Matroid Intersection: A Pseudo-Deterministic Parallel Reduction from Search to Weighted-Decision

Sumanta Ghosh
Indian Institute of Technology Bombay, India

Rohit Gurjar
Indian Institute of Technology Bombay, India

Abstract

We study the matroid intersection problem from the parallel complexity perspective. Given two matroids over the same ground set, the problem asks to decide whether they have a common base and its search version asks to find a common base, if one exists. Another widely studied variant is the weighted decision version where with the two matroids, we are given small weights on the ground set elements and a target weight W, and the question is to decide whether there is a common base of weight at least W. From the perspective of parallel complexity, the relation between the search and the decision versions is not well understood. We make a significant progress on this question by giving a pseudo-deterministic parallel (NC) algorithm for the search version that uses an oracle access to the weighted decision.

The notion of pseudo-deterministic NC was recently introduced by Goldwasser and Grossman [19], which is a relaxation of NC. A pseudo-deterministic NC algorithm for a search problem is a randomized NC algorithm that, for a given input, outputs a fixed solution with high probability. In case the given matroids are linearly representable, our result implies a pseudo-deterministic NC algorithm (without the weighted decision oracle). This resolves an open question posed by Anari and Vazirani [2].

2012 ACM Subject Classification Mathematics of computing → Matroids and greedoids; Mathematics of computing → Probabilistic algorithms

Keywords and phrases Linear Matroid, Matroid Intersection, Parallel Complexity, Pseudo-deterministic NC

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2021.41

Category RANDOM

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2021/121/

Acknowledgements We thank the anonymous reviewers for pointing towards the relevant literature on lattice families.

1 Introduction

Most often, a search problem can be efficiently solved using an oracle for a closely related decision problem. For example, if you have access to a decision oracle that tells you whether a given graph has a perfect matching, you can efficiently construct a perfect matching in a given graph using the decision oracle. Such search-to-decision reductions usually involve self-reducibility and make a linear number of oracle calls sequentially. However such reductions do not fit into the framework of parallel complexity, where one can make multiple oracle calls in parallel, but wants poly-logarithmic time complexity. For a more detailed discussion on the difference in parallel complexity of search and decision problems, see [25].

Graph matching and related problems like linear matroid intersection and linear matroid matching were one of the first problems to be studied from the parallel complexity perspective [26, 5]. The decision versions of these problems ask to decide the existence of the respective combinatorial substructures:

- Matching: Does a given graph contain a perfect matching – a set of disjoint edges that cover all the vertices in the graph?
- Linear Matroid Intersection: Given two sets of m vectors each, is there a set of indices $B \subseteq [m]$ that corresponds to a basis set in each of the two sets?
- Linear Matroid Matching/Parity: Given a set of pairs of vectors, is there a subset of pairs whose union will give a basis for the union of all pairs?

The search versions of these problems ask for constructing the respective combinatorial substructures (if one exists). The matching problem in bipartite graphs is a special case of all the three problems above (see Figure 1). A bipartite graph is a graph whose vertices can be partitioned into two parts such that every edge connects a vertex from one part to one in the other part. Even in the special case of bipartite matching, the questions of the exact parallel complexity of decision and search and whether decision and search are equivalent in a parallel sense still remain unresolved.

$$\begin{tikzcd}
& \text{Linear Matroid Matching/Parity} & \\
\text{Linear Matroid Intersection} \arrow[ur] & & \text{General Matching} \arrow[ul] \\
& \text{Bipartite Matching} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Figure 1 Reductions among the four problems. $A \to B$ represents that problem A reduces to problem B.

The first efficient randomized parallel algorithms for the three decision problems above followed from the results of Lovász [26]. Lovász gave randomized algorithms for these problems by first reducing these decision questions to testing whether the determinant of a certain symbolic matrix is nonzero, as a polynomial. Then he used the fact that the zeroness of a polynomial can be tested efficiently by just evaluating it at a random point [32, 36, 12, 29]. Hence, the questions were basically reduced to computing determinant of a randomly generated matrix. Interestingly, there are efficient parallel (NC) algorithms for computing the determinant of a matrix [3, 7, 4]. An NC algorithm is one which uses polynomially many parallel processors and takes only polylogarithmic time. Thus, the algorithms of Lovász [26] can be viewed as randomized parallel (RNC) algorithms for the three decision problems. However, this did not imply any parallel algorithms for the search versions.

