Parallel Polynomial Permanent Mod Powers of 2 and Shortest Disjoint Cycles

Samir Datta ✉
Chennai Mathematical Institute, India

Kishlaya Jaiswal ✉
Chennai Mathematical Institute, India

Abstract

We present a parallel algorithm for permanent mod 2^k of a matrix of univariate integer polynomials. It places the problem in $\oplus\mathsf{L} \subseteq \mathsf{NC}^2$. This extends the techniques of Valiant [26], Braverman, Kulkarni and Roy [3] and Björklund and Husfeldt [2] and yields a (randomized) parallel algorithm for shortest two disjoint paths improving upon the recent (randomized) polynomial time algorithm [2].

We also recognize the disjoint paths problem as a special case of finding disjoint cycles, and present (randomized) parallel algorithms for finding a shortest cycle and shortest two disjoint cycles passing through any given fixed number of vertices or edges.

2012 ACM Subject Classification Theory of computation → Parallel algorithms

Keywords and phrases permanent mod powers of 2, parallel computation, graphs, shortest disjoint paths, shortest disjoint cycles

Digital Object Identifier 10.4230/LIPIcs.MFCS.2021.36

Related Version *Full Version*: `https://arxiv.org/abs/2106.00714`

Funding *Samir Datta*: Partially funded by a grant from Infosys foundation and SERB-MATRICS grant MTR/2017/000480.

Acknowledgements We would like to thank Eric Allender for a discussion regarding what was known about field operations in $\oplus\mathsf{L}$. We would also like to thank Partha Mukhopadhyay for his comments on a preliminary version of the paper. We thank the anonymous referees for helping us improve the presentation of the paper.

1 Introduction

The problem of computing the determinant of a matrix has been a very well studied problem in the past, and several fast (both sequential and parallel) algorithms are known. On the contrary, Valiant in his seminal paper [26] showed that computing permanent of a matrix, an algebraic analogue of determinant, is hard. However modulo 2, determinant and permanent are equal and so building up on this, he presented an algorithm for computing permanent of an integer matrix modulo small powers of 2. The algorithm uses Gaussian elimination which is known to be highly sequential and so it is desirable to have a parallel algorithm. This was resolved by [3] who presented a $\oplus\mathsf{L} \subseteq \mathsf{NC}$ algorithm.

Moreover, NC algorithms for computing determinant of matrices over arbitrary commutative rings are also known, e.g. [18]. We would like to ask a similar question for the permanent. One natural extension would be to consider the ring of polynomials with integer coefficients. In this paper, we present an NC algorithm to compute permanent of matrices over integer polynomials modulo 2^k for any fixed k.

▶ **Theorem 1.** *Let $k \geq 1$ be fixed and A be an $n \times n$ matrix of integer polynomials, such that the degree of each entry is at most $\mathsf{poly}(n)$. We can compute* $\mathrm{perm}(A) \pmod{2^k}$ *in* $\oplus\mathsf{L} \subseteq \mathsf{NC}^2$

46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021).
Editors: Filippo Bonchi and Simon J. Puglisi; Article No. 36; pp. 36:1–36:22
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

In the second half of the paper, we consider some applications of our parallel polynomial permanent algorithm. One direct consequence is that we are now able to parallelize the shortest two disjoint paths problem [2]. Furthermore, we generalize this problem by adding more constraints on the disjoint paths - that the paths should pass through any given set of edges. This can also be viewed as a problem of finding two disjoint cycles, for which we present a randomized parallel algorithm, using the techniques from [28] and [2].

▶ **Theorem 2.** *Let $k \geq 1$ be fixed and G be an undirected graph with k marked vertices. We can find shortest two disjoint cycles passing through the marked vertices in* $\oplus\mathsf{L}/\mathsf{poly}$ *(and* RNC*).*

Finally, we notice that a similar approach gives us an algorithm to compute Hafnians modulo 2^k of symmetric matrices of integers. Unfortunately, unlike the case of the permanent, we weren't able to extend this to a parallel algorithm. But nevertheless it gives a direct proof of the fact that counting number of perfect matchings modulo 2^k, in any general graph, is in P, as proved in [3].

1.1 Historical Survey

The problem of computing permanent of an integer matrix was first shown to be NP-hard by Valiant, where he also presented a $O(n^{4k-3})$ running time algorithm to compute permanent modulo 2^k. It was also shown that computing permanent modulo any odd prime still remains hard. Zanko [29] gave a proof for hardness of permanent under many-one reductions strengthening the result from the weaker Turing reductions used by Valiant. Later, Braverman, Kulkarni and Roy [3] presented a parallel $\oplus\mathsf{SPACE}(k^2 \log n)$ algorithm for computing permanent modulo 2^k. Björklund and Husfeldt [2] presented a $d^3 n^{O(k)}$ time algorithm to compute permanent modulo 2^k of matrices over integer polynomials where the entries are of degree at most d.

Finding k disjoint paths in a graph has been a well studied problem in the past: given a graph (undirected/directed) and k pairs of terminals $(s_i, t_i)_{1 \leq i \leq k}$, find k pairwise vertex-disjoint paths P_i from s_i to t_i, if they exist.

When k is not fixed (and is part of input) then the problem is known to be NP-hard even for undirected planar graphs [16]. Linear time algorithms are known when further restricting directed planar graphs to the case: when all terminals lie on outer face [25], or when all the s_i-terminals lie on one common face while all the t_i-terminals lie on another common face [22]. If we further ask for paths with minimal total length in the latter problem, then [7] presented a $O(kn \log n)$ running time algorithm to achieve the same.

When k is fixed, the problem remains NP-hard for directed graphs, even for $k = 2$ [11], who had also given given a poly time algorithm for the restricted case of directed acyclic graphs. In the restriced setting of directed planar graphs, [24] presented a $n^{O(k)}$ running time algorithm, which was further improved to a fixed parameter tractable algorithm by [4].

Shifting our focus to undirected graphs, the celebrated work of Robertson and Seymour [23] gave a $O(n^3)$ algorithm for finding k disjoint paths in an undirected graph, for any fixed k. [6] gave a parallel algorithm for class of planar graphs where all the terminals lie either on one or two faces. All this while, the question of finding *shortest* disjoint paths in general undirected graphs, remained open for many years until recently, Björklund and Husfeldt [2] gave a polynomial time algorithm for finding the shortest two disjoint paths. For general k, this problem still remains open. Björklund and Husfeldt also gave a parallel algorithm to count shortest two disjoint paths but only for cubic planar graphs [1].

1.2 Our Techniques

To compute permanent of a matrix A over integer polynomials, we closely follow the analysis of [3] but immediately hit an obstacle. They give a reduction from $\mathrm{perm}(A) \pmod 4$ to several computations of $\mathrm{perm}(.) \pmod 2$, which crucially uses the fact that $\mathbb{Z}_2$ is a field. More precisely, when mimicking the proof, firstly it is required to find a non-trivial solution of $Av = 0$ with the property that at least one of the entries of this vector is invertible. This fails[1] over $\mathbb{Z}_2[x]$. Moreover, their algorithm also uses the fact that a non-singular matrix admits a LU decomposition iff all the leading principal minors are non-zero, which is known to hold in general only for matrices over fields.

Therefore, replacing $\mathbb{Z}_{2^k}$ with $\mathbb{Z}_{2^k}[x]$ in their analysis doesn't work as $\mathbb{Z}_2[x]$ isn't a field while $\mathbb{Z}_2$ is. Furthermore, any finite field $\mathbb{F}$ of characteristic 2 only corresponds to modulo 2 arithmetic. We need a way to extend the field structure so that it supports modulo 2^k arithmetic as well. If $\mathbb{F}$ was realized as $\mathbb{Z}_2[x]/(p(x))$ where $p(x)$ is irreducible over $\mathbb{Z}_2$ then a possible candidate is the ring $\mathbb{Z}[x]/(2^k, p(x))$. Therefore, the appropriate algebraic structure to consider would be the ring $\mathfrak{R} = \mathbb{Z}[x]/(p(x))$

Now we see that replacing $\mathbb{Z}$ with $\mathfrak{R}$ solves the above mentioned problems in the analysis, primarily because of the fact that $\mathfrak{R} \pmod 2$ is a finite characteristic 2 field. With a slight bit of modification in the proof, we achieve that: given a matrix A over $\mathfrak{R}$, we can find $\mathrm{perm}(A) \pmod{2^k}$ or in other words if A is a matrix over $\mathbb{Z}[x]$, we can compute $\mathrm{perm}(A) \pmod{2^k, p(x)}$.

We are still not done because our aim was to compute $\mathrm{perm}(A) \pmod{2^k}$ over $\mathbb{Z}[x]$. To achieve that, we choose $p(x)$ such that its degree is larger than the degree of polynomial $\mathrm{perm}(A)$. This requires doing computations over a large field. Alternatively, we develop a new way of interpolation over $\mathfrak{R}$, which allows us to choose $p(x)$ such that its degree is of logarithmic order of degree $\mathrm{perm}(A)$, but with a tradeoff of computing several (polynomially many) more permanents. We present this technique for its novelty.

Wahlström [28] addressed the question of finding a cycle passing through given vertices. We ask if we can also find shortest such cycle. And furthermore, can we also find shortest two disjoint cycles passing through these vertices? We combine techniques of [28] and [2] to answer the above questions, by reducing them to computing permanents modulo 2 of 2^{k-1} and modulo 4 of $2^{k-1} + 2^{k-2}$ matrices respectively. These matrices are adjacency matrix of what we refer to as *pattern* graphs. Notice that for $k = 2$ finding shortest two disjoint cycles corresponds to finding shortest two disjoint paths (by connecting each pair of terminals with a common vertex), and in this case our pattern graphs are exactly those presented in [2].

1.3 Organization of the Paper

In section 2, we first introduce the preliminaries and the notation that we shall be using throughout this paper. In the next section 3, we present proof of our main theorem 1 about computing permanent modulo 2^k, following which we also discuss the complexity of certain computations over the ring $\mathfrak{R}$ which shows that our algorithm is in $\oplus\mathsf{L}$. We also present an alternative proof for our main theorem in section 4 which uses new techniques. Then we present applications of our result that is finding shortest disjoint cycles, in section 5.

