Electronic Colloquium on Computational Complexity, Report No. 30 (1999)

A Combinatorial Algorithm for Pfaffians[1]

Meena Mahajan[2]
Institute of Mathematical Sciences, Chennai, INDIA.

P. R. Subramanya & **V. Vinay**[3]
Dept. of Computer Science & Automation, Indian Institute of Science, Bangalore, INDIA.

Abstract

The Pfaffian of an oriented graph is closely linked to Perfect Matching. It is also naturally related to the determinant of an appropriately defined matrix. This relation between Pfaffian and determinant is usually exploited to give a fast algorithm for computing Pfaffians.

We present the first completely combinatorial algorithm for computing the Pfaffian in polynomial time. Our algorithm works over arbitrary commutative rings. Over integers, we show that it can be computed in the complexity class GapL; this result was not known before. Our proof techniques generalize the recent combinatorial characterization of determinant [MV97] in novel ways.

As a corollary, we show that under reasonable encodings of a planar graph, Kasteleyn's algorithm [Kas67] for counting the number of perfect matchings in a planar graph is also in GapL. The combinatorial characterization of Pfaffian also makes it possible to directly establish several algorithmic and complexity theoretic results on Perfect Matching which otherwise use determinants in a roundabout way.

We also present hardness results for computing the Pfaffian of an integer skew-symmetric matrix. We show that this is hard for $\sharp$L and GapL under logspace many-one reductions.

1 Introduction

The main result of this paper is a combinatorial algorithm for computing the Pfaffian of an oriented graph. This is similar in spirit to a recent result of Mahajan & Vinay [MV97] who give a combinatorial algorithm for computing the determinant of a matrix. In complexity theoretic terms, we establish that computing the Pfaffian of a graph is in the class GapL.

GapL is the class of functions that are logspace reducible to computing the integer determinant of a matrix. It is known that computing the determinant of a matrix is equivalent to taking the difference of two $\sharp$L functions [Vin91, Dam91, Val92, Tod91]. In other words, GapL is the

[1]An extended abstract describing most of these results appeared in the Proceedings of the Fifth Annual International Computing and Combinatorics Conference COCOON 1999, in the Springer-Verlag Lecture Notes in Computer Science series Volume 1627, pp. 134–143.

[2]Part of this work was done when this author was supported by the NSF grant CCR-9734918 on a visit to Rutgers University during summer 1999.

[3]This work was initiated when this author was visiting DIMACS at Rutgers University during summer 1998.

ISSN 1433-8092

class of functions that can be expressed as the difference in the number of accepting paths of two nondeterministic logspace machines. They define a space analog of the important counting classes GapP and $\sharp\mathsf{P}$.

Pfaffians are intimately connected to determinants. For example, it is known that the square of the Pfaffian of a graph is equal to the determinant of a related matrix. This is, however, not adequate to imply our GapL algorithm as we do not know if GapL is closed under square roots (of positive integers).

One of the motivations for this work is to understand the complexity of Perfect Matching. Perfect Matching is not known to be in NC, but is known to be in RNC [MVV87]. This result has been recently improved by Allender & Reinhardt [AlRe98] who show that Perfect Matching is in the class SPL (non-uniformly). (SPL is that subclass of GapL where functions take the value 0 or 1. Refer to [AlRe98] for details.) Interestingly, Allender & Reinhardt make use of the Mahajan-Vinay clow sequences [MV97] critically to establish their result. We hope that our combinatorial characterization of Pfaffian will be a key in resolving the vexed question of the complexity of Perfect Matching. This is indeed our main motivation for this work.

Pfaffians arise naturally in the study of matchings; the pfaffian of an oriented graph is just the sum over all possible perfect matchings except that each matching has an associated sign as well, dictated by the orientation. This gives it a flavour similar to that of a determinant. In the absence of the sign, they would calculate the number of perfect matchings in a graph, a problem that is well-known to be complete for $\sharp\mathsf{P}$ [Val79]. Also, in the case of special graphs, it is known that the graph may be oriented in such a way that all the terms of the pfaffian turn out to be positive. This obviously means there would be no cancellation and hence the pfaffian would count the number of perfect matchings in the underlying graph. Such orientations of graphs are called Pfaffian orientations.

It is easy to construct graphs which *do not* admit a pfaffian orientation; $K_{3,3}$ is one such graph. A celebrated result of Kasteleyn [Kas67] proves that all planar graphs admit a pfaffian orientation. This result was subsequently improved by [Lit74] who showed that all $K_{3,3}$-free graphs admit a pfaffian orientation. Finding such an orientation was shown to be in NC by Vazirani [Vaz89]. In this paper, we partially improve Vazirani's result to show that for planar graphs presented by reasonable encodings, a pfaffian orientation can be found in L. Combining this with our combinatorial algorithm for pfaffians, we thus show that under reasonable encodings of planar graphs, the problem of counting the number of perfect matchings in a planar graph is in GapL as well. The problem of extending our result to $K_{3,3}$-free graphs remains to be investigated.

In section 2, a few preliminaries and definitions are stated. In section 3, we set up the combinatorial framework for pfaffians. Section 4 focuses on the combinatorial algorithm for computing Pfaffians. We show in section 5 that finding pfaffian orientations of planar graphs is in L, and hence counting the number of perfect matchings in a planar graph, is in GapL. In section 6 we show that computing the Pfaffian of an integer skew-symmetric matrix is hard for both $\sharp\mathsf{L}$ and GapL.

2 Preliminaries & Definitions

Let D be an $n \times n$ matrix. S_n is the permutation group on $\{1, 2, \ldots, n\}$ (denoted $[n]$). The permanent and determinant of D, $per(D)$ and $det(D)$, are defined as,

$$per(D) = \sum_{\sigma \in S_n} \prod_i d_{i\sigma(i)} \qquad det(D) = \sum_{\sigma \in S_n} sgn(\sigma) \prod_i d_{i\sigma(i)}$$

where $sgn(\sigma)$ is -1 if σ has an odd number of inversions, $+1$ otherwise. An equivalent definition of the sign of a permutation is in terms of the number of cycles in its cycle decomposition.

We associate with the matrix D the graph G_D, which is the complete directed graph on n vertices (with self-loops), having the matrix elements as edge weights. Every permutation $\sigma \in S_n$ can be decomposed into a set of cycles in G_D. The cycles are non-intersecting (i.e. simple), disjoint and they cover every vertex in the graph, i.e. these are **cycle covers**. The sign of a cycle cover is defined in terms of the number of even length cycles constituting it. The sign is $+1$ if there are an even number of such cycles, else it is -1.

A **clow** in G_D is a walk that starts at some vertex (called *head*), visits vertices larger than the head any number of times, and returns to the head. This cycle in G_D is not always a simple cycle. Formally,

Definition 1 [MV97]

1. *A clow is an ordered sequence of edges $C = \langle e_1, e_2, \ldots, e_m \rangle$ such that $e_i = \langle v_i, v_{i+1} \rangle$ and $e_m = \langle v_m, v_1 \rangle$, $v_1 \neq v_j$ for $j \in \{2, 3, \ldots, m\}$ and $v_1 = min\{v_1, \ldots, v_m\}$. The vertex v_1 is called the head of the clow and denoted $h(C)$. The length of the clow is $|C| = m$, and the weight of the clow is $wt(C) = \prod_{i=1}^{m} wt(e_i)$. [Note: $C = \langle e \rangle$ where $e = \langle v, v \rangle$, i.e. a* self-loop*, is also a clow, of length one.]*
2. *A clow sequence is an ordered sequence of clows $\mathcal{C} = (C_1, \ldots, C_k)$ such that $h(C_1) < \ldots < h(C_k)$ and $\sum_{i=1}^{k} |C_i| = n$.*

Pfaffians were introduced by Kasteleyn [Kas67] to count the number of dimer coverings of a lattice graph. We define matchings and Pfaffians more formally.