Randomized parallel (RNC) algorithms for the search versions of these problems were obtained some years later [24, 27, 28]. However, these results did not go via a parallel search-to-decision reduction. Instead, they gave randomized parallel (RNC) reductions from the search version to a variant of the decision problem, namely *weighted decision*. For example, the weighted decision version for perfect matchings asks: given a graph with *small* weights on edges and a target weight W, is there a perfect matching of weight at most W (or at least W). Here the weight of a perfect matching is defined to be the sum of the weights of the edges in the perfect matching. It turns out that Lovász's RNC algorithms can be appropriately modified to solve the weighted decision versions as well, when the given weights are small. The search-to-weighted-decision reductions together with the weighted decision algorithms implied randomized parallel search algorithms for the three problems. We elaborate a bit on the reductions.

Reductions from search to weighted-decision

Karp, Upfal and Wigderson [24] do not explicitly talk about weights, but their reduction is from finding a perfect matching to a subroutine that can be viewed as weighted decision with 0-1 weights on the edges. From the perspective of our current investigation, the result of Mulmuley, Vazirani, and Vazirani [27] is much more interesting. They showed that using the weighted decision oracle, one can compute a perfect matching with just two rounds of parallel calls to the oracle. The crucial ingredient in their algorithm was the powerful Isolation Lemma which states that if the edges of a graph are assigned random weights from a polynomially bounded range uniformly and independently then with high probability, there is a *unique* minimum weight perfect matching in the graph. Once we have such a weight assignment, we can first find the minimum weight w^* of a perfect matching by calling the weighted decision oracle for each possible target value W in a polynomially bounded range. Then for each edge e in parallel, delete e and ask the oracle if there is a perfect matching of weight at most w^*. The answer will be no if and only if e is a part of the unique minimum weight perfect matching. Thus, in two rounds of polynomially many parallel oracle calls, we can compute the unique minimum weight perfect matching.

The amazing thing about the Isolation Lemma is that it applies to not just the family of perfect matchings in a graph, but to arbitrary families of subsets. Thus, the above described search-to-weighted-decision reduction of [27] can be made to work for any problem that admits a similar self-reducibility property. Narayan, Saran, and Vazirani [28] used the same Isolation Lemma based reduction to give RNC algorithms for the search versions of linear matroid intersection and linear matroid matching.

Derandomization

Since the work of Lovász [26], it has been a big open question to derandomize these results i.e., to find deterministic parallel (NC) algorithms for these problems. While derandomization results have been obtained for the matching problem in many special classes of graphs [11, 35, 10, 20, 1], the question remains open even for bipartite graphs. Only recently, there was a significant progress made when a quasi-NC algorithm was obtained for finding a perfect matching in a bipartite graph [16, 15]. A quasi-NC algorithm runs in polylogarithmic time but can use quasipolynomially ($2^{\log^{O(1)} n}$) many parallel processors, so this result brought the problem quite close to the class NC. Similar quasi-NC algorithms were later obtained for linear matroid intersection [22] and matching in general graphs [34] as well.

In the quest of understanding the deterministic parallel complexity of these problems, an interesting question one can ask is whether there is a deterministic parallel (NC) search-to-decision reduction. An easier question would be to ask for an NC reduction from search to weighted-decision, i.e., derandomizing the reductions of [27, 24, 28] described above. Soon after the quasi-NC result for bipartite matching [16], Goldwasser and Grossman [19] started quite an interesting line of enquiry, where they answered the above question positively for bipartite matching. They observed that the quasi-NC algorithm can be modified to give a deterministic parallel (NC) search-to-weighted-decision reduction for bipartite matching. Their main result was, what they call, a pseudo-deterministic NC algorithm for bipartite matching, which followed from this reduction.

Pseudo-determinism

The notion of pseudo-deterministic algorithms was introduced by Gat and Goldwasser [17] which is applicable only for search problems. For a given instance of a search problem, a randomized algorithm can possibly give different outputs for different choices of the random seed. Pseudo-deterministic algorithms are randomized algorithms which give a fixed output for a given input with high probability. Note that the earlier described RNC algorithm of [27] for matching is not pseudo-deterministic because for a given graph, it will output different perfect matchings for different possibilities of the randomly chosen weight assignment.

It is not hard to see that if one gives a *deterministic* reduction from a search problem to a decision problem that is known to have a randomized algorithm, then one immediately gets a pseudo-deterministic algorithm for the search problem (see [18, Theorem 2.2]). That is why the NC search-to-weighted-reduction for bipartite matching [19] implied a pseudo-deterministic NC algorithm for bipartite matching, i.e., an RNC algorithm that, for a given graph, outputs the same perfect matching with high probability. One interesting implication of this result is that if one finds an NC algorithm for the weighted-decision of bipartite matching, one will get an NC algorithm for the search version as well.