Finally, we discuss an example of our permanent algorithm and we show that using the same techniques as above we can also compute Hafnians and hence it gives an alternate proof of the already known result that counting perfect matchings modulo 2^k is in P. These are listed in the appendix for lack of space.

[1] Let $A = \begin{pmatrix} x & x+1 \\ x & x+1 \end{pmatrix}$ then there does not exist any null vector of the form $\begin{pmatrix} f \\ 1 \end{pmatrix}$ or $\begin{pmatrix} 1 \\ f \end{pmatrix}$ for any $f \in \mathbb{Z}_2[x]$

2 Preliminaries

We begin by stating the definition of the complexity class $\oplus\mathsf{L}$.

▶ **Definition 3.** *$\oplus\mathsf{L}$ is the class of decision problems solvable by an NL machine such that*

- *If the answer is "yes", then the number of accepting paths is odd.*
- *If the answer is "no", then the number of accepting paths is even.*

By a slight abuse of notation, we say a function class is in $\oplus\mathsf{L}$ if each of its bits can be computed in $\oplus\mathsf{L}$. We will have the occasion to use only constantly many bits because we are working mod 2^k.

Given an $n \times n$ matrix $A = (a_{ij})_{i,j\in[n]}$, determinant and permanent of A are defined as

$$\det(A) = \sum_{\sigma\in S_n} \mathrm{sgn}(\sigma) \prod_{i=1}^{n} a_{i\sigma(i)} \qquad \mathrm{perm}(A) = \sum_{\sigma\in S_n} \prod_{i=1}^{n} a_{i\sigma(i)}$$

The permanent of a matrix can be regarded as the weighted sum of cycle covers of an undirected graph. This gives a combinatorial interpretation to a seemingly pure algebraic quantity. We shall use this bridge to illustrate an application of our parallel polynomial permanent.

Let G be a weighted undirected graph (not necessarily loopless) with the associated weight function w.

▶ **Definition 4.** *We say $C \subseteq V(G) \times V(G)$ is a cycle cover of G if*

- $(u, v) \in C \implies \{u, v\} \in E(G)$
- *C is a union of vertex-disjoint simple directed cycles in G*
- *every vertex is incident to some directed edge in C*

Note: loops are allowed as simple cycles in the above definition.

▶ **Definition 5.** *For any cycle cover C we denote the weight of C by $\tilde{w}(C) = \prod_{e\in C} w(e)$*

The above definition is well-defined because any directed edge (u, v) or (v, u) in our cycle cover correspond to the same edge $\{u, v\}$ in our underlying undirected graph. And hence both these directed edges get the same weight, that is $w((u, v)) = w((v, u)) = w(\{u, v\})$. In literature, such type of weight functions are commonly referred to as *symmetric* weight functions.

▶ **Definition 6.** *Let $V(G) = [n]$ then we say $A = (a_{ij})_{i,j\in[n]}$ is the $(n \times n)$ adjacency matrix of G if*

$$a_{ij} = \begin{cases} w(e) & \text{if } e = \{i, j\} \in E(G) \\ 0 & \text{otherwise} \end{cases}$$

▶ **Observation 7.** $\mathrm{perm}(A) = \sum \tilde{w}(C)$ *where the sum is taken over all cycle covers C of G*

3 Permanent over $\mathfrak{R}$ Mod 2^k

To begin with, we fix some general notation. Let $p(x)$ be an irreducible polynomial over $\mathbb{Z}_2[x]$ such that $\deg(p(x))$ is at most $\mathsf{poly}(n)$. Denote by $\mathbb{F}$ the finite field of characteristic 2 which is realized as $\mathbb{Z}_2[x]/(p(x))$ and by $\mathfrak{R}_k = \mathbb{Z}[x]/(2^k, p(x))$. In particular $\mathfrak{R}_1 \cong \mathbb{F}$. Now as already discussed, we essentially replace $\mathbb{Z}$ by $\mathfrak{R}$ in the algorithm of [3]. We are ready to state the main theorem.

▶ **Theorem 8.** *Let $k \geq 1$ be fixed and $A \in \mathfrak{R}^{n\times n}$. We can compute* $\mathrm{perm}(A) \pmod{2^k}$ *in* $\oplus\mathsf{L}$

Proof is by induction on k. We start with the base case $k = 1$. Note that $\mathrm{perm}(A) \equiv \det(A) \pmod 2$. Using corollary 17 we can find $\mathrm{perm}(A) \pmod 2$ in $\oplus\mathsf{L}$. Now suppose $k > 1$, we shall reduce it to computing several such determinants modulo 2, all of which can be computed in parallel. In doing so, we first illustrate an algorithm which is sequential and then we shall see how to parallelize it.

3.1 Sequential algorithm for computing permanent modulo 2^k

We present the algorithm from [3] for computing permanent modulo 2^k but translated within our framework. Let $A = (a_{ij})_{i,j\in[n]} \in \mathfrak{R}^{n\times n}$ be such that $\det(A) \equiv 0 \pmod 2$. Therefore we can find a non-zero vector $v \in \mathbb{F}^n$ such that $A^T v = 0$ over $\mathbb{F}$. Assume without loss of generality $v_1 = 1$.

Let r_i denote the i^{th} row of A and define A' to be the matrix where the 1^{st} row in matrix A is replaced with $\sum_i v_i r_i$. Now if we expand the permanent along the first row then we get

$$\mathrm{perm}(A') = \sum_{i=1}^{n} v_i \mathrm{perm}(A[1 \leftarrow i]) = \mathrm{perm}(A) + \sum_{i=2}^{n} v_i \mathrm{perm}(A[1 \leftarrow i]) \tag{1}$$

where $A[i \leftarrow j]$ is the matrix A but with i^{th} row replaced with the j^{th} row. For $I, J \subseteq [n]$ denote by $A[\widehat{I}, \widehat{J}]$ the matrix obtained from A by deleting rows indexed by I and columns indexed by J. With this equation, modulo 2^k computation reduces to modulo 2^{k-1} computations of the minors as follows:

$$\mathrm{perm}(A') = \sum_{j=1}^{n} \left(\sum_{i=1}^{n} v_i a_{ij} \right) \mathrm{perm}(A[\widehat{\{1\}}, \widehat{\{j\}}])$$

Since $A^T v = 0 \pmod 2$, we can write $\sum_i v_i a_{ij} = 2b_j \pmod{2^k}$ for some $b_j \in \mathfrak{R}_k$, therefore, we can re-write the above permanents as:

$$\mathrm{perm}(A') \pmod{2^k} = 2 \left(\sum_{j=1}^{n} b_j \mathrm{perm}(A[\widehat{\{1\}}, \widehat{\{j\}}]) \pmod{2^{k-1}} \right)$$

Similarly, expanding $\mathrm{perm}(A[1 \leftarrow i])$ along the 1^{st} and i^{th} rows, we get the reduction:

$$\mathrm{perm}(A[1 \leftarrow i]) = \sum_{j \neq l} a_{ij} a_{il} \mathrm{perm}(A[\widehat{\{1, i\}}, \widehat{\{j, l\}}])$$

$$\mathrm{perm}(A[1 \leftarrow i]) \pmod{2^k} = 2 \left(\sum_{j<l} a_{ij} a_{il} \mathrm{perm}(A[\widehat{\{1, i\}}, \widehat{\{j, l\}}]) \pmod{2^{k-1}} \right)$$

Substituting these equations back in 1, we get

$$\begin{aligned}\mathrm{perm}(A) \pmod{2^k} = {} & 2 \left(\sum_{j=1}^{n} b_j \mathrm{perm}(A[\widehat{\{1\}}, \widehat{\{j\}}]) \pmod{2^{k-1}} \right) \\ & - 2 \sum_{i=2}^{n} v_i \left(\sum_{\substack{j,l=1 \\ j<l}}^{n} a_{ij} a_{il} \mathrm{perm}(A[\widehat{\{1, i\}}, \widehat{\{j, l\}}]) \pmod{2^{k-1}} \right)\end{aligned}$$

Since addition and multiplication in $\mathfrak{R}_k$ is in $\oplus\mathsf{L}$ (see 14) we get that $\text{perm}(A) \pmod{2^k}$ $\oplus\mathsf{L}$-reduces to $\text{perm}(.) \pmod{2^{k-1}}$. Hence by induction, we can compute $\text{perm}(A) \pmod{2^k}$ in $\oplus\mathsf{L}$, provided that $\text{perm}(A) \equiv 0 \pmod 2$.

Let us see how to drop this assumption. We expand the permanent of A along the i^{th} row, then

$$\text{perm}(A) = \sum_j a_{ij}\text{perm}(A[\widehat{\{i\}}, \widehat{\{j\}}])$$

If $\text{perm}(A) \not\equiv 0 \pmod 2$, then $\exists i, j$ such that $\text{perm}(A[\widehat{\{i\}}, \widehat{\{j\}}]) \not\equiv 0 \pmod 2$. We can find this pair (i, j) by running over all n^2 possibilities and evaluating $\text{perm}(A[\widehat{\{i\}}, \widehat{\{j\}}]) \pmod 2$ using corollary 17, as already discussed above. Consider the matrix C where all entries are same as A except the $(i, j)^{th}$ entry which is replaced with $a_{ij} + y$. Then, we get $\text{perm}(C) = \text{perm}(A) + y\text{perm}(A[\widehat{\{i\}}, \widehat{\{j\}}])$. Notice that $\text{perm}(A) + y\text{perm}(A[\widehat{\{i\}}, \widehat{\{j\}}]) \equiv 0 \pmod 2$ is a linear equation in y over the field $\mathbb{F}$ and so there exists a unique $y \in \mathbb{F}$ which satisfies this equation, which is $y_0 = \text{perm}(A)\text{perm}(A[\widehat{\{i\}}, \widehat{\{j\}}])^{-1} \pmod 2$ and can be computed using corollaries 14, 15 and 17. Setting $y = y_0$ we get $\text{perm}(C) \equiv 0 \pmod 2$, so we can compute $\text{perm}(C) \pmod{2^k}$ and then compute $\text{perm}(A[\widehat{\{i\}}, \widehat{\{j\}}])$ recursively as $A[\widehat{\{i\}}, \widehat{\{j\}}]$ is a smaller $(n-1) \times (n-1)$ size matrix. Hence we obtain $\text{perm}(A) = \text{perm}(C) - y_0\text{perm}(A[\widehat{\{i\}}, \widehat{\{j\}}])$ $\pmod{2^k}$. This yields a sequential algorithm for computing permanent modulo 2^k over $\mathfrak{R}$.

