (to appear in Theory of Computing Systems).

Lemma 13. *There exist a logspace computable weight function that assigns polynomially bounded weights to the edges of a directed graph such that: (a) the weights are skew symmetric, i.e., w(u,v) = - w(v,u), and (b) the sum of weights along any (simple) directed cycle is non-zero.*

Lemma 14. *In any class of graphs closed under the subdivision of edges,* Theorem 10 *implies the hypothesis of* Lemma 13.

Proof. Given an undirected graph G, construct a *bipartite* graph G' as follows: replace every undirected edge $\{u, v\}$ by a path $u - w - v$ of length two. Use Lemma 13 to assign weights to the edges of G'. Suppose that the weight assigned to the undirected edge $\{u, w\}$ in G' is a and the weight of $\{w, v\}$ is b. Let $\overrightarrow{G}$ denote the directed graph obtained from G by considering each undirected edge as two directed edges in opposite directions. Now we assign the weights to the edges of $\overrightarrow{G}$ as follows: directed edge (u, v) gets weight $a - b$; whereas the directed edge (v, u) will get weight $b - a$. The circulations of the cycles in G' being non-zero will translate into the sum of the edges along any cycle in the directed graph $\overrightarrow{G}$ being non-zero. $\square$

4.1 Proof of Main Theorem

Proof of Theorem 10. For a graph $G \in$ K-ORI-GG, we define W is a linear combination of the following $4g+1$ weight functions defined below. This is possible in logspace since g is constant.

Define $4g+1$ weight functions as follows:

- For each $i \in [2g]$,

$$w_i(e) = \begin{cases} 1 & \text{if } e \text{ lies on the segment } S_i \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

- For each $i \in [2g]$,

$$w'_i(e) = \begin{cases} j & \text{if } e \text{ lies on the segment } S_i \text{ at index } j \text{ from the head of } S_i \text{ and } j \text{ is odd} \\ -j & \text{if } e \text{ lies on the segment } S_i \text{ at index } j \text{ from the head of } S_i \text{ and } j \text{ is even} \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

-

$$w''(e) = \begin{cases} 0 & \text{if one end of } e \text{ lies on the boundary of the grid} \\ 0 & \text{if } e \text{ does not lie on the boundary and } e \text{ is a vertical edge} \\ (-1)^{i+j}(i+j-1) & \text{if } e \text{ is the } j\text{th horizontal edge from left, lying in row } i \\ & \text{from bottom, and not lying on the boundary} \end{cases} \tag{4.3}$$

Note that if e does not lie on the boundary of the grid then $w''(e)$ is same as the weight function defined in [DKR08].

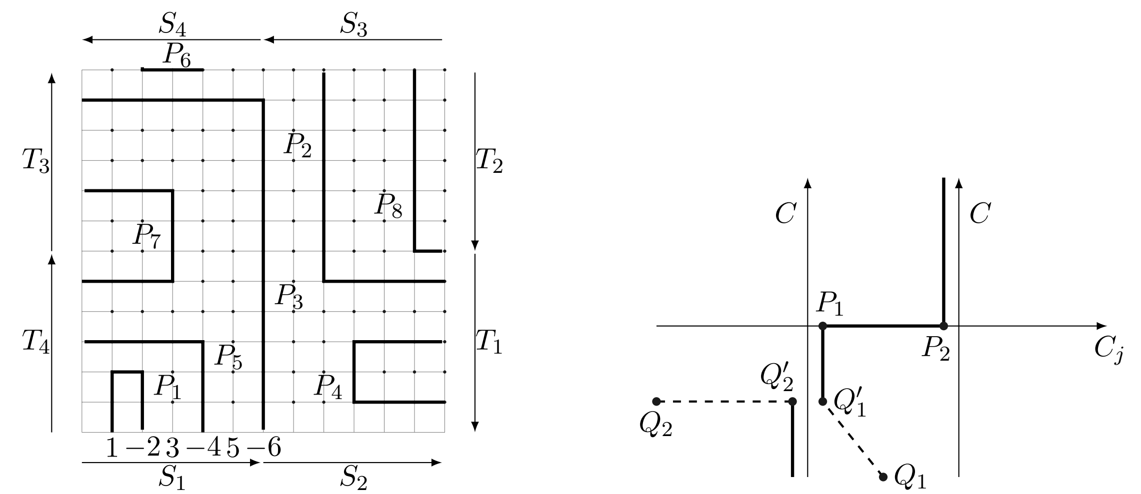
If C is a cycle in G, we denote the circulation of C with respect to a weight function w as $circ_w(C)$. For any subset $E' \subseteq C$, $circ_w(E')$ is the value of the circulation restricted to the edges of E'. An example of a cycle on a grid is given in Figure 10(a).