A natural question arises: can we similarly modify the quasi-NC algorithms for linear matroid intersection [22] and matching in general graphs [34] into NC search-to-weighted-decision reductions, and thus, get pseudo-deterministic NC algorithms for the search versions? It looks quite possible because one can can extract out an abstract framework from [19] for converting these quasi-NC algorithms into pseudo-deterministic NC algorithms. But as we discuss below, a straightforward application of this framework does not work out for linear matroid intersection or matching in general graphs. A key step in [19] is to *compute a succinct description* of the set of all (possibly exponentially many) minimum weight perfect matchings in a weighted bipartite graph in NC, given the weighted-decision oracle. However, it is not immediately clear how to solve the analogous question in NC for linear matroid intersection or matching in general graphs. Interestingly, in an earlier work in a different context, Cygan, Gabow, and Sankowski [9] had already solved this question for matching in general graphs. They had designed a procedure based on LP duality to compute a succinct description of the set of all minimum weight perfect matchings, via the weighted-decision oracle. Moreover, as observed in [30], this procedure can also be parallelized using standard techniques. Armed with this heavy hammer, Anari and Vazirani [2] give an NC search-to-weighted-decision reduction, and thus, get a pseudo-deterministic NC algorithm for perfect matching in general graphs. Anari and Vazirani [2] put it as an open question to obtain similar results for linear matroid intersection. In this work, we take up this challenge.

Our contributions

In the setting of linear matroid intersection, the analogue of a perfect matching is referred as a *common base* – a set of indices that corresponds to a basis in both the sets of vectors. For the weighted version, it is well understood how to succinctly describe the set of minimum or maximum weight common bases, i.e., the minimizing/maximizing face of the common base polytope; see e.g., [31, Chapter 41]. Any face of the common base polytope is characterized by its tight sets. Suppose that M_1 and M_2 are two matroids over the same ground set E. Then, a subset S of E is called a tight set for a maximizing face (of the common base polytope), if for some matroid M_i the following holds: for every maximum weight common base B, the set $S \cap B$ spans the set S. Note that the number of tight sets of a maximizing face can be exponentially large. However, they are known to have succinct representations. We give a randomized NC algorithm to compute a succinct and unique representation for the tights sets of a maximizing face.

▶ **Theorem 1** (Informal version of Theorem 5)**.** *There exists a randomized NC algorithm to compute a succinct and* unique *description for the tight sets of a maximizing face of the common base polytope, given the weighted-decision oracle.*

For a maximizing face of the common base polytope, all the tight sets for some matroid M_i forms a lattice family, and our description for tight sets is motivated by the succinct representation of lattice families based on the partial order of its prime subsets (also known as irreducible subsets). We construct a digraph in bottom-up fashion, using bases from the maximizing face of the common base polytope, such that it contains the necessary information regarding the tight sets of maximizing face. From this digraph we shall be able to compute the succinct description. See Section 3.1 for more details. Here, we would like to mention that the succinct representation of lattice families using the partial order of its prime subsets is well known and has been used in multiple previous algorithms [31, Chapter 49], [23, 14, 6]. However, all these applications do not fall in the category of parallel computation.

Note that the uniqueness of the description is important because then this RNC algorithm is by default pseudo-deterministic, as there is only one possible output. Once we have designed this heavy hammer, it is relatively easier to combine the procedure of [22] with the abstract framework provided by [19] and obtain a pseudo-deterministic NC search-to-weighted-decision reduction. This leads to our first main result.

▶ **Theorem 2.** *The search version of the linear matroid intersection problem has a pseudo-deterministic NC algorithm.*

General Matroid Intersection

Our main technical contributions are applicable to not just linear matroid intersection but also to matroid intersection. In the general matroid intersection problem, instead of two sets of vectors, we are given two matroids on the same ground set and the goal is to find a set of elements that forms a base in each of the two matroids. In this problem, the matroids are not given explicitly but only via a independence or rank oracle. Thus, it does not makes sense to talk about NC or RNC algorithms for this problem. One can however consider a parallel oracle model where we can make polynomially many queries to the oracle in parallel (see [25]). To the best of our knowledge, there is no such parallel algorithm known for the decision or the search version of matroid intersection, even with sub-linear number of rounds of parallel oracle calls. This makes the question all the more interesting whether decision and search are equivalent in a parallel sense.

Interestingly, the search-to-weighted-decision reduction of [28] applies to general matroid intersection as well and can be said to be in RNC. Our results make a significant progress on this question by giving a pseudo-deterministic NC reduction from search to weighted decision. Formally, we can show the following.