3.2 Parallel algorithm for computing permanent modulo 2^k

The bottleneck was finding i, j such that $A[\widehat{\{i\}}, \widehat{\{j\}}]$ is non-singular over $\mathbb{F}$. We fix this by again appealing to the fact that we are working over a field, and modifying A such that all leading principal minors are non-zero. This modification essentially derives from the following fact.

▶ **Theorem 9** ([14, Corollary 1]). *Let A be an invertible matrix over a field $\mathbb{F}$, then all leading principal minors are non-zero iff A admits an LU decomposition*

Every invertible matrix admits a PLU factorization [14] so let $A = PLU$. Denote by $Q = P^{-1}$, then $QA = LU$. Since Q is also a permutation matrix, we get that $\text{perm}(QA) = \text{perm}(A)$ (because permanent is invariant under row swaps). Therefore, it suffices to give a $\oplus\mathsf{L}$ algorithm to find Q so that we can replace A by QA which is an invertible matrix such that all leading principal minors are non-zero. Thus computing $\text{perm}(A) \pmod{2^k}$ reduces to the problem of computing (in parallel) permanent modulo 2^k of $n-1$ matrices with $\text{perm} \equiv 0 \pmod 2$. This gives a $\oplus\mathsf{L}$ algorithm to compute permanent modulo 2^k over $\mathfrak{R}$.

To find Q, we closely follow [9]. For each $1 \le i \le n$, let A_i be the matrix formed from A by only taking the first i columns. Let A_i^j matrix obtained from A_i by only taking the first j rows. We construct a set $S_i \subseteq [n]$ inductively as follows:

- Base case: $l \in S_i$ if $rank(A_i^l) = 1$ and $rank(A_i^k) = 0$ for all $k < l$
- Include $j \in S_i$ iff $rank(A_i^j) = 1 + rank(A_i^{j-1})$

Since $rank(A_i) = i$, we get $|S_i| = i$. Furthermore note that $S_i \subset S_{i+1}$. So let $S_1 = \{s_1\}$ and for each $i \ge 2$, denote by $s_i \in S_i \setminus S_{i-1}$. Consider the following permutation $Q = (n, s_n) \dots (2, s_2)(1, s_1)$. Thus Q is our desired permutation, such that QA has all leading principal minors non-zero.

As a corollary, we immediately get our desired result.

▶ **Corollary 10** (Theorem 1 restated). *Given an $n \times n$ matrix $A = (a_{ij})_{i,j \in [n]}$ over $\mathbb{Z}[x]$ with $\deg(a_{ij})$ at most $\mathsf{poly}(n)$, we can compute $\mathrm{perm}(A) \pmod{2^k}$ in $\oplus\mathsf{L}$ for any fixed $k \geq 1$*

Proof. Let $N = n \max\{\deg(a_{ij})\} + 1$ and choose $l = \lceil \log_3(N/2) \rceil$. Consider $p(x) = x^{2 \cdot 3^l} + x^{3^l} + 1$ which is irreducible over $\mathbb{Z}_2[x]$ (see [27] Theorem 1.1.28)

Since $\deg(p(x)) \geq N > \deg(\mathrm{perm}(A))$, using this $p(x)$ in above theorem, we get $\mathrm{perm}(A) \pmod{2^k}$ for any fixed k. ◀

▶ Remark. See appendix A for a complete worked out example of the above algorithm.

3.3 Complexity Analysis

We discuss the complexity results for arithmetic operations over the ring $\mathfrak{R}_k$ and matrix operations over the field $\mathbb{F}$, which were required in our above algorithm. To begin with, we state a well-known fact about integer polynomials matrix multiplication modulo 2. This shall form our basis for showing computations over $\mathbb{F}$ in $\oplus\mathsf{L}$.

▶ **Lemma 11** (Folklore [5]). *Let $A_1, A_2, \ldots A_n \in \mathbb{Z}_2[x]^{n \times n}$ then the product $A_1 A_2 \ldots A_n$ can be computed in $\oplus\mathsf{L}$*

To obtain an analogous result over $\mathbb{F}$ we first perform multiplication over $\mathbb{Z}_2[x]$ and then divide all entries by $p(x)$, using the following polynomial division, as demonstrated by Hesse, Allender and Barrington in [12], to get that iterated matrix product over $\mathbb{F}$ is in $\oplus\mathsf{L}$

▶ **Lemma 12** ([12, Corollary 6.5]). *Given $g(x), p(x) \in \mathbb{Z}[x]$ of degree at most $\mathsf{poly}(n)$, we can compute $g(x) \pmod{p(x)}$ in $\mathsf{DLOGTIME} - $uniform $\mathsf{TC}^0 \subseteq \mathsf{L}$*

In particular, it follows that given an irreducible polynomial $p(x)$ (over $\mathbb{Z}_2[x]$), then for any $g(x) \in \mathbb{Z}[x]$ of degree at most $\mathsf{poly}(n)$ we can find $g(x) \pmod{2^k, p(x)}$ in $\oplus\mathsf{L}$, for any fixed $k \geq 1$.

▶ **Corollary 13.** *Let $A_1, A_2, \ldots A_n \in \mathbb{F}^{n \times n}$ such that the degree of each entry is at most $\mathsf{poly}(n)$ then the product $A_1 A_2 \ldots A_n$ can be computed in $\oplus\mathsf{L}$*

Now we discuss arithmetic over $\mathfrak{R}_k$

▶ **Lemma 14.** *Let $k \geq 1$ be fixed then the following operations can be done in $\oplus\mathsf{L}$*

- *Multiplication : Given $a, b \in \mathfrak{R}_k$ compute ab*
- *Iterated Addition: Given $c_1, c_2, \ldots, c_n \in \mathfrak{R}_k$ compute $\sum_i c_i$*

Proof. We use the fact that the arithmetic operations mentioned in the statement of lemma, but over $\mathbb{Z}_{2^k}$ are in $\oplus\mathsf{L}$ (see for e.g. [12])

- Let $a(x), b(x) \in \mathfrak{R}_k$ and write $a(x) = \sum_{i=0}^{D} a_i x^i$ and $b(x) = \sum_{i=0}^{D'} b_i x^i$, then $a(x)b(x) = \sum_{i=0}^{D+D'} \left(\sum_{j=0}^{i} a_j b_{i-j} \right) x^i$. Finally, using lemma 12, divide $a(x)b(x)$ by $p(x)$ to obtain $ab \in \mathfrak{R}_k$
- Similarly, let $c_1(x), c_2(x), \ldots, c_n(x) \in \mathfrak{R}_k$ and write $c_i(x) = \sum_{j=0}^{D_i} c_{ij} x^j$ for each $i \in [n]$, then $\sum_{i=1}^{n} c_i(x) = \sum_{j=0}^{\max\{D_i\}} \left(\sum_{i=1}^{n} c_{ij} \right) x^j$ where we assume $c_{ij} = 0$ if $j > D_i$. ◀

Our algorithm also requires computing inverse of non-zero elements. To compute inverse over $\mathbb{F}^*$ we adopt the techniques from Fich and Tompa [8, 10]. Since $\mathbb{F} = \mathbb{Z}_2[x]/(p(x))$ with $N = \deg(p(x))$ which is at most $\mathsf{poly}(n)$, then given $a \in \mathbb{F}^*$, we observe that $a^{-1} = a^{q-2}$ where $q = 2^N = |\mathbb{F}|$.

We interpret this equation over $\mathbb{Z}_2[x]$, that is we need to compute $a(x)^{q-2} \pmod{p(x)}$ over $\mathbb{Z}_2[x]$. First we show how to compute $a(x)^2 \pmod{p(x)}$. Construct the $N \times N$ matrix Q whose i^{th} row $(Q_{i,0}, Q_{i,1}, \ldots, Q_{i,N-1})$ is defined as:

$$\sum_{j=0}^{N-1} Q_{i,j} x^j = x^{2i} \pmod{p(x)}$$

for each $0 \le i \le N-1$. Matrix Q can be computed in $\oplus\mathsf{L}$ using the divison lemma 12. Then the elements of the row vector $(a_0, a_1, \ldots, a_{N-1})Q$ are the coefficients of $a(x)^2 \pmod{p(x)}$ as explained in section 3 of [10]. Furthermore, the coefficients of $a(x)^{2^k} \pmod{p(x)}$ are given by $(a_0, a_1, \ldots, a_{N-1})Q^k$. From lemma 11 we get that $a(x)^{2^k} \pmod{p(x)}$ can be computed in $\oplus\mathsf{L}$, for any k bounded by $\mathsf{poly}(n)$. Therefore. writing $q-2 = (c_0, c_1, \ldots, c_{N-1})$ in binary,

$$a(x)^{q-2} \pmod{p(x)} = \prod_{i=0}^{N-1} a(x)^{c_i 2^i} \pmod{p(x)}$$

can be computed in $\oplus\mathsf{L}$, which gives us a^{-1}.

