Electronic Colloquium on Computational Complexity, Report No. 35 (2007)

Parity Problems in Planar Graphs

Mark Braverman*
Dept. of Comp. Sci.
University of Toronto

Raghav Kulkarni
Dept. of Comp. Sci.
University of Chicago

Sambuddha Roy†
India Research Lab
IBM India Pvt. Ltd.

Abstract

We consider the problem of counting the number of spanning trees in planar graphs. We prove tight bounds on the complexity of the problem, both in general and especially in the modular *setting. We exhibit the problem to be complete for Logspace when the modulus is 2^k, for constant k. On the other hand, we show that for any other modulus and in the non-modular case, our problem is as hard in the planar case as for the case of arbitrary graphs. This completely settles the question regarding the complexity of modular computation of the number of spanning trees in planar graphs. The techniques used are algebraic topological that may be useful in many other problems involving planar or higher genus graphs – such as higher genus graph recognition in Logspace. Prior work in this area was reported by H. Shank and others who considered the problems, only in the simplest planar setting. Our results vastly extend theirs, while our methods rely heavily on algebraic-topology.*

In the spirit of counting problems modulo 2^k, we also exhibit a highly parallel $\oplus\mathbf{L}$ algorithm for finding the value of a Permanent modulo 2^k. Previously, the best known result in this direction was Valiant's result that this problem lies in **P**. *We also show that we can count the number of perfect matchings in an arbitrary graph in* **P**. *This extends Valiant's result for the Permanent, since the Permanent may be modeled as counting the number of perfect matchings in bipartite graphs.*

1 Introduction and Previous Work

Enumeration and counting problems are of paramount importance in both mathematics and computer science. In addition to being interesting on their own right, they give us fundamental insights as to the complexity of the decision problem underlying the counting problem, and at times the sophisticated methods employed to perform the counting lead to beautiful mathematics. Modular counting involves counting objects with a certain property *modulo* some number. Modular counting plays a significant role in complexity theory – a few instances are afforded by Toda's Theorem [18], and also by Valiant's result [19] stating that if the Permanent modulo 3 were tractable, then the class of unambiguous polynomial time (**UP**) would collapse to **P** – this last being unlikely since it would contradict widely believed cryptographic assumptions.

The upshot is that most enumeration problems are intractable, although some examples are known where the counting problem can be resolved in polynomial time. A few instances of the latter case occurring are as follows: counting the number of spanning trees in an arbitrary undirected graph [8] , counting the number of perfect matchings in planar undirected graphs [12, 17], counting the number of simultaneous source to sink paths in a directed acyclic graph with n sources and n sinks [7]. Valiant in his holographic algorithms paradigm borrows the result about counting perfect matchings in planar graphs in a nontrivial way to give instances of several other problems where the counting version lies in polynomial time.

It has been observed that many of the counting problems which lie in polynomial time reduce to a computation of the determinant of a suitably defined matrix. Determinant computation effectively captures the complexity of the parallel class **GapL**, and it contains the class of nondeterministic logspace, **NL** (which in turn contains **L**). It is also closely related to the class #**L**, which is the natural counting class that relates to **L** in the same way as #**P** relates to **P**.

Let us take this opportunity to describe known results about a close relative of the Determinant, namely, the Permanent. The permanent problem was shown to be #**P**-hard by Valiant in his seminal paper [19]. Valiant also showed how the Permanent modulo (small)

[*] partially supported by an NSERC postgraduate scholarship

[†] supported in part by NSF Grant CCF-0514155, while at Rutgers University

ISSN 1433-8092

powers of 2 is solvable in **P** – but with no further bounds on the parallel complexity of this last problem.

We will consider two (modular) counting problems in this paper, one of which reduces to a determinant computation in arbitrary graphs, and one that reduces to a permanent computation.

First, let us give an instance of a situation where a counting problem reduces to the computation of the determinant of a suitably defined matrix. The classical Matrix Tree Theorem [8] by Kirchoff (1847) states that the number of spanning trees in a graph can be found by computing the determinant of (the minor of) a matrix, namely the *Laplacian* of the graph. The Laplacian matrix of a graph is easily derived from the adjacency matrix of a graph, and appears ubiquitously in expanders, connectivity computations [15], etc. We can show that computation of the number of spanning trees in a graph has the same complexity as that of the determinant. Given this, we may thereby ask as to whether this complexity reduces for specific graph classes, say for instance, the class of planar graphs. Does the complexity of modular counting reduce thereby? Somewhat surprisingly, the answer depends on the modulus.

Secondly, let us consider the problem of counting the number of perfect matchings in a graph. If the graph is bipartite, it is easy to see that the permanent of its adjacency matrix exactly captures the (square of the) number of perfect matchings in the graph, and thus, counting the number of perfect matchings in a bipartite graph is also #**P**-hard [19]. Valiant proved that finding the permanent of a matrix modulo small powers of 2 can be done in **P**. We extend this result in two respects. First, we show that the permanent modulo constant powers of two can be computed in $\oplus$**L**, thus settling the complexity of the problem. We then consider the problem of finding the number of perfect matchings in an *arbitrary* (not necessarily bipartite) graph modulo small powers of 2. To the best of our knowledge, there is no obvious way to model the number of matchings in an arbitrary graph as the permanent of a suitable matrix. We show that this problem can be solved in **P**.

In light of the above, this paper considers the following three problems:

- computation of the number of spanning trees in planar graphs modulo 2^k;
- computation of the permanent of an integer matrix modulo 2^k;
- computation of the number of perfect matchings in arbitrary graphs modulo 2^k.

In the mid-70s, H. Shank [16] formulated the theory of so-called left-right cycles in planar graphs (this concept will be defined later in the paper). There is a connection between left-right cycles in planar graphs and the Laplacians of planar graphs (and thereby to modular counting of spanning trees) that is implicit in [8]. To the best of our knowledge, this connection has not been made explicit before this paper. For instance, Eppstein [5] gives combinatorial and algebraic characterizations for graphs with an even number of spanning trees – but the connection to left-right cycles is not observed therein.

We start by giving our own proof for the basic connection between left-right cycles and parity of the number of spanning trees in planar graphs in Section 2, as an illustration of the basic technique we build upon in Section 3. Henceforth, we make modular counting our principal focus, and having resolved the complexity of finding out the parity of the number of spanning trees in planar graphs in **L**, we move on to higher powers of 2, and to other prime moduli. We prove that we can find out the number of spanning trees in planar graphs modulo 2^k (for constant k) in **L**. On the other hand, we are able to prove tight lower bounds for the same computation modulo primes other than 2. This is a situation common in computer science, and especially in planar graphs where duality may make circumstances simpler for modulus 2 compared to other moduli.

Next, we consider another counting problem in graphs, namely the number of perfect matchings. We consider the number of perfect matchings in *bipartite* graphs, which can be modeled as the permanent of a suitable matrix. This enables us to consider just the permanent of matrices. While Valiant [19] gives a polynomial time algorithm for computing the permanent modulo 2^k, for a constant k, his method is akin to Gaussian Elimination and does not have an obvious parallelization. In this paper, we exhibit the complexity of computing the permanent modulo 2^k in a highly parallel class, namely $\oplus$**L**. In fact, $\oplus$**L**-hardness of the permanent modulo 2^k proves our algorithm to be optimal. We also use the techniques for the above to prove that the number of matchings in arbitrary graphs modulo 2^k (for constant k) can be found in polynomial time.

It should be mentioned that the Permanent problem enjoys a special status with regard to its *easiness* modulo 2^k. Let #SAT denote the problem of counting the number of satisfying assignments of a formula. It is known that #SAT mod 2 is $\oplus$**P**-hard; note that $\oplus$**P** is a relatively large class – the whole of the polynomial hierarchy (**PH**) randomly reduces to $\oplus$**P** [18]!

Main results and structure of the paper

We start by giving the basic definitions and presenting our basic techniques for modular counting of spanning trees in planar graphs in Section 2. In Section 3, we expand on these techniques using tools from algebraic topology to prove our main result that counting spanning trees in planar graphs modulo 2^k (for constant k) can be done in $\mathbf{L}$.

Theorem 17 *Given an integer k and a planar graph G, the number of spanning trees $\tau(G) \bmod 2^k$ can be computed in space $O(k^2 \log n)$.*

After this, we look at other moduli and prove tight hardness results for prime moduli $p > 2$ in Section 4.

Theorem 22 *For prime $p > 2$, finding out whether $\tau(G) \equiv 0 \bmod p$ for a planar graph G is complete for* $\mathrm{Mod}_p\mathbf{L}$.

Denote the number of spanning trees in a graph by τ. The main results about the complexity of computing τ are summarized in the table below.

Problem	**General G**	**Planar G**
$\tau(G)$	DET	DET
$\tau(G)$ modulo prime $p > 2$	$\mathrm{Mod}_p\mathbf{L}$	$\mathrm{Mod}_p\mathbf{L}$
$\tau(G)$ modulo 2^k	$\oplus\mathbf{L}$	$\mathbf{L}$

In Section 5, we consider another counting problem modulo 2^k, we prove that

Theorem 27 *Finding out the* PERMANENT *of a matrix modulo 2^k (for constant k) is complete for* $\oplus\mathbf{L}$.

Another way of stating the above is that we can find the last k bits of the permanent of a matrix (for constant k) in $\oplus\mathbf{L}$.

The same techniques also prove the following:

Theorem 32 Finding out the number of perfect matchings in a graph G modulo 2^k (for constant k) can be done in $\mathbf{P}$.

We end with some conclusions and open problems in Section 7.

Technical Contributions

Given that counting the number of spanning trees in a planar graph modulo 2 is in $\mathbf{L}$, it is perhaps natural to conjecture the same modulo 2^k – for instance, it is known that computing the determinant of a matrix modulo 2^k is no harder than computing it modulo 2 [1]. The question of modular counting of the spanning trees in planar graphs appears to be of surprising difficulty – and seems to require the use of algebraic topological techniques. An interesting feature is that to compute the number of spanning trees in a *planar* graph modulo 2^k, one has to take recourse, in the current proof, to *higher* genus realms! The proof uses a variety of techniques from algebraic topology, such as universal covers and homology groups. We believe techniques developed here may be applicable to a variety of other problems on small genus graphs, and maybe even – as in this case – on planar graphs.

We also show how another modular counting problem, namely the number of matchings in arbitrary bipartite graphs modulo 2^k (which is essentially the permanent of a suitable matrix modulo 2^k) is complete for $\oplus\mathbf{L}$, using LUP-decompositions. While the proof outlined in [1] for a similar question about the determinant seems to involve some ad hoc techniques, our proof for permanent modulo 2^k gives a more uniform approach to such problems – in particular we get a new, arguably more transparent proof for the result that determinants of matrices modulo 2^k are computable in $\oplus\mathbf{L}$.

2 Definitions and Basic Techniques

For definitions of logspace and related complexity classes, we refer the reader to [1].

In the following, we will use linear algebra over finite fields, mostly $\mathbb{Z}_p$ for prime p. For definitions of rank, kernel, dimension, we refer the reader to any linear algebra text; see [11]. For definitions of planar graphs and their duals, spanning trees, refer to any standard graph theory text see [8, 3].

Given a continuous closed curve C in the plane, and a point P not lying on C, we can define a winding number of C with respect to P: it is informally the number of times the curve C *winds* around the point P. This number is called the *winding number* of C with respect to P. For a formal definition, refer to any text in algebraic topology, say [6, 9].

We denote the (*geometric*) dual of a planar graph G by G^*. We denote the number of spanning trees in a graph G by $\tau(G)$. The adjacency matrix of the graph will be denoted by $A(G)$, and the **Laplacian** matrix of a graph G (denoted by $L(G)$) is defined as the matrix $L(G) = D(G) - A(G)$, where $D(G)$ is a diagonal matrix consisting of the degree of vertex v_i of the graph G in its ii^{th} entry. In this paper we will be dealing mostly with connected graphs G.

For instance, the Laplacian matrix of the complete graph on three vertices, K_3 is

$$\begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix}$$

NOTE : We will allow multiple edges in the graphs we consider in this paper, so for instance, the Laplacian matrix may have off-diagonal entries that are not 0 or -1.

The Laplacian of a graph has several other remarkable properties, for instance the Kirchoff's Matrix Tree Theorem:

Theorem 1. *Given the Laplacian matrix $L(G)$ of a graph G, the number of spanning trees $\tau(G)$ in G equals the determinant of* any *minor of $L(G)$.*

We now proceed with the definition of *left-right cycles* [8].

Definition 1. *Let us consider a special kind of walk in a planar graph G. View each vertex of G as a small disk, and each edge as a thin strip. Since each edge is a thin strip, it has two distinct sides and we can visualize traveling along the side of an edge. Select a starting point on the graph where the side of a strip meets the boundary of a disk. Let us form triples (v, e, s) where v is a vertex, e is an edge, and s is a side of the edge. We call such a (v, e, s) triple a* flag. *From there, walk along the side of the edge crossing to the opposite side of the edge when you reach the point on the edge halfway between its endpoints. On reaching the neighboring vertex, walk around the boundary of the disk representing the vertex, leaving the vertex along the side of the edge lying in the* same *face as the side of the edge you have just arrived on. Extend the walk by using the same rules of negotiating edges and vertices. A* **left-right walk** *is the alternating sequence of vertices and edges encountered during such a walk, together with the starting flag.*

A **closed left-right walk** *is a left-right walk that starts and ends at the same flag. A* **left-right cycle** *is an equivalence class of closed left-right walks under rotation and reversal. Thus, in a left-right cycle, the cyclic order of the vertices and edges is important and which sides of the edges are used is important, but the direction and the starting vertex are not.*

Let $c(G)$ denote the number of left-right cycles in a graph G.

See Figure 1(a) for an illustration. One fact worth noting is that the underlying sequence of the vertices and edges in a left-right walk is a walk in the usual sense, but distinct left-right walks may have the same underlying walk if they start at flags on opposite sides of the same edge. Also, it can be seen that the number of left-right cycles is independent of the embedding of the planar graph G. Having defined left-right cycles for planar graphs, we see that we can extend the definition to *any* graph embedded on a surface.

Throughout this paper, when we consider equations such as $Lx = 0$ over $\mathbb{Z}_2$, for L being the Laplacian of a graph G, we will view a solution vector x as a $0-1$ *weighting* or *labeling* of the vertices of G.

From Theorem 17.3.5 and Lemma 14.15.3 of [8], it follows that:

Theorem 2. *Given a planar graph G, the number of left-right cycles in G is exactly equal to the co-rank of the Laplacian L of G (over $\mathbb{Z}_2$). In fact, each left-right cycle C corresponds to an element in $\{0,1\}^{|V(G)|}$ which is a* basis *element of the kernel of the Laplacian as follows:*

- *Considering a specific left-right cycle C, we have to give labels to every vertex v of G: Given C as a closed curve in the plane, which* winds *around the vertices of G, find the* winding number *of C with respect to a vertex v. The parity of this winding number is the label we give to vertex v.*

By the above, we thereby get a vector $x \in \{0,1\}^{|V(G)|}$, and this is a basis element of the kernel of L.

Defining a vector of labels x thus, corresponding to a left-right cycle C, we say that C *realizes* x. Given a vector x, and a collection of left-right cycles $\mathcal{C} = C_1, C_2, \cdots, C_r$, we say that $\mathcal{C}$ *realizes* x if there exist $x_1, x_2, \cdots, x_r$ such that $x = \Sigma_1^r x_i$ and C_i realizes x_i.

We give our own proof of the above theorem, which we is extended in Section 3 to obtain new results. The proof will follow from two claims.