Let C be a simple cycle in G. If C is surface non-separating, then $circ_{w_i}(C) \neq 0$ for some i. If C is surface separating and crosses the boundary of the grid at some vertex v, then $circ_{w'_i}(C) \neq 0$ for i, such that v lies in the segment S_i. If C does not intersect any of the boundary segments, then C does not have any edge on the boundary since there are no edges along the boundary by definition of K-ORI-GG. Therefore $circ_{w''}(C) \neq 0$ by [DKR08].

Without loss of generality, assume C intersects segment S_1. Let E_1^C be the set of edges of C that intersect S_1. Note that $circ_{w_1}(C) = circ_{w_1}(E_1^C)$ (same thing holds for w'_1 as well. We can assume that $|E_1^C|$ is even since otherwise $circ_{w_1}(E_1^C)$ is odd and hence non-zero. By Lemma 16 it follows that the edges of E_1^C, alternate between going out and coming into the grid. Then using Lemma 17 we get that $circ_{w'_1}(E_1^C) \neq 0$ and thus $circ_{w'_1}(C) \neq 0$. (See below for Lemma 16 and 17) □

To establish Lemma 16 we use an argument (Lemma 15) from homology theory. For two cycles (directed or undirected) C_1 and C_2, let $I(C_1, C_2)$ denote the number of times C_1 and C_2 cross each other (that is one of them goes from the left to the right side of the other, or vice versa).

Next we adapt the following Lemma from Cabello and Mohar [CM07]. Here we assume we are given an orientable surface (Cabello and Mohar gives a proof for a graph on a surface).



(a) Example of a cycle on the grid that crosses each segment an even number of times with the weights w_1'

(b) Construction of a path from Q_1 to Q_2 in $\Gamma \setminus C$ (the dotted path is the shortest path between Q_1 and Q_1' (resp. between Q_2 and Q_2').

Figure 10:

Lemma 15 ([CM07])**.** *Given a genus g orientable, surface Γ, let $\mathcal{C} = \{C_i\}_{i\in[2g]}$ be a set of cycles that generate the first homology group $H_1(\Gamma)$. A cycle C in Γ in non-separating if and only if there is some cycle $C_i \in \mathcal{C}$ such that $I(C, C_i) \equiv 1 (\mod 2)$.*

Proof. Let $\tilde{C}$ be some cycle in Γ. We can write $\tilde{C} = \sum_{i\in[2g]} t_i C_i$ since $\mathcal{C}$ generates $H_1(\Gamma)$. Define $I_{\tilde{C}}(C) = \sum_{i\in[2g]} t_i I(C, C_i) (\mod 2)$. One can verify that $I_{\tilde{C}} : \mathcal{C}_1(\Gamma) \to \mathbb{Z}_2$ is a group homomorphism. Now since $\mathcal{B}_1(\Gamma)$ is a normal subgroup of $\mathcal{B}_1(\Gamma)$, $I_{\tilde{C}}$ induces a homomorphism from $H_1(\Gamma)$ to $\mathbb{Z}_2$.