▶ **Theorem 3.** *There is a pseudo-deterministic NC algorithm for finding a common base of two matroids M_1 and M_2 on the same ground set E, provided that the algorithm has an oracle access to the following decision question: given two matroids with polynomially bound (in $|E|$) weights on the ground set elements and a target weight W, is there a common base of weight at least W? Furthermore, the oracle calls need to be made only for the following pairs of matroids: $\langle M_1, M_2\rangle$, $\langle M_1, M_1\rangle$, and $\langle M_2, M_2\rangle$.*

Note that in the above theorem, as there is no explicit input, the ground set size is taken as the input size.

Discussion

There are many natural open questions that are highlighted by our work. The big question is whether there is an NC algorithm for linear matroid intersection. Going to the more general setting, is there some kind of parallel algorithm for matroid intersection? Another question which can generate some new ideas is whether there is an NC reduction from search to decision for linear matroid intersection. For general matroid intersection, it would be interesting to find a parallel search to decision reduction even with the use of randomization.

The third question mentioned in the beginning, that is, linear matroid matching is completely open, in the sense that not even a quasi-NC algorithm is known for it. Given the wide applicability of the Isolation Lemma, the randomized parallel search-to-weighted-decision reduction of Mulmuley, Vazirani, and Vazirani [27] would work for any combinatorial problem with an appropriate self-reducibility property, including NP-hard problems like maximum independent set. An intriguing meta-question is – what is the most general setting where we can find deterministic or pseudo-deterministic parallel search-to-weighted-decision reductions.

2 Previous works

We start by briefly describing the techniques of previous works [28, 22, 19] that will be helpful in both comprehending as well as describing our work. Wherever these works talk about a minimization problem, we will describe it in terms of maximization, just for convenience. We will be using the following notations for the two versions of the matroid intersection problem.

- search-MI: Given two matroids on a common ground set, compute a common base.
- weighted-decision-MI: Given two matroids on the same ground set, polynomially bounded weights on the ground set elements, and a target weight W, is there a common base of weight at least W?

Whenever we are in a setting where the matroids are not given explicitly, we will consider the ground set size as the input size.

The result of Narayanan, Saran, and Vazirani [28] can be interpreted as an RNC reduction from search-MI to weighted-decision-MI. The first step of this reduction is to assign weights to the ground set elements, randomly and independently from a small range. Then from the Isolation Lemma [27], one can say that there is a unique maximum weight common base of the two matroids, with high probability. Here, the *weight of a common base* is defined to be sum of the weights of the elements in the common base. We can first find the maximum weight w^* of a common base by calling the weighted-decision-MI oracle for each possible target value W in a small range. Then for each ground set element e in parallel, increase its weight by one and find out the new maximum weight. The maximum weight increases if and only if e is a part of the unique maximum weight common base. This way we can find the unique maximum weight common base.

Note that the uniqueness property is crucial for this construction and that is the only place where randomness is needed. And this construction is not pseudo-deterministic because for different choices of random weights, we will get a different maximum weight common base. There has been several efforts to *deterministically* construct a weight assignment in NC that *isolates* a common base, i.e., ensures unique maximum weight common base, but this goal has not been achieved till now. A recent work [22] came quite close to this goal and constructed an isolating weight assignment in quasi-NC. This work generalizes the ideas used to do the same for bipartite matching in [16]. We build on their ideas to construct an isolating weight assignment in pseudo-deterministic NC. We first give a brief description of their result.

Isolating a common base in quasi-NC

Suppose that $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ are two matroids over the same ground set E where $\mathcal{B}_1$ and $\mathcal{B}_2$ are the family of bases of M_1 and M_2, respectively. Let $m = |E|$ and r_1 and r_2 be the rank functions of the matroids. The main idea of [22] is to isolate a common base in $\log m$ rounds, where in each round they significantly reduce the set of maximum weight common bases, and finally bring it down to just one maximum weight common base. In each of these rounds, they deterministically propose a set of poly(m) weight assignments, one of which will do the desired reduction in the set of maximum weight common bases. In a round, they have no way of figuring out which one out of these poly(m) weight assignments will do the job. So, they have to try all poly$(m)^{\log m}$ combinations of these weight assignments. Moreover, for any particular combination, they have to combine the $\log m$ weight assignments on different scales, which means their weights become as large as poly$(m)^{\log m}$. Due to these two factors, their construction is in quasi-NC and not in NC.