Claim 3. *For every left-right cycle, C, the labeling given to the vertices v of G via the winding numbers as in the statement of the theorem is a solution to $Lx = 0$ over $\mathbb{Z}_2$ (where L is the Laplacian matrix of G). Hence it follows that every collection of left-right cycles $\mathcal{C}$ gives a solution to $Lx = 0$.*

Proof. Denote the set of vertices that get label 1 via the winding numbers by A and the set of vertices that get label 0 by B. We need to show that for every $v \in A$, the number of neighbors w of v that belong to B is even; also that for every $v \in B$, the number of neighbors w of v that belong to A is even – this being a restatement of $Lx = 0 \bmod 2$.

Consider the vertex v and let the edges incident on v be $e_1, e_2, \cdots, e_d$ where d is the degree of v in G. Let these edges also be ordered according to the planar

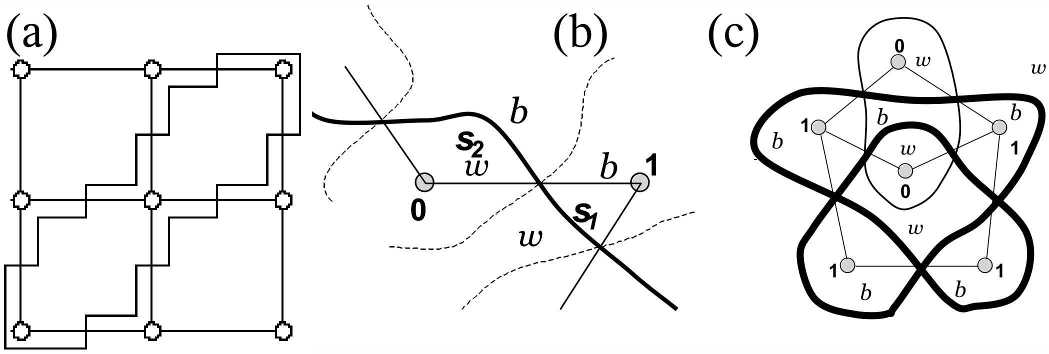

Figure 1. A left-right cycle and consistent colorings

layout of G in the neighborhood of v. Now consider the left-right cycle C, and we observe that any time the curve C crosses an edge e_i only once, the two endpoints of the edge e_i (one of them being v) get different winding numbers $(\bmod 2)$ and since v belongs to A (by assumption) the other endpoint belongs to B. So we are left to argue that the number of edges e_i which C crosses only once is even.

This last is now obvious once we note that whenever the curve C *approaches* v via some edge e_i it has to leave via some other edge e_j (j may equal i). Hence, the total number of (e_i, C) incidences is even. These incidences can be counted differently as the number of edges e_i which are crossed singly by C and twice the number of edges e_j which are crossed twice by C (no edge is crossed more than twice by any left-right cycle). So the number of interest, the number of edges e_i that are crossed singly by C is even. □

The other direction of the proof reads

Claim 4. *For every solution x of $Lx = 0$, there is a collection of left-right cycles $\mathcal{C}$ that realizes it.*

Proof. Given that x is a solution to $Lx = 0$, we know that for each vertex v in G which get label 1, the number of neighbors of v which get label 0 is even; likewise, for every vertex v which gets label 0, the number of neighbors of v which get label 1 is even. Let us define $x(v)$ to be the label that vertex v receives under the labeling x.

Given an edge e of the graph G endowed with the labeling x, call e *monochromatic* under labeling x if the two endpoints of e receive the same value under the labeling x, otherwise call e *bichromatic.*

Also if two vertices get labels 0 and 1 in a labeling x, we will refer to them as having *opposite* labels.

Let us take some embedding of the graph G on the plane, and draw all the left-right cycles. Each edge of G is crossed twice by this collection (maybe even by one left-right cycle).

The left-right cycles decompose the plane into *regions.* Each vertex of G belongs to some region; some regions do not contain any vertices and are enclosed entirely in some face of G. We call the regions that contain vertices *vertex regions* and the other regions *face regions.* There are as many vertex regions as there are vertices, and as many face regions as there are faces. We color the vertex region of a vertex v in black if $x(v) = 1$ and in white otherwise. We color the infinite (face) region white. We color two adjacent faces the same color if any of the edges that separate them is monochromatic, and different colors if any of these edges is bichromatic. If this coloring procedure is possible without any inconsistencies, we would consider each *segment* and consider the XOR of the colors of the two regions adjacent to it (one vertex region and one face region). We would include such a segment in the collection of left-right cycles that we are trying to construct from x, only if the XOR is 1. For brevity, we call this collection of segments $\mathcal{S}$.

We have to prove two things:

1. The coloring in the procedure does not lead to any inconsistencies.
2. Given a consistent coloring, we can extract out a collection of left-right cycles by the latter part of the procedure. In other words, $\mathcal{S}$ forms a disjoint collection of left-right cycles. Furthermore, the vertices v for which the winding number of $\mathcal{S}$ around v is odd, are exactly the ones for which $x(v) = 1$.

A consistent coloring is illustrated in Figure 1(c).

First, we prove the second item: what we need to prove is that if a segment s_1 of a left-right cycle C is included in $\mathcal{S}$, then the segment s_2 on C following s_1 is also in $\mathcal{S}$. This would ensure that the whole of

C is in $\mathcal{S}$. This is easily done by considering cases. We only consider the case of a bichromatic edge; the monochromatic case is similar. Suppose edge $e = (a, b)$ is such that a gets label 1, b gets label 0. Then by the procedure, the vertex regions corresponding to a and b get colors black and white respectively. Suppose that, s_1 and s_2 are segments of some left-right cycle which crosses e as in Figure 1(b). Then clearly the face region bordering s_1 has to be colored white (or else s_1 would not belong to $\mathcal{S}$). But then the procedure outlined above implies that the face region bordering s_2 has to be colored black, so that s_2 also belongs to $\mathcal{S}$. It is not hard to see that by the construction $x(v) = 1$ iff $\mathcal{S}$ has an odd winding number around v: $\mathcal{S}$ will cross a monochromatic edge either 0 or 2 times, and any other edge exactly once.

Now we prove the first item. If we are unable to color the face regions consistently, it implies that there is a simple closed walk γ along which the inconsistency occurs. In other words, γ crosses an odd number of bichromatic edges, and thus the color of the face is supposed to change an odd number of times along the closed curve γ.

Suppose such a γ existed. Let I be the set of vertices which are inside the region enclosed by γ. Consider the bichromatic edges that are crossed by γ. This number is supposed to be odd. But the number of such bichromatic edges (mod2) can also be summed up as: $\Sigma_{v \in I} \#\{\text{vertices of opposite labels neighboring } v\}$ and this is 0 mod 2 since we assumed that x is a solution to $Lx = 0$ and thus every v has an even number of neighbors of opposite color. This implies that a contradiction cannot occur. □

This completes the proof of Theorem 2. As a corollary, Theorem 2 yields:

Corollary 5. *[8] Given a planar graph G, the number of spanning trees is odd iff there is exactly one left-right cycle in G.*

We also record the following:

Corollary 6. *Given a planar graph G, if matrix B is a minor of the Laplacian $L(G)$ of G, then the co-rank of B is exactly equal to the number of left-right cycles in G minus 1.*

3 Computing the number of spanning trees modulo 2^k

In this section we generalize the construction to compute for a given planar graph G, the value of $\tau(G)$ modulo 2^k for a constant k in **L**. We first show how to determine whether $\tau(G)$ is divisible by 2^k. The strategy is to reduce the problem to the problem of computing the parity of $\tau(G')$ for a graph G' embedded into a constant genus surface.

3.1 Background: surfaces and homology groups

We make use of some basic facts about genus g surfaces $\mathcal{S}$ and their first homology group modulo 2, $H_1(\mathcal{S})_2$. A comprehensive study of the surfaces and their properties can be found in any introductory topology text, such as [6, 14, 9]. We concentrate on genus g orientable surfaces. For any g, such a surface $\mathcal{S}_g$ is just a sphere with g "handles". In particular, the sphere is a genus 0 surface and the torus is a genus 1 surface.

One way to view a genus g surface is by looking at it as a polygon with $4g$ edges that are glued to each other is a certain way. This gluing is usually defined by putting letters on the edges so that each letter appears twice. The surface is obtained by gluing the corresponding letters with an appropriate direction. The converse is also partially true: if we take a polygon and glue its edges in pairs in any fashion, there are very few possible outcomes.

Theorem 7. *(Theorem 77.5 in [14]) Let X be the quotient space obtained from a polygonal region in the plane by pasting its edges together in pairs. Then X is homeomorphic either to the sphere S^2, to the n-fold torus T_n, or to the m-fold projective plane P_m.*

It can be seen that if the edges that are pasted to each other are always facing in opposite directions on the polygon then the surface is orientable, and the resulting surface cannot be a projective plane, and will have to be a genus g orientable surface. We will use this fact later in the section. For our purpose, one can present a genus g surface $\mathcal{S}_g$ as a gluing of finitely many triangles. A closed curve γ on $\mathcal{S}_g$ is just a closed polygon on the surface, or a collection of several such polygons. Since our analysis is carried modulo 2, we are not concerned with the direction of the curves in γ, because a "positive direction" (+1) is the same as the "opposite direction" (−1).

For a genus g surface $\mathcal{S}_g$, its homology group $H_1(\mathcal{S})_2$ is isomorphic to $\mathbb{Z}_2^{2g}$. Informally, for any curve, or collection of curves γ in $\mathcal{S}_g$ there is a corresponding element $h(\gamma) = (x_1, x_2, \ldots, x_{2g}) \in H_1(\mathcal{S})_2 \cong \mathbb{Z}_2^{2g}$. The x_i's can be thought of as the mod 2 "winding" numbers of γ around the $2g$ different curves $\beta_1, \ldots, \beta_{2g}$ in $\mathcal{S}_g$. We say that a curve γ is simple if the set of points covered by γ more than once is discrete. We will use the following properties of the homology group:

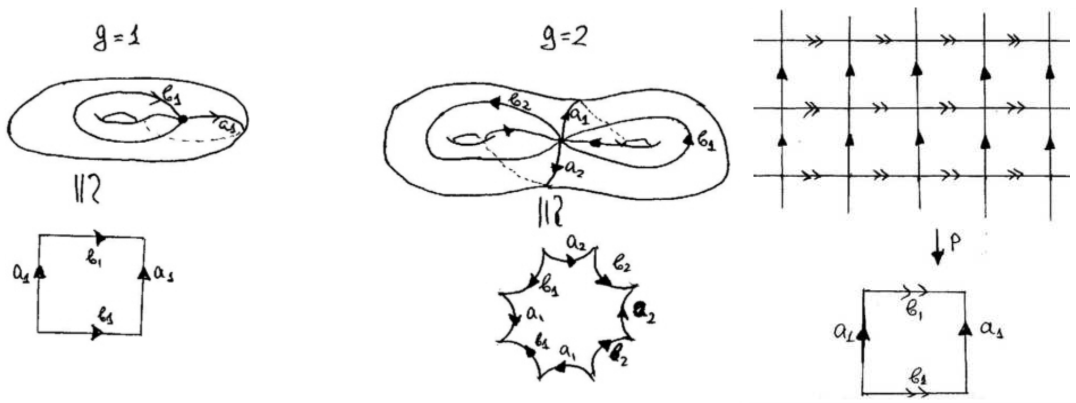

Figure 2. Examples of genus 1 and genus 2 tori (left) and of the universal cover of the torus (right)

Theorem 8. *1. For two collections of curves γ_1 and γ_2, if $\gamma = \gamma_1 \cup \gamma_2$ then $h(\gamma) = h(\gamma_1) + h(\gamma_2)$;*

2. for a simple γ, $h(\gamma) = 0$ if and only if there is a subregion A of $\mathcal{S}$ such that γ is the boundary of A, that is, the points covered by γ are exactly ∂A.

Theorem 8 provides us with an algorithmic tool for checking whether a given simple collection of curves γ has homology 0. This is done by checking whether the graph of faces which are obtained by the subdivision of γ on $\mathcal{S}_g$ is 2-colorable in black and white. In such a coloring, the black faces exactly correspond to the set A from Theorem 8.

3.2 The surface $\mathcal{S}_g$ and its universal cover

As mentioned earlier, one standard description of the surface $\mathcal{S}_g$ is by a $4g$-gon with gluing performed on its edges in the following order $a_1 b_1 a_1^{-1} b_1^{-1} a_2 b_2 a_2^{-1} b_2^{-1} \dots\ a_g b_g a_g^{-1} b_g^{-1}$. That is, the first edge is glued with the reverse third edge, the second edge is glued with the reverse fourth edge etc. Presentations of $\mathcal{S}_1$ (the torus) and $\mathcal{S}_2$ can be seen on Fig. 2. Note that the edges $a_1, b_1, \dots, a_g, b_g$ correspond to $2g$ curves on the surface. These curves are called the *generators* of $\mathcal{S}_g$. If these curves are removed, we get the original $4g$-gon.

For any surface $\mathcal{S}_g$ there is a map $p : \mathbb{R}^2 \to \mathcal{S}_g$ called the *universal cover* of $\mathcal{S}_g$. Every point x in $\mathcal{S}_g$ has infinitely many preimages $\widetilde{x}$ under p. These preimages are called *lifts*. For any such $\widetilde{x}$, p is a local homeomorphism between a neighborhood of $\widetilde{x}$ and a neighborhood of x. Furthermore, for any two preimages $\widetilde{x}_1$ and $\widetilde{x}_2$ of x, there is a unique *deck transformation* t such that $t(\widetilde{x}_1) = \widetilde{x}_2$ and $p \circ t = p$. The universal cover of $\mathcal{S}_g$ can be viewed as an infinite lamination of $\mathbb{R}^2$ with $4g$-gons such that every two neighbors share exactly one of the edges. This is illustrated on Fig. 2 (right).

Finally, we define the following operation that turns a genus g surface into a genus $2g-1$ surface.

Definition 2. *For a genus g surface T, and for a function $f : \{a_1, b_1, \dots, a_g, b_g\} \to \{0, 1\}$, the* doubling *of T by f, $T\#_f T$, or T^f in short, is defined as follows. If $f \equiv 0$ then $T^f := T$. Otherwise, consider the first generator on which f is not 0. Without loss of generality suppose that $f(a_i) = 1$ for some i. Consider two copies of the $4g$-gon of T. Denote them by T_1 and T_2. We glue a_i in T_1 to a_i^{-1} in T_2, thus obtaining a $8g-2$-gon T'. We then proceed by gluing the rest of the edges of T' as follows. For an edge x_j in T_1, if $f(x_j) = 0$, then x_j is glued to x_j^{-1} in T_1. If $f(x_j) = 1$, then it is glued to x_j^{-1} in T_2. The gluing is done similarly for x_j in T_2.*

It is not hard to see that T^f is well defined. By Theorem 7, T_f is a surface, and since it is orientable (we always glue opposite facing edges), T_f must be a genus k surface for some k. We can use Euler's Characteristic to compute k. We know that

$$2 - 2k = \chi(T^f) = F - E + V = 2 - 4g + 2 = 4 - 4g.$$

Thus $k = 2g - 1$. A sample construction of T^f is illustrated on Fig. 3.

3.3 Solving linear equations on a surface

We are now ready to prove the main technical lemma of the section.

Lemma 9. *For any g and k and a graph G embedded in a genus g surface $T \cong \mathcal{S}_g$, there is a machine that uses $O(\log n + g + k \log(k+g))$ space and either*

1. finds vectors $v_1, \dots, v_j$ spanning $\ker L(G)$, with $j \leq k$; or

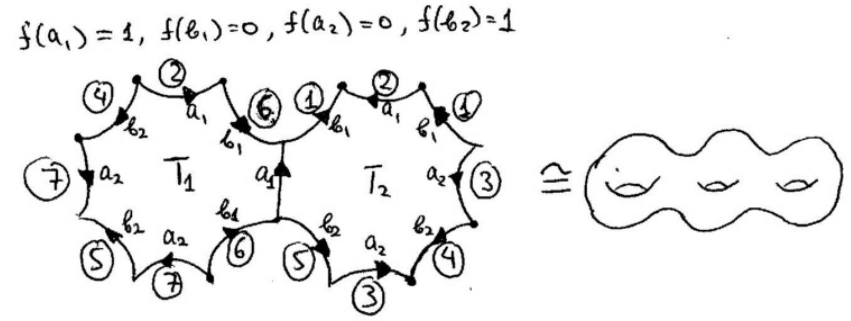

Figure 3. An example of T^f where $g = 2$; the resulting surface is isomorphic to $\mathcal{S}^3$

2. outputs "$\dim\ker L(G) > k$*".*

The remainder of current subsection is dedicated to proving Lemma 9. First, we give an algorithm and then prove that it works.