Any cycle is separating if and only if it is homologous to the empty set. Therefore if C is separating, then $C \in \mathcal{B}_1(\Gamma)$ and thus every homomorphism from $H_1(\Gamma)$ to $\mathbb{Z}_2$ maps it to 0. Hence for every $i \in [2g]$, $I(C, C_i) \equiv I_{C_i}(C) = 0$.

Suppose C is non-separating. One can construct a cycle C' on Γ, that intersects C exactly once. Let $C' = \sum_{i\in[2g]} t_i' C_i$. Now $1 \equiv I_{C'}(C) \equiv \sum_{i\in[2g]} t_i' I(C, C_i) (\mod 2)$. This implies that there exists $i \in [2g]$ such that $I(C, C_i) \equiv 1 (\mod 2)$. □

Lemma 16. *Let C be a simple directed cycle on a genus g orientable surface Γ and let $\mathcal{C} = \{C_i\}_{i\in[2g]}$ be a system of $2g$ directed cycles on Γ, having exactly one point in common and $\Gamma \setminus \mathcal{C}$ is the fundamental polygon, say Γ'. If $I(C, C_i)$ is even for all $i \in [2g]$ then for all $j \in [2g]$, C alternates between going from left to right and from right to left of the cycle C_j in the direction of C_j (if C crosses C_j at all).*

Proof. Suppose there exists a $j \in [2g]$ such that C does not alternate being going from left to right and from right to left with respect to C_j. Thus if we consider the ordered set of points where C intersects C_j, ordered in the direction of C_j, there are two consecutive points (say P_1 and P_2) such that at both these points C crosses C_j in the same direction.

Let Q_1 and Q_2 be two points in $\Gamma \setminus C$. We will show that there exists a path in $\Gamma \setminus C$ between Q_1 and Q_2. Consider the shortest path from Q_1 to C. Let Q_1' be the point on this path that is as close to C as possible, without lying on C. Similarly define a point Q_2'

corresponding to Q_2. Note that it is sufficient for us to construct a path between Q_1' and Q_2' in $\Gamma \setminus C$. If both Q_1' and Q_2' locally lie on the same side of C, then we get a path from Q_1' to Q_2' not intersecting C, by traversing along the boundary of C. Now suppose Q_1' and Q_2' lie on opposite sides (w.l.o.g. assume that Q_1' lies on the right side) of C. From Q_1' start traversing the cycle until you reach cycle C_j (point P_1 in Figure 10(b)). Continue along cycle C_j towards the adjacent intersection point of C and C_j, going as close to C as possible, without intersecting it (point P_2 in Figure 10(b)). Essentially this corresponds to switching from one side of C to the other side without intersecting it. Next traverse along C to reach Q_2'. Thus we have a path from Q_1' to Q_2' in $\Gamma \setminus C$. We give an example of this traversal in Figure 10(b). This implies that C is non-separating.

It is well known that $\mathcal{C}$ forms a generating set of $H_1(\Gamma)$, the first homology group of the surface. Now from Lemma 15 it follows that $I(C, C_l) \equiv 1(\mod 2)$ for some $l \in [2g]$, which is a contradiction.

□

Lemma 17. *Let G be a graph in* K-ORI-GG *with C being a simple cycle in G and E_1^C being the set of edges of C that intersects segment S_1. Assume $|E_1^C|$ is even and the edges in E_1^C alternate between going out and coming into the grid. Let $i_1 < i_2 < \ldots < i_{2p-1} < i_{2p}$ be the distinct indices on S_1 where C intersects it. Then*

$$\left|circ_{w_1'}(E_1^C)\right| = \left|\sum_{k=1}^{p}(i_{2k} - i_{2k-1})\right|$$

and thus non-zero unless E_1^C is empty.