▶ **Corollary 15.** *Let $a \in \mathbb{F}$ then $a^{-1} \in \mathbb{F}$ can be computed in $\oplus\mathsf{L}$*

Mahajan and Vinay [18] describe a way to reduce the computation of a determinant over a commutative ring to a semi-unbounded logarithmic depth circuit with addition and multiplication gates over the ring. In fact, the following is an easy consequence of their result:

▶ **Lemma 16** (Mahajan-Vinay [18]). *Let $A \in R^{n\times n}$ be a matrix over a commutative ring. Then there exist $M \in R^{(2n^2)\times(2n^2)}$ and two vectors $a, b \in R^{2n^2}$ such that $\det(A) = a^T M^n b$. Moreover, each entry of the matrix M_{ij} and the vectors a, b is one of the entries $A_{i',j'}$ or a constant from $\{0,1\}$ and the mapping ϕ where for every $(i,j) \in [2n^2] \times [2n^2]$, $\phi(i,j) \in A_{[n]\times[n]} \cup \{0,1\}$ is computable in Logspace.*

Proof. (Sketch) In [18], given a matrix A they construct a graph H_A whose vertex set is $\{s, t_+, t_-\} \cup Q$ where $Q = \{[p,h,u,i] : p \in \{0,1\}, h, u \in [n], i \in \{0,\ldots,n-1\}\}$. Moreover, the edges are one of the following forms $(s,q), (q,q'), (q,t_+)$ and (q,t_-) where $q, q' \in Q$ and have weights $w(q,q')$ that each depend on a single entry of A or are one of the constants $0, 1$. Moreover the mapping is very simple to describe. Let us focus on the induced subgraph $H_A[Q]$. Notice that $|Q| = 2n^3$ and each "layer" of $H_A[Q]$ is identical. In other words, $e_i = \langle [p,h,u,i], [p',h',u',i+1] \rangle$ is an edge in $H_A[Q]$ iff $e_j = \langle [p,h,u,j], [p',h',u',j+1] \rangle$ is an edge in $H_A[Q]$ and both have the same weights for every $i, j \in \{0,\ldots,n-1\}$. Thus define the matrix M by putting $M_{[p,h,u],[p',h',u']}$ as the weight of any of the edges e_i.

Finally to define a, b: let $a_{[n \bmod 2,1,1]} = 1$ and $a_q = 0$ for all other q. $b_{[1,h,u]} = a_{uh}$ and $b_{[0,h,u]} = -a_{uh}$. The correctness of our Lemma then follows from the proof of Lemma 2 of [18]. ◀

Using above lemma 16, we reduce determinant over $\mathbb{F}$ to matrix powering over $\mathbb{F}$, which can be computed in $\oplus\mathsf{L}$ using corollary 13. Hence we get

▶ **Corollary 17.** *Let $A \in \mathbb{F}^{n\times n}$ then $\det(A)$ can be computed in $\oplus\mathsf{L}$*

Now we demonstrate two results: computing rank and a null vector a matrix over $\mathbb{F}$ in $\oplus\mathsf{L}$. We use Mulmuley's algorithm [19], which requires finding determinant over the ring $\mathbb{F}[y,t]$, which reduces to matrix powering over $\mathbb{F}[y,t]$ by the above result. We shall further reduce this to matrix powering over $\mathbb{F}$ as follows: Let R be an arbitrary commutative ring. We associate with each polynomial $f(x) = \sum_{i=0}^{d-1} f_i x^i \in R[x]$ a $d \times d$ lower-triangular matrix

$$P(f) = \begin{bmatrix} f_0 & & & & \\ f_1 & f_0 & & & \\ f_2 & f_1 & f_0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ f_{d-1} & f_{d-2} & f_{d-3} & \dots & f_0 \end{bmatrix} \in R^{d\times d}$$

Suppose we have two polynomials $f(x)$ and $g(x)$ of degree d_1 and d_2 respectively. We can interpret them as degree $d_1 + d_2$ polynomials (with higher exponent coefficients as 0). Then we have that $P(f + g) = P(f) + P(g)$ and $P(fg) = P(f)P(g)$.

▶ **Theorem 18.** *Let R be any commutative ring and $A_1, A_2, \dots A_n$ be $n \times n$ matrices over $R[x]$ such that the degree of each entry is at most $\mathsf{poly}(n)$. Denote by $\mathbf{A} = \prod A_l$. There exists $\mathsf{poly}(n) \times \mathsf{poly}(n)$ matrices $B_1, B_2, \dots B_n$ over R such that the coefficient of x^k in $\mathbf{A}_{ij}$ is equal to $\mathbf{B}_{\psi(i,j,k)}$ where $\mathbf{B} = \prod B_l$ and ψ is logspace computable.*

In other words, iterated matrix multiplication over $R[x]$ is logspace reducible to iterated matrix multiplication over R.

Proof. Let $N = n \max_{i,j,k\in[n]}\{\deg((A_i)_{jk})\}$ where $(A_i)_{jk}$ denotes the $(j,k)^{\text{th}}$ entry of A_i. By our assumption, N is at most $\mathsf{poly}(n)$. Now for each $1 \leq i \leq n$, compute the matrix $B_i \in R^{nN\times nN}$ obtained from A_i by replacing each polynomial $(A_i)_{jk}$ with the $N \times N$ matrix $P((A_i)_{jk})$. These matrices B_i can be computed in log space. Now the coefficient of x^k in $\mathbf{A}_{ij}$ can be read from the entry $\mathbf{B}_{\psi(i,j,k)}$ where $\psi(i,j,k) = ((i-1)N + k + 1, (j-1)N + 1)$ is logspace computable. The correctness follows from our observation $P(fg) = P(f)P(g)$. ◀

▶ Remark. This gives us an alternate proof of the fact that iterated matrix multiplication over $\mathbb{Z}_2[x]$ is in $\oplus\mathsf{L}$, as it follows immediately from the definition of $\oplus\mathsf{L}$ that iterated matrix multiplication over $\mathbb{Z}_2$ is in $\oplus\mathsf{L}$.

▶ **Lemma 19** ([19]). *Let $A \in \mathbb{F}^{m\times n}$ then $rank(A)$ can be computed in* $\oplus\mathsf{L}$

Proof. We can assume that A is a square $(n \times n)$ symmetric matrix because otherwise replace A with $\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}$ which has rank twice that of A. Let Y be an $n \times n$ diagonal matrix with the $(i,i)^{th}$ entry as y^{i-1}. And let m be the smallest number such that t^m has a non-zero coefficient in the characteristic polynomial of YA, that is $\det(tI - YA)$. Then rank of $A = n - m$.

Suffices to show that $\det(tI - YA)$ can be computed in $\oplus\mathsf{L}$. Notice that $(tI - YA) \in \mathbb{F}[y,t]^{n\times n}$ and so $\det(tI - YA)$ is logspace reducible to matrix powering over $\mathbb{F}[y,t]$. Using the canonical isomorphism $\mathbb{F}[y,t] \cong \mathbb{F}[y][t]$, repeated application of theorem 18 logspace reduces it to matrix powering over $\mathbb{F}$. ◀

▶ **Observation 20.** *Let $A \in \mathbb{F}^{n\times n}$ be an invertible matrix then A^{-1} can be computed in* $\oplus\mathsf{L}$

This follows from the fact that computing A^{-1} involves computing the determinant of A and n^2 cofactors, that is determinants of n^2 matrices of size $(n-1) \times (n-1)$. Notice that this also requires inverting the determinant, an element of $\mathbb{F}^*$, which has been explained above.

▶ **Corollary 21.** *Let $A \in \mathbb{F}^{n\times n}$ then finding a non-trivial null vector (if it exists) is in* $\oplus\mathsf{L}$

Proof. Let $rank(A) = m$, then permute the rows and columns of A so that we can express $A = \begin{pmatrix} B & C \\ D & E \end{pmatrix}$ such that B is an invertible $m \times m$ matrix. Let $\begin{pmatrix} x \\ y \end{pmatrix}$ be a column vector where $x \in \mathbb{F}^m$ and $y \in \mathbb{F}^{n-m}$, such that

$$A \begin{pmatrix} x \\ y \end{pmatrix} = 0 \implies \begin{pmatrix} B & C \\ D & E \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = 0$$

This reduces to the set of equations: $Bx + Cy = 0$ and $Dx + Ey = 0$. But the later is a redundant set of equations because $\begin{pmatrix} D & E \end{pmatrix}$ can be written in terms of $\begin{pmatrix} B & C \end{pmatrix}$. More precisely, there exists a matrix $V \in \mathbb{F}^{n-m \times m}$ such that $D = VB$ and $E = VC$ and so $Dx + Ey = VBx + VCy = V(Bx + Cy) = 0$. Therefore setting $x = \mathbf{1}$ and $y = -B^{-1}C\mathbf{1}$, gives us the desired null vector. So it suffices to give a $\oplus$L algorithm to transform A to the form as specified above, which follows from [9]. Let A_i be the matrix formed from first i rows of A. We construct a set $S \subseteq [n]$ as follows:

- Base case: $i \in S$ if $rank(A_i) = 1$ and $rank(A_j) = 0$ for all $j < i$
- Include $k \in S$ iff $rank(A_k) = 1 + rank(A_{k-1})$

It follows that $|S| = m$ and let $S = \{i_1 < i_2 < \cdots < i_m\}$ and P_r be the permutation matrix described by $(m, i_m) \ldots (2, i_2)(1, i_1)$. Then $P_r A$ is the required matrix having first m rows as linearly independent. Next, consider the matrix $A' = (P_r A)^T$ and apply the above algorithm to get a permutation matrix P_c such that first m rows of $P_c A'$ are linearly independent. Then $P_r A P_c^T$ is the required matrix such that the leading principal $m-$minor is non-singular. ◀

4 Permanent via Interpolation

We now demonstrate another technique to compute permanent modulo 2^k, which doesn't resort to computations over exponentially sized fields. This proceeds by choosing small degree polynomial $p(x)$. The techniques developed in this section are new and hence interesting by themself.

First we mention a result from [15] used to interpolate the coefficients of a polynomial.

▶ **Lemma 22** ([15, Lemma 3.1]). *Let $\mathbb{F}$ be a finite, characteristic 2, field of order q.*

$$\sum_{a \in \mathbb{F}^*} a^m = \begin{cases} 0 & \text{if } q-1 \nmid m \\ 1 & \text{otherwise} \end{cases}$$

This dichotomy allows us to extract coefficients of any integer polynomial.

▶ **Lemma 23** ([15, Corollary 3.2]). *Let $f(x) = \sum_{i=0}^{d} c_i x^i$ be a polynomial with integer coefficients and $q > d + 1$, then for any $0 \le t \le d$,*

$$\sum_{a \in \mathbb{F}^*} a^{q-1-t} f(a) = c_t \pmod 2$$

But this gives us the coefficients modulo 2 only. How do we get coefficients modulo 2^k?

The crucial observation here is that the above sum was computed over $\mathbb{F}$. So instead we do so over $\mathfrak{R}$ by identifying a copy of $\mathbb{F}^* \hookrightarrow \mathfrak{R}$, and then we have

$$\sum_{a \in \mathbb{F}^*} a^m = \begin{cases} 2\alpha_m & \text{if } q-1 \nmid m \\ 2\beta_m + 1 & \text{otherwise} \end{cases}$$

where $\alpha_m, \beta_m \in \mathfrak{R}$.