Let $X = \{x_1, x_2, \ldots, x_{2g}\}$ be generator curves on T. For each of the 2^{2g} functions $f : X \to \{0,1\}$ we consider the surface T^f. If $f \neq 0$, then there is a natural $2n$-vertex graph G^f in T^f obtained by taking the union of the two copies of G such that the edges are connected according to the new gluing in T^f. The algorithm proceeds as follows:

1. For all possible $f : X \to \{0,1\}$, compute all the left-right curves in G^f embedded into T^f;

2. let $A(f)$ be the collection of the left-right curves in G^f;

3. if $|A(f)| > 4g + 2k$, return "$\dim\ker L(G) > k$";

4. otherwise, try all the possible $2^{|A(f)|}$ combinations of curves in $A(f)$;

5. for each combination a of elements in $A(f)$ check whether there is a 2-coloring of the vertices of G^f such that vertices separated by a curve are colored in different colors; denote the set of vertices colored 1 by b_a; b_a can naturally be viewed as a vector in $\{0,1\}^{V(G^f)}$;

6. let $B(f)$ be a collection of all such vectors; note that $|B(f)| \leq 2^{|A(f)|}$;

7. if $f = 0$, let $C(f) = B(f)$, otherwise there is a natural way to view vectors in $B(f)$ as vectors in $\{0,1\}^{V(G)+V(G)}$, as $V(G^f)$ consists of two copies of $V(G)$; let
$$C(f) = \{v \ : \ (v,v) \in B(f)\};$$

8. $\bigcup_f C(f)$ spans $\ker L(G)$, a basis $v_1, \ldots, v_j$ can be found in space $O(\log n + k\log(k+g) + g)$ using Gaussian elimination.

All steps except step 8 take $O(\log n + k + g)$ space, because there are 2^{2g} possible f's and we exit if $|A(f)| > 4g + 2k$. It remains to see that the algorithm is correct.

Claim 10. *If for some f, $|A(f)| > 4g + 2k$, then* $\dim\ker L(G) > k$.

Proof. We first deal with the case when $f \neq 0$. Suppose there are $a = |A(f)| > 4g + 2k$ left-right curves of G^f in T^f. Denote the curves by $\gamma_1, \gamma_2, \ldots, \gamma_a$. Each of the curves corresponds to an element of the homology group $H_1(T^f)_2 \cong \mathbb{Z}_2^{4g-2}$. For a collection of curves $\beta_1, \ldots, \beta_d$ we denote by $span\{\beta_1, \ldots, \beta_d\}$ the set of all 2^d possible sums from the set $\{\beta_1, \ldots, \beta_d\}$. For some $\ell \leq 4g-2$ there is a collection of ℓ γ's such that the subgroup of $H_1(T^f)_2$ they span is equal to the subgroup of $H_1(T^f)_2$ all the γ's span. Without loss of generality we say that those are $\gamma_1, \ldots, \gamma_\ell$.

Any element in the span of $B = \{\gamma_{\ell+1}, \ldots, \gamma_a\}$ corresponds to an element of $H_1(T^f)_2$ that is also spanned by some elements of $\{\gamma_1, \ldots, \gamma_\ell\}$. Thus any element γ in the span of B can be completed to an element γ' in the span of $A(f)$ that corresponds to 0 in $H_1(T^f)_2$. Note that $|B| = a - \ell > 2k+2$. Each such γ' introduces a subdivision of the surface T^f and also of the graph G^f. Such a subdivision corresponds to an element of $\ker L(G^f)$. It may be the zero element only if no curves are present. Thus there are at least $2^{a-\ell}$ different elements in $\ker L(G^f)$, and $\dim\ker L(G^f) \geq a - \ell > 2k+2$.

Observe that if $(s_1, s_2) \in \ker L(G^f)$ then $s_1 + s_2$ is in $\ker L(G)$. Also if $(s,s) \in \ker L(G^f)$, then $s \in \ker L(G)$. Consider the operator $M : \ker L(G^f) \to \{0,1\}^{V(G)}$, $(s_1, s_2) \mapsto s_1 + s_2$. Then $Im(M) \subseteq \ker L(G)$, and thus $\dim Im(M) \leq \dim\ker L(G)$. On the other hand, $\ker M$ consists of elements of the form $(s,s) \in \ker L(G^f)$, and hence $\dim\ker(M) \leq \dim\ker L(G)$. Together, we obtain

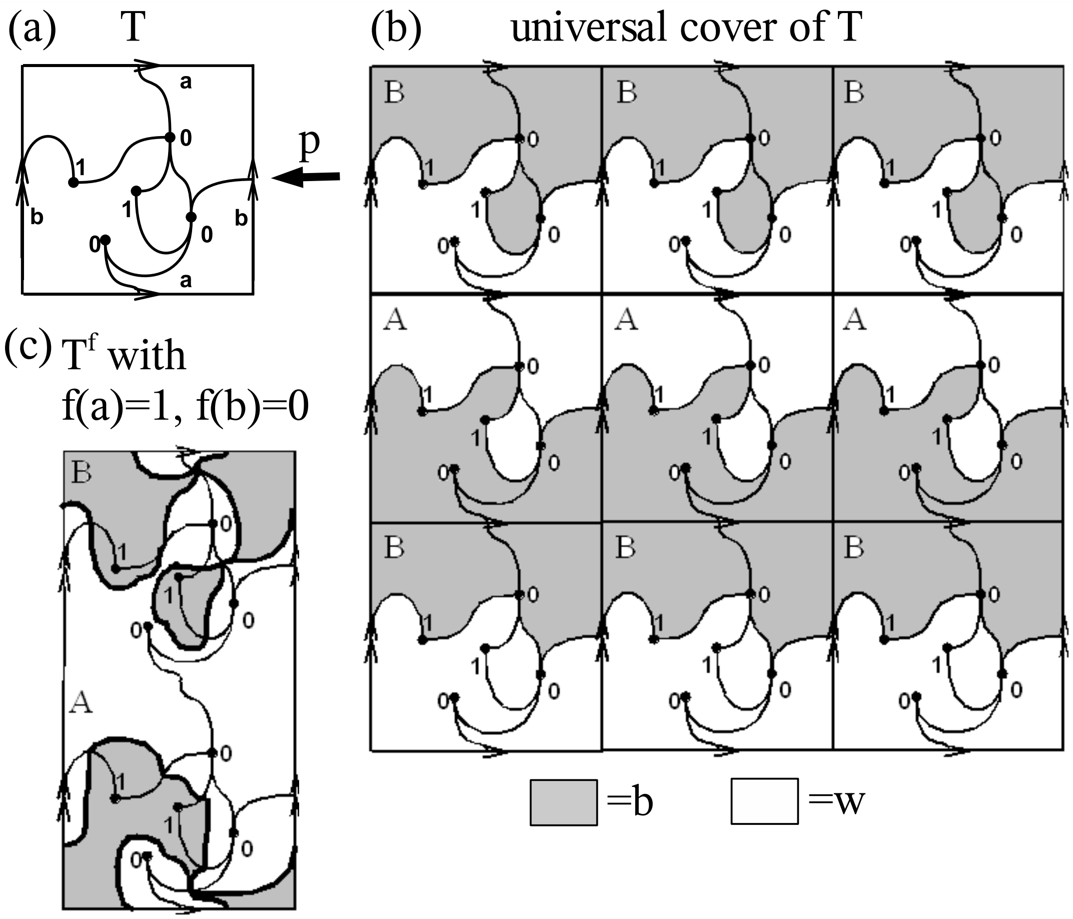

Figure 4. An example of T and a solution of $L(G)x = 0$ (a), the corresponding coloring of face regions in the covering of T (b), and the resulting left-right cycle that divides T^f into two regions producing the solution (x, x) (c)

$$\dim\ker L(G) \geq \frac{1}{2}(\dim Im(M) + \dim\ker(M)) = \frac{1}{2} \cdot \dim\ker L(G^f) > k + 1.$$

The case when $f = 0$ is more straightforward, as there any combination of curves that corresponds to 0 in $H_1(T)_2$ gives rise to a different element of $\ker L(G)$. □

It is not hard to see that every element in any $C(f)$, and thus all the elements the algorithm outputs are in $\ker L(G)$. It is trickier to see that any element of $\ker L(G)$ can be obtained this way.

Claim 11. *For any $x \in \ker L(G)$ there is an f such that $x \in C(f)$, and thus is obtained by the algorithm.*

Proof. The main idea is that an element x of $\ker L(G)$ gives rise to an element of the kernel on the infinite graph which is the lift $\widetilde{L(G)}$ of $L(G)$ to the universal cover of T. The lift and the solution is illustrated on Fig. 4(a,b). Note that the lift $\widetilde{L(G)}$ is embedded into the simply-connected plane $\mathbb{R}^2$, and the analysis from the proof of Claim 4 holds here.

In particular, we can start with the vertex regions colored in w if x is 0 on the vertex and b otherwise, and then color the face regions consistently as in the proof of Claim 4, so that when we take XOR of the "b" regions we obtain a valid left-right walk. Note that once we decide on the color of one face, the colors of all other faces follow. Denote the resulting collection of left-right walks by W.

We now consider T as a $4g$-gon P with its edges glued. The covering plane is laminated with pre-images of P such that every two adjacent pre-images share the pre-image of one of the edges of P. Every copy of P contains one copy of the graph G and a part of the collection W. Every pre-image $\widetilde{P}$ of P is colored in b

and w in a certain way. Note that the color of one face region of $\widetilde{L(G)}$ within $\widetilde{P}$ dictates the coloring of the entire plane. In particular, there are only two distinct ways in which the pre-images of P may be colored. There are two cases.

Case 1: All the pre-images have the same color scheme. This means that $W \cap \widetilde{P}$ is the same for all pre-images $\widetilde{P}$, and hence W projects to a collection of left-right walks on $L(G)$ embedded in T. Hence $x \in C(f)$ for $f = 0$.

Case 2: There are two different color schemes, we call them A and B. In this case the color schemes A and B for the face regions must be exact negations of each other, because if A and B disagree in the color of one face region, they will be forced to disagree in the color of all face regions. Furthermore, if a is one of the edges of P, then two preimages $\widetilde{P}_1$ and $\widetilde{P}_2$ of P that share a preimage of a must either always have the same color scheme or the opposite color scheme. This is because any two copies of a may be copied to each other through a deck transformation on the covering space, and a deck transformation can either keep all the coloring schemes, or flip all of them.

We take two copies of polygon P, P_A and P_B and we glue them as follows. Let the edges of P_A be labeled with $\{a_1^A, b_1^A, \ldots, a_g^A, b_g^A\}$ and the edges of P_B with $\{a_1^B, b_1^B, \ldots, a_g^B, b_g^B\}$. Each label appears exactly twice. We glue an edge a_i^A (or b_i^A) to the other edge with the same label in P_A if copies of P sharing a_i^A have the same color scheme. In this case we define $f(a_i) := 0$. Otherwise we glue a_i^A with the corresponding edge in P_B and define $f(a_i) := 1$. It is not hard to see that by definition the resulting surface is T^f. By the construction, the curves W project to a collection of right-left walks in T^f giving rise to the projected solution (x, x). This implies that $x \in C(f)$, and completes the proof.

The last stages of the proof are illustrated on Fig. 4. When the solution is lifted into the universal cover, we see that there are two possible colorings of each fundamental domain, labeled A and B. When we cross a we alternate between A and B, when we cross b, we do not. Thus $f(a) = 1$ and $f(b) = 0$. On Fig. 4(c) we see the solution on the surface T^f with the left-right curve that yields the solution (x, x). As before, the left-right cycle is obtained by XORing the color of the face with the color of the vertex. □

3.4 Solving divisibility by 2^k

We can now apply results from Section 3.3 to solve divisibility of $\tau(G)$ modulo 2^k for a planar G, as well as some other related algebraic problems. In the case of divisibility by 2, the fact that we can compute the basis for the kernel of the Laplacian matrix $L(G)$ was sufficient. Here we will need more.

Lemma 12. *For any k, let A be the adjacency matrix of an even-regular planar graph G (that is, a graph for which $G = L(G) \bmod 2$), and let A' be the matrix obtained from A by removing k rows. Then there is a Turing Machine that uses space $O(\log n + k \log k)$ and either*

1. *finds a basis $v_1, v_2, \ldots, v_s$ for $\ker A'$ with $s \le 2k$; or*

2. *outputs "$\dim \ker A' > 2k$".*

Proof. Suppose that A' is obtained from A by removing rows corresponding to vertices $v_1, v_2, \ldots, v_k$. Let G' be the graph G with the vertices $v_1, \ldots, v_k$ removed. Consider the graph $\widetilde{G}$ on $2n - k$ vertices depicted on Fig. 5. It is obtained from two copies of G' with one copy of the vertices $v_1, \ldots, v_k$ attached to both copies of G' as in G. Denote its adjacency matrix by $\widetilde{A}$. Then

$$\widetilde{A} = \left[\begin{array}{cc|c|c} \multicolumn{3}{c|}{A'} & 0 \\ \hline * & * & * & \}k \text{ rows} \\ \hline 0 & \multicolumn{3}{|c}{A'} \end{array}\right].$$

The first n and the last n entries of any element of $\ker \widetilde{A}$ form a vector in $\ker A'$, hence $\dim \ker \widetilde{A} \le 2 \cdot \dim \ker A'$. On the other hand, for every $w \in \ker A'$ there is a corresponding vector in $\ker \widetilde{A}$. It is obtained by assigning the vertices in the two copies of G' and the vertices $v_1, \ldots, v_k$ in $\widetilde{G}$ according to their corresponding values in w. Hence the projection of $\ker \widetilde{A}$ on the first n entries contains $\ker A'$ as a subspace.

Next, we observe that $\widetilde{G}$ can be easily embedded into a genus k surface. This is done by putting two identical copies of G' on two parallel planes, and for each face of G' that contains a v_i (or several v_i's) attaching a "tube" between the faces in the two copies and putting v_i in the middle between the two planes. This is illustrated on Fig 5.

By Lemma 9 in space $O(\log n + k \log k)$ we can either find a basis of $\ker \widetilde{A}$ or decide that $\dim \ker \widetilde{A} > 4k$, in which case $\dim \ker A' > 2k$. From a basis for $\ker \widetilde{A}$ with at most $4k$ elements we can compute a basis for $\ker A'$ in space $O(\log n + k)$. □

The following lemma generalizes Lemma 12.