To measure the progress in each round, they need a succinct way to describe the current set of maximum weight common bases. The most convenient way to understand the set of maximum weight common bases is through the *common base polytope*. The common base polytope $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ is a polytope formed by taking convex hull of the 0-1 indicator vectors of the sets in $\mathcal{B}_1 \cap \mathcal{B}_2$. For any weight assignment $\mathbf{w} \in \mathbb{R}^E$, the weight of a common base B is defined as a linear function, and thus, one can obtain the maximum weight common bases by maximizing the function $\sum_{e\in E} \mathbf{w}_e \mathbf{x}_e$ over $P(\mathcal{B}_1 \cap \mathcal{B}_2)$. In particular, the set of maximum weight common bases will always be the set of corners of a face of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$.

Edmonds [13] gave a nice description of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ using the rank functions r_1 and r_2. He showed that a point $\mathbf{x} \in \mathbb{R}^E$ is in $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ if and only if it satisfies the following constraints:

$$\mathbf{x}_e \geq 0 \quad \forall e \in E, \tag{1}$$

$$\mathbf{x}(S) = \sum_{e\in S} \mathbf{x}_e \leq r_i(S) \quad \forall S \subset E,\ i = 1, 2, \tag{2}$$

$$\mathbf{x}(E) = \sum_{e\in E} \mathbf{x}_e = r_1(E) = r_2(E). \tag{3}$$

The construction in [22] crucially uses the description of the common base polytope $P(\mathcal{B}_1 \cap \mathcal{B}_2)$. In terms of the polytope, their construction of the weight assignment is such that in each round, the maximum weight face of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ gets significantly smaller and after $\log m$ rounds, the maximum weight face is simply a corner point. The key notions they introduced to measure the improvement in each iteration are *cycles* with respect to a face and their *circulations* with respect to a weight assignment.

Suppose that F is a face of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$. A subset S of E is called a *tight set* of M_i with respect to F if the corresponding inequality in (2) is tight for F i.e, for all $\mathbf{x} \in F$, $\mathbf{x}(S) = r_i(S)$. Then [22] showed that for every face F, we have two partitions of E, denoted by $\mathsf{partition}_1[F]$ and $\mathsf{partition}_2[F]$, such that every tight set of M_i with respect to F is a union of the sets from $\mathsf{partition}_i[F]$. The partitions of E naturally induce a bipartite graph, denoted by $\mathcal{G}[F]$, with the left vertex set $\mathsf{partition}_1[F]$, the right vertex set $\mathsf{partition}_2[F]$ and the edge set E: the edge corresponding to an element $e \in E$ is incident on the vertex corresponding to a set $v \in \mathsf{partition}_i[F]$ if and only if $e \in v$. A sequence of distinct elements $(e_1, \ldots, e_k)$ from E is called a *cycle* with respect to F if it forms a cycle in the graph $\mathcal{G}[F]$.

Let $\mathcal{C}_F$ denotes the set of cycles with respect to a face F of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$. Then [22] showed that for face F, if $\mathcal{C}_F = \emptyset$ then F is a corner point of the polytope $P(\mathcal{B}_1 \cap \mathcal{B}_2)$. Their idea was to keep eliminating cycles via appropriate modification of the weight assignment and get smaller and smaller maximizing face of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ to eventually reach a corner point. For a weight assignment $\mathbf{w}$ on E, define the *circulation* for a (even length) cycle as the absolute value of the difference of weights in the two sets of alternating edges. Let C be a cycle, say with respect to $F = P(\mathcal{B}_1 \cap \mathcal{B}_2)$, and let $\mathbf{w}$ be a weight assignment such that the circulation of C is non-zero w.r.t. $\mathbf{w}$. Then they showed that the cycle C does not appear in the maximizing face with respect to $\mathbf{w}$. Now if the weight assignment $\mathbf{w}$ gives non-zero circulation to *all* the cycles in $P(\mathcal{B}_1 \cap \mathcal{B}_2)$, then all the cycles in the maximizing face F will be eliminated, i.e. $\mathcal{C}_F = \emptyset$, and F will be a corner. However, with polynomially bounded weights, one cannot expect to give nonzero circulation to all the cycles at once, since the number of cycles can be exponentially large.

One of the key ideas in [22, 16] was to eliminate the cycles in rounds. In each round, they double the length of the eliminated cycles and reach to face of a smaller dimension. Thus, in $\log m$ rounds, one can eliminate all the cycles and reach a corner point of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$. They used the fact that if in a graph all the cycles have length greater than 2^i, then there are at most m^4 many cycles of length at most 2^{i+1} [33]. This implies that, at each iteration, we have to give nonzero circulation to at most m^4 many cycles. Using a hashing technique (for example see [16, Lemma 2.3]), one can give nonzero circulation for each of these m^4 many cycles. Formally, Gurjar and Thierauf [21, Lemma 3.11 and 3.12] showed the following property for faces F (of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$) having no cycle of length $\leq r$.