Definition 2 *Given an undirected graph $G = (V, E)$ with $V = \{1, 2, \ldots, n\}$, we define*

1. *A* **matching** *$\mathcal{M}$, is a subset of the edges of G such that no two edges have a vertex in common. That is, $\mathcal{M} \subseteq E(G)$ such that $e_1, e_2 \in \mathcal{M}$, $e_1 = (i_1, j_1), e_2 = (i_2, j_2)$ and $i_1 = i_2 \Leftrightarrow j_1 = j_2$.*
2. *A matching $\mathcal{M}$ is a* **perfect matching** *if every vertex $i \in V(G)$ occurs as the end-point of some edge in $\mathcal{M}$.*

Thus a perfect matching is a partition of the vertices of G into $\frac{n}{2}$ unordered pairs, where each pair is an edge. We will in the sequel prove our results for graphs with integer weights on edges, although our results can easily be seen to hold over arbitrary commutative rings. The weight of a matching is the product of the weights of its constituent edges.

Given an undirected graph G, assign orientations to edges of G to get a directed graph $\vec{G}$. The *Tutte Matrix* associated with $\vec{G}$ is the skew-symmetric adjacency matrix defined as

$$\begin{array}{rclr l} A_s(\vec{G})_{ij} & = & w(i,j) & \text{if } \langle i,j\rangle \text{ is an edge in } \vec{G} \\ & = & -w(i,j) & \text{if } \langle j,i\rangle \text{ is an edge in } \vec{G} \\ & = & 0 & \text{if } (i,j) \text{ is not an edge in } G \end{array}$$

Here $w(i,j)$ refers to the weight of the undirected edge (i,j) in G.

(Note that a skew-symmetric matrix does not automatically give us an orientation unless we assume that weights are non-negative, i.e. it does not precisely tell us the edge weights in the underlying undirected graph.)

The pfaffian of a skew-symmetric matrix D, or equivalently, of an orientation of an undirected graph, is defined as,

$$\mathsf{Pf}(D) = \sum_{\mathcal{M}} p(\mathcal{M})$$

where the sum ranges over all perfect matchings $\mathcal{M}$. In order to define the pfaffian term, $p(\mathcal{M})$, corresponding to a perfect matching $\mathcal{M}$, we require a few preliminaries.

Let σ be a permutation in S_n. We can think of σ as representing the matching $\{\langle\sigma(1),\sigma(2)\rangle, \langle\sigma(3),\sigma(4)\rangle, \ldots, \langle\sigma(n-1),\sigma(n)\rangle\}$. Several permutations correspond to a matching because the edges in the matching are neither oriented nor ordered (in fact, there are exactly $2^{\frac{n}{2}} \cdot (\frac{n}{2})!$ permutations that represent a matching). The standard definition of the sign of a permutation holds. That is, $sgn(\sigma)$ is $+1$ if an even number of transpositions convert σ to the identity matching, and -1 otherwise. The weight of the permutation is defined as

$$w(\sigma) = \prod_{i=1}^{\frac{n}{2}} D_{\sigma(2i-1)\sigma(2i)}$$

Consider a matching $\mathcal{M}$. Irrespective of which permutation σ one chooses to represent $\mathcal{M}$, the term $sgn(\sigma)w(\sigma)$ is invariant. That is, let σ and σ' both represent $\mathcal{M}$. If σ differs from σ' in one edge being flipped, then the signs of the permutations are different but so are their weights. If σ and σ' differ in the arrangement of edges, then the number of transpositions to convert σ to σ' is even, and therefore their signs are the same.

So, the pfaffian term corresponding to a matching $\mathcal{M}$ is defined to be $p(\mathcal{M}) = sgn(\sigma)w(\sigma)$, where σ is any permutation representing $\mathcal{M}$.

The canonical permutation for any matching $\mathcal{M}$, denoted $\sigma_{\mathcal{M}}$, is the permutation where edges are from smaller to larger vertices and are listed in increasing order of the smaller vertices in each edge, i.e. $\sigma_{\mathcal{M}}(2l-1) < \sigma_{\mathcal{M}}(2l)$ for $l = 1, \ldots, \frac{n}{2}$, and $\sigma_{\mathcal{M}}(1) < \sigma_{\mathcal{M}}(3) < \ldots < \sigma_{\mathcal{M}}(n-1)$. Using these, the pfaffian of the skew-symmetric matrix D may be defined as

$$\mathsf{Pf}(D) = \sum_{\mathcal{M}} sgn(\sigma_{\mathcal{M}}) \cdot w(\sigma_{\mathcal{M}})$$

where the sum ranges over all perfect matchings $\mathcal{M}$.

Consider the graph given in Figure 1. Distinct variables are used to represent the edge weights, still unspecified. The associated matrix D for the graph is

$$D = \begin{bmatrix} 0 & a & b & e \\ -a & 0 & c & 0 \\ -b & -c & 0 & d \\ -e & 0 & -d & 0 \end{bmatrix}$$

Possible Matchings	**Terms**
(1 2) (3 4)	$+1.a.d$
(1 4) (2 3)	$+1.e.c$
(1 3) (2 4)	$-1.b.0$

$$\begin{aligned} \mathsf{Pf}(D) &= a \cdot d + c \cdot e - 0 \cdot b \\ &= a \cdot d + c \cdot e \end{aligned}$$

Each Pfaffian term corresponds to a possible perfect matching in the graph. The non-vanishing terms correspond to feasible perfect matchings.

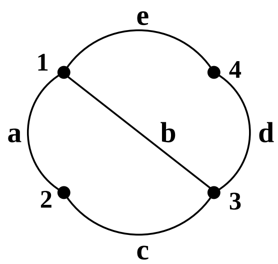

Figure 1: An Example Graph

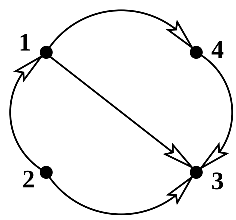

Figure 2: An Oriented Example Graph

Fig 2 imposes an orientation on the graph in Fig 1. Assuming that all the edge weights are $+1$, this amounts to assigning ± 1 to the variables, and results in the matrix D given below. Now, comparing $per(D)$, $det(D)$ and $\mathsf{Pf}(D)$, we have

$$D = \begin{bmatrix} 0 & -1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ -1 & -1 & 0 & -1 \\ -1 & 0 & 1 & 0 \end{bmatrix} \qquad \begin{array}{c} per(D) = 2 \\ det(D) = 4 \\ \mathsf{Pf}(D) = 2 \end{array}$$

Using Linear Algebra we can prove the following properties of skew symmetric matrices D.

- If D has an odd number of rows, then $det(D) = 0$.
- If D has an even number of rows, then $det(D) = (\mathsf{Pf}(D))^2$.

Let G be a directed acyclic graph with integer weights on its edges, and with three special vertices s, t_+ and t_-. Consider a function f defined as,

$$f = \sum_{\rho:\ s \rightsquigarrow t_+} wt(\rho) \quad - \sum_{\eta:\ s \rightsquigarrow t_-} wt(\eta)$$

where ρ iterates over all paths from s to t_+, η over all s to t_- paths, and $wt(\rho)$ or $wt(\eta)$ is the weight of the path (i.e. the product of the weights of all the edges on the path). GapL is precisely the class of functions that can be formulated in this fashion. Stated somewhat differently, GapL consists of those functions that are the difference of two $\sharp$L functions, where $\sharp$L is the counting class for NL. As stated earlier, GapL is the class of languages logspace reducible to computing the integer determinant. The Mahajan & Vinay GapL algorithm for the determinant [MV97] formulates the determinant as described above. This GapL algorithm can be used to compute $det(D)$, viz. $[\mathsf{Pf}(D)]^2$. However, this does not immediately yield a GapL algorithm for the Pfaffian itself, because GapL is not known to be closed under square roots.