Proof. Let $e_j = (u_j, v_j)$ for $j \in [2p]$ be the $2p$ edges of G lying on the segment S_1. Assume without loss of generality that the vertices v_j's lie on S_1. Assign an orientation to C such that e_1 is directed from u_1 to v_1. Also assume that i_1 is even and the circulation gives a positive sign to the edge e_1. Therefore $circ_{w_1'}(\{e_1\}) = -i_1$.

Now consider any edge e_j such that j is even. By Lemma 16, the edge enters the segment S_1. Suppose i_j is odd. Then consider the following cycle C' formed by tracing C from u_j to u_1, without the edges e_1 and e_j and then moving along the segment S_1 back to u_j. Since i_j is odd therefore the latter part of C' has odd length. Note that C' need not be a simple cycle. By Lemma 9, $|C'|$ is even, therefore the part of C' from u_1 to u_j also has odd length. This implies that the circulation gives a positive sign to the edge e_j. Therefore, $circ_{w_1'}(\{e_j\}) = i_j$. Similarly, if i_j is odd, then the part of C' from u_1 to u_j will have even length. Thus the circulation gives a negative sign to the edge e_j and therefore $circ_{w_1'}(\{e_j\}) = -(-i_j) = i_j$.

If j is odd, the above argument can be applied to show that $circ_{w_1'}(\{e_j\}) = -i_j$. Therefore we have,

$$circ_{w_1'}(E_1^C) = \sum_{k=1}^{p}(i_{2k} - i_{2k-1}).$$

Now removing the assumptions at the beginning of this proof would show that the LHS and RHS of the above equation is true modulo absolute value as required. □

It is interesting to note here that similar method does not show that bipartite matching in non-orientable constant genus graphs is in SPL. The reason is that Lemma 16 crucially uses the fact that the surface is orientable. In fact, one can easily come with counterexample to the Lemma if the surface is non-orientable.

5 Reducing the non-orientable case to the orientable case

Let G be a bipartite graph embedded on a genus g non-orientable surface. As a result of Theorem 6 we can assume that we are given a combinatorial embedding (say Π) of G on a (non-orientable) polygonal schema, say $\Lambda(\Gamma)$, in the normal form with $2g'$ sides. (Here g' is a function of g.)

Let $Y = (X_1, X_2)$ be the cyclic ordering of the labels of the sides of $\Lambda(\Gamma)$, where X_2 is the 'orientable part' and X_1 is the 'non-orientable part'. More precisely, for the polygonal schema in the normal form, we have: X_1 is either (σ, σ) (thus corresponds to the projective plane) or it is $(\sigma, \tau, \bar{\sigma}, \tau)$ (thus corresponds to the Klein bottle). See Figure 11.

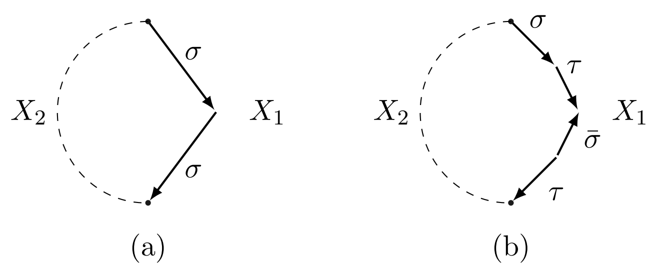


Figure 11: **(a)** $\Lambda(\Gamma)$ when the surface is a sum of an orientable surface and the projective plane. **(b)** $\Lambda(\Gamma)$ when the surface is a sum of an orientable surface and the Klein bottle

.

Now let G be a bipartite graph embedded on a non-orientable polygonal schema $\Lambda(\Gamma)$ with $2g'$ sides. We will construct a graph G' embedded on an *orientable* polygonal schema with $4g'-2$ sides such that G has a perfect matching iff G' has a perfect matching. Moreover, given a perfect matching in G' one can retrieve in logspace a perfect matching in G. This is illustrated in the following Theorem.