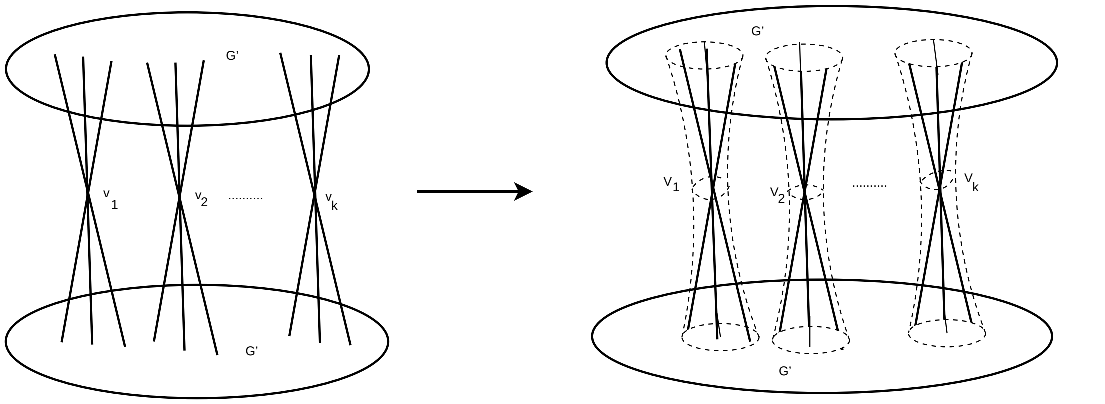

Figure 5. The graph $\widetilde{G}$ in the proof of Lemma 12 and its embedding into a genus k surface

Lemma 13. *For any k, let A be the adjacency matrix of a planar even-regular graph G. Let A' be obtained from A by*

1. *removing a set S of rows, with $|S| \le k$;*
2. *removing a set T of columns, with $|T| \le k$;*
3. *adding a set B of columns, $|B| \le k$.*

Then there is a Turing Machine that uses space $O(\log n + k \log k)$ in case $|B| = 0$ and $O(k \log n)$ otherwise, and either

1. *outputs the basis for* $\ker A'$*; or*
2. *outputs "*$\dim\ker A' > 2k$*".*

Proof. We first prove the lemma under the assumption $|B| = 0$, and then show how to use this special case to solve the general case where $|B| \le k$.

The case $|B| = 0$**.** Let $\widetilde{A}$ be the matrix A with the set S of rows removed. Without loss of generality assume that the first $|T|$ columns of A are removed. For $v \in \ker A'$ the vector $(\underbrace{0,\dots,0}_{|T|}, v)$ is in $\ker\widetilde{A}$. Hence $\ker\widetilde{A}$ contains a copy of $\ker A'$ as a subspace.

On the other hand, if $a = \dim\ker\widetilde{A}$, then $\ker\widetilde{A}$ has a subspace K_1 of dimension $\ge a - |T|$ that has $|T|$ zeros in the beginning of each vector. Hence $\dim\ker A' \ge a - k$, and if $\dim\ker\widetilde{A} > 3k$, then we may output "$\dim\ker A' > 2k$".

Apply Lemma 12 on the matrix $\widetilde{A}$ with $2k$ instead of k. If $\dim\ker\widetilde{A} > 4k$, then we know that $\dim\ker A' > 2k$. Otherwise, we obtain $4k$ vectors that span $\ker\widetilde{A}$. From these vectors we can compute a basis for $\ker A'$ in space $O(\log n + k)$. This completes the case when $|B| = 0$. As a special case, what we have proved so far allows us to compute the determinant of any minor $(n-k) \times (n-k)$ or bigger of A modulo 2 is space $O(\log n + k \log k)$.

The general case. Denote $b = |B|$. Without loss of generality, assume that the columns in B are the first b columns in A'. Denote the matrix A' without the columns B added by A''. By the previous case we can compute $\ker A''$. Note that if $v \in \ker A''$, then $(\underbrace{0,\dots,0}_{b}, v)$ is in $\ker A'$. In particular, $\dim\ker A'' \le \dim\ker A'$. We now need to find those elements of $\ker A'$ that are not all-0 in the first b positions. For every $z \in \{0,1\}^b$ we will check whether there is an element of the form (z, v) in $\ker A'$, and find one if it exists. Since there are just $2^b \le 2^k$ possible z's to check, this would allow us to compute a basis for $\ker A$ in $O(\log n + k)$ extra space.

For a fixed z we want a linear combination of the columns of A'' that adds up to $b' = \sum_{i=1}^{b} b_i z_i$. In other words, we are trying to solve the equation $A''x = b'$. Using brute force, in space $O(k \log n)$ we can find a square minor M in A'' such that $\mathrm{DET}(M) = 1$ and $Rank\ M = Rank\ A''$ (we know that $coRank\ A'' \le 2k$). For simplicity assume that M occupies the first $n - \ell$ rows and columns of A''. Then the first $n - \ell$ columns span the column space of A'', and it is enough to try to find a linear combination of these columns that yields b'. In particular, we obtain the linear equation $Mx_{[1..n-\ell]} = b'_{[1..n-\ell]}$. This last equation can be solved using Cramer's Rule to obtain the unique possible first $n-\ell$ entries of x. We can do this because by the special case of $b = 0$ we can compute determinants of minors of A'', and thus of M. Finally, a simple check would determine whether the vector $x = (x_{[1..n-\ell]}, \underbrace{0,\dots,0}_{\ell})$ satisfies $A''x = b'$. □

Finally, we are ready to prove the main theorem of the section.

Theorem 14. *Given a planar graph G and a number k, in space $O(k^2 \log n)$ we can output either*

1. *An $\ell \leq k$ such that 2^ℓ is the highest power of 2 dividing $\tau(G)$; or*
2. *"$2^{k+1}|\tau(G)$" (the power is too big to determine).*

Proof. Let $A = L(G)$, and let A_0 be its minor. We know that $\tau(G) = \text{DET}(A_0)$, hence we need to evaluate the biggest power of 2 that divides $\text{DET}(A_0)$. We do this by iteratively applying Lemma 13 at most k times, thus obtaining an algorithm that runs in space $O(k^2 \log n)$.

On the i-th iteration we have a matrix A_i that differs from A_0 in at most i rows such that the highest power of 2 dividing $\text{DET}(A_i)$ is equal to the highest power of 2 dividing $\text{DET}(A_0)$ minus i. Thus we will need at most k iterations before concluding that 2^{k+1} divides $\text{DET}(A_0)$.

On iteration i we apply Lemma 13 to A_i^T thus obtaining a linear combination of rows of A_i that adds up to a row that only has even entries. Suppose that the rows that yield this sum have indexes $i_1, i_2, \ldots, i_m$. Denote the rows of A_i by $v_1, v_2, \ldots, v_{n-1}$. Let A'_i be obtained from A_i by replacing v_{i_1} with $v_{i_1} + v_{i_2} + \ldots + v_{i_m}$, then $\text{DET}(A'_i) = \text{DET}(A_i)$, and the i_1-th row of A'_i has all-even entries. Let A_{i+1} be obtained from A'_i by dividing the i_1-th row by 2. Then A_{i+1} differs from A_0 in at most $i+1$ rows, and $\text{DET}(A_{i+1}) = \frac{1}{2} \cdot \text{DET}(A_i)$.

This process continues until we either reach A_{k+1} and return "$2^{k+1}|\tau(G)$", or until we reach A_ℓ such that $\ker A_\ell = \{0\}$, so that $\text{DET}(A_\ell)$ is odd, and we can return 2^ℓ as the highest power of 2 dividing $\text{DET}(A_0) = \tau(G)$. □

We note that the results in this section hold in a slightly more general setting where G is a constant-genus rather than a planar graph. The key to this claim is an analogue of Lemma 12.

Lemma 15. *For any k, let A be the adjacency matrix of an even-regular graph G* that is given with its embedding into a genus $c \leq k$ surface, *and let A' be the matrix obtained from A by removing k rows. Then there is a Turing Machine that uses space $O(\log n + k \log k)$ and either*

1. *finds a basis $v_1, v_2, \ldots, v_s$ for $\ker A'$ with $s \leq 2k$; or*
2. *outputs "$\dim \ker A' > 2k$".*

Proof. The proof is exactly the same as the proof of Lemma 12. The only difference is that G' is now embeddable into a genus $2c + k \leq 3k$ surface instead of genus k surface. Thus the result carries. □

Corollary 16. *Given a number k and a graph G embedded into a genus $c \leq k$ surface, in space $O(k^2 \log n)$ we can output either*

1. *An $\ell \leq k$ such that 2^ℓ is the highest power of 2 dividing $\tau(G)$; or*
2. *"$2^{k+1}|\tau(G)$" (the power is too big to determine).*

Proof. The corollary follows from Lemma 15 in the same way Theorem 14 follows from Lemma 12. Note that the proofs of Lemma 13 and Theorem 14 follow from Lemma 12 in a completely algebraic fashion. Thus the proofs work with a G of genus c instead of a planar G. □

3.5 Computing $\tau(G) \bmod 2^k$

In the previous section we have shown how to compute the highest power of 2 (up to k) that divides $\tau(G)$ for a planar or low-genus G in **L**. For example given a graph G, with $k = 3$ we could decide in which set $\tau(G) \bmod 8$ belongs: $\{1,3,5,7\}, \{2,6\}, \{4\}, \{0\}$. We had no way, however, of determining whether $\tau(G) \bmod 8$ is 2 or 6, for example. In this section we show how to compute the actual value of $\tau(G) \bmod 2^k$. The constructions are stated for a planar G, but work as well for graphs embedded into a low-genus ($\leq k$) surface.

Theorem 17. *Given an integer k and a planar graph G, $\tau(G) \bmod 2^k$ can be computed in space $O(k^2 \log n)$.*

The remainder of the section consists of the proof of Theorem 17. As a first step, we show that it suffices to deal with graphs whose degree is bounded by 3.

Lemma 18. *Given a planar graph G, one can compute a planar graph G' in space $O(\log n)$ so that $\tau(G') \equiv \tau(G) \bmod 2^k$, and the degrees of vertices in G' are bounded by 3.*

Proof. We replace every degree d vertex in G with the $(2d-1)2^k$-edge gadget T_d shown on Figure 6. T_d is a tree with d leafs, and every leaf corresponds to one of the "exits" from v. Note that contracting all the T_d's will yield the original graph G. Thus the number of spanning trees of G' that contain *all* the gadgets T_d is exactly $\tau(G)$. By symmetry, it is not hard to see that the number of spanning trees of G' for which at least one of the edges from the gadgets is missing is divisible by 2^k. Thus $\tau(G') \equiv \tau(G) \bmod 2^k$. □

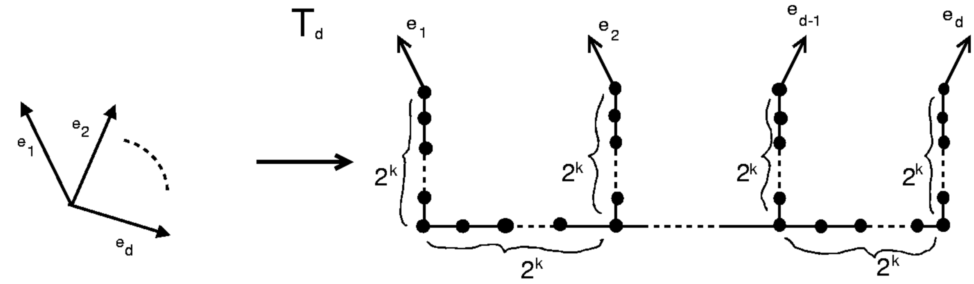

Figure 6. The gadget T_d from the proof of Lemma 18

From now on, we will assume that G has degrees bounded by 3. The strategy of the proof is as follows. First, we assume that $\tau(G)$ is odd. We find a sequence of planar graphs $G_{n-1}, G_{n-2}, \ldots, G_1$ computable from G in space $O(\log n)$ such that the following conditions hold.

1. for each i, G_i has $i+1$ vertices;
2. for each i, G_i and G_{i+1} differ from each other by one vertex and a constant (≤ 10) number of edges;
3. G differs from G_{n-1} by a constant (≤ 3) number of edges;
4. for each i, $\tau(G_i)$ is odd (recall that we assume here that $\tau(G)$ is odd).

Then we will show that computing $\tau(H_1)/\tau(H_2) \bmod 2^k$ for "similar" graphs H_1 and H_2 with odd $\tau(H_1), \tau(H_2)$ can be done in space $O(k^2 \log n)$. $\tau(G_1)$ is trivial to compute (as it only has two nodes) and the computations of $\tau(G_{i+1})/\tau(G_i) \bmod 2^k$ will be performed in parallel. Finally, we will have

$$\tau(G) = \frac{\tau(G)}{\tau(G_{n-1})} \cdot \frac{\tau(G_{n-1})}{\tau(G_{n-2})} \cdot \ldots \cdot \frac{\tau(G_2)}{\tau(G_1)} \times \tau(G_1) \bmod 2^k.$$

First, we define the graphs G_i. We order the vertices of G, $\{v_1, v_2, \ldots, v_n\}$ so that if we remove $A_i = \{v_{i+1}, \ldots, v_n\}$ from G, then in the residual graph G' all the vertices that have neighbors in A_i are on the same outside face. There are many ways to accomplish this. For example, if the graph G is drawn on the plane, then ordering the vertices from left to right accomplishes this goal. Furthermore, we assume there is some arbitrary order $\prec$ on the edges of G.

The vertices of G_i are $V_i = \{v_1, v_2, \ldots, v_i, v^*\}$, where v^* is a special vertex on the outside of the graph. It is added to make sure that $\tau(G_i)$ is odd.

Let the graph G_i' be obtained from G by removing $A_i = \{v_{i+1}, \ldots, v_n\}$. Consider G_i' along with the edges from G_i' to A_i as on Fig. 7. Consider the (pieces of) left-right walks in G_i'. If there are s edges from G_i' to A_i then there will be s such pieces – one for each "loose edge". Denote the "loose edges" by $e_1 \prec e_2 \prec \ldots \prec e_s$. For each edge e_j there are two ends of the left-right walk to trace. If either one of them reaches an end of some e_ℓ with $\ell < j$, we just continue the loop from the other end on e_ℓ. If, as a result, we reach e_ℓ with $\ell > j$, we remove the edge e_j. If we reach the other end of e_j without reaching any higher-ranking edge, we connect the other end of e_j to v^*. In the example on Fig. 7, curves starting at e_1 hit e_2, hence e_1 is discarded. Curves starting at e_2 go through e_1 and back to e_2, so e_2 connects to v^*. Similarly, e_3 also connects to v^*.

We need to see that the resulting G_i will have one left-right cycle, and thus $\tau(G_i)$ is odd. Note that every left-right walk in G_i' has its ends on one of the edges e_j, because by the assumption $\tau(G)$ is odd, and thus G has one left-right walk. After we discard all the edges e_j for which the left-right walk originating at e_j reaches some e_ℓ for $\ell > j$, we are left with t loose edges, $e_{i_1}, e_{i_2}, \ldots, e_{i_t}$ such that a left-right walk starting at e_{i_j} ends on the other end on e_{i_j}. It is straightforward to see that connecting v^* to the edges $e_{i_1}, \ldots, e_{i_t}$ results in a graph with a single left-right cycle.

It remains to see that G_{i+1} and G_i are similar to each other and that G_{n-1} is similar to G. First of all, by the construction, in G_{n-1} the node v^* may only be connected to vertices to which v_n was connected. Since *deg* $v_n \leq 3$ this means that G_{n-1} differs from G by ≤ 3 edges.