Let F_1 and F_2 be perfect matchings in a graph G. Their superposition, $F_1 \cup F_2$, is the graph obtained by including all closed walks along edges alternately from F_1 and F_2. Start at a vertex and walk along its matched edge in F_1. Next, walk along an adjacent edge in F_2. If this closes a cycle, pick an unvisited vertex and start the closed walks on the remaining vertices. Else, continue walking till there are no more matched edges. $F_1 \cup F_2$ is a cycle cover of G where each cycle is an alternating cycle and has even length. Note that each cycle in $F_1 \cup F_2$ can be routed in either of two possible directions. Generalizing Kasteleyn's notation for cycle covers on the regular 2-D lattice, we call the two possible routings clockwise and anti-clockwise. By clockwise routing we mean that routing where the first vertex is the smallest vertex in the cycle and the first edge of a cycle is picked from F_1.

Suppose we impose an orientation on the edges of G to get a directed graph $\vec{G}$. A cycle C in $F_1 \cup F_2$, when routed in any particular way, may traverse some edges according to their orientation in $\vec{G}$ and some edges in a direction opposite to their orientation. C is said to have an even orientation with respect to $\vec{G}$ if the number of properly oriented edges along any routing of C is even. Otherwise, C has an odd orientation. As every cycle in the superposition of two matchings is of even length, the orientation of a cycle is independent of the routing (clockwise or anti-clockwise).

3 A Combinatorial Setting for Pfaffians

In this section, we build the combinatorial framework for pfaffians using a variant of *clow sequences*. We also provide a new characterization for the sign of a pfaffian term. We shall utilize this characterization in our combinatorial algorithm for computing pfaffians.

We will require a variant of a standard lemma (See Lemma 8.3.1 from [LovPlu86]) for the cases when the edges have arbitrary integer weights.

Lemma 3 *Let $\vec{G}$ be an arbitrary orientation of an undirected graph G. Let F_1 and F_2 be two* perfect matchings *of G. Let k be the number of evenly oriented alternating cycles in $F_1 \cup F_2$. Then,*

$$p(F_1) \cdot p(F_2) = (-1)^k \cdot w(F_1) \cdot w(F_2)$$

Here $w(F) = \prod_{e \in F} w(e)$, where D is the skew-symmetric adjacency matrix of $\vec{G}$, and if $e = (i,j)$, and $\vec{G}$ orients the edge (i,j) from i to j, then $w(e) = D_{ij}$.

Sketch of Proof: $F_1 \cup F_2$ consists of cycles of even length. Consider the case when $F_1 \cup F_2$ consists of just one non-trivial cycle C. Choose the clockwise routing of C. Represent F_1 and F_2 by those permutations π and τ respectively, where the edges in C are listed in the order in which they appear in this clockwise routing, and the other edges are listed identically. Now it is clear that to go from the permutation π to the permutation τ, we need an odd number of transpositions. So the signs of these permutations are opposing.

(For instance, let $F_1 = (1,2)(3,4)(5,6)(7,8)$ and $F_2 = (1,6)(2,3)(4,5)(7,8)$. The cycle 123456 in $F_1 \cup F_2$ implies
$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix}$ and $\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 4 & 5 & 6 & 1 & 7 & 8 \end{pmatrix}$.
In order to convert π to τ, 1 has to be moved right over *five* vertices.)

As regards the weights, the edges for trivial cycles contribute the same to the weights of both permutations, and so the product is identical to the product of the weights of these edges in the matchings. A non-trivial cycle contributes the remaining weight with a -1 thrown in for each edge traversed wrongly. So, if the number of wrongly traversed edges is odd (i.e. the cycle is oddly oriented with respect to $\vec{G}$), then the net contribution is a -1 over and above the weights of the matchings. i.e. $w(\pi) \cdot w(\tau) = -w(F_1) \cdot w(F_2)$. This -1 will nullify the corresponding -1 in the product of the signs. On the other hand, if the number of edges traversed wrongly is even, then $w(\pi) \cdot w(\tau) = w(F_1) \cdot w(F_2)$, and the -1 in the sign is not nullified.

So, if the lone non-trivial cycle C in $F_1 \cup F_2$ is oddly oriented, then $p(F_1) \cdot p(F_2) = w(F_1) \cdot w(F_2)$. It is when C is evenly oriented that a -1 is introduced, i.e. $p(F_1) \cdot p(F_2) = -w(F_1) \cdot w(F_2)$. Extending this argument, it is evident that if there are several non-trivial cycles in $F_1 \cup F_2$, then a -1 is introduced by each evenly oriented cycle. This proves the lemma.

■

The idea is to compute all pfaffian terms of a skew-symmetric matrix D with respect to the base matching $\mathcal{I}$ corresponding to the identity permutation. Consider the matching $\mathcal{M}$ and its superposition with $\mathcal{I}$. Given a skew-symmetric matrix D, select the orientation G^f where each edge is oriented from its smaller endpoint to its larger endpoint. With respect to this orientation, we will consider only canonical permutations to represent each matching. Using Lemma 3, we can show the following.

Corollary 4

$$sgn(\sigma_{\mathcal{M}}) = (-1)^k$$

where k is the number of cycles in $\mathcal{M} \cup \mathcal{I}$ that are evenly oriented with respect to G^f.

Proof: From Lemma 3, we have

$$\begin{aligned} sgn(\sigma_{\mathcal{M}}) \cdot wt(\sigma_{\mathcal{M}}) \cdot wt(\sigma_{\mathcal{I}}) &= [sgn(\sigma_{\mathcal{M}}) \cdot wt(\sigma_{\mathcal{M}})] \cdot [sgn(\sigma_{\mathcal{I}}) \cdot wt(\sigma_{\mathcal{I}})] \\ &= p(\mathcal{M}) \cdot p(\mathcal{I}) \\ &= (-1)^k \cdot wt(\sigma_{\mathcal{M}}) \cdot wt(\sigma_{\mathcal{I}}) \end{aligned}$$

Therefore, $sgn(\sigma_{\mathcal{M}}) = (-1)^k$. ■

We show another characterization of the sign of the canonical permutation of a partition.

Lemma 5

Let $\mathcal{M}$ be a partition of $[n]$ into $\frac{n}{2}$ unordered pairs and $\sigma_{\mathcal{M}}$ be its canonical permutation. Let $\mathcal{I}$ be the base partition corresponding to the identity permutation. Let $\mathcal{M} \cup \mathcal{I}$ have l cycles, and let $\mathcal{C}$ denote the cycle cover obtained by the clockwise routing of each cycle in $\mathcal{M} \cup \mathcal{I}$. The pfaffian of a skew-symmetric matrix D is given by

$$\mathsf{Pf}(D) = \sum_{\mathcal{M}} w(\sigma_{\mathcal{M}}) \cdot (-1)^{|\{\langle i,j \rangle \;:\; \langle i,j \rangle \;\in\; \mathcal{C},\; i<j\}| \;\;+\;\; l}$$

Proof: The standard way to characterize the sign of a pfaffian term is by the number of evenly oriented cycles. The claim is that the number of cycles plus the number of properly oriented edges also characterizes the sign.