Now we use repeated squaring method to obtain our desired modulo 2^k result.

▶ **Lemma 24.** $\forall m \geq 0, k \geq 1$

$$\sum_{a_1,\ldots,a_{2^{k-1}} \in \mathbb{F}^*} (a_1 a_2 \cdots a_{2^{k-1}})^m = \left(\sum_{a \in \mathbb{F}^*} a^m\right)^{2^{k-1}} = \begin{cases} 0 \pmod{2^k} & \textit{if } q-1 \nmid m \\ 1 \pmod{2^k} & \textit{otherwise} \end{cases}$$

Note: We remind the reader that the computation here is done over $\mathfrak{R}_k$

Proof. Fix any $m \geq 0$. Clearly $(2\alpha_m)^{2^{k-1}} \equiv 0 \pmod{2^k}$.

Suffices to show $(2\beta_m + 1)^{2^{k-1}} \equiv 1 \pmod{2^k}$. This follows from induction on k. For $k = 1$ the result holds as stated above. Assume that for some $k \geq 1$, the result holds. Then we have $(2\beta_m + 1)^{2^{k-1}} = 2^k \gamma_{m,k} + 1$ where $\gamma_{m,k} \in \mathfrak{R}$

$$(2\beta_m + 1)^{2^k} = \left((2\beta_m + 1)^{2^{k-1}}\right)^2 = (2^k \gamma_{m,k} + 1)^2 = 1 \pmod{2^{k+1}}$$ ◀

Using this we can interpolate coefficients of an integer polynomial as follows:

$$\sum_{a_1,\ldots,a_{2^{k-1}} \in \mathbb{F}^*} (a_1 \ldots a_{2^{k-1}})^{q-1-t} f(a_1 \ldots a_{2^{k-1}}) = c_t \pmod{2^k}$$

Finally let $A(x)$ be an $n \times n$ matrix of integer polynomials and the permanent polynomial be

$$\operatorname{perm}(A(x)) = \sum_{i=0}^{N} c_i x^i$$

From the above lemma it follows that

$$\sum_{a_1, a_2 \ldots \in \mathbb{F}^*} (a_1 a_2 \cdots)^{q-1-t} \operatorname{perm}(A(a_1 a_2 \cdots)) = c_t \pmod{2^k}$$

provided that our field $\mathbb{F}$ is of order at least $N + 2$. For this, fix $l = \lceil \frac{\log \log N}{\log 3} \rceil$ so that $2^{2.3^l} > N + 1$. Hence the field obtained from the irreducible polynomial $p(x) = x^{2 \cdot 3^l} + x^{3^l} + 1$ ([27] Theorem 1.1.28) serves the purpose. It suffices to compute $|\mathbb{F}^*|^{2^{k-1}} = O(N^{2^{k-1}})$ many permanents over $\mathfrak{R}_k$ to obtain all the coefficients c_t modulo 2^k, all of which can be computed in parallel. Hence, we can compute the permanent of A modulo 2^k over $\mathbb{Z}[x]$ in $\oplus$L.

5 Shortest Disjoint Cycles

Now that we have a $\oplus$L algorithm to compute permanent mod 2^k for matrices over $\mathbb{Z}[x]$, we are all set to demonstrate a parallel algorithm for shortest two disjoint paths. But we notice that we can place this problem in a more general framework of shortest disjoint cycles. Let us first formally define these problems.

$SDP(k)$: Given a weighted undirected graph with k pairs of marked vertices $\{(s_i, t_i) \mid 1 \leq i \leq k\}$, find the minimum of sum of weight of paths between each pair s_i and t_i such that all paths are pairwise disjoint.

$SDC(l, k)$: Given a weighted undirected graph with k marked vertices, find the minimum of sum of weight of l cycles such that they pass through all the marked vertices and are pairwise disjoint and each cycle is incident to at least one of the marked vertices.

▶ Note. We only consider *non-trivial cycles* that is we don't consider self-loops in the above problem.

Given an instance of the $SDP(2)$ problem, join the pairs of vertices (s_1, t_1) and (s_2, t_2) with new vertices u_1 and u_2 respectively, as show in Figure 1. Notice that any two disjoint cycles passing through u_1 and u_2 give us two disjoint paths between (s_1, t_1) and (s_2, t_2).

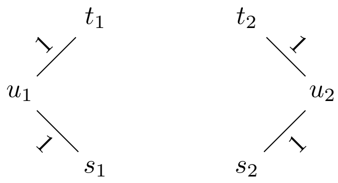


Figure 1 Converting an instance of $SDP(2)$ to $SDC(2, 2)$.

Similarly, connecting the k-pairs of vertices via another new vertex and edges of weight $x^0 = 1$, gives us a reduction from k disjoint paths to k disjoint cycles via k vertices. Since this reduction preserves the weight of the path/cycle, it is indeed a reduction from $SDP(k)$ to $SDC(k, k)$.

To apply the techniques of [2] to disjoint cycles problem, we instead consider the following variant $SDCE(l, k)$: Given a weighted undirected graph with k marked *edges*, find the minimum of sum of weight of l cycles such that they pass through all the marked edges and are pairwise disjoint.

It can be easily seen that for a fixed k there is a logspace reduction from $SDC(l, k)$ to $SDCE(l, k)$ as follows: Let $(G, \{v_1, v_2, \ldots, v_k\})$ be an instance of $SDC(l, k)$. Assume without loss of generality that the marked vertices form an independent set, or otherwise split the edge into two by introducing a new vertex in the middle. For each i, choose a vertex u_i, a neighbour of v_i, such that $i \neq j \implies u_i \neq u_j$, we solve $(G, \{e_1, e_2, \ldots e_k\})$ an instance of $SDCE(l, k)$ where $e_i = \{u_i, v_i\}$ and output the smallest solution amongst all the instances of $SDCE$ thus created. Since for each i, $\deg(v_i) < n$, number of instances of $SDCE$ created are bounded by $O(n^k)$ all of which can be solved in parallel as k is fixed.

5.1 Pre-processing

Given a graph $G = (V, E, w)$ and k marked edges $\{e_i = \{s_i, t_i\}\}_{i \leq k}$, assign weight $x^{w(e)}$ to the edge e of G and add self loops (weight 1) on all vertices except $\{s_i, t_i\}_{i \leq k}$. Observe that all the non-zero terms appearing in the permanent of adjacency matrix correspond to a cycle cover in G. To force these k-edges in our cycle cover, we direct these edges in a certain way which we shall call as a *pattern*.

Formally, define a pattern P as an ordered pairing of terminals of given edges $\{s_i, t_i \mid 1 \leq i \leq k\}$. Furthermore, we view each undirected edge $\{u, v\}$ in G as two directed edges (u, v) and (v, u) with the same weight. For any pattern P, define a pattern graph G_P with the same vertex/edge set as of G but such that if $(u, v) \in P$ then all outgoing edges from u, except edge (u, v), are deleted. We denote by A_P the adjacency matrix of G_P.

Now we shall show how to solve the $SDCE(1, k)$ problem for any $k \geq 1$. This algorithm essentially follows from the work of [28]. Next, we also present how to solve the $SDCE(2, k)$ problem for any $k \geq 2$. As far as we know, no algorithm (better than brute force) was known apriori to our work for $k \geq 3$.

5.2 Shortest Cycle

Let $\{e_i = \{s_i, t_i\}\}_{i \le k}$ be given k-edges. For each binary sequence $b = (b_1, b_2, \dots, b_{k-1})$ of length $k-1$, consider the following pattern P_b:

- $(s_1, t_1) \in P_b$
- $\forall 2 \le i \le k$, if $b_{i-1} = 0$ then $(s_i, t_i) \in P_b$ else $(t_i, s_i) \in P_b$

So $\{P_b\}_b$ is the collection of patterns with the orientation of e_1 fixed and all possible orientations of the other edges $\{e_i\}_{i \ge 2}$, as dictated by each binary sequence.

▷ Claim 25. Under the assumption that the shortest cycle is unique, the smallest exponent with non-zero coefficient in $f_1(x) \pmod 2$ is the weight of unique shortest cycle passing through the given edges, where

$$f_1(x) = \sum_b \text{perm}(A_{P_b})$$

Proof. Let C be any cycle cover which consists of at least 2 non-trivial cycles. Consider the cycle in C which doesn't contain edge e_1 - there are two ways of orienting this cycle, namely clockwise and counter-clockwise. So this cycle cover contributes to $f_1(x)$ for at least two such b-sequences and so it vanishes modulo $f_1 \pmod 2$.

Thus the only terms that survive in $f_1 \pmod 2$ are the cycle covers which consist of one cycle passing through all the given edges and self-loops on the remaining vertices, and furthermore number of cycles of this weight must be odd.

Since the shortest weight cycle was unique by our assumption, we get the desired result. ◁

To drop the assumption that a unique minimum weight solution exists, we instead assign modified weights $2nmw(e) + w'(e)$ where $n = |V(G)|$, $m = |E(G)|$, $w(e)$ is the given weight of edge e and $w'(e) \in \{0, 2, \dots, 2m-1\}$ is choosen independently and uniformly at random for each edge e. Then isolation lemma [20] tells us that, with probability $\frac{1}{2}$, the minimum weight cycle is unique. Hence if the term x^j survives as the smallest exponent with non-zero coefficient term in $f_1(x) \pmod 2$ then we get the weight of shortest cycle as $\lfloor j/2nm \rfloor$.

But this gives us only a randomized ⊕L algorithm (that is RNC algorithm). To further show that a common poly weight scheme exists for all graphs of size n, we use the well-known result of [21] which immediately places this problem in ⊕L/poly. For completeness sake, we provide a proof of this fact in the appendix B.