▶ **Lemma 4.** *Let F be a face of the polytope $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ such that $\mathcal{C}_F$ has no cycle of length r. Then one can construct a set of $O(m^6)$ many weight assignments with weights bounded by $O(m^6)$ in NC such that one of the weight assignment will give nonzero circulation to all the cycles in $\mathcal{C}_F$ of length at most $2r$.*

Now, as described earlier, we consider all possible combinations of weight assignments from different rounds to get a family of $\text{poly}(m^{\log m})$ many weight assignments with weights bounded by $\text{poly}(m^{\log m})$ such that for any two matroids on a ground set of size m, at least one weight assignment isolates a common base.

In this paper, we give a *pseudo-deterministic NC* reduction from search-MI to weighted-decision-MI. This line of work was started by Goldwasser and Grossman [19]. One can extract an abstract framework from [19] with the following two steps to get a pseudo-deterministic NC search-to-weighted-decision reduction: 1) Like [16, 22], an iterative approach of designing an isolating weight assignment family, 2) Succinct representation of the maximum weight faces of the underlying polytope with an RNC algorithm to compute it, assuming the oracle access to the weighted decision. For example, a face of the bipartite matching polytope is completely described by the set edges that participate in some perfect matching in that face, and [19] gives an NC algorithm to compute it using the respective weighted decision oracle.

The faces of the perfect matching polytope for general graphs are more complicated than their bipartite counterpart. Here, any face is described by a maximal laminar family of tight odd cuts. The work of [8, 30] give an NC procedure, with the oracle access to the weight decision problem, to compute a maximal laminar family of tight odd cuts. This result supplies the second ingredient of the [19] framework, which helped [2] give an NC reduction from search to weighted decision for general perfect matching.

Our reduction also follows the abstract framework of [19]. We use the iterative approach developed by [22]. On top of that, we need an RNC algorithm (using the oracle access to weighted-decision-MI) to compute a succinct representation for a maximum weight face of the common base polytope $P(\mathcal{B}_1 \cap \mathcal{B}_2)$. However, none of the previous ideas help to answer this question, and we need something completely new.

3 Proof techniques

In this section, we briefly describe the proof ideas of our results. Our proofs strongly rely on some structural properties of lattice families over finite sets. Therefore, we briefly discuss the necessary notations and facts about lattice families. For a finite set E, a family of subsets $\mathcal{L}$ of E is called a *lattice family* over E if it is closed under set union and intersection and for every element $a \in E$ there exists a set in $\mathcal{L}$ containing a. For every element $a \in E$ there exists a *unique* smallest set in $\mathcal{L}$ containing a. Such sets are called as *prime sets* of $\mathcal{L}$. All the sets in a lattice family can be written as a union of its prime sets. Every lattice family $\mathcal{L}$ over E induces a *unique partition* $\mathcal{P}$ of E such that every set in $\mathcal{L}$ is a disjoint union of sets in $\mathcal{P}$. Moreover, the sets in $\mathcal{P}$ can be written as a sequence $(S_1, \dots, S_\ell)$ with the following property: for all $k \in [\ell]$, $\cup_{j=1}^{k} S_j$ is in $\mathcal{L}$. A family $\mathcal{L}' \subseteq \mathcal{L}$ is called a *sublattice* of $\mathcal{L}$, if $\mathcal{L}'$ is also a lattice family over E. The partition $\mathcal{P}$ is a refinement of the partition $\mathcal{P}'$ induced by $\mathcal{L}'$, that is for all $S \in \mathcal{P}'$, the sets in $\mathcal{P}$ having a nonempty intersection with S form a partition of S. For proof of these properties, one can see Section 4.2 of the full version.

3.1 Proof Idea of Theorem 1

We discuss a succinct representation for the maximum weight face of the common base polytope and an RNC algorithm to compute it. First, we define some notations. Supppose that $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ are two matroids with $\mathcal{B}_1$ and $\mathcal{B}_2$ as their family of the bases and r_1 and r_2 as the rank functions, respectively. Let $m = |E|$. Let $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ be the common base polytope of M_1 and M_2 defined by the equations (1), (2), (3), and F be a face of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$. Then a subset S of E is called a *tight set* for M_i (with respect to F) if for all $\mathbf{x} \in F$, $\mathbf{x}(S) = r_i(S)$. For all $i \in [2]$, let $\textsf{tight-sets}_i[F]$ denote the family of all tight sets for M_i with respect to the face F. Edmonds [13] showed that for all $i \in [2]$, $\textsf{tight-sets}_i[F]$ forms a lattice family over E.