Let $\mathcal{C}$ have k even cycles and m odd cycles with respect to the forward orientation; $l = k + m$. Define $E = \{\langle i, j\rangle : \langle i, j\rangle \in \mathcal{C}, i < j\}$, the set of properly oriented edges. Let the contributions to $|E|$ from each of the even and odd oriented cycles be e_i and o_j for $1 \leq e_i \leq k$ and $1 \leq j \leq m$. Thus $|E| = \sum_{i=1}^{k} e_i + \sum_{j=1}^{m} o_j$. Note that each e_i is even and each o_j is odd. Thus $|E| + l = \sum_{i=1}^{k} e_i + \sum_{j=1}^{m} o_j + k + m = \sum_{i=1}^{k} e_i + \sum_{j=1}^{m} (o_j + 1) + k \equiv k \bmod 2$. Now the result follows from Corollary 4. ■

Corollary 6

Let $\mathcal{M}$ be a partition and $\mathcal{I}$ be the identity permutation. The sign of $\mathcal{M}$ in the pfaffian is,

$$sgn(\sigma_{\mathcal{M}}) = (-1)^{|FE|+|BO|+l}$$

where l is the number of cycles in $\mathcal{M} \cup \mathcal{I}$, $\mathcal{C}$ is the orientation of $\mathcal{M} \cup \mathcal{I}$ with each cycle routed in the clockwise sense and, FE and RO are sets of edges of $\mathcal{M}$ defined as,

$$\begin{aligned} FE &= \{\langle i, 2j\rangle \;:\; \langle i, 2j\rangle \;\in\; \mathcal{M} \cap \mathcal{C},\; i < 2j \;\} \\ RO &= \{\langle i, 2j-1\rangle \;:\; \langle i, 2j-1\rangle \;\in\; \mathcal{M} \cap \mathcal{C},\; i > 2j-1 \;\} \end{aligned}$$

Proof: Lemma 5 tells us that we need to keep track of the parity of properly oriented (i.e. forward) edges in the clockwise routing of cycles in $\mathcal{M} \cup \mathcal{I}$. Instead here, let us focus on the edges of $\mathcal{M}$ alone, and hold an edge of $\mathcal{M}$ responsible for the following $\mathcal{I}$ edge. Each edge of $\mathcal{M}$ contributes 0, 1 or 2 forward edges to $\mathcal{M} \cup \mathcal{I}$. For instance, suppose the clockwise routing encounters edge $\langle i, 2j-1\rangle$ from $\mathcal{M}$, where $i < 2j-1$. Then it also encounters the edge $\langle 2j-1, 2j\rangle$ from $\mathcal{I}$, and both these edges are properly oriented. The other cases can be argued similarly.

Edge of $\mathcal{M}$ as in $\mathcal{C}$	condition	no. of forward edges in $\mathcal{C}$
$\langle i, 2j-1\rangle$	$i < 2j-1$	2
$\langle i, 2j-1\rangle$	$i > 2j-1$	1
$\langle i, 2j\rangle$	$i < 2j$	1
$\langle i, 2j\rangle$	$i > 2j$	0

So, to evaluate the parity of forward edges, it suffices to keep track of the edges for the middle two cases, and none for the first and last cases. The sets FE and RO precisely do this. ■

We need a variant of clows called **pclows**[4] for our combinatorial setting for pfaffians.

Definition 7

- *A pair of edges $E = (e_1, e_2)$ is a* **p-edge** *if for some $i \in [1, n]$ either,*
  1. $e_1 = \langle i, 2j\rangle$ *and* $e_2 = \langle 2j, 2j\text{-}1\rangle$, *or*
  2. $e_1 = \langle i, 2j\text{-}1\rangle$ *and* $e_2 = \langle 2j\text{-}1, 2j\rangle$.
- *A* **pclow** *is a clow with its ordered sequence of edges being $P = \langle E_1, E_2, \ldots, E_m\rangle$ where each E_i is a p-edge. The length of the pclow is $2m$. A pclow traversal begins from its smallest vertex (called the* head*).*
- *A* **pclow sequence** *is an ordered sequence of pclows, $\mathcal{P} = \langle P_1, \ldots, P_k\rangle$ with heads in strictly increasing order, and with $\sum_{i=1}^{k} |P_i| = n$.*
- *Define the* **sign of a pclow sequence** *to be the parity of the number of evenly oriented pclows (with respect to G^f).*
- *The* **weight of a** *p*-**edge** *$E = (e_1, e_2)$ is the weight of the edge e_1 in its forward direction, i.e. if $e_1 = (i, j)$, its weight is w_{ij} if $i < j$ and w_{ji} otherwise. The second edge, e_2, always contributes a* **1** *to the weight of E. The* **weight of a pclow** *is the product of the p-edge weights. The* **weight of a pclow sequence** *is the product of the weights of its pclows.*

Thus, if a pclow sequence $\mathcal{P}$ actually represents a perfect matching $\mathcal{M}$, then its weight is $w(\sigma_{\mathcal{M}})$, and its sign is the sign of $\sigma_{\mathcal{M}}$. The results of Lemma 5 and Corollary 6 generalize to pclow sequences as well; Lemma 5 tells us that the sign of a pclow sequence is the parity of the number of pclows in it plus the number of edges traversed in the *forward* direction.

[4]**Pclows** expand to Pfaffian Closed Walks and **p-edge** stands for pfaffian-edge.

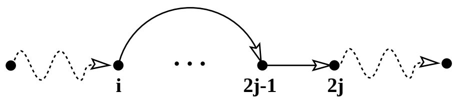

Figure 3: Selecting $\langle i, 2j-1 \rangle$ from i.

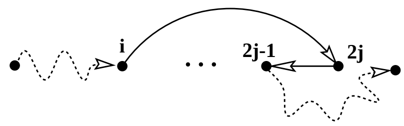

Figure 4: Selecting $\langle i, 2j \rangle$ from i.

Figures 3 & 4 indicate the cases when a pair of consecutive edges are p-edges. Using pclow sequences, we prove a novel and powerful characterization of the Pfaffian. This provides the basis for our combinatorial algorithm for computing pfaffians.

Theorem 8

$$\mathsf{Pf}(D) = \sum_{\mathcal{W}:\ pclow\ \ sequence} sgn(\mathcal{W}) \cdot wt(\mathcal{W})$$

<u>Proof:</u> Pclow sequences that are cycle covers are the superposition of the base matching with a perfect matching. We need to show that pclow sequences that are not cycle covers do not contribute to the summation. We establish an involution on the set of pclow sequences. Non-cycle covers get mapped onto non-cycle covers of opposite signs. The fixed points of the involution are the cycle covers.

Our technique would be to pair a pclow sequence with another having the same set of edges but with an opposite sign. Consequently, they cancel each other's contribution to the summation.

Note that all pclows are, by definition, even in length. However, a given sequence could have an odd length simple cycle in a pclow as shown in Fig 5 & 6. To pair such sequences, pick the pclow with the smallest head that has an odd simple sub-cycle. Walk down this pclow from its head, until you realize that you have gone around an odd cycle. Simply reverse the orientation of all the edges in this cycle. This defines a new pclow sequence. Conversely, starting with the new sequence, our mapping will consider the same (sub-)cycle and reversing its edges will give us the old pclow sequence; so they pair. Their total contribution is zero, since reversing an odd number of edges changes the parity of the number of properly oriented edges and so contributes a negative sign. The pclows in Figures 5 & 6 are an example of the above bijection.

We are left with pclow sequences in which all sub-cycles in all pclows are even. Let $\mathcal{P} = \langle P_1, \ldots, P_k \rangle$ be such a pclow sequence. Pick the smallest i such that P_{i+1} to P_k are disjoint simple cycles. If $i = 0$, then $\mathcal{P}$ is a cycle cover and $\mathcal{P}$ maps onto itself. Else, traverse P_i till one of the following happens,

1. We hit a vertex that meets one of P_{i+1} to P_k.
2. We hit a vertex that completes an even length simple cycle in P_i.

Let v be this vertex. Note that, these two conditions are mutually exclusive because of the way we have traversed P_i. We never hit a vertex that simultaneously satisfies both the conditions.