5.3 Shortest Two Disjoint Cycles

We shall first prove the following stronger result:

▶ **Theorem 26.** *Given a set of k-edges $\{e_i\}_{i \le k}$, we can find weight of the shortest two disjoint cycles passing through these edges such that e_1 and e_2 appear in different cycles in* ⊕L/poly *(and* RNC*)*

Let $\{P_b\}_b$ be the patterns as defined above. Furthermore, for each binary sequence $c = (c_1, c_2, \dots, c_{k-2})$ of length $k-2$, define pattern Q_c as

- $(s_1, s_2) \in Q_c$
- $(t_1, t_2) \in Q_c$
- $\forall 3 \le i \le k$, if $c_{i-2} = 0$ then $(s_i, t_i) \in Q_c$ else $(t_i, s_i) \in Q_c$

With a combination of these patterns, we can get our desired cycle covers. We claim that the non-zero terms appearing in

$$f_2(x) = \sum_b \operatorname{perm}(A_{P_b}) - \sum_c \operatorname{perm}(A_{Q_c})$$

correspond to cycle covers in G_{P_b} such that edges e_1 and e_2 appear in different cycles.

To prove our claim, we need to argue that the cycle covers of G_{P_b} in which e_1 and e_2 appear in the same cycle are exactly the cycle covers of G_{Q_c} Let

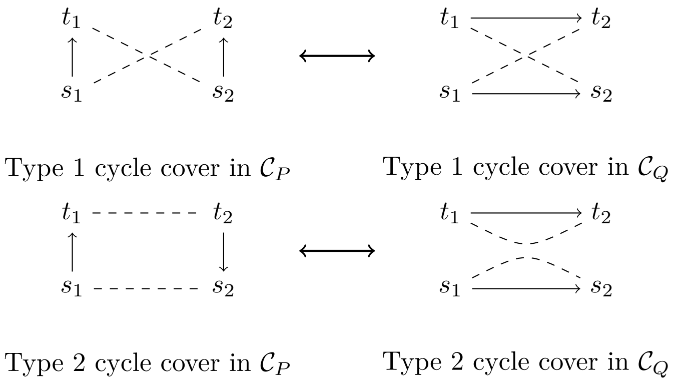
$$\mathcal{C}_Q = \bigsqcup_c \text{ cycle covers of } G_{Q_c}$$

$$\mathcal{C}_P = \bigsqcup_b \text{ cycle covers of } G_{P_b} \text{ such that edges } e_1 \text{ and } e_2 \text{ appear in the same cycle}$$

where each cycle cover is counted with repetitions in $\mathcal{C}_P$ and $\mathcal{C}_Q$.

▷ Claim 27. There is a one-one correspondence between $\mathcal{C}_P$ and $\mathcal{C}_Q$.

Proof. We define the mapping $\varphi : \mathcal{C}_P \to \mathcal{C}_Q$ as follows. Given a cycle cover in $\mathcal{C}_P$, remove the edges e_1 and e_2 and add edges (s_1, s_2) and (t_1, t_2), refer to Figure 2 below.



Type 1 cycle cover in $\mathcal{C}_P$ Type 1 cycle cover in $\mathcal{C}_Q$

Type 2 cycle cover in $\mathcal{C}_P$ Type 2 cycle cover in $\mathcal{C}_Q$

Figure 2 Bijection between $\mathcal{C}_P$ and $\mathcal{C}_Q$.

To show that this a well-defined mapping and indeed a bijection, we partition $\mathcal{C}_P$ into type 1 and type 2 cycle covers, depending upon the orientation of the edge e_2. Consider the cycle in which e_1 and e_2 appear together. Then if the edge e_2 is oriented as (s_2, t_2) then we call it type 1 cycle cover otherwise we call it a type 2 cycle cover.

Similarly, we partition $\mathcal{C}_Q$ into type 1 and type 2 cycle covers. If the edges $\{s_1, s_2\}$ and $\{t_1, t_2\}$ appear in the same cycle then we call it a type 1 cycle cover otherwise we call it a type 2 cycle cover.

Fix a type 1 cycle cover of $\mathcal{C}_P$. Then it contains a cycle of the form

$$(s_1 \xrightarrow{e_1} t_1 \to P_1 \to s_2 \xrightarrow{e_2} t_2 \to P_2 \to s_1)$$

Applying φ to this cycle cover we get the cycle

$$(s_1 \xrightarrow{e_1} s_2 \to P_1^{\text{reverse}} \to t_1 \xrightarrow{e_2} t_2 \to P_2^{\text{reverse}} \to s_1)$$

and the other cycles remain intact. This constitutes a type 1 cycle cover in $\mathcal{C}_Q$

Similarly, consider a type 2 cycle cover of $\mathcal{C}_P$ with the cycle

$$(s_1 \xrightarrow{e_1} t_1 \to P_1 \to t_2 \xrightarrow{e_2} s_2 \to P_2 \to s_1)$$

Applying φ to this cycle cover we get two cycles

$$(s_1 \xrightarrow{e_1} s_2 \to P_2^{\text{reverse}} \to s_1)$$

$$(t_1 \xrightarrow{e_2} t_2 \to P_1^{\text{reverse}} \to t_1)$$

and the other cycles remain intact. This constitutes a type 2 cycle cover in $\mathcal{C}_Q$.

Therefore, φ is a well-defined mapping and furthermore type i cycle covers of $\mathcal{C}_P$ are mapped to type i cycle covers of $\mathcal{C}_Q$, $i \in \{1,2\}$. Now consider $\psi : \mathcal{C}_Q \to \mathcal{C}_P$ defined as follows. Given a cycle cover in $\mathcal{C}_Q$, remove the edges (s_1, s_2) and (t_1, t_2) in the cycle and insert edges e_1 and e_2 with the orientation decided by the type. By an similar argument as above, we get that ψ is well-defined and clearly ψ is inverse of φ. ◁

Now suppose C is a cycle cover of G such that edges e_1 and e_2 appear in different cycles. We have two cases:

Case 1: number of non-trivial cycles in C is more than 2. Consider any two non-trivial cycles in C such that e_1 is not incident on them. We can orient these two cycles in both clockwise and anti-clockwise direction and so we get that C is a cycle cover in G_{P_b} for at least 4 b-sequences. Hence, the term corresponding to C cancels out in $f_2 \pmod 4$

Case 2: number of non-trivial cycles is exactly two. Then C is a cycle cover in G_{P_b} for exactly two b-sequences , that is the the cycle passing through e_2 has two possible orientations whereas cycle passing through e_1 has a fixed orientation (as orientation of e_1 remains fixed in all G_{P_b}). Hence, the term corresponding to C appears with a coefficient of two in $f_2 \pmod 4$. Therefore, the non-zero terms in $f_2 \pmod 4$ correspond to only the cycle covers in which edges e_1 and e_2 appear in different cycles and number of non-trivial cycles is exactly 2. Assuming a unique shortest two disjoint cycle exists, it's weight can be obtained from the smallest exponent with a non-zero coefficient in $f_2 \pmod 4$. Finally, to drop this assumption, we again assign random weights as done previously to ensure that the minimum weight solution is unique, with high probability. Furthermore, as in the previous case, we can again follow the proof of [21] to obtain a $\oplus$L/poly algorithm. This completes the proof of Theorem 26.

▶ **Corollary 28** (Theorem 2 restated). *Given a set of k-edges $\{e_i\}_{i\leq k}$, we can find weight of the shortest two cycles passing through these edges in $\oplus$L/poly (and RNC)*

Proof. For each pair of edges e_i and e_j, we can find weight of the shortest cycles separating them using the above algorithm. Hence taking the minimum over all pairs, we get our desired result. ◀

5.4 Constructing Cycles

We remark that under the assumption that the shortest cycle(s) are unique, we can recover these cycles C just from the knowledge of their weight $w(C)$. This follows the standard strategy of solving search via isolation as in [20]: for each edge $e \notin \{e_1, \ldots, e_k\}$, delete the edge e and call the resulting graph G_e. Running our algorithm on $(G_e, \{e_1, \ldots, e_k\})$, if the shortest cycle(s) weight is more than $w(C)$, then discard e otherwise add e to the set C, which gives us the required cycle(s). Doing this procedure in parallel for each edge e, it can be easily seen that we can recover C in $\oplus$L/poly (or RNC).

6 Conclusion

We started by recognizing the appropriate algebraic structure $\mathfrak{R}$ over which we can present a parallel algorithm to compute permanent modulo 2^k. Then we saw two techniques to get permanent over $\mathbb{Z}[x] \pmod{2^k}$ from $\mathfrak{R} \pmod{2^k}$. First method was to choose a large enough irreducible polynomial for our ring $\mathfrak{R}$. Another method was to interpolate over the ring $\mathfrak{R}$, which was an extension of the commonly known interpolation over finite fields.

Then we considered applications for parallel polynomial permanent. This includes a direct parallelization of the shortest two disjoint paths problem as given by [2]. Another direct application, although which required some modification, was finding shortest cycle passing through given vertices [28]. We further presented a common framework to view the above mentioned problems. This also aided us in further generalizing and obtaining a parallel algorithm to find shortest two disjoint cycles in any weighted undirected graph.

The more general question of computing permanent over arbitrary commutative rings of characteristic 2^k for $k \geq 2$ still remains open. On the other hand, using the framework we presented, we ask if it is possible to obtain shortest k disjoint cycles for $k \geq 3$ in RNC or even perhaps in randomized polynomial time?

References

1 Andreas Björklund and Thore Husfeldt. Counting shortest two disjoint paths in cubic planar graphs with an NC algorithm. In Wen-Lian Hsu, Der-Tsai Lee, and Chung-Shou Liao, editors, *29th International Symposium on Algorithms and Computation, ISAAC 2018, December 16-19, 2018, Jiaoxi, Yilan, Taiwan*, volume 123 of *LIPIcs*, pages 19:1–19:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ISAAC.2018.19.

2 Andreas Björklund and Thore Husfeldt. Shortest two disjoint paths in polynomial time. *SIAM J. Comput.*, 48(6):1698–1710, 2019. doi:10.1137/18M1223034.

3 Mark Braverman, Raghav Kulkarni, and Sambuddha Roy. Space-efficient counting in graphs on surfaces. *Comput. Complex.*, 18(4):601–649, 2009. doi:10.1007/s00037-009-0266-4.

4 Marek Cygan, Dániel Marx, Marcin Pilipczuk, and Michal Pilipczuk. The planar directed k-vertex-disjoint paths problem is fixed-parameter tractable. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 197–206. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.29.