For an arbitrary i, we first add a vertex we call v_{i+1}'

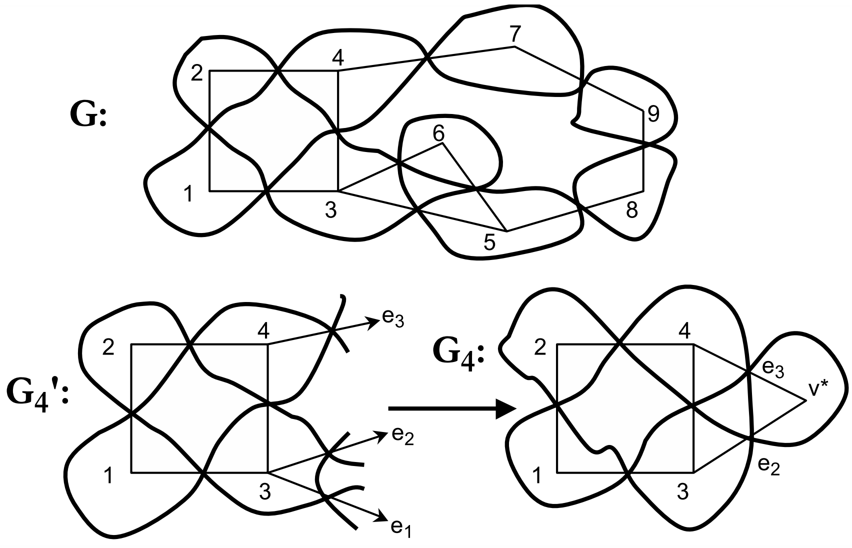

Figure 7. An example of obtaining G_i from G

to G_i and connect it to v^*. The resulting graph $\widetilde{G_i}$ has $i+2$ vertices, and can be directly compared to G_{i+1}. v'_{i+1} is a degree 1 vertex, and thus does not affect the left-right cycle in G_i. We also have $\tau(G_i) = \tau(\widetilde{G_i})$. We claim that G_{i+1} and $\widetilde{G_i}$ differ by at most 10 edges. The differences between G_{i+1} and $\widetilde{G_i}$ are: (1) the edges leaving v_{i+1} in G_{i+1} are different from edges leaving v'_{i+1} in G_i. The degree of v_{i+1} is ≤ 3 and the degree of v'_{i+1} is 1 – hence the difference amounts to ≤ 4 edges; (2) the edges leaving v^* may be different in G_{i+1} and in G_i.

We need to bound the number of edges in (2). If there is an edge from v_j to v^* in G_{i+1} but not in G_i, it means that the left right cycle that yielded the edge has been disturbed by the removal of v_{i+1} in transition from G_{i+1} to G_i. Thus it must have crossed one of the edges touching v_{i+1}. There are at most three such cycles, and thus at most 3 of the v_j's are affected.

If there is an edge from v_j to v^* in G_i but not in G_{i+1}, it means that the left-right path that caused v_j to connect to v^* in G_i is not valid in G_{i+1}. For this to happen such a path should cross one of edges that connect v_{i+1} to its neighbors. Up to three paths may become invalid.

Overall, we see that $\widetilde{G_i}$ differs from G_{i+1} in $\leq 4+3+3 = 10$ edges. Now we need to show that $\tau(G_{i+1})/\tau(G_i) \bmod 2^k$ can be computed in space $O(k^2 \log n)$. It is here that we use Corollary 16.

Lemma 19. *For two planar graphs G_1 and G_2 on n vertices that differ in $\leq c$ edges for some constant c, and such that $\tau(G_1)$ and $\tau(G_2)$ are odd, we can compute $\tau(G_1)/\tau(G_2) \bmod 2^k$ in space $O(k^2 \log n)$.*

Proof. Denote the edges in which G_1 and G_2 are different by $e_1, \ldots, e_c$. We start by creating a "hybrid" graph G where all the edges from both G_1 and G_2 appear. While G_1 and G_2 are planar, G may not be. However, it is obtained from either G_1 or G_2 by adding at most $c/2$ edges. Hence by adding a "handle" for each newly added edge we see that G can always be embedded into a genus $c/2$ surface.

We replace every edge e_i in G with the gadget $g(\alpha_i, \beta_i)$ depicted on Fig. 8. It consists of one "chain" of α_i edges and $\beta_i - 1$ more regular edges connecting the endpoints. The idea is that if in G there were B spanning trees containing e_i and A spanning trees excluding e_i, then with the gadget it will have $B\beta_i$ spanning trees where edges from the gadget connect the endpoints and $A\alpha_i$ spanning trees where they disconnect the endpoints to a total of $A\alpha_i + B\beta_i$.

For every possible combination of $\alpha_i, \beta_i \in \{1, 2, \ldots, 2^k\}$ we consider $\tau(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c)$ – the number of spanning trees modulo 2^k of the graph G with each e_i replaced with $g(\alpha_i, \beta_i)$. According to Corollary 16 we can compute the highest power of 2 dividing $\tau(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c)$ for all possible combinations in space $O(k^2 \log n)$.

Consider $\tau(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c)$ as a function of the α's and β's. If we fix all the variables but α_i and β_i for some i, we have seen that the expression for $\tau(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c)$ will have the form $A\alpha_i + B\beta_i$. This implies that τ is multilinear in the α's and β's, and moreover each of its additive terms contains exactly

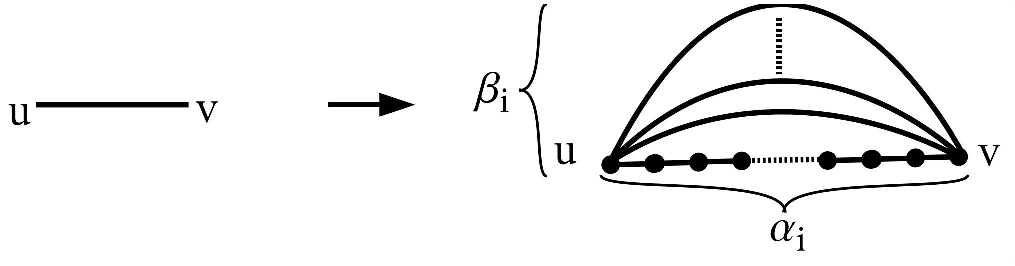

Figure 8. The gadget $g(\alpha_i, \beta_i)$

one of $\{\alpha_i, \beta_i\}$ for all i. Thus

$$\tau(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c) = \sum_{f:\{1..c\}\to\{0,1\}} A_f \gamma_1^f \gamma_2^f \ldots \gamma_c^f \quad \text{where}$$

$$\gamma_i^f = \begin{cases} \alpha_i & \text{if } f(i) = 0 \\ \beta_i & \text{if } f(i) = 1 \end{cases} \tag{1}$$

From now on we consider all equalities to be modulo 2^k. The coefficients A_f thus can be always taken from $\{0, 1, \ldots, 2^k - 1\}$. Note that there are 2^c coefficients. This number is constant ($\leq 2^{10}$), and thus the entire calculation is easily done on the main tape.

Note that if one multiplies all the coefficients A_f by some odd integer, then for each possible choice of the α's and β's the highest power of 2 dividing $\tau(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c)$ is not affected. Thus there is no hope of finding the actual values of the A_f. Fortunately, we do not need to. We will show how to find coefficients $c_f \in \{0, \ldots, 2^k - 1\}$ such that $A_f = c_f A_0$ for some A_0 (recall that all equalities are modulo 2^k). We know that there is an assignment of α's and β's that gives a graph equivalent to G_1, for which τ is odd. Thus A_0 must be odd. Once we find the coefficients c_f, we can compute $\tau(G_1)/\tau(G_2)$. Let $(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c)$ be the assignment corresponding to G_1, and $(\widetilde{\alpha_1}, \widetilde{\beta_1}, \ldots, \widetilde{\alpha_c}, \widetilde{\beta_c})$ the assignment corresponding to G_2. Then

$$\frac{\tau(G_1)}{\tau(G_2)} = \frac{\sum_{f:\{1..c\}\to\{0,1\}} A_f \gamma_1^f \gamma_2^f \ldots \gamma_c^f}{\sum_{f:\{1..c\}\to\{0,1\}} A_f \widetilde{\gamma}_1^f \widetilde{\gamma}_2^f \ldots \widetilde{\gamma}_c^f} = \frac{\sum_{f:\{1..c\}\to\{0,1\}} c_f \gamma_1^f \gamma_2^f \ldots \gamma_c^f}{\sum_{f:\{1..c\}\to\{0,1\}} c_f \widetilde{\gamma}_1^f \widetilde{\gamma}_2^f \ldots \widetilde{\gamma}_c^f},$$

thus if we know the c_f's, we can compute $\tau(G_1)/\tau(G_2)$. To complete the proof, we need (everything is the claim and in what follows is modulo 2^k):

Claim 20. *Given an expression of the form* (1), *and given for each assignment of α's and β's the highest power of* 2 *dividing $\tau(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c)$ we can compute c_f such that $A_c = A_0 c_f$ for some common A_0. Furthermore, at least one of the c_f's can be made odd.*

We prove the claim by induction on c. It is obvious for $c = 0$, as there is only one coefficient A_0, and we can take $c_f = 1$. For the step we will be using the following claim.

Claim 21. *Suppose that τ_1 and τ_2 are given by two formulas as in* (1). *Suppose that the highest power of* 2 *dividing all the coefficients of τ_1 is 2^{d_1}, and of τ_2, 2^{d_2}. Then there is an assignment $\vec{\alpha}, \vec{\beta}$ such that (simultaneously) the highest power of* 2 *dividing $\tau_1(\vec{\alpha}, \vec{\beta})$ is 2^{d_1}, and the highest power of* 2 *dividing $\tau_2(\vec{\alpha}, \vec{\beta})$ is 2^{d_2}.*

Now we can do the induction step for Claim 20. Write

$$\tau(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c) = \alpha_c \tau_1(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1}) + \beta_c \tau_2(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1}).$$

Setting $\alpha_c = 1, \beta_c = 0$ we can use the induction hypothesis to compute d_f such that $A_f = A_1 d_f$ for some A_1 and for all f with $f(c) = 0$ and such that at least one of these d_f's is odd. Similarly, we can compute d_f such that $A_f = A_2 d_f$ for some A_2 and for all f with $f(c) = 1$. Without loss of generality assume that the power of 2 dividing A_2 is greater or equal to the power of 2 dividing A_1. To complete the proof, we need to find a d such that $A_2 = d \cdot A_1$ (recall that the equality if modulo 2^k). Then we choose $A_0 = A_1$, and

$$c_f = \begin{cases} d_f & \text{if } f(c) = 0 \\ d \cdot d_f & \text{if } f(c) = 1 \end{cases}$$

By Claim 21 there is an assignment of $(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1})$ for which the highest power of 2 dividing $\tau_1(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1})$ is the same as the highest power of 2 dividing A_1, and the

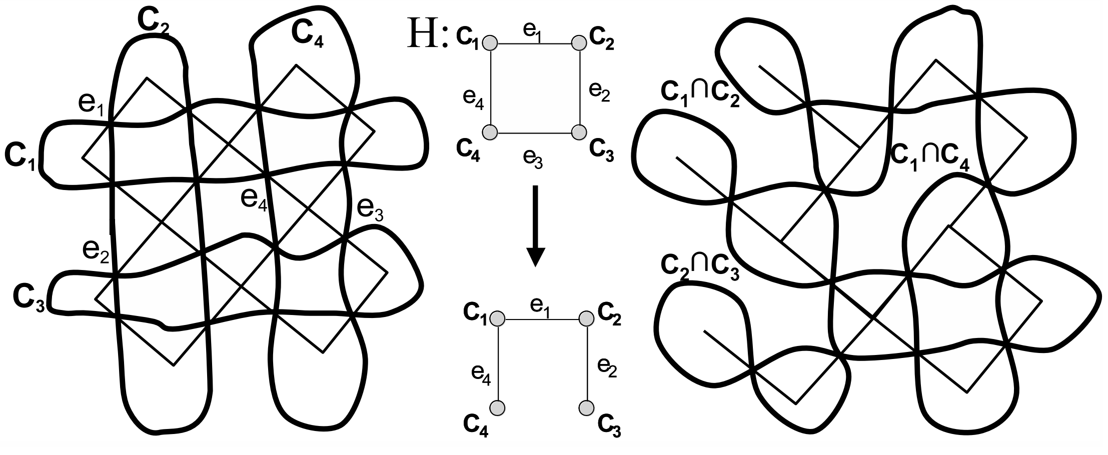

Figure 9. Making $\tau(G)$ odd by removing m edges, $\{e_1, e_2, e_4\}$ in this case

same is true for τ_2. By looking at all possible assignments we can find one with this property. After substituting the assignment we will have $\tau_1(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1}) = A_1 s_1$, and s_1 must be odd. Similarly $\tau_2(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1}) = A_2 s_2$ for an odd s_2. Fixing $(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1})$ at this assignment we have

$$\tau(\alpha_c, \beta_c) = A_1 s_1 \alpha_c + A_2 s_2 \beta_c.$$

By fixing $\beta_c = -1$ and trying all possible α_c, we can find α_c such that

$$A_1 s_1 \alpha_c - A_2 s_2 = 0.$$

Thus $A_2 = (s_1 \alpha_c s_2^{-1}) \cdot A_1$. This completes the proof. □

To finish the proof of Lemma 19, it remains to prove Claim 21.

Proof. (of Claim 21) Once again, we prove the claim by induction on c. The statement is trivial for $c = 0$. Assume it is true for $c - 1$. Write

$$\begin{cases} \tau_1(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c) = \\ \qquad \alpha_c \tau_3(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1}) + \\ \qquad\qquad \beta_c \tau_4(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1}) \\ \tau_2(\alpha_1, \beta_1, \ldots, \alpha_c, \beta_c) = \\ \qquad \alpha_c \tau_5(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1}) + \\ \qquad\qquad \beta_c \tau_6(\alpha_1, \beta_1, \ldots, \alpha_{c-1}, \beta_{c-1}) \end{cases}$$

There are three cases.

Case 1: There is a coefficient of the form $2^{d_1} \cdot \mathit{odd}$ in $\tau_3(\cdot)$ and a coefficient of the form $2^{d_2} \cdot \mathit{odd}$ in $\tau_5(\cdot)$. In this case we can set $\alpha_c = 1, \beta_c = 0$, and the assignment exists by the induction hypothesis applied to the pair τ_3, τ_5.

Case 2: There is a coefficient of the form $2^{d_1} \cdot \mathit{odd}$ in $\tau_4(\cdot)$ and a coefficient of the form $2^{d_2} \cdot \mathit{odd}$ in $\tau_6(\cdot)$. This case is exactly symmetric to case 1.

Case 3: The cases above do not hold. Then all the coefficients of τ_3 must be divisible by 2^{d_1+1} and all the coefficients of τ_6 must be divisible by 2^{d_2+1} (or, a similar statement is true for τ_4 and τ_5, a case which is dealt with in exactly the same fashion). Set $\alpha_c = \beta_c = 1$. By the induction hypothesis with τ_4, τ_5 there is an assignment for which τ_4 has a form $2^{d_1} \cdot \mathit{odd}$, and hence τ_1 has this form (because τ_3 is divisible by 2^{d_1+1}), and τ_5 has a form $2^{d_2} \cdot \mathit{odd}$, and hence τ_2 has this form. □

So far we have seen how to compute $\tau(G) \bmod 2^k$ in space $O(k^2 \log n)$ in the case $\tau(G)$ is odd. To complete the proof of Theorem 17 we need to show how to deal with all other cases. Let ℓ be the highest power of 2 dividing $\tau(G)$. If $\ell \geq k$ we can output "0" and we do not need any further computations. Otherwise, it is not hard to see that we must have $\dim \ker L(G) \leq \ell + 1$. Thus G has at most $\ell + 1$ left-right cycles. Furthermore, we can assume that G is connected, because otherwise $\tau(G)$ is trivially 0. Suppose that G has $m + 1 \leq \ell + 1$ left-rights cycles. As a first step we show how to remove m edges $\{e_1, \ldots, e_m\}$ from G to obtain a G' with one left-right cycle (and hence an odd $\tau(G')$).

First of all, note that if C_1 and C_2 are two different left-right cycles, which intersect at an edge e, then the effect of removing e from the graph is that C_1 and C_2 are merged and become one cycle. The same effect is achieved if e is replaced by a chain of even length. Let $C_1, \ldots, C_{m+1}$ be the left-right cycles in G. We construct an adjacency graph H for cycles, where two cycles C_i and C_j are connected if and only if they share an edge e'. We label the edge (C_i, C_j) in H with e'. H is connected because G is connected. Thus we can find an m-edge spanning tree T in H. Let the edges of T be labeled with $e_1, \ldots, e_m$. It is not hard to prove by induction on the size of T that removing $e_1, \ldots, e_m$ from G will result in a graph G' where all the cycles $C_1, \ldots, C_m$ are merged into one, and hence $\tau(G')$ is odd. The transition is illustrated on Fig. 9.