Theorem 18. *Let G be a bipartite graph given with its embedding on a non-orientable polygonal schema in normal form $\Lambda(\Gamma)$, with $2g'$ sides as above. One can construct in logspace, another graph G' together with its embedding on the polygonal schema of an orientable surface Γ' of genus $4g'-2$ such that: G has a perfect matching iff G' has a perfect matching. Moreover, given a perfect matching in G', one can construct in logspace a perfect matching in G.*

Proof. We first show the case when Γ is the sum of an orientable surface and a Klein bottle. Consider the polygonal schema formed by taking two copies of $\Lambda(\Gamma)$ and glueing the side τ of one copy with its partnered side τ of the other copy. We relabel the edge labelled σ in the second copy with some unused symbol δ to avoid confusion. The entire reduction is shown in Figure 12. Let G' be the resulting graph.

Note that the polygonal schema obtained as a result represents an orientable surface and has constantly many sides. Also every vertex and edge in G has exactly two copies in G' and G' is also bipartite. Let M be a matching in G. Let M' be the union of the edges of M from both the copies of G . Its easy to see that M' is a matching in G'. Now consider a matching M' in G'. The *projection* of M' to G gives a subgraph of G where every vertex has degree (counted with multiplicity) exactly two. Since G is bipartite, one can obtain a perfect matching within this subgraph.

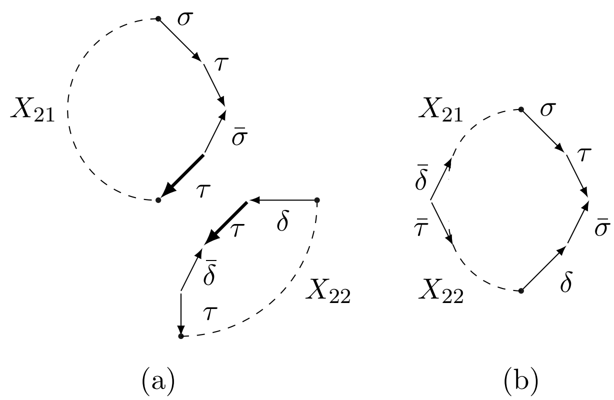


Figure 12: *Klein bottle.* **(a)** The two copies of $\Lambda(\Gamma)$ with the side that is being glued shown in dark. **(b)** Polygonal schema obtained after the glueing operation.

Now consider the case when Γ is the 'sum' of an orientable surface and a projective plane, i.e., following the notation above X_1 corresponds to the labels of a polygonal schema for the projective plane and X_2 corresponds to the labels of a polygonal schema of an orientable surface. Take two copies of $\Lambda(\Gamma)$, and glue σ of one copy with its partner σ in the other copy. We show this operation in Figure 13. The rest of the proof is similar to the Klein bottle

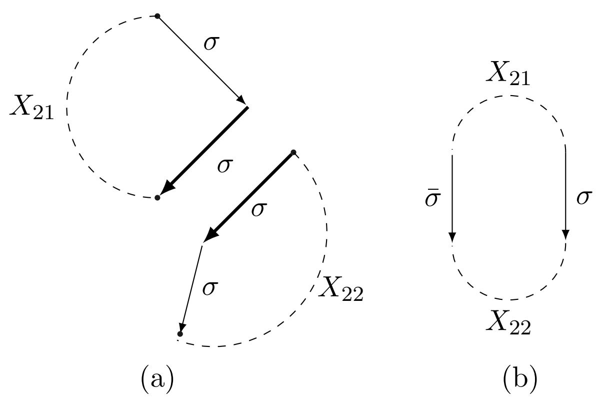


Figure 13: *Projective plane.* **(a)** The two copies of $\Lambda(\Gamma)$ with the two pair of sides that are being glued shown in dark. **(b)** Polygonal schema obtained after the glueing operation.

case. □

Thus we see that the non-orientable case can be reduced to the orientable case. The resulting polygonal schema need not be in the normal form. Once again we apply Theorem 6 to get a combinatorial embedding on a polygonal schema in the normal form.