5 Carsten Damm. Problems complete for parityl. *Information Processing Letters*, 36(5):247–250, 1990. doi:10.1016/0020-0190(90)90150-V.

6 Samir Datta, Siddharth Iyer, Raghav Kulkarni, and Anish Mukherjee. Shortest k-disjoint paths via determinants. In Sumit Ganguly and Paritosh K. Pandya, editors, *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018, December 11-13, 2018, Ahmedabad, India*, volume 122 of *LIPIcs*, pages 19:1–19:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.FSTTCS.2018.19.

7 Éric Colin de Verdière and Alexander Schrijver. Shortest vertex-disjoint two-face paths in planar graphs. *ACM Trans. Algorithms*, 7(2):19:1–19:12, 2011. doi:10.1145/1921659.1921665.

8 W. Eberly. Very fast parallel matrix and polynomial arithmetic. In *25th Annual Symposium onFoundations of Computer Science, 1984.*, pages 21–30, 1984. doi:10.1109/SFCS.1984.715897.

9 W. Eberly. Efficient parallel independent subsets and matrix factorizations. In *Proceedings of the Third IEEE Symposium on Parallel and Distributed Processing*, pages 204–211, 1991. doi:10.1109/SPDP.1991.218278.

10 Faith E. Fich and Martin Tompa. The parallel complexity of exponentiating polynomials over finite fields. *J. ACM*, 35(3):651–667, 1988. doi:10.1145/44483.44496.

11 Steven Fortune, John E. Hopcroft, and James Wyllie. The directed subgraph homeomorphism problem. *Theor. Comput. Sci.*, 10:111–121, 1980. doi:10.1016/0304-3975(80)90009-2.

12 William Hesse, Eric Allender, and David A. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *J. Comput. Syst. Sci.*, 65(4):695–716, 2002. doi:10.1016/S0022-0000(02)00025-9.

13 Hiroshi Hirai and Hiroyuki Namba. Shortest (a+b)-path packing via hafnian. *Algorithmica*, 80(8):2478–2491, 2018. doi:10.1007/s00453-017-0334-0.

14 Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, USA, 2nd edition, 2012.

15 Ce Jin, Nikhil Vyas, and Ryan Williams. Fast low-space algorithms for subset sum. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1757–1776. SIAM, 2021. doi:10.1137/1.9781611976465.106.

16 James F. Lynch. The equivalence of theorem proving and the interconnection problem. *SIGDA Newsl.*, 5(3):31–36, 1975. doi:10.1145/1061425.1061430.

17 Meena Mahajan, P. R. Subramanya, and V. Vinay. The combinatorial approach yields an nc algorithm for computing pfaffians. *Discret. Appl. Math.*, 143(1-3):1–16, 2004. doi:10.1016/j.dam.2003.12.001.

18 Meena Mahajan and V. Vinay. Determinant: Combinatorics, algorithms, and complexity. *Chic. J. Theor. Comput. Sci.*, 1997, 1997. URL: http://cjtcs.cs.uchicago.edu/articles/1997/5/contents.html.

19 Ketan Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Comb.*, 7(1):101–104, 1987. doi:10.1007/BF02579205.

20 Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, page 345–354, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/28395.383347.

21 Klaus Reinhardt and Eric Allender. Making nondeterminism unambiguous. *SIAM J. Comput.*, 29(4):1118–1131, 2000. doi:10.1137/S0097539798339041.

22 Heike Ripphausen-Lipa, Dorothea Wagner, and Karsten Weihe. Linear-time algorithms for disjoint two-face paths problems in planar graphs. *Int. J. Found. Comput. Sci.*, 7(2):95–110, 1996. doi:10.1142/S0129054196000087.

23 N. Robertson and P.D. Seymour. Graph minors .xiii. the disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995. doi:10.1006/jctb.1995.1006.

24 Alexander Schrijver. Finding k disjoint paths in a directed planar graph. *SIAM J. Comput.*, 23(4):780–788, 1994. doi:10.1137/S0097539792224061.

25 Hitoshi Suzuki, Takehiro Akama, and Takao Nishizeki. Finding steiner forests in planar graphs. In David S. Johnson, editor, *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, 22-24 January 1990, San Francisco, California, USA*, pages 444–453. SIAM, 1990. URL: http://dl.acm.org/citation.cfm?id=320176.320230.

26 Leslie G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979. doi:10.1016/0304-3975(79)90044-6.

27 J.H. van Lint. *Introduction to Coding Theory*. Graduate Texts in Mathematics. Springer Berlin Heidelberg, 2013. URL: https://books.google.co.in/books?id=6dbqCAAAQBAJ.

28 Magnus Wahlström. Abusing the tutte matrix: An algebraic instance compression for the k-set-cycle problem. In *STACS*, pages 341–352, 2013.

29 Viktória Zankó. #p-completeness via many-one reductions. *Int. J. Found. Comput. Sci.*, 2(1):77–82, 1991. doi:10.1142/S0129054191000066.

A Examples

Example 1

Let $A = \begin{pmatrix} 1 & x+1 & x+2 \\ x & x^2 & x^2+x \\ x^2 & 3 & x^2+3 \end{pmatrix}$, $p(x) = x^6 + x^3 + 1$ be the irreducible polynomial and we want to evaluate $\text{perm}(A) \pmod 4$ over the ring $\mathfrak{R} = \mathbb{Z}[x]/(p(x))$. First of all, a direct computation gives us $\text{perm}(A) = 2x^5 + 6x^4 + 2x^3 + 12x^2 + 12x$. Now we demonstrate the steps taken by our algorithm.

Step 1: We start by evaluating $\text{perm}(A) \pmod 2$. We directly notice here that last column is the sum of first two columns and so $\det(A) = 0 \implies \text{perm}(A) \equiv 0 \pmod 2$

Step 2: Since $\det(A) \equiv 0 \pmod 2$, we solve the equation $A^T v = 0$ over $\mathbb{F}$ by our method as follows: $\begin{pmatrix} 1 & x & x^2 \\ x+1 & x^2 & 1 \\ x & x^2+x & x^2+1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = 0$

Since rank of the principal 2×2 submatrix is already 2, we set $v_3 = 1$ and solve the equation: $\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = -\begin{pmatrix} 1 & x \\ x+1 & x^2 \end{pmatrix}^{-1} \begin{pmatrix} x^2 \\ 1 \end{pmatrix} 1$ to get $v_1 = x^3 + 1$ and $v_2 = x^5 + x$.

Step 3: For each $j = 1, 2, 3$, we find b_j such that $\sum_i v_i a_{ij} = 2b_j \pmod 4$

$$\begin{aligned}
j = 1: &\qquad (x^3+1) + x(x^5+x) + x^2 = 2x^2 \\
j = 2: &\qquad (x+1)(x^3+1) + x^2(x^5+x) + 3 = 2x^3 \\
j = 3: &\qquad (x+2)(x^3+1) + (x^2+x)(x^5+x) + x^2 + 3 = 2x^3 + 2x^2
\end{aligned}$$

Step 4: We have the formula

$$\begin{aligned}
\text{perm}(A) \pmod 4 = 2 &\left(\sum_{j=1}^{3} b_j \text{perm}(A[\widehat{\{3\}}, \widehat{\{j\}}]) \pmod 2 \right) \\
&- 2 \sum_{i=1}^{2} v_i \left(\sum_{\substack{j,k=1 \\ j<k}}^{3} a_{ij} a_{ik} \text{perm}(A[\widehat{\{3, i\}}, \widehat{\{j, k\}}]) \pmod 2 \right)
\end{aligned}$$

Step 4.1:

$$\text{perm}(A[\widehat{\{3\}}, \widehat{\{1\}}]) = \text{perm} \begin{pmatrix} x+1 & x+2 \\ x^2 & x^2+x \end{pmatrix} = x \pmod 2$$

$$\text{perm}(A[\widehat{\{3\}}, \widehat{\{2\}}]) = \text{perm} \begin{pmatrix} 1 & x+2 \\ x & x^2+x \end{pmatrix} = x \pmod 2$$

$$\text{perm}(A[\widehat{\{3\}}, \widehat{\{3\}}]) = \text{perm} \begin{pmatrix} 1 & x+1 \\ x & x^2 \end{pmatrix} = x \pmod 2$$

$$\implies \sum_{j=1}^{3} b_j \text{perm}(A[\widehat{\{3\}}, \widehat{\{j\}}]) = ((x^3) + (x^4) + (x^4 + x^3)) = 0 \pmod 2$$

Step 4.2:

$$\sum_{\substack{j,k=1\\ j<k}}^{3} a_{1j}a_{1k}\mathrm{perm}(A[\widehat{\{1,3\}},\widehat{\{j,k\}}])$$
$$= (x+1)(x^2+x)+(x+2)x^2+(x+1)(x+2)x = x^3+x^2+x \pmod 2$$
$$\sum_{\substack{j,k=1\\ j<k}}^{3} a_{2j}a_{2k}\mathrm{perm}(A[\widehat{\{2,3\}},\widehat{\{j,k\}}])$$
$$= x^3(x+2)+x(x^2+x)(x+1)+x^2(x^2+x) = x^4+x^3+x^2 \pmod 2$$

$$\sum_{i=1}^{2} v_i \left(\sum_{\substack{j,k=1\\ j<k}}^{3} a_{ij}a_{ik}\mathrm{perm}(A[\widehat{\{3,i\}},\widehat{\{j,k\}}]) \pmod 2 \right) = x^5+x^4+x^3 \pmod 4$$

Therefore, $\mathrm{perm}(A) \pmod 4 = 2x^5+2x^4+2x^3$ which matches with our direct computation.