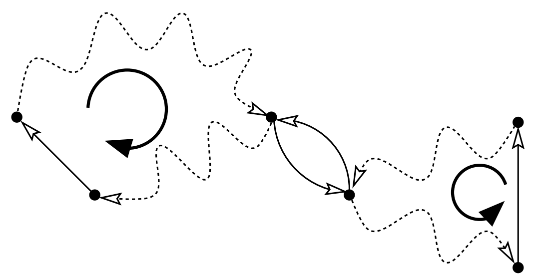

Figure 5: Pclow with odd sub-cycle

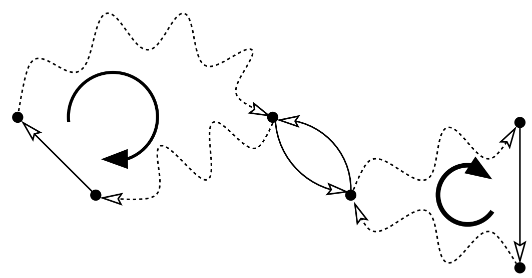

Figure 6: Pclow with odd sub-cycle reversed

<u>Case 1:</u> Suppose v touches P_j. The successor edge of v in P_i must be from the base matching (read ahead to see why). Let w be this vertex. If v is a odd numbered vertex, then w is $v+1$; otherwise, w is $v-1$. Either, the predecessor or the successor of v in P_j has to be w. (If w had been the predecessor of v in P_i, then we would have stopped our traversal at w itself.) The orientation of the (v, w) edge in P_j gives rise to two cases.

1. *If the edge in P_j is from v to w:* (v,w) is identically oriented in P_i and P_j. We simply stick P_j into P_i at v. Formally, map $\mathcal{P}$ to a pclow sequence

$$\mathcal{P}' = \langle P_1, \ldots, P_{i-1}, P_i', P_{i+1}, \ldots, P_{j-1}, P_{j+1}, \ldots P_k \rangle$$

   P_i' is obtained from P_i by inserting into it the simple cycle P_j at the first occurrence of v. Figure 7 illustrates this case.
2. *If the edge in P_j is from w to v:* (v,w) has opposite orientations in P_i and P_j. We cannot stick P_j into P_i as is, beause then P_i would lose the alternating property; it would use two edges from the base matching consecutively. So first reverse the orientation of all edges in P_j, and then insert this pclow into P_i. Figure 8 shows the mapping.

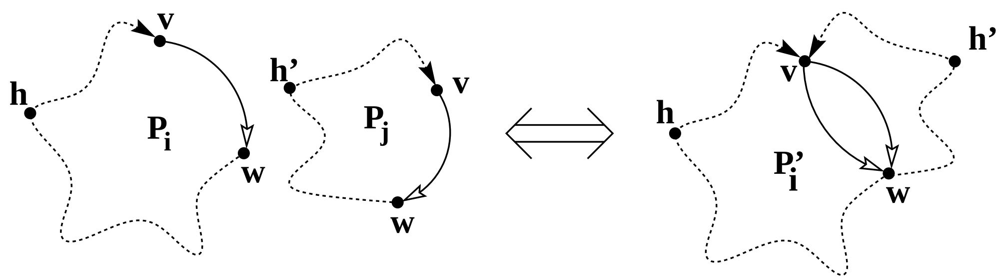

Figure 7: P_i and P_j have (v,w) oriented the same way.

<u>Case 2:</u> Suppose v completes a simple cycle P in P_i. P must be disjoint from all the later cycles. We modify the pclow sequence $\mathcal{P}$ by plucking out P from P_i and introducing it as a new pclow. P's position will be to the right of P_i as P_i's head would be smaller than P's. However,

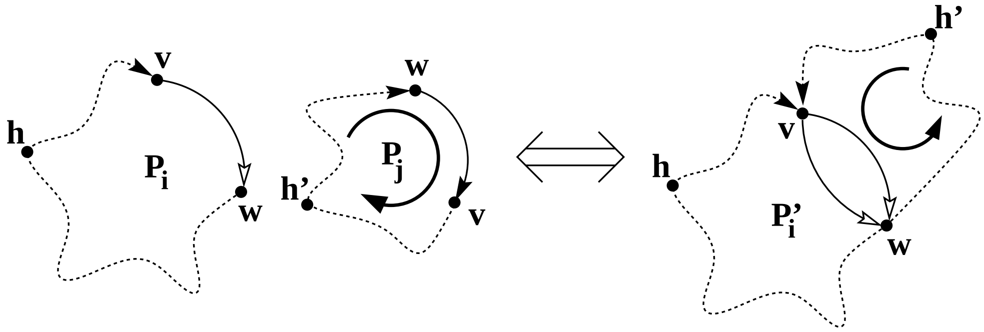

Figure 8: P_i and P_j have (v,w) oriented differently.

one additional change may be necessary. The definition of a p-edge demands that the out-going edge from the head be a matched edge. This *may not* be so with P, in which case P is not a valid pclow. However, merely reversing the orientations of all the edges in P gives a valid pclow, which we then insert into an appropriate position.

We need to argue the correctness of these mappings. It should be clear that the new sequences map back to the original sequences and hence the mapping is an involution. We now show that the mapped pclow sequences have opposing signs, and as their weights are identical, they cancel each other's contribution.

Recall that the sign is characterized by the number of pclows and the number of properly oriented edges. In finding the mapped sequences, we change the parity of the number of pclows. The parity of the number of properly oriented edges remains unchanged, because the reversal of a pclow or an even length sub-cycle preserves this. Thus, the mapped pclow sequences indeed have opposing signs.

Pclow sequences arising from the superposition of the identity permutation with some perfect matching map onto themselves. These are the sole survivors. ■

The above theorem and proof appear similar to Theorem 1 in [MV97]. After all, pclow sequences are a subclass of clow sequences, and so one would expect that the cancellative involution over clow sequences described in [MV97], or perhaps a slight modification, may be the desired involution over pclow sequences. However, this is not the case. The definitions of the sign and the weight of a pclow sequence are quite different from the corresponding definitions in [MV97], and for this setting we need an altogether different involution. The major departure is reflected in the way odd sub-cycles are handled.

4 A Combinatorial Algorithm for Pfaffians

In this section we describe a combinatorial algorithm for computing the Pfaffian. We construct a layered directed acyclic graph H_D with three special vertices s, t_+ and t_-. We show that,

$$\mathsf{Pf}(D) = \sum_{\rho:\ s \rightsquigarrow\ t_+} wt(\rho) \quad - \sum_{\eta:\ s \rightsquigarrow\ t_-} wt(\eta)$$

In this model of computation, all $s \rightsquigarrow t_+$ ($s \rightsquigarrow t_-$) paths of positive (negative) sign are in 1-1 correspondence with pclow sequences of positive (negative) sign.

H_D has the vertex set, $\{s, t_+, t_-\} \cup \{[p, h, u, i] | p \in \{0, 1\}, h, u \in [1, n], i \in \{0, \ldots, n-1\}\ \}$. A path from s to $[p, h, u, i]$ indicates that in the pclow sequence being constructed along this path, p is the parity of the pclow sequence, h is the head of the current pclow, u is the current vertex on the pclow and i is the number of edges seen so far. A $s \rightsquigarrow t_+$ ($s \rightsquigarrow t_-$) path corresponds to a pclow sequence having a positive (negative) sign.

H_D has n layers and layer i has vertices of the form $[\_, \_, \_, i]$. The edges from layer ($2j$-1) to layer $2j$ are fixed and independent of D. The edges in H_D are:

1. $\langle s, [0, h, h, 0] \rangle$ for $h = 2i - 1$, where $i \in [1, \frac{n}{2}]$; edge weight is 1.
2. $\langle [p, h, u, 2i], [\bar{p}, h, v, 2i + 1] \rangle$, $v > h$, $v > u$, $i \in [0, \frac{n}{2} - 1]$; edge weight is d_{uv}.
3. $\langle [p, h, u, 2i], [p, h, v, 2i + 1] \rangle$, $v > h$, $v < u$, $i \in [0, \frac{n}{2} - 1]$; edge weight is d_{vu}.
4. $\langle [p, h, 2j - 1, 2i - 1], [\bar{p}, h, 2j, 2i] \rangle$ if $2j - 1 > h$, $2i < n$; edge weight is 1.
5. $\langle [p, h, 2j, 2i - 1], [p, h, 2j - 1, 2i] \rangle$ if $2j - 1 > h$, $2i < n$; edge weight is 1.
6. $\langle [p, h, h + 1, 2i - 1], [\bar{p}, h', h', 2i] \rangle$ if $h' > h$, h' is odd, $2i < n$; edge weight is 1.
7. $\langle [0, h, h + 1, n - 1], t_- \rangle$ and $\langle [1, h, h + 1, n - 1], t_+ \rangle$ if $h = 2i - 1$, $i \in [1, \frac{n}{2}]$; edge weight is 1.