For a function $f : \{1 \ldots m\} \to \{0, 1\}$ let G_f be obtained from G by

1. removing edges e_i when $f(i) = 0$;

2. replacing edges e_i with a chain of 2^k edges if $f(i) = 1$.

The effect of replacing e_i with a chain of 2^k edges is that the cycles that intersect at e_i are merged. Hence, like G', G_f always has a single left-right cycle, and thus $\tau(G_f)$ is odd and can be computed modulo 2^k. For a spanning tree T in G and an $f : \{1 \ldots m\} \to \{0, 1\}$:

1. if T contains some e_i for which $f(i) = 0$, then T corresponds to 0 trees in $\tau(G_f)$; otherwise

2. if T omits $t \geq 1$ e_i's for which $f(i) = 1$, then T corresponds to $(2^k)^t$ trees in $\tau(G_f)$; otherwise

3. T corresponds to exactly one tree in G_f.

Hence, modulo 2^k, $\tau(G_f)$ is equal to the number of spanning trees in G which include all the e_i's for which $f(i) = 1$ and exclude all other e_i's. Thus

$$\tau(G) = \sum_{f:\{1\ldots m\}\to\{0,1\}} \tau(G_f) \mod 2^k,$$

and by evaluating the right-hand-side we complete the computation of $\tau(G)$ modulo 2^k.

4 Hardness of the Laplacian rank modulo primes $p > 2$

In this section we show how 2 is special when it comes to divisibility properties of $\tau(G)$ even for planar G. It is not hard to show that computing $\tau(G) \bmod 2$ for arbitrary G is $\oplus\mathbf{L}$-complete. We have seen that this is not the case for *planar G* (unless $\mathbf{L} = \oplus\mathbf{L}$). On the other hand, we have the following:

Theorem 22. *For prime $p > 2$, finding out whether $\tau(G) \equiv 0 \bmod p$ for a* planar *graph G is complete for* $\mathrm{Mod}_p\mathbf{L}$.

The general idea for proving this is the following:

We will show the following chain of reductions from the computation of rank of a general symmetric matrix to the computing the rank of the Laplacian of a planar graph:

$$\begin{aligned}\mathrm{RANK}_{\mathrm{Adjacency}} \leq \mathrm{RANK}_{\mathrm{Laplacian}} \\ \leq \mathrm{RANK}_{\mathrm{PlanarLaplacian}}\end{aligned}$$

where all the RANKs are being considered over $\mathbb{Z}_p$. The reductions will be such that if we start with an adjacency matrix whose co-rank is 0, we will get a Laplacian matrix with co-rank 1. If we start with an adjacency matrix with co-rank at least 1, then we will get the Laplacian matrix with co-rank at least 2, all co-ranks being considered modulo the prime p. Then the planarizing gadgets will transform arbitrary Laplacian into a planar Laplacian while preserving the co-rank modulo p. Overall, the singularity testing of a matrix modulo p will be reduced to testing whether co-rank of a planar Laplacian is 1 or more, i.e. whether a planar $\tau(G)$ is divisible by p or not. The idea hence is: given an arbitrary graph Laplacian $L(G)$, first *transform* it into a graph Laplacian with every vertex degree $0 \bmod p$. In this transformation, we would want to "preserve" the rank; i.e. given the rank of the new Laplacian, we should be able to retrieve the rank of the original graph Laplacian, and vice versa. But now that the transformed graph (call this H) has all degrees $0 \bmod p$, its Laplacian matrix is *essentially* its adjacency matrix too!

Next, we *replace* the crossovers in this graph H to get a planar graph H' which has the following properties:

- H' preserves co-rank of H. That is if x is a vector such that $Hx = 0$ (over $\mathbb{Z}_p$), then there corresponds a vector y of suitable length such that $H'y = 0$. Vice versa, for every y, there corresponds an x so that the transformation *preserves co-ranks.*

- every vertex in H' has degree $0 \bmod p$.

So, the (planar) graph H' again has its adjacency matrix (essentially) the same as its Laplacian (over $\mathbb{Z}_p$). Hence, this would prove that finding the rank of planar Laplacian matrices (over $\mathbb{Z}_p$) is hard for $\mathrm{Mod}_p\mathbf{L}$.

$\mathbf{RANK_{Adjacency}} \leq \mathbf{RANK_{Laplacian}}$: Note SINGULARITY and $\mathrm{RANK}_{\mathrm{Adjacency}}$ for matrices over $\mathbb{Z}_p$ are complete for $\mathrm{Mod}_p\mathbf{L}$, see [1].

We begin with a

Lemma 23. SINGULARITY mod$p$ *reduces to computation of rank of arbitrary graph Laplacians (over $\mathbb{Z}_p$).*

Proof. Consider an arbitrary matrix A. We convert that into a *Laplacian* matrix L by describing a minor of L first (call this minor L'):

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & A \\ 0 & 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & A & 0 & 0 \\ 0 & 0 & A^t & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 & 0 & 0 \\ A^t & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

where there are p A's and p A^t's on the diagonal (A^t being the transpose of A).

Let L be now obtained from the above matrix L' by adding one row and one column, so that sum of entries in every row and column is 0. Clearly, L is the Laplacian matrix of some graph G. Since we have p copies of A and p copies of A^t, the $(1, 1)$ entry of L is $0 \bmod p$, which means that for the graph G, every vertex degree is $0 \bmod p$ (all the other diagonal entries of L are zero since A has 0 on its diagonal). Note that if A has full rank (i.e. co-rank 0), then L has co-rank 1. If $\dim\ker A \geq 1$, then then $\dim\ker L' \geq p$, so $\dim\ker L \geq (p - 1)$. So if we could determine the rank of L, we could find out if A is singular or not (over $\mathbb{Z}_p$). □

For the future, we record the following direct corollary of the Matrix Tree Theorem:

Claim 24. *Given a graph G with Laplacian matrix L, $\tau(G)$ is not divisible by p if and only if the co-rank of L is* 1.

$\mathbf{RANK_{Laplacian}} \leq \mathbf{RANK_{PlanarLaplacian}}$: Now we transform a non planar graph G with every vertex degree $0 \bmod p$ into a planar graph H with every vertex degree $0 \bmod p$ while preserving the co-rank. Since the vertex degrees concerned are all $0 \bmod p$, the Laplacians are the same as the adjacency matrices.

Let A, B be the adjacency matrices of G, H respectively. Since in the following we are working over $\mathbb{Z}_p$, we will not mention $\mathbb{Z}_p$ for brevity's sake unless otherwise necessary.

The construction consists of two *stages*:

1. **Stage 1:** Make all the intersections in the graph simple, so that each edge would intersect at most one other edge.

2. **Stage 2:** Replace simple intersections with planar gadgets.

STAGE 1 : The gadget we construct preserves the property that every vertex has degree $0 \bmod p$, and is shown in Figure 10. Some remarks about the diagram are in order: we allow the edge intersections (of the original graph) to take place only at the **bold** lines in Figure 10. Since an edge of the original graph G might have at most n intersections with other edges (where n is the number of vertices of the graph), we have to extend each edge of G into a path of length $2n-1$ with $n-1$ "subgadgets" as enclosed in between the dotted lines in Figure 10. Solutions x to $Ax = 0$ translate to "weights" on each vertex, so that the sum of the weights on all the neighbors of any vertex is 0. The squiggly double arrows in Figure 10 with $p - 1$ written above refer to the multiplicity of the corresponding edges of the graph H'.

Having done this, it is clear that the objective of Stage 1 is fulfilled: all intersections in the resulting graph are simple.

Suppose the graph resulting from the above (in which every intersection is simple) is H' with adjacency matrix B'. Since we are trying to preserve the co-rank of the adjacency matrix in this transformation, we will assume that we are given a vertex labeling by values over $\mathbb{Z}_p$ (i.e. a $\mathbb{Z}_p$-valued *weighting* on the vertices) which encodes a solution x to $Ax = 0$, and make such a solution x correspond to a solution y' of $B'y' = 0$ (and vice versa). Given "weights" on each vertex as above, solutions x to $Ax = 0$ translate to weightings where for any vertex the sum of the weights on all the neighbors of that vertex is 0.

Now it is clear from the figure that there is a unique way of extending a solution x to $Ax = 0$ to produce a solution to $B'y' = 0$. On the other hand, the process is invertible, so that for every y' there corresponds an x. The easiest way to see that the successive values are as marked in the figure is via induction. We leave out the details of this easy induction.

STAGE 2 : Now, we replace each simple intersection in H' by a gadget as shown in Figure 11. Call this final graph H, and its adjacency matrix B.

Again, we note that the initial values at the endpoints of an edge (and the neighbors of the endpoints) corresponding to a solution y' for $B'y' = 0$ uniquely extend to a solution for $By = 0$. The easiest way to see this is by induction, as before.

Altogether, at the end of the two stages, we have transformed G into a *planar* graph H which has the same co-rank as G, and has every vertex degree 0 mod p. Hence Theorem 22 is proved.

We also observe that the graphs produced by the transformation can be made bipartite if the original graphs are. To this end, note that there are always two ways to apply Stage 2 to an intersection, and one of them will keep the graph bipartite.

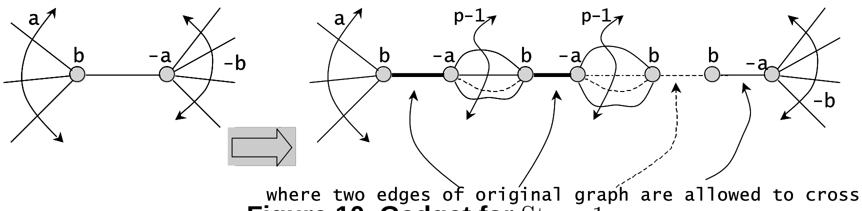

Figure 10. Gadget for Stage 1

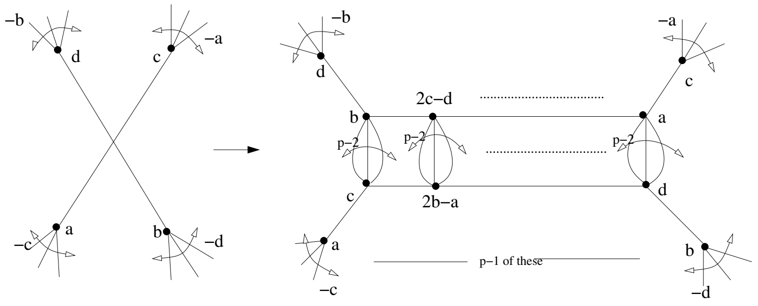

Figure 11. Gadget for Stage 2

The modular results yield the following corollary for the hardness of computing $\tau(G)$ for a planar G.

Corollary 25. *The problem of computing the value of $\tau(G)$ for a planar G is complete for* DET *under a Logspace Turing reduction.*

Proof. In fact the reduction can be made into a *Logtime*-uniform $\mathbf{TC}^0$ Turing reduction. Given an integer matrix A we need to reduce the computation of $\mathrm{DET}(A)$ to a series of computations of $\tau(G_i)$ for some planar G_i's. Let $(p_1, p_2, \ldots, p_k) = (3, 5, \ldots)$ be an enumeration of the first $k = n^{O(1)}$ primes, starting with 3. We may assume that A is symmetric, since computing the determinant of symmetric matrices is complete for DET.

By Theorem 22, computing $\mathrm{DET}(A)$ modulo p_i is reduced to verifying whether $\tau(G_{ij}) = 0$ modulo p_i for at most p_i planar graphs G_{ij}. This obviously reduces to computing the actual value of $\tau(G_{ij})$. Finally, the calculations of $\tau(G_{ij}) \mod p_i$ and the reconstruction of $\mathrm{DET}(A)$ from its residues modulo $p_1, \ldots, p_k$ can be done in *Logtime*-uniform $\mathbf{TC}^0$ according to [10], which completes the proof. □

As the last item in this section, we prove the following contrapuntal result for $p = 2$.

Theorem 26. *Finding out if $\tau(G)$ for a planar graph G* given along with its planar embedding *is odd is* **L**-*complete under* $\mathbf{AC}^0[\mathbf{2}]$ *reductions.*

Proof. Since we have already shown that the problem is contained in **L**, we need to show hardness for **L**.

The proof idea is simple: we reduce SCP – Single Cycle Permutation (cf. [2]) to the above problem. The problem SCP is the following: Given a permutation presented pointwise, determine whether the permutation consists of a single cycle. Equivalently, we are given the edges of a 2-regular graph H listed as vertex pairs (a, b) and we are to determine if it consists of a single cycle. The intuition is as follows: a planar graph G has an odd number of spanning trees iff it has exactly one left-right cycle. Given graph H, we will output a graph G such that G is planar with an explicit embedding, and H is essentially the graph derived from the left-right cycles in G.

The main challenge of the proof is to get a G that is given with an explicit planar embedding. The graph H itself, for example, is 2-regular and thus planar, but we do not have and explicit embedding of H into the plane. Note that [2] prove SCP to be complete for **L** under $\mathbf{NC}^1$ reductions, but we can easily verify that their proof in fact gives completeness under $\mathbf{AC}^0$ reductions.

Place n points corresponding to the n vertices of H on a circle. Consider all the edges between the n points joined as according to H. The edges of the circle are absent unless they are specified as being in H. We can always arrange the points so that no three edges intersect at the same point. These edges divide the plane up into regions, which are bounded by *segments.* Call two regions *crossing* if they intersect only in a point, and do not share a segment. Let us color the regions in two colors, black and white. Let the regions adjacent to the vertices of H be colored black. Complete the coloring such that two regions which share an edge get opposite colors. This is always possible. Now we create the graph G. Place a vertex v_r inside each black region r. We say that v_r *corresponds* to region r and vice versa. We place an edge between v_1 corresponding to

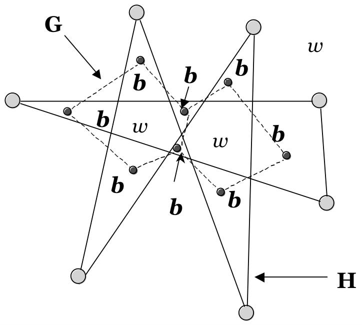

Figure 12. Graph G from graph H

black region r_1 and v_2 corresponding to black region r_2 iff the two regions r_1 and r_2 are crossing in the layout (because they have the same color, they clearly cannot share a segment). Performing this procedure for all vertices v_r, we get our graph G. See Figure 12. It is clear from construction that G has the cycles of H as its left-right cycles. So G has an odd number of spanning trees iff H has exactly one cycle.

Note in the above, that if we had placed a vertex in the unbounded region, which is colored white and produced a graph G' by connecting up vertices in the white regions (like we did above for the black regions), we would have created the planar dual of the graph G (which has the same number of spanning trees as G).

The reduction above can be implemented in $\mathbf{AC}^0[\mathbf{2}]$, because all we need to color the regions in black and white is a parity gate. To make sure that we get one representative v_r for each black region r we begin with a collection of $\Theta(n^3)$ points P, such that any potential region contains at least one point. We create an equivalence relation on P so that $p, q \in P$ are in the same class iff they are on the same region. Every bounded region is convex, and hence $p \sim q$ iff there is no line between two vertexes of H intersecting the segment pq. Thus the relation can be computed in $\mathbf{AC}^0$, and we can obtain a unique representative v_r for every black region r. □

5 The Permanent modulo powers of 2

Given a matrix A, it is clear that the permanent of A (denoted $\text{PERM}(A)$) is of the same parity as that of the determinant (denoted $\text{DET}(A)$). For definitions of the permanents and determinants of matrices, see for instance, [13]. Valiant proves in his seminal paper [19], that finding out the value of the PERM of a matrix modulo 2^k (for constant k) is in $\mathbf{P}$, however the method he uses is akin to Gaussian elimination, and is inherently sequential. Here, we prove

Theorem 27. *Finding out the* PERM *of a matrix modulo 2^k (for constant k) is complete for $\oplus\mathbf{L}$.*

Hardness for $\oplus\mathbf{L}$ follows from the fact that for $k = 1$, it corresponds to singularity of the DET (over $\mathbb{Z}_2$). Hence we have to prove containment in $\oplus\mathbf{L}$.