Acknowledgment

The third author would like to thank Prof. Mark Brittenham from the Mathematics department at the University of Nebraska-Lincoln, for numerous discussions that they had and for providing valuable insight into topics in algebraic topology.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 1 edition, 2009.

[ABC+09] Eric Allender, David A. Mix Barrington, Tanmoy Chakraborty, Samir Datta, and Sambuddha Roy. Planar and grid graph reachability problems. *Theory Comput. Syst.*, 45(4):675–723, 2009.

[AM08] V. Arvind and Partha Mukhopadhyay. Derandomizing the isolation lemma and lower bounds for circuit size. In *Proceedings of RANDOM '08*, pages 276–289, 2008.

[ARZ99] Eric Allender, Klaus Reinhardt, and Shiyu Zhou. Isolation, matching, and counting: Uniform and nonuniform upper bounds. *Journal of Computer and System Sciences*, 59:164–181, 1999.

[Bra21] H. R. Brahana. Systems of circuits on two-dimensional manifolds. *The Annals of Mathematics*, 23(2):144–168, 1921.

[BTV09] Chris Bourke, Raghunath Tewari, and N. V. Vinodchandran. Directed planar reachability is in unambiguous log-space. *ACM Trans. Comput. Theory*, 1(1):1–17, 2009.

[CM07] Sergio Cabello and Bojan Mohar. Finding shortest non-separating and non-contractible cycles for topologically embedded graphs. *Discrete Comput. Geom.*, 37(2):213–235, 2007.

[DH07] Max Dehn and Poul Heegaard. Analysis situs. *Enzyklopädie der mathematischen Wissenschaften mit Einschluß ihrer Anwendungen*, III.AB(3):153–220, 1907.

[DKR08] Samir Datta, Raghav Kulkarni, and Sambuddha Roy. Deterministically isolating a perfect matching in bipartite planar graphs. In *25th International Symposium on Theoretical Aspects of Computer Science*, pages 229–240, 2008.

[GW96] Anna Gal and Avi Wigderson. Boolean complexity classes vs. their arithmetic analogs. *Random Structures and Algorithms*, 9:1–13, 1996.

[Hoa09] Thanh Minh Hoang. On the matching problem for special graph classes. In *Electronic Colloquium on Computational Complexity*, number TR09-091, 2009.

[KMV08] Raghav Kulkarni, Meena Mahajan, and Kasturi R. Varadarajan. Some perfect matchings and perfect half-integral matchings in NC. *Chicago Journal of Theoretical Computer Science*, 2008(4), September 2008.

[KV09] Jan Kynčl and Tomáš Vyskočil. Logspace reduction of directed reachability for bounded genus graphs to the planar case. In *Electronic Colloquium on Computational Complexity*, number TR09-050, 2009.

[Mas91] William S. Massey. *A Basic Course in Algebraic Topology*. Springer-Verlag, 1991.

[MT01] Bojan Mohar and Carsten Thomassen. *Graphs on Surfaces*. John Hopkins University Press, 2001.

[MVV87] Ketan Mulmuley, Umesh Vazirani, and Vijay Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7:105–113, 1987.

[RA00] Klaus Reinhardt and Eric Allender. Making nondeterminism unambiguous. *SIAM Journal of Computing*, 29:1118–1131, 2000. An earlier version appeared in FOCS 1997, pp. 244–253.

[Tho89] C. Thomassen. The graph genus problem is np-complete. *J. Algorithms*, 10(4):568–576, 1989.

[Vol99] Heribert Vollmer. *Introduction to Circuit Complexity - A Uniform Approach*. Springer-Verlag, 1999.

[VY90] Gert Vegter and Chee-Keng Yap. Computational complexity of combinatorial surfaces. In *Proceedings of the 6th Annual Symposium on Computational Geometry*, pages 102–111, 1990.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de