Theorem 9
Given a $n \times n$ skew symmetric matrix D, let H_D be the graph described above. Then,

$$\mathsf{Pf}(D) = \sum_{\rho:\ s \rightsquigarrow\ t_+} wt(\rho) - \sum_{\eta:\ s \rightsquigarrow\ t_-} wt(\eta)$$

Proof: We show a one-to-one correspondence between $s \rightsquigarrow t_+$ ($s \rightsquigarrow t_-$) paths and pclow sequences of positive (negative) sign. Then, from Theorem 8 the result is immediate.

We utilize our characterization of the sign of a pfaffian term as stated in Lemma 5.

Let $\mathcal{W} = \langle P_1, \ldots, P_k \rangle$ be a pclow sequence. Let h_i be the head of pclow P_i, n_i the number of forward edges in P_i, $p_i = (i + \sum_{j=1}^{i} n_j) mod(2)$ the parity of the pclow sequence $\langle P_1, \ldots, P_i \rangle$, and m_i the total number of edges of the pclow sequence $\langle P_1, \ldots, P_i \rangle$. The path we construct for $\mathcal{W}$ goes through the vertices $[p_i, h_{i+1}, h_{i+1}, m_i]$. We use an inductive argument to prove our result.

Suppose, after traversing $\langle P_1, \ldots, P_i \rangle$, we are at the vertex $[p_i, h_{i+1}, h_{i+1}, m_i]$. In order to establish the inductive argument, it suffices to show that starting the traversal of P_{i+1} from this vertex, we will correctly reach $[p_{i+1}, h_{i+2}, h_{i+2}, m_{i+1}]$.

Let $P_{i+1} = \langle h_{i+1}, v_1, \ldots, v_l \rangle$. As P_{i+1} is a valid pclow, there is an edge from $[p_i, h_{i+1}, h_{i+1}, m_i]$ to $[p_i, h_{i+1}, v_1, m_i{+}1]$ in H_D. As we traverse P_{i+1}, there will be vertices of the form $[p, h_{i+1}, v_j, m_i + j]$ where p is the parity of p_i and the number of forward edges upto v_j in P_{i+1}. When we reach the last vertex $v_l = h_{i+1} + 1$ of P_{i+1}, we would have changed signs as many as $n_{i+1} - 1$ times. The last edge of any pclow is always wrongly oriented and we reach $[p_{i+1}, h_{i+2}, h_{i+2}, m_{i+1}]$. Lemma 5 tells us that this is the proper way to calculate the sign of a pclow.

At layer n, depending on whether p_n is $+1$ or -1, H_D will have an edge to t_+ or t_-.

To show the other direction, consider a path $s \rightsquigarrow t_+$. If we were to list out the path, it will be a non-decreasing sequence with respect to the second component of each vertex. Segments having the same second component correspond to a pclow whose head is the second component. The number of parity changes along this segment will exactly equal the number of forward edges along the path plus one. This generates a pclow sequence corresponding to the $s \rightsquigarrow t_+$ path and of even orientation parity. Similarly, each $s \rightsquigarrow t_-$ path corresponds to a pclow sequence of odd orientation parity. ■

Using simple dynamic programming techniques we can evaluate $\mathsf{Pf}(D)$ in polynomial time. The algorithm proceeds in n stages, where in the i^{th} stage we compute the sum of the weighted paths from s to any vertex x in layer i. Layer n has vertices t_+ and t_-, and we compute the difference of the weighted paths from s to t_+ and t_-. This algorithm looks at an edge in H_D once and hence is a polynomial-time algorithm ($O(n^4)$ ring operations).

It is clear that we can parallelize the above computation and thus computing pfaffian is in NC. Over integers, we can design an NL machine which nondeterministically traces out paths in H_D; the number of its accepting (rejecting, respectively) paths precisely computes $\sum_{\rho:\ s \rightsquigarrow t_+} wt(\rho)$ ($\sum_{\eta:\ s \rightsquigarrow t_-} wt(\eta)$ respectively). This is a GapL algorithm for computing the pfaffian. For more details about the efficient parallel implementations and the GapL implementation, see Sections 6.1 and 6.2 in [MV97], where similar algorithms for counting all clow sequences (with a somewhat different definition of sign and weight) is described. Thus, the main results of this section are:

Theorem 10 *Computing the pfaffian of a skew-symmetric matrix over integers is in* GapL.

Theorem 11 *The Pfaffian of a skew-symmetric $n \times n$ matrix over any commutative ring can be computed by an arithmetic circuit with $O(n^4)$ gates and depth $O(\log n)$. The gates of the circuit are of two types: (1) unbounded fanin gates computing ring addition, and (2) bounded fanin gates computing ring multiplication. Alternatively, the pfaffian can be computed by an*

OROW PRAM performing $O(n^6)$ work and running in $O(\log^2 n)$ parallel time, assuming unit cost per ring operation.

5 Finding admissible orientations for Planar graphs

Counting the number of perfect matchings in a graph requires:

1. Finding an admissible orientation of the graph.
2. Computing the Pfaffian of the associated matrix.

We know how to do the latter from the previous section. We also know that the general problem of counting the number of perfect matchings in a graph is $\sharp$P-Complete. In this section, we show that by restricting ourselves to planar graphs, we can find admissible orientations to them in GapL. This means that counting perfect matchings in planar graphs is in GapL.

Definition 12 [Kas67] *Given a skew symmetric matrix D,*

1. *An* orientation *of D is an assignment of signs to its matrix elements.*
2. *The* orientation parity *of a cycle is the number of edges that are correctly oriented with respect to the underlying orientation on D.*
3. *An orientation is* admissible *if every superposition cycle has odd orientation parity.*

Admissible orientations ensure that each pfaffian term is positive, and we end up computing the number of perfect matchings in the graph. We show that an admissible orientation for a planar graph can be found in GapL using a variant of Kasteleyn's algorithm.

Lemma 13
Finding an admissible orientation of a planar graph is logspace reducible to the problem of evaluating a **parity tree**.

Proof: We assume that the planar graph is so encoded that the faces seen so far form a simple connected component (i.e. each face has at least one edge not shared with the earlier faces).[5] This is required by Kasteleyn's algorithm [Kas67] to uncover an admissible orientation, which we now describe without proof. Start with any face and do the following,

1. Orient all unoriented edges except one arbitrarily.
2. For the last edge, pick an orientation so that the cycle has odd orientation parity when traversed clockwise.

[5]One way of finding such an ordering of the faces, given any planar embedding, is described in [LovPlu86]; construct a spanning tree in the dual graph, and then enumerate the vertices of the dual in the order of their distance from the tree center.

3. Continue if there are unoriented faces remaining. Pick an adjoining face such that this face together with the other oriented faces form a simply connected region. Go to step 1.

Planar graphs have the property that no edge is common to more than two faces. We utilize this property in our *logspace* reduction.

Suppose we are given the faces of the input planar graph. We can order them in many ways as per the requirements of Kasteleyn's algorithm. Fix one such ordering of the faces, say $C_1, C_2, \ldots, C_k$. With respect to this ordering and Kasteleyn's scheme, each face can be uniquely associated with an edge that has to have a specific orientation in order to maintain odd orientation parity. We denote such an edge as the **critical edge** for the face with respect to the face ordering. Figure 9 provides an illustration of critical edges associated with faces.