Suppose that $\mathbf{w}$ is a weight assignment on E. Let $F_{\mathbf{w}}$ be the maximizing face of the common base polytope $P(\mathcal{B}_1 \cap \mathcal{B}_2)$, with respect to $\mathbf{w}$. The face $F_{\mathbf{w}}$ can be uniquely represented by $\textsf{tight-sets}_1[F_{\mathbf{w}}]$ and $\textsf{tight-sets}_2[F_{\mathbf{w}}]$. However, we can not compute them explicitly with our limited computational resources since the size of each family can be exponentially large. On the other hand, since $\textsf{tight-sets}_i[F_{\mathbf{w}}]$ is a lattice family over E, each $\textsf{tight-sets}_i[F_{\mathbf{w}}]$ has a succinct representation using partial order defined on its prime sets. More specifically, one can define a pre-order $\preceq_i$ (that is, reflexive and transitive) on E as follows: for all $a, b \in E$, $a \preceq_i b$ if and only if in $\textsf{tight-sets}_i[F_{\mathbf{w}}]$, the prime set containing b is a subset of the prime set containing a. The pre-order $\preceq_i$ gives a succinct representation of $\textsf{tight-sets}_i[F_{\mathbf{w}}]$, that is for every $S \subseteq E$, S is in $\textsf{tight-sets}_i[F_{\mathbf{w}}]$ if and only if S is transitively closed under $\preceq_i$. Such succinct representation for lattice familes is well known (see [31, Chapter 49] [1]). For any $a \in E$, the transitive closure of a in $\preceq_i$ is same as the prime set in $\textsf{tight-sets}_i[F_{\mathbf{w}}]$ containing a. Also, the collection of all maximal subsets of E which are symmetric under $\preceq_i$ is same as the partition E induced by $\textsf{tight-sets}_i[F_{\mathbf{w}}]$. If one consider the digraph representaion of $\prec_i$, (that is (a, b) is an edge if and only if $a \preceq_i b$) then in $\textsf{tight-sets}_i[F_{\mathbf{w}}]$, the prime set containing a is same the set of vertices reachable from a in the digraph and the partition of E induced by $\textsf{tight-sets}_i[F_{\mathbf{w}}]$ is same as the set of strongly connected components. Thus, the prime sets of $\textsf{tight-sets}_i[F_{\mathbf{w}}]$ contain all the information

[1] Our definition of $\preceq_i$ is exactly opposite to the definition used [31, Chapter 49], that is according to their definition, $a \preceq_i b$ if and only if the prime set containing a is a subset of the prime set containing b.

regarding it. In our context, we compute the following succinct objects related to $F_{\mathbf{w}}$: $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}}]$ and $\mathsf{partition}_i[F_{\mathbf{w}}]$ for all $i \in [2]$, where $\mathsf{prime\text{-}sets}_i[F]$ be the set of all primes sets of the lattice family $\mathsf{tight\text{-}sets}_i[F_{\mathbf{w}}]$ and $\mathsf{partition}_i[F_{\mathbf{w}}]$ denote the partition of E induced by $\mathsf{tight\text{-}sets}_i[F]$. Recall from [22, Section 3.3] that the cycles of the bipartite graph induced by $\mathsf{partition}_1[F_{\mathbf{w}}]$ and $\mathsf{partition}_2[F_{\mathbf{w}}]$ define the cycles with respect to the face $F_{\mathbf{w}}$. And, the tight constraints coming from sets in $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}}]$ serve as a basis for all the tight constraints from $\mathsf{tight\text{-}sets}_i[F_{\mathbf{w}}]$. Here, we would like to mention that basis forming families of tight sets are well studied (see [31]). However, to best of our knowledge, no efficient parallel algorithm is known to compute them. Also, the succinct representation of lattices using the partial order of its prime sets has been widely used to design algorithms for different optimization problems. For example, computing optimal stable matching [23], problems in computational geometry [14, 6], submodular function minimization [31, Chapter 49]. However, the context of these applications are very different from parallel computation.