The structure of the proof will be as follows: we first show how the question $4|\text{PERM}(A)$ can be resolved in $\oplus\mathbf{L}$. We use this, along with facts about LUP-decompositions (cf. [4]) to show how we can find out $\text{PERM}(A) \bmod 4$ in $\oplus\mathbf{L}$. After we have accomplished this, we can easily see how to find out $\text{PERM}(A) \bmod 2^k$ (for constant k) in $\oplus\mathbf{L}$. As a first step, we prove that finding out whether $4|\text{PERM}(A)$ can be done in $\oplus\mathbf{L}$.

Given a $n \times n$ matrix A, we first check if $\text{DET}(A)$ is even. Having passed this check, we proceed to find a solution $x \in \{0,1\}^n$ for $A^T x = 0$ (over $\mathbb{Z}_2$), and this can be done in $\oplus\mathbf{L}$. Let $x^t = (x_1, x_2, \cdots, x_n)$. This means that the sum of the rows of A corresponding to the x_i's which are 1 is $0 \bmod 2$. Without loss of generality we may assume $x_1 = 1$. If r_i denotes the i^{th} row of A, then replace the 1^{st} row of A by the sum of rows $\Sigma_i x_i \cdot r_i$ to get a new matrix A'. The first row of A' consists only of even entries.

Let A_i denote the matrix A with the 1^{st} row replaced by the i^{th} row of A (for instance, $A_1 = A$). We can write that $\text{PERM}(A') = \Sigma_i \text{PERM}(A_i) \cdot x_i$.

Note that each matrix A_i (for $i > 1$) has two rows equal, and hence $\text{PERM}(A_i)$ is even.

For each $i > 1$ and for each (j, k), build matrix B_{ijk} as follows: from matrix A_i, delete the 1^{st} and i^{th} rows (these rows actually being equal), and delete the j^{th} and k^{th} columns. Find out $\text{PERM}(B_{ijk}) \bmod 2$. Then we can use linearity of the permanent function to figure out the value of $\text{PERM}(A_i) \bmod 4$.

Matrix A' is of a slightly different form: it has its first row which consists wholly of even entries. Let C_i denote the matrix obtained from A' by deleting the first row and column i. We can find out $\text{PERM}(C_i) \bmod 2$, and then use linearity of the permanent to find out the value of $\text{PERM}(A') \bmod 4$. Finally, we have $\text{PERM}(A) = \text{PERM}(A_1) = \text{PERM}(A') - \Sigma_{i>1}\text{PERM}(A_i) \cdot x_i$, which allows us to compute $\text{PERM}(A)$.

Note that the above algorithm for divisibility of the permanent by 4 actually gives us the modulus when $\text{PERM}(A)$ is even. Therefore, we have to devise an algorithm only for the case when $\text{PERM}(A)$ is odd – we reduce this case to the case of the Permanent being even. We prove

Lemma 28. *We can find out the exact value of* $\text{PERM}(A) \bmod 4$ *in* $\oplus\mathbf{L}$.

Proof. Since we have dealt with the situation that if PERM(A) is even, we can find out its value mod 4, here we need to deal with the situation where PERM(A) is not even.

So, suppose PERM(A) is odd. We want to get hold of a suitable *cofactor* of A (call this cofactor $A_{i,j}$) such that PERM($A_{i,j}$) is odd too. Clearly, if PERM(A) is odd, then some minor $A_{i,j}$ of A also has odd determinant (hence odd permanent).

We observe first that given A, we can always find a matrix C (depending on i,j) such that PERM(C) = PERM(A) + PERM($A_{i,j}$). This is easy to do: just increase the $(i,j)^{th}$ entry of matrix A by 1 to get matrix C. Expanding the matrix C by its i^{th} row and using the fact that the permanent function is linear, we get that PERM(C) is equal to sum of the permanents of two matrices, one of which has the same i^{th} row as that of A (and is equal to A), and the other has its i^{th} row consisting wholly of 0's except for the j^{th} entry which is a 1. Expanding the second matrix by its i^{th} row, we find that its permanent is exactly PERM($A_{i,j}$). This proves the equation above.

Given the above, we give a sequential algorithm for finding PERM(A) mod 4, and then we comment on how to parallelize it suitably. Suppose we start with the matrix A with odd Permanent. We can find (by checking all minors of dimension $n-1$) a minor $A_{i,j}$ such that PERM($A_{i,j}$) is also odd. By the above, we can find a matrix C such that PERM(C) equals the sum of the odd permanents, PERM(A) and PERM($A_{i,j}$). Since PERM(C) is even, we can find out its value modulo 4. Thereby we can find out whether PERM(A) $\equiv$ +PERM($A_{i,j}$) or $\equiv$ −PERM($A_{i,j}$) mod 4. Now we can continue with $A_{i,j}$ in order to get a new matrix A_2 (formed by removing *two* rows and *two* columns from A) such that PERM($A_{i,j}$) + PERM(A_2) is even. We continue this process until we reach a matrix of constant dimensions, for which we can evaluate the permanent directly. Thus, we have a sequential process for finding out the Permanent of a matrix modulo 4.

Let us now turn to an efficient parallelization of the above sequential algorithm. Any matrix that is nonsingular (over the relevant field where the entries of the matrix live) admits a decomposition of the form LUP, where L is a lower triangular matrix, U an upper triangular matrix, and P a permutation matrix. It is well known that a matrix has a LU-decomposition over a field , cf. [4], if and only if all its *principal* minors are nonzero in the field. Over $\mathbb{Z}_2$, this translates to all the principal minors being odd. In the above sequential process, we note that if we started with a matrix which has a LU-decomposition, then it is easy (in Logspace) to find out its permanent modulo 4. This is because given a matrix A' (which has odd permanent) in the procedure, we will not have to do any work in order to find a minor of A' which also has odd permanent – the principal minor of A' would already do the job.

Now all that is left to do is the following: given an invertible matrix M, find a permutation matrix P such that MP has a LU decomposition. In other words, we want to find a permutation matrix P such that all the principal minors of MP are odd; and the procedure for finding this P should be in $\oplus\mathbf{L}$.

For this, we closely follow the reduction given in [4] from the above problem to the Determinant. We show thereby that over $\mathbb{Z}_2$, the reduction can be implemented in $\oplus\mathbf{L}$. Eberly [4] reduces the problem of finding a suitable permutation matrix P to rank computations thus: suppose a matrix $A = M^t$ is nonsingular over $\mathbb{Z}_2$ (i.e. has DET $\equiv 1$ mod 2), let A_i be the $n \times i$ matrix formed from the first i columns of A, and let $S_i \subseteq \{1, 2, \cdots, n\}$ be the set of the lexicographically first maximal independent subset of the rows of A_i for $1 \le i \le n$ (i.e S_i consists of the *indices* of the rows of A_i which constitute the lexicographically first maximal independent subset).

For this purpose, let the n rows of A_i be $r_1^i, r_2^i, \cdots, r_n^i$. Consider the matrices A_i^k which have rows $r_1^i, r_2^i, \cdots, r_k^i$ (for instance, A_i^1 consists of just the single row r_1^i, $A_i^n = A_i$). Find out the ranks of these matrices (over $\mathbb{Z}_2$) for $1 \le k \le n$. These can be found in parallel in $\oplus\mathbf{L}$.

Given these ranks, the lexicographically first maximal independent subset of the rows is obtained as follows:

1. *The base case:* Include row r_1^i in the independent subset if and only if rank(A_i^1) = 1;

2. Include row r_j^i in the independent subset if and only if rank(A_i^j) =rank(A_i^{j-1}) + 1 (clearly, since in the matrices A_i^k, we are increasing by a row at a time, the difference of two adjacent ranks can be at most 1).

It is easy to see that this set of j's constitutes the *lexicographically first* maximal independent subset of the rows of A_i. Thereby, we find S_i for each $1 \le i \le n$ in $\oplus\mathbf{L}$.

Now $|S_i| = i$ for $1 \le i \le n$ and $S_i \subset S_{i+1}$ for $1 \le i < n$ (since each S_i is the lexicographically *first* maximal independent subset of the rows of A_i). Set j_1 to be the unique element of S_1, and set j_i to be the unique element of $S_i - S_{i-1}$ for $2 \le i \le n$. Then the desired permutation (matrix) P is such that the i^{th} row of $P^T A$ is the ${j_i}^{th}$ row of A, and can be easily computed in $\mathbf{L}$. Here all the principal minors of $P^t A = P^t M^t$ are odd. Thus we get the required permutation matrix P, which we wanted, such that, MP has all the principal minors odd.

This completes the proof that finding $\mathrm{PERM}(A) \bmod 4$ is in $\oplus\mathbf{L}$. □

Now it is easy to see how we can find out $\mathrm{PERM}(A) \bmod 8$ (say) in $\oplus\mathbf{L}$: we would first check whether A has an even permanent, if it does, then we find the decomposition as in the algorithm outlined above for finding out whether $4|\mathrm{PERM}(A)$, finding out all the values of the submatrices modulo 4 (this gives us the value of $\mathrm{PERM}(A) \bmod 8$). If A has an odd permanent, we reduce it to the even case as above, by finding a suitable P (as in the LUP-decomposition: note however that we do not need to find the L, U factors), and solving the implicit system of equations mod 8. This procedure clearly generalizes to any power 2^k for constant k. This completes the proof of Theorem 27.

Note that the same method as above gives us another proof that $\mathrm{DET}(A) \bmod 2^k$ is in $\oplus\mathbf{L}$. In the end, we note down the following theorem.

Theorem 29. *Finding out the* DET *and* PERM *of a matrix modulo 2^k can be done in complexity $\oplus(SPACE(k^2 \log n))$. Hence for constant k,* DET *and* PERM *modulo 2^k is in $\oplus\mathbf{L}$.*

6 The number of perfect matchings modulo 2^k

In Section 5, we proved that the Permanent of a matrix modulo 2^k (for constant k) can be found in $\oplus\mathbf{L}$. In this section, we want to generalise this result in two directions. As we will see, the techniques of Section 5 may be applied to handle these generalizations. The generalizatoins are motivated by the following questions.

- Is the modulus 2 special?
- Noting that the Permanent exactly embodies the number of perfect matchings in *bipartite* graphs, we want to answer the same question for *arbitrary* graphs, viz. can we find out the number of perfect matchings in arbitrary graphs modulo 2^k in $\oplus\mathbf{L}$?

The first question is slightly speculative, in that, for the Permanent it is indeed true that 2 is special, since modulo 2, the permanent and the determinant of a matrix are the same. But this is the only point where 2 is important as we show below. Note also that Valiant [19] proves hardness results for the Permanent with respect to moduli other than powers of 2.

First we construct other matrix functions which are related to the determinant function.

Given a matrix $A = (a_{ij})$ we define the function

$$h(A) = \Sigma_{even\sigma\in S_n} 2 \cdot a_{1\sigma_1} a_{2\sigma_2} \cdots a_{n\sigma_n} + \Sigma_{odd\sigma\in S_n} a_{1\sigma_1} a_{2\sigma_2} \cdots a_{n\sigma_n} \quad (2)$$

We have thereby created a new matrix function similar to the permanent. First we make the

Claim 30. *The matrix function h is $\#\mathbf{P}$-hard.*

Proof. Given a matrix A consider the two quantities

$$X = \Sigma_{even\sigma\in S_n} a_{1\sigma_1} a_{2\sigma_2} \cdots a_{n\sigma_n} \quad (3)$$

$$Y = \Sigma_{odd\sigma\in S_n} a_{1\sigma_1} a_{2\sigma_2} \cdots a_{n\sigma_n} \quad (4)$$

Then it is transparent that $h(A) = 2X + Y$ while $\mathrm{DET}(A) = X - Y$. If h were computable in polynomial time, so would $2h - \mathrm{DET} = 3X + 3Y$. But $3(X + Y)$ is exactly three times the Permanent of A, which is known to be $\#\mathbf{P}$-hard. □

Now we may observe that modulo 3, the function h satisfies

$$h(A) \equiv -\mathrm{DET}(A). \quad (5)$$

Hence modulo 3, the function h can be computed in $\mathbf{P}$ (in fact, it is complete for $\mathrm{Mod}_3\mathbf{L}$). We may then ask the question as to whether it is easy to compute over all small powers of 3. This is answered by

Claim 31. *The function h modulo 3^k for constant values of k lies in $\mathrm{Mod}_3\mathbf{L}$.*

Proof. We will only give a proof sketch since it is similar to the proof of Theorem 27.

It is clear that finding out the value of $h(A)$ for a matrix A modulo 3 is in $\mathrm{Mod}_3\mathbf{L}$. $\mathrm{Mod}_3\mathbf{L}$-hardness follows from Equation (5).

The proof follows the same route as for Theorem 27: first we prove that finding out whether 9 divides $h(A)$ can be done in $\mathrm{Mod}_3\mathbf{L}$, and then this fact is used to get the value of $h(A)$ modulo 9, which is then used to find out if 27 divides $h(A)$ and so on.

The details are omitted. □

Now we answer the second question regarding the number of perfect matchings in arbitrary graphs modulo 2^k. For brevity, we denote the number of perfect matchings in a graph G by $m(G)$.

We prove the following

Theorem 32. *Finding out the number of perfect matchings in a graph G modulo 2^k (for constant k) can be done in $\mathbf{P}$.*

Proof. As in Theorem 27, we will work in two stages:

- First, we prove that we can find out if 4 divides the number of perfect matchings $m(G)$ in a graph G in $\oplus\mathbf{L} \subseteq \mathbf{P}$.
- Given that we can find out "$m(G) \overset{?}{\equiv} 0 \bmod 4$" in $\mathbf{P}$, we want to find out $m(G) \bmod 4$ in $\mathbf{P}$, which we would then use to resolve $m(G) \overset{?}{\equiv} 0 \bmod 8$ and so on.

We begin with the following

Observation 33. *Given a undirected graph with no self-loops G, the number of perfect matchings in the graph is of the same parity as the determinant of its adjacency matrix $A(G)$.*

Proof. We use the fact that the adjacency matrix of an undirected graph is symmetric. Let the ij^{th} entry of the adjacency matrix A be denoted by a_{ij}.

Consider a typical term in the expression for the determinant of the adjacency matrix: $t_\sigma = a_{1\sigma_1} \cdot a_{2\sigma_2} \cdots a_{n\sigma_n}$, where $\sigma \in S_n$ is a permutation of $\{1, 2, \cdots, n\}$. Here, we are forgetting the sign of the permutation σ since we are working over $\mathbb{Z}_2$.

Decompose the chosen σ into cycles: if one of its constituent cycles is of length > 2, then we derive a companion σ' which is of the same length and also contributes to the determinant mod 2. This σ' is just the inverse of σ: we essentially want to get to the term $t_{\sigma'} = a_{\sigma_1 1} \cdot a_{\sigma_2 2} \cdots a_{\sigma_n n}$. It is easy to see that this term corresponds to σ' being the inverse of σ. Because matrix A is symmetric, $t_{\sigma'}$ has the same value as t_σ. Since we are computing the determinant mod 2, terms t_σ and $t_{\sigma'}$ pair up and vanish. We claim that the only terms remaining are the ones in which every constituent cycle in σ is a 2-cycle.