Consider a cycle C_i in Fig 9. There can be three types of edges on C_i that determine the orientation of its critical edge.

1. Non-critical edges whose orientations were fixed in cycles C_j, $j < i$.
2. Critical edges from the earlier cycles C_j, $j < i$.
3. Unoriented (or fresh) edges other than the critical edge of C_i.

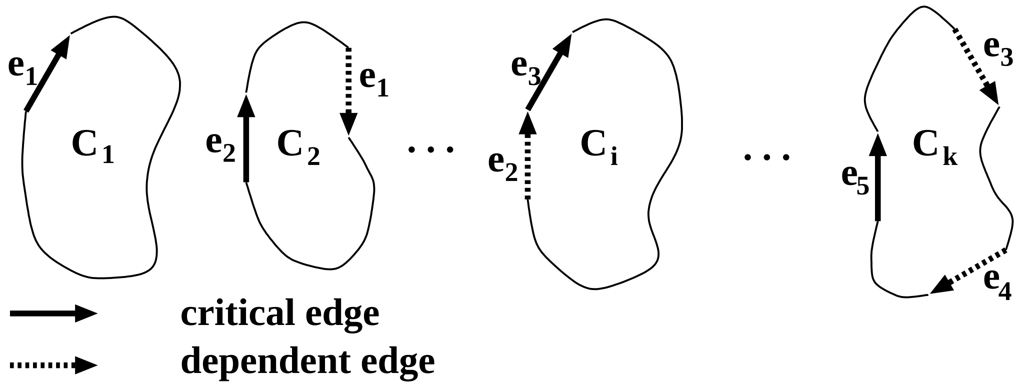

Figure 9: Critical Edges associated with each cycle

We have reformulated the problem of finding an admissible orientation to a planar graph to one of finding the orientations of critical edges so that each face has an odd number of properly oriented edges. Let us pick a cycle and find the orientation for its critical edge.

- Non-critical edges: As their orientations are fixed, we need know the parity of those among them that are properly oriented.
- Fresh edges: We orient these clockwise, and hence we need to know the parity of such edges.

- Critical edges: The orientations of these are fixed in earlier cycles. The difference being that we need to recompute them. Once computed, we take their parity.

Computing the orientation of the critical edge of a cycle requires us to know the parity of the properly oriented edges in the cycle. During this computation, on encountering a critical edge of an earlier cycle, its orientation is the outcome of another parity computation on the edges constituting its cycle. This structure repeats along every critical edge to give us a computation graph. We shall show that this graph is actually a tree where every internal node is a parity node. Figure 10 illustrates this structure.

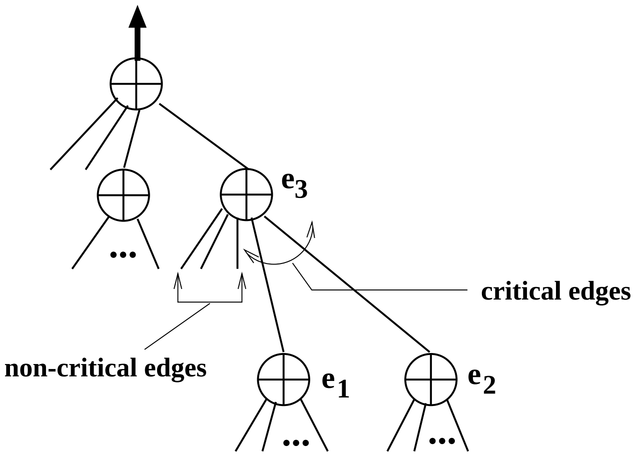

Figure 10: Parity Tree for finding the orientation of critical edges on cycles

Let e_j be the critical edge of an earlier face C_j appearing in face C_i (i.e. $j < i$). Finding e_j's orientation requires us to do a parity on the properly oriented edges in C_j. Consider some other critical edge e_l also appearing in C_i. The computation path from C_i along the parity node corresponding to e_l will never encounter e_j. This is because our input is a planar graph, and hence an edge is common to at most two faces. Therefore, all paths from a parity node are non-intersecting, and we have our parity tree.

We need to show that this is a *logspace* reduction. Note that we are assuming that the input is nicely encoded. Determining whether the current edge of a face is critical, non-critical or fresh can be easily done in *logspace* by scanning the preceding input. Therefore, given a parity node of the tree, we can identify the incoming arcs to it within *logspace*.

We deviate from Kasteleyn's scheme of identifying the critical edge on a cycle. We make the first unoriented edge on a cycle as the critical edge. By doing this, things are simpler because we now do not need to spend valuable computational resources to identify the last edge on the cycle. ■

We now show that parity tree evaluation itself is not a problem of high complexity; in fact

Lemma 14
Parity Tree Evaluation can be done in logspace.

Proof: Parity is associative and commutative. Evaluating the parity of a sequence of elements requires one to remember only the parity of the elements seen so far. Hence, the parity tree can be collapsed. The parity of the leaves is therefore that of the tree. Systematically finding these leaves can be done by a logspace machine. ■

From Theorem 10 and lemmas 13 and 14, we have

Theorem 15
Given a planar graph G, counting the number of perfect matchings in G can be done within GapL.

6 Hardness of the Pfaffian

We complete our tour of computing the Pfaffian of an integer skew-symmetric matrix by pinpointing its hardness. We show that this problem is ♯L-Hard and GapL-Hard.

Theorem 16 *Computing the pfaffian of a skew-symmetric integer matrix is hard for #L. (In fact, all entries of the matrix are from $\{0, +1, -1\}$.)*

Proof: We will show a reduction from the following canonical #L-complete problem.

Instance: A directed acyclic graph G, with vertices numbered $\{0, 1, 2, \ldots, n\}$ and all edges directed from i to j, $j > i$.
Question: Find the number of paths from vertex 0 to vertex n in G.

The following is a reduction from the above problem to that of **counting perfect matchings**. (Chandra, Stockmeyer and Vishkin [CSV84] describe a reduction from directed s, t connectivity to testing for the existence of a perfect matching, and attribute part of the construction independently to Feather and Pippenger. The reduction below is essentially the same, and is easily seen to be parsimonious.) Construct an undirected graph H from G as follows.

- Retain vertex 0 and replace each i, $1 \leq i \leq n-1$ by two vertices $\mathbf{2i-1}$ and $2i$.
- Replace vertex n with a new vertex $\mathbf{2n-1}$.
- For any edge $\langle i, j\rangle$ in G, insert an edge $\langle \mathbf{2i}, \mathbf{2j-1}\rangle$ in H.

Claim 17 *Paths in G from 0 to n are in* **1-1** *correspondence with perfect matchings in H.*

Claim 18 *The forward orientation on H (i.e. all edges H are oriented to go from a smaller to a larger vertex), denoted H^f, is a pfaffian orientation.*

Claim 19 *Computing the pfaffian of the skew-symmetric adjacency matrix of H^f is hard for #L.*

Claim 17 is easy to see. There is only one way to convert a 0 to n path in G into a matching in H and vice-versa. Claim 19 basically rephrases Claim 18. We shall prove Claim 18.

Let $\mathcal{M}$ be a matching in H. Consider the following scheme for choosing a permutation to represent $\mathcal{M}$.

- List the edges of the path in G corresponding to $\mathcal{M}$.
- List these out in the order in which they appear in G.
- List the remaining edges of $\mathcal{M}$ in increasing order of vertices.
- Sort the complete list of edges based on the first vertex of each edge.

To illustrate, consider the graph G shown in Figure 11 with $n = 8$. The path in G corresponding to the matching in H is $0 \to 2 \to 4 \to 7 \to 8$. The matched edges listed in sequence for the path are **0-3**, **4-7**, **8-13**, **14-15**. Appending the remaining edges and then sorting the whole list based on the smaller vertex of each edge, we get the sequence **0-3 1-2 4-7 5-6 8-13 9-10 11-12 14-15**. This is the permutation chosen for the matching.