Example 2

Consider $A = \begin{pmatrix} 1 & x & x^2 \\ x & x^2 & 1 \\ 1 & x^2 & x \end{pmatrix}$, and so $\mathrm{perm}(A) = x^5+x^4+x^2+x$. Therefore, we now have $\det(A) \not\equiv 0 \pmod 2$

Step 1: Find Q such that QA has all leading principal minors are non-zero. In this case, we will get $Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \implies QA = \begin{pmatrix} 1 & x & x^2 \\ 1 & x^2 & x \\ x & x^2 & 1 \end{pmatrix}$

Step 2: Consider matrix C whose all entries are same as A except the last one which is incremented by y, that is $C = \begin{pmatrix} 1 & x & x^2 \\ 1 & x^2 & x \\ x & x^2 & 1+y \end{pmatrix}$, then $\mathrm{perm}(C) = \mathrm{perm}(A) + y\mathrm{perm}(A[\widehat{\{3\}},\widehat{\{3\}}])$. Again construct C' same as $A[\widehat{\{3\}},\widehat{\{3\}}]$ but replace the last entry incremented by y', that is $C' = \begin{pmatrix} 1 & x \\ x & x^2+y' \end{pmatrix} \implies \mathrm{perm}(C') = \mathrm{perm}(A[\widehat{\{3\}},\widehat{\{3\}}]) + y'\mathrm{perm}(A[\widehat{\{2,3\}},\widehat{\{2,3\}}])$. Written as one equation, we get

$$\mathrm{perm}(A) = \mathrm{perm}(C) - y\,(\mathrm{perm}(C') - y'a_{11})$$

In this equation, both C, C' are matrices with $\det \equiv 0 \pmod 2$ with the correct choice of y, y', which were:

$$y_0 = \mathrm{perm}(A)\mathrm{perm}(A[\widehat{\{3\}},\widehat{\{3\}}])^{-1} \pmod 2 = (x^5+x^4+x^2+x)(x^4+x^3+x^2) = x^3+1$$
$$y_0' = \mathrm{perm}(A[\widehat{\{3\}},\widehat{\{3\}}])\mathrm{perm}(A[\widehat{\{2,3\}},\widehat{\{2,3\}}])^{-1} \pmod 2 = x^2+x$$

So we can compute $\mathrm{perm}(C)$ and $\mathrm{perm}(C')$ by above method and substitute it into previous equation to get $\mathrm{perm}(A) \pmod 4$.

B Weights as a Polynomial Advice

Our proof is based on [21]. Call a weighted undirected graph (G, w) (w is the given weight function on edges) *min-k-unique*, if for any k marked edges on G, there exists unique shortest l disjoint cycles passing through these k edges. Our goal is to show for each $n > 0$ there exists a set of n^2 weight functions $w_1, \dots, w_{n^2}$ such that given a graph G on n vertices, (G, w_i) is *min-k-unique* for some $i \in [1, n^2]$.

Given a graph G on n vertices and k marked edges $e_1, \dots, e_k$, let $\mathcal{F}(e_1, \dots, e_k)$ be the family of all l disjoint cycles passing through $e_1, \dots, e_k$. Using isolation lemma [20], if w is a random weight function, that is each edge is assigned a weight from $[1, 4n^{2k+2}]$ independently and uniformly at random, then probability that $\mathcal{F}(e_1, \dots, e_k)$ has a unique minimum weight element is at least $1 - 1/4n^{2k}$. Therefore, probability that (G, w) is not *min-k-unique* for a random weight function w is at most

$$\Pr[\exists e_1, \dots, e_k : \mathcal{F}(e_1, \dots, e_k) \text{ doesn't have a minimum weight element}] \le \sum_{e_1,\dots,e_k} \frac{1}{4n^{2k}} \le 1/4$$

Now we claim that there exists a set of n^2 weight functions $W = (w_1, \dots, w_{n^2})$ such that for any given graph G on n vertices, (G, w_i) is *min-k-unique* for some $1 \le i \le n^2$. We say *W is bad* if it doesn't meet this criteria and in particular *W is bad for G*, if none of (G, w_i) is *min-k-unique*. For a randomly choosen W, that is each w_i is chosen independently and uniformly at random, then

$$\Pr[W \text{ is bad for } G] \le \Pr[\forall i : (G, w_i) \textit{ is not min-}k\textit{-unique}] \le \left(\frac{1}{4}\right)^{n^2}$$

$$\implies \Pr[W \text{ is bad}] \le \Pr[\exists G : W \text{ is bad for } G] \le 2^{n^2} \left(\frac{1}{4}\right)^{n^2} < 1$$

Hence there exists some $W = (w_1, \dots, w_{n^2})$ which satisfies the above property and so W is the required poly advice. To complete the argument for $SDCE(1, k), SDCE(2, k) \in \oplus\mathsf{L}$, we obtain the weight of shortest cycle(s), using each of the weight functions w_i and output the minimum amongst them.

C Hafnians and counting perfect matchings modulo 2^k

Similar to permanent and determinant, another pair of well-studied algebraic analogous functions on a matrix are hafnians and pfaffians. Let $A = (a_{ij})$ be a symmetric $2n \times 2n$ matrix over integers, hafnian is defined as

$$\mathrm{hf}(A) = \frac{1}{2^n n!} \sum_{\sigma \in S_{2n}} \prod_{j=1}^{n} a_{\sigma(2j-1),\sigma(2j)} \tag{2}$$

Note that the diagonal entries of A don't contribute in the calculation of hafnians and hence we can assume them to be 0. Let $B = (b_{ij})$ be a skew-symmetric $2n \times 2n$ matrix, pfaffian is defined as

$$\mathrm{pf}(B) = \frac{1}{2^n n!} \sum_{\sigma \in S_{2n}} \mathrm{sgn}(\sigma) \prod_{j=1}^{n} b_{\sigma(2j-1),\sigma(2j)} \tag{3}$$

But notice that $\mathrm{hf}(A) \equiv \mathrm{pf}(A) \pmod 2$. [17] have shown that $\mathrm{pf}(A)$ can be computed in NC and hence as an immediate consequence we get that $\mathrm{hf}(A) \pmod 2$ can be computed in NC. We can reduce the computation of hafnian to several hafnians of smaller submatrices using the following lemma. Denote by $A[i,j]$ the matrix obtained from A by deleting rows i and j, columns i and j.

▶ **Lemma 29** ([13, Lemma 2.2])**.**

$$\mathrm{hf}(A) = \sum_{j:j\neq i} a_{ij}\mathrm{hf}(A[i,j])$$

$$\mathrm{hf}(A) = a_{ij}\mathrm{hf}(A[i,j]) + \sum_{pq:p,q\notin\{i,j\},p\neq q} (a_{ip}a_{jq} + a_{iq}a_{jp})\mathrm{hf}(A[i,j,p,q])$$

Assume $\mathrm{pf}(A) \equiv 0 \pmod 2$, then $\det(A) \equiv 0 \pmod 2$ and we can find a vector $v \in \mathbb{Z}_2^{2n}$ such that $Av = A^T v = 0 \pmod 2$. Assume without loss of generality $v_1 = 1$.

Let r_i, c_i denote the i^{th} row and i^{th} column of A respectively.

- Construct A' by replacing first row with $\sum v_i r_i$ and then replacing first column with $\sum v_i c_i$
- Construct A_i by replacing first row with r_i and first column with c_i.

Then we check that

$$\begin{aligned}
\mathrm{hf}(A') &= \sum_{j>1}\left(\sum_{i\geq 1} v_i a_{ij}\right)\mathrm{hf}(A[1,j]) \\
&= \sum_{i\geq 1} v_i \left(\sum_{j>1} a_{ij}\mathrm{hf}(A[1,j])\right) \\
&= \sum_{j>1} a_{1j}\mathrm{hf}(A[1,j]) + \sum_{i>1} v_i \left(\sum_{j>1} a_{ij}\mathrm{hf}(A[1,j])\right) \\
&= \mathrm{hf}(A) + \sum_{i>1} v_i \mathrm{hf}(A_i)
\end{aligned}$$

Computing $\mathbf{hf(A')}$: since $A^T v = 0 \pmod 2 \implies \sum_{i\geq 1} v_i a_{ij} = 2b_j \pmod{2^k}$ for some $c_j \in \mathbb{Z}$ and hence

$$\begin{aligned}
\mathrm{hf}(A') &= \sum_{j>1}\left(\sum_{i\geq 1} v_i a_{ij}\right)\mathrm{hf}(A[1,j]) \\
&= \sum_{j>1} 2b_j \mathrm{hf}(A[1,j])
\end{aligned}$$

$$\implies \mathrm{hf}(A') \pmod{2^k} = 2\left(\sum_{j>1} b_j \mathrm{hf}(A[1,j]) \pmod{2^{k-1}}\right)$$

Computing $\mathrm{hf}(A_i)$:

$$\mathrm{hf}(A_i) = a_{ii}\mathrm{hf}(A[1,i]) + \sum_{pq:p,q\notin\{1,i\},p\neq q} 2a_{ip}a_{iq}\mathrm{hf}(A[1,i,p,q])$$

$$\implies \mathrm{hf}(A_i) \pmod{2^k} = 2\left(\sum_{pq:p,q\notin\{1,i\},p\neq q} a_{ip}a_{iq}\mathrm{hf}(A[1,i,p,q]) \pmod{2^{k-1}}\right)$$

Thus, we can compute $\mathrm{hf}(A) \pmod{2^k}$ provided $\mathrm{pf}(A) \equiv 0 \pmod 2$.

Now if $\mathrm{pf}(A) \not\equiv 0 \pmod 2$, then we can find $(i,j), i \neq j$ such that $\mathrm{hf}(A[i,j]) \not\equiv 0 \pmod 2$. Consider the matrix C where all entries are same as in A except a_{ij} is replaced with $a_{ij}+1$, then we get $\mathrm{hf}(C) = \mathrm{hf}(A)+\mathrm{hf}(A[i,j])$. Since $\mathrm{hf}(C) \equiv 0 \pmod 2$, we can compute $\mathrm{hf}(C) \pmod{2^k}$ as described above and since $\mathrm{hf}(A[i,j])$ is a $(n-2)\times(n-2)$ matrix, we compute it's hafnian recursively modulo 2^k. Therefore, we can compute $\mathrm{hf}(A) = \mathrm{hf}(C) - \mathrm{hf}(A[i,j]) \pmod{2^k}$.

This gives us a P algorithm for computing hafnians modulo 2^k.

Counting perfect matchings modulo 2^k

Let G be an undirected graph and A_G denote the adjacency matrix of the graph G. If G has odd number of vertices, then clearly there aren't any perfect matchings. Otherwise it is straight-forward to see that number of perfect matchings in G is same as the value $\mathrm{hf}(A_G)$. Hence the result follows.