This can be made clearer by a diagram as in Figure 13. It might make more sense to view a permutation as a *directed* cycle cover of the ground set $\{1, 2, \cdots, n\}$. Given a cycle $(a_1, a_2, \cdots, a_k)$ in permutation σ, we draw edges directed from a_i to a_{i+1} (with wrapping around modulo the length of the cycle). We see that if σ consists entirely of 2-cycles, then the new (directed) graph we get from reversing the *orientations* of the edges is the same: in short, the σ's consisting just of 2-cycles are the fixed points of the action of taking inverse (of a permutation).

But the σ's where every constituent cycle is a 2-cycle correspond exactly to perfect matchings in G. □

This indicates that we have to "use" the adjacency matrix $A(G)$ of the graph G to handle mod 2^k calculations.

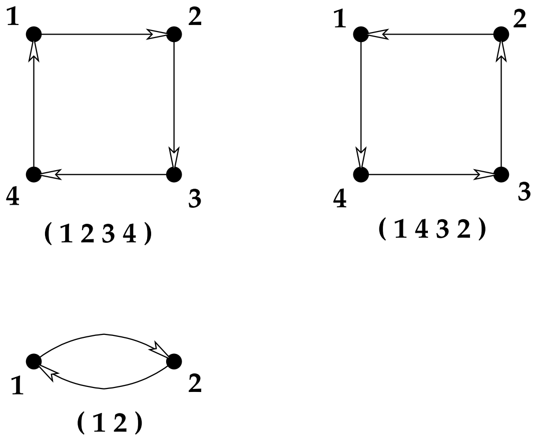

Figure 13. A 4-cycle and a 2-cycle

We proceed to prove that $m(G) \overset{?}{\equiv} 0 \bmod 4$ is in $\oplus\mathbf{L}$. Consider the determinant of the adjacency matrix $A(G)$. If it is odd then it cannot be divisible by 4 and we are done. If the determinant of $A(G)$ is even then we proceed. In that case suppose that rows r_{i_1}, r_{i_2}, ... r_{i_l} sum up to zero modulo 2, then we consider a new graph $\tilde{G}$ which is on the same vertex set as graph G, but vertex v_{i_1} is joined to all the neighbors of vertices $v_{i_2}, \ldots, v_{i_l}$ in G (with multiplicities).

Note that the adjacencies of all the vertices (but for v_{i_1}) are the same across G and $\tilde{G}$. Also observe that even if G did not have loops, there may be a few loops introduced in $\tilde{G}$ (on vertex v_{i_1}). Remove all of these loops from $\tilde{G}$ to get a graph G' – removal of these loops makes sense because a loop on a vertex can never participate in a perfect matching. Note that $\tilde{G}$ (or G') may have multiple edges too, even if the original graph G were simple.

Since G, $\tilde{G}$ and G' all share the same vertex set, we will call the common vertex set V.

Call the set $v_{i_1}, \ldots, v_{i_k}$ the set $X \subseteq V(G') = V$. Denote the neighborhood set of a vertex $v \in V$ by $N(v)$ (the graph in which we are considering the neighborhood relation will be clear from the context). Then, the subset of vertices X in the graph G has the following property: for every $v \in V$, $N(v) \cap X$ is even. This translates to the following property for the graph G': For every $v \in V - \{v_{i_1}\}$, v is joined to v_{i_1} by an *even* number of edges.

Now we state an easy lemma that corresponds to the multilinearity of the Permanent, but applies to perfect matchings in graphs:

Lemma 34. *Given a graph H and a vertex $v \in V(H)$. Look at the set of edges E_v incident on v and consider a partition $E_v = E_1 \cup E_2 \cup \cdots E_k$ (for any suitable value*

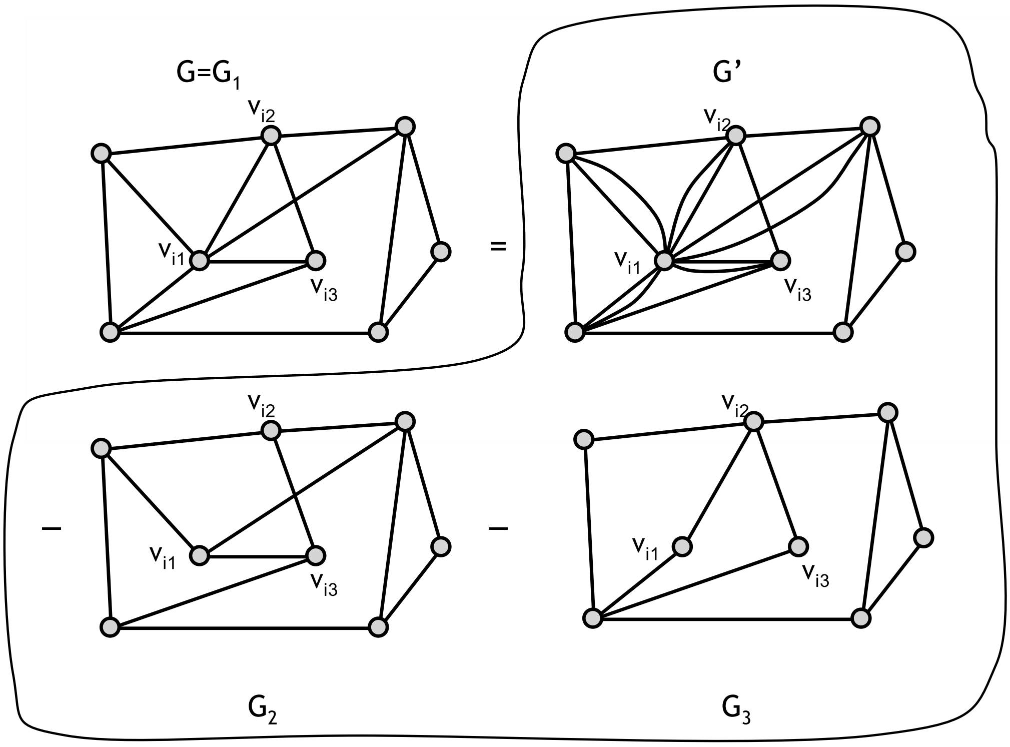

Figure 14. Partitioning the edges of G_1

of k). Define k new graphs H_k as follows: each H_k is defined on the vertex set $V(H)$, and the incidence relation for each pair of vertices $w, z \in V(H) - \{v\}$ is the same as in H. The edges incident on v are exactly the edges in E_k. Then

$$m(H) = m(H_1) + m(H_2) + \cdots m(H_k) \qquad (6)$$

The idea is now to partition the edges incident on v_{i_1} in G' and *allocate* them to several subgraphs so that each of the new subgraphs have

1. an even number of perfect matchings; and
2. and that this can be easily *certified* (this last term being admittedly vague).

Consider the set E' of edges incident on v_{i_1} in G'. The set E' will be partitioned into sets of edges, with it being implicitly understood that an edge set corresponds to a subgraph of G'. We will describe k such subgraphs $G_1, G_2, \cdots, G_k$. The construction would be such that in G_j $(1 \leq j \leq k)$ there is no edge between v_{i_1} and v_{i_j}; also, the neighborhoods of v_{i_1} and v_{i_j} are the same.

Note, that across all the G_j's, the adjacencies of all the vertices (but for v_{i_1}) are the same (since the subgraphs G_j arise from partitioning of edges incident on v_{i_1}). Define G_j as follows: keep an edge between v_{i_1} and some vertex v iff there is an edge between v_{i_j} and v in G.

Observe that $G_1 = G$ by this construction.

It is easy to convince oneself that this describes a partition of E'; see Figure 14.

Thereby, we get a collection of graphs $G_1 = G, G_2, G_3, \cdots, G_k$.

Note that Equation 6 now reads

$$\begin{aligned} m(G') &= m(G_1) + m(G_2) + m(G_3) + \cdots m(G_k) \\ &= m(G) + m(G_2) + m(G_3) + \cdots m(G_k) \qquad (7) \end{aligned}$$

While the graph G' has the property that it has a vertex which shares an even number of edges with every other vertex, the graphs $G_2, G_3 \cdots G_k$ have the property that two vertices in G_j $(2 \leq j \leq k)$ have the same neighbourhood set.

Each of the graphs $G', G_2, G_3, \cdots, G_k$ have $m(\cdot) \equiv 0$ mod 2. The easiness *certificate* for G' is the vertex v_{i_1} which shares an even number of edges with every vertex $v \in V - \{v_{i_1}\}$. The easiness *certificate* for G_j $(2 \leq j \leq k)$ are the two vertices v_{i_1} and v_{i_j} which have the same neighborhood set. Intuitively, the easiness certificate for a graph X with $m(X) \equiv 0$ mod 2 gives a short reason as to why $m(X)$ is even.

In either of the above cases, it is easy to find out $m(\cdot)$ mod 4 in $\oplus\mathbf{L}$ (as this computation reduces to a few mod 2 computations) – this is similar to the corresponding calculation in the proof of Theorem 27. Altogether from Equation (7), we see that we can resolve the question $m(G) \overset{?}{\equiv} 0$ mod 4 in $\oplus\mathbf{L}$.

Now we consider the problem of getting the exact value of $m(G)$ mod 2^k (for constant k). As in Theorem 27, we will show how to get the value of $m(G)$ mod 4, then use this to resolve $m(G) \overset{?}{\equiv} 0$ mod 8 and continue to find out the value mod 2^k (for constant k).

As a first step,

Lemma 35. *Suppose graph G has $m(G)$ odd, then we can find a subgraph G^1 of G in polynomial time, such that $m(G^1)$ is odd.*

Proof. Consider a vertex $v \in V(G)$, and look at the set of edges incident on it. Apply Lemma 34 to $m(G)$, with the corresponding partition consisting of single edges. Equation (6) then describes $m(G)$ as the sum of some other $m(\cdot)$'s. Since $m(G)$ is odd, some term on the RHS of Equation 6 has to be odd too. □

In fact, the subgraph G^1 has a single edge going out of v (v becomes a *pendant* vertex); say that this single edge is the edge (v, u). Note that the edge (v, u) has to be present in any perfect matching of G^1, so G^1 may also be thought of as $G - \{u, v\}$ (for purposes of consideration of the perfect matchings).

We can iterate this process to yield the following: Given a graph G with $m(G)$ odd, we can find in polynomial time subgraphs $G^1, G^2, \cdots$ (G^{i+1} is a subgraph of G^i) each of which has an odd number of perfect matchings.

Now we claim that we can construct a graph H_1 such that $m(H_1) = m(G) + m(G^1)$. In the graph G^1, the vertex v has degree 1; let the only edge incident on v be (v, u). Construct H_1 as follows: in the graph G, add a path of length 3 between vertices v and u. It is easy to observe that $m(H_1) = m(G) + m(G^1)$. Likewise we can construct H_2 such that $m(H_2) = m(G^1) + m(G^2)$ and so on.

For each of the graphs $H_1, H_2, \cdots$, we can find out $m(\cdot)$ mod 4 (given that $m(H_i)$ for $i \geq 1$ is even). Finally we can solve this simple system of linear equations in $m(G), m(G^1), \cdots$ in $\mathbf{P}$. This gives us the value of $m(G)$ mod 4 in $\mathbf{P}$. By iterating the above process, the result generalizes for any constant k following the proof of Theorem 27.

Note that it is the step which involves Lemma 35 that causes the whole procedure to lie in $\mathbf{P}$ rather than $\oplus\mathbf{L}$. □

7 Conclusion and Open Problems

We have shown that the Laplacian matrix for a planar graph encodes useful information for computing the number of spanning trees, modulo small powers of 2, but does not reduce the complexity of the same computation for odd prime moduli. One may ask, how about the adjacency matrix? Does planarity help in computing say, the rank of the adjacency matrix of a graph over $\mathbb{Z}_2$? We can also show that this is not the case; in fact, computing the rank of the adjacency matrix of a cubic planar graph (over $\mathbb{Z}_2$) is hard for $\oplus\mathbf{L}$. Details will appear in the full version of the paper.

As we mentioned earlier, our proof that computing $\tau(G)$ mod 2^k for planar G seems to take recourse to graphs of higher genera. We also have a distinct proof for the special case of $k = 2$; one that does not go through higher genus surfaces. This last is purely graph theoretic; unfortunately, it does not seem to extend to higher values of k. Is there a proof of Theorem 17 that is purely graph theoretic and does not involve higher genus surfaces when we are restricting ourselves to planar graphs?

As mentioned in the Introduction, the PERM function is but an incarnation of the number of perfect matchings in bipartite graphs. Taking our cue from here, we may also ask – can we count the number of perfect matchings in *arbitrary* graphs modulo 2^k (for constant k) in $\oplus\mathbf{L}$? Note that the problem for non-bipartite graphs doesn't translate to a permanent computation. Yet, we have shown that this counting problem lies in $\mathbf{P}$. Also, it is easy to see that we can indeed count them modulo 2 in $\oplus\mathbf{L}$. This would seem to imply that this question may be answered affirmatively. The complexity of computing the permanent modulo non-constant powers of 2 remains open.

Acknowledgments. We would like to thank Dror Bar-Natan for his advice on Algebraic Topology and for referring us to Theorem 7. We are grateful to Eric Allender for encouragement to work on this problem and thorough perusal of some of the proofs. We would like to thank Stephen Cook for his advice on the reduction in [2] being in $\mathbf{AC}^0$. We also thank Vinayaka Pandit for carefully reading through the introduction.

References

[1] G. Buntrock, C. Damm, U. Hertrampf, and C. Meinel. Structure and importance of logspace-MOD-classes. In *Symposium on Theoretical Aspects of Computer Science*, pages 360–371, 1991.

[2] S. A. Cook and P. McKenzie. Problems complete for deterministic logarithmic space. *J. Algorithms*, 8(3):385–394, 1987.

[3] R. Diestel. *Graph Theory.* Springer Verlag, Heidelberg, 3 edition, 2005.

[4] W. Eberly. Efficient parallel independent subsets and matrix factorizations, 1991.

[5] D. Eppstein. On the parity of graph spanning tree numbers. Technical Report 96-14, Univ. of California, Irvine, Dept. of Information and Computer Science, Irvine, CA, 92697-3425, USA, 1996.

[6] W. Fulton. *Algebraic Topology: A First Course.* Number 153 in Graduate Texts in Mathematics. Springer-Verlag, New York, NY, 1995.

[7] I. M. Gessel and X. G. Viennot. Determinants, paths, and plane partitions.

[8] C. Godsil and G. Royle. *Algebraic Graph Theory.* Springer Verlag, New York, 1st. edition, 2001.

[9] A. Hatcher. *Algebraic Topology.* Cambridge University Press, 1st. edition, 2002.

[10] W. Hesse, E. Allender, and D. A. M. Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *Journal of Computer and System Sciences*, 65:695–716, 2002.

[11] K. Hoffman and R. Kunze. *Linear Algebra.* Prentice Hall, USA, 2 edition, 1971.

[12] P. W. Kasteleyn. Graph theory and crystal physics. *Graph Theory and Theoretical Physics*, pages 44–110, 1967.

[13] H. Minc. *Permanents.* Encyclopaedia of Mathematics and its Applications. Cambridge University Press, Cambridge, UK, New Ed edition, 1984.

[14] J. R. Munkres. *Topology.* Prentice Hall; 2nd edition, USA, 1999.

[15] O. Reingold. Undirected st-connectivity in log-space. In *Proceedings 37th Symposium on Foundations of Computer Science*, pages 376–385. IEEE Computer Society Press, 2005.

[16] H. Shank. The theory of left-right paths. *Combinatorial Mathematics III, Proceedings of 3rd Australian Conference, St. Lucia; Lecture Notes in Mathematics*, 452:42–54, 1975.

[17] H. N. V. Temperley and M. E. Fisher. Dimer problem in statistical mechanics - an exact result. *Philosophical Magazine*, 6:1061–1063, 1961.

[18] S. Toda. PP is as hard as the polynomial hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991.

[19] L. G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/