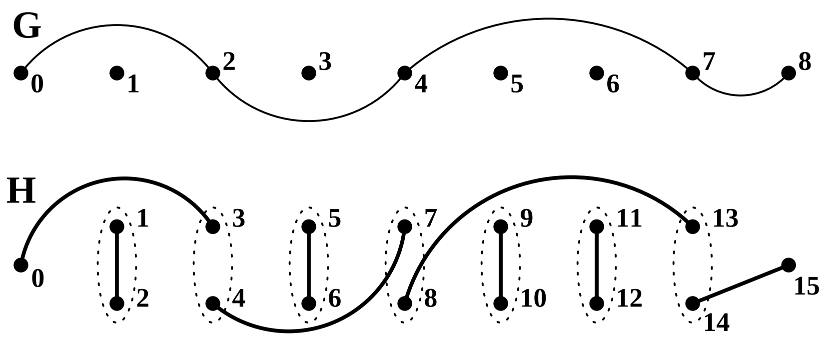

Figure 11: Example for the reduction from st-path to perfect matchings

As all the edges are forward going, the sign of the permutation comes only from transpositions. How many transpositions are needed to reach identity? Look only at the destination vertices of the path edges. Each one of these will need to be moved over an even number (maybe, 0) of non-path edge vertices. Therefore, moving each one of these destination vertices to their correct place requires on the whole an even number of transpositions. Hence the sign of the permutation, and therefore the sign of each pfaffian term, is positive.

■

Theorem 20 *Computing the pfaffian of a skew-symmetric integer matrix is hard for* GapL. *(In fact, all entries of the matrix are from* $\{0, +1, -1\}$*.)*

Proof: We use a construction that is similar to the one in Theorem 16. The canonical GapL-complete problem from which we show the reduction is,

Instance: A directed acyclic graph G, with vertices numbered $\{0, 1, 2, \ldots, n, n+1\}$ and all edges directed from i to j, $j > i$.
Question: Find $\mathbf{p} - \mathbf{q}$, where
$p =$ The number of paths in G from vertex 0 to vertex $n+1$.
$q =$ The number of paths in G from vertex 0 to vertex n.

(Without loss of generality, we can assume that there is no edge from n to $n+1$.)

We construct the undirected graph H as in Theorem 16 with a few changes.

- Vertex 0 goes to 0 and each $i \in \{1, 2, \ldots, n-1\}$ goes to $2i-1$ and $2i$.
- Vertex n goes to $2n-1$, vertex $n+1$ goes to $2n+1$ and introduce a new vertex $2n$.
- Include edges as in the construction of Theorem 16.
- Introduce 2 new edges, $e = \langle 2n-1, 2n \rangle$ and $f = \langle 2n+1, 2n \rangle$. Note that the edge f is not forward.

The orientation of H that we will use is: *all edges of H except f are forward edges.* A matching $\mathcal{M}$ in H will be represented by the permutation using the same scheme as in Theorem 16 except for the edges e and f. Any matching will use precisely one of e or f. This edge is enumerated last in the permutation depending on its orientation.

For instance, consider the situation when $n = 8$ and the matching in H corresponds to the positive path **0-2-4-7-9**. We can list the matching $\mathcal{M}$ and the permutation $\sigma_{\mathcal{M}}$ chosen to represent it as,

$\mathcal{M} =$ **0-3 4-7 8-13 14-17 1-2 5-6 9-10 11-12 15-16**
$\sigma_{\mathcal{M}} =$ **0-3 1-2 4-7 5-6 8-13 9-10 11-12 14-17 15-16**

Let a negative path be **0-2-4-7-8**. The corresponding matching and permutation are,

$\mathcal{M}' =$ **0-3 4-7 8-13 14-15 1-2 5-6 9-10 11-12 17-16**
$\sigma_{\mathcal{M}'} =$ **0-3 1-2 4-7 5-6 8-13 9-10 11-12 14-15 17-16**

Every 0 to $(n+1)$ path in G gives a matching as earlier plus the forward edge e. The additional transpositions required here are for moving $2n+1$ right over edge e, i.e. two extra transpositions. Therefore, the pfaffian term is positive.

On the other hand, every 0 to n path in G also gives a matching as earlier plus the reverse edge f. We get back the identity with just one additional transposition, the one to rewrite the vertices of f forward. Therefore, the pfaffian term is negative.

■

7 Discussion

We have shown that given a "reasonable" encoding of a planar graph, counting the number of perfect matchings in it is in GapL. However, accepted versions of "reasonableness" differ. What would be more satisfying is to know the complexity of counting the number of perfect matchings in a graph, given that the graph is planar. A relaxed version of this would also give some planar embedding of the graph, though not necesarily one suitable for the above algorithm.

A related question that immedately arises is: what is the complexity of planarity testing itself? Can this be done in GapL? The best known result so far is that planarity testing can be done on a CRCW PRAM in $O(\log n)$ time [RR94], and hence is in AC^1.

Of course, the big question still remains open: what exactly is the complexity of both the decision and counting versions of perfect matchings?

Acknowledgments

The authors would like to thank Günter Rote for very helpful comments on an earlier version of the paper.

References

[AlRe98] E. Allender and K. Reinhardt, *Isolation, Matching, and Counting*, in IEEE Conference on Computational Complexity, pp 92-100, 1998.

[CSV84] A. Chandra, L. Stockmeyer and U. Vishkin, *Constant Depth Reducibility*, SIAM Journal on Computing, vol. 13(2), pp 423–439, 1984.

[Dam91] C. Damm. DET=$\mathrm{L}^{(\#\mathrm{L})}$. Technical Report Informatik–Preprint 8, Fachbereich Informatik der Humboldt–Universität zu Berlin, 1991.

[Kas67] P. W. Kasteleyn, *Graph Theory and Crystal Physics*, in Graph Theory and Theoretical Physics, ed: F. Harary, Academic Press, pp 43-110, 1967.

[Lit74] C. H. C. Little, *An extension of Kasteleyn's method of enumerating the 1-factors of planar graphs*, in Combinatorial Mathematics, Proc. 2^{nd} Australian Conference, ed: D. Holton, *Lecture Notes in Mathematics*, 403, pp 63-72, 1974.

[LovPlu86] L. Lovasz and M. Plummer, *Matching Theory*, Annals of Discrete Mathematics 29, North-Holland, 1986.

[MV97] M. Mahajan and V. Vinay, *Determinants: Combinatorics, algorithms, and complexity*, Chicago Journal of Theoretical Computer Science, vol. 5, 1997.

[MVV87] K. Mulmuley, U. V. Vazirani and V. V. Vazirani, *Matching is as easy as Matrix Inversion*, Combinatorica, vol. 7, pp 105-113, 1987.

[RR94] V. Ramachandran and J. Reif, *Planarity Testing in Parallel*, Journal of Computer and System Sciences, vol. 49, pp 517-561, 1994.

[Tod91] S. Toda. *Counting problems computationally equivalent to the determinant.* manuscript, 1991.

[Val79] L. G. Valiant, *The complexity of computing the permanent*, Theoretical Computer Science, vol. 8, pp 189-201, 1979.

[Val92] L. G. Valiant. Why is boolean complexity theory difficult? In M. S. Paterson, editor, *Boolean Function Complexity.* Cambridge University Press, 1992. London Mathematical Society Lecture Notes Series 169.

[Vaz89] V. V. Vazirani, *NC algorithms for computing the number of perfect matchings in $K_{3,3}$-free graphs and related problems*, Information and Computation, vol. 80(2), pp 152 - 164, 1989.

[Vin91] V. Vinay, *Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits*, Proc. 6th IEEE Structure in Complexity Theory Conference, pp 270-284, 1991.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'