With the above two objects, we also need the following characterization: for all $i \in [2]$, there exists a function $N_i^{F_{\mathbf{w}}}$ from $\mathsf{partition}_i[F_{\mathbf{w}}]$ to $\mathbb{Z}_{\geq 0}$ such that a base $B \in \mathcal{B}_1 \cap \mathcal{B}_2$ is in the face $F_{\mathbf{w}}$ if and only if for all $i \in [2]$ and $S \in \mathsf{partition}_i[F_{\mathbf{w}}]$, we have $|S \cap B| = N_i^{F_{\mathbf{w}}}(S)$. Here, we would like to mention that both the notions of partition and the function $N_i^{F_{\mathbf{w}}}$ and the criteria we just mentioned were already introduced in [22], but were a bit weaker in the following ways: Our criteria is an exact characterization, however, they showed it for one direction. Our partition has an additional "chain property" ensured by the structural properties of the lattice families. All these additional points will be useful in our proofs. For details see Section 4.5 of the full version and [22, Section 3.2].

Now we briefly discuss about our RNC algorithm to compute $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}}]$ and $\mathsf{partition}_i[F_{\mathbf{w}}]$ for all $i \in [2]$. One important point is that our algorithm is equipped with the oracle access to weighted-decision-MI. Our idea is the following: We first compute a random vertex, equivalently a random base, B in the face $F_{\mathbf{w}}$. The base B can be computed in RNC using the oracle access to weighted-decision-MI. Then iteratively construct a chain of subsets of bases from $F_{\mathbf{w}}$

$$\{B\} = \mathcal{B}_0 \subseteq \mathcal{B}_1 \subseteq \cdots \subseteq \mathcal{B}_\ell$$

such that the minimal face containing $\mathcal{B}_\ell$ is same as $F_{\mathbf{w}}$ and $\ell = \lceil \log m \rceil$. Next we briefly discuss how to construct the set $\mathcal{B}_j$ from $\mathcal{B}_{j-1}$ and compute $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}}]$ and $\mathsf{partition}_i[F_{\mathbf{w}}]$ from the set of common bases $\mathcal{B}_\ell$.

For all $j \in \{0, \ldots, \ell\}$, let F_j denotes the minimal face containing $\mathcal{B}_j$. For all $j \in [\ell]$, the set $\mathcal{B}_j$ contains the elements in $\mathcal{B}_{j-1}$ with the following extra elements: For all $i \in [2]$, $A \in \mathsf{partition}_i[F_{j-1}]$, we add a common base $B_{ij}^{(A)}$ (if it exists) from the face $F_{\mathbf{w}}$ with the property

$$|A \cap B_{ij}^{(A)}| \neq N_i^{F_{j-1}}(A). \tag{4}$$

We know that for all $i \in [2]$ $A \in \mathsf{partition}_i[F_{j-1}]$, every base in F_{j-1} contains exactly $N_i^{F_{j-1}}(A)$ many elements from A. However, our property on $B_{ij}^{(A)}$ says that we want a base from $F_{\mathbf{w}}$ which violates that condition, and if exists, we can compute such a base in RNC using the oracle access to weighted-decision-MI. Next, we discuss how to compute $\mathsf{partition}_i[F_j]$ in NC. Note that, after computing $\mathsf{partition}_i[F_j]$, $N_i^{F_j}$ can be computed in NC by computing $|B \cap A|$, for some $B \in \mathcal{B}_j$, in parallel for all $A \in \mathsf{partition}_i[F_j]$.

The set families $\mathsf{tight\text{-}sets}_i[F_j]$ for all $i \in [2]$ form lattice families over E, and given B_j, we are interested to compute $\mathsf{prime\text{-}sets}_i[F_j]$ and $\mathsf{partition}_i[F_j]$ in NC. As we mentioned earlier, every lattice family has a digraph representation based on the partial order on primes sets of lattice family. Given this digraph representation of $\mathsf{tight\text{-}sets}_i[F_j]$, one can compute $\mathsf{prime\text{-}sets}_i[F_j]$ and $\mathsf{partition}_i[F_j]$ in NC. However, given $\mathcal{B}_j$, it is not clear how to construct the digraph representation of the lattice family $\mathsf{tight\text{-}sets}_i[F_j]$ in NC. We show that, instead of this digraph, it would sufficient for us if we work with a subgraph $G_i[\mathcal{B}_j]$ defined as follows: the vertex set is same as the ground set E and for all $a, b \in E$, (a, b) is an edge of $G_i[\mathcal{B}_j]$ if and only if

there exists a base $B \in \mathcal{B}_j$ such that $b \in B$ and $(B \setminus \{b\}) \cup \{a\}$ is also a base of M_i.

More specifically, we prove that for every $a \in E$ the prime set in $\mathsf{tight\text{-}sets}_i[F_j]$ containing a is same as the set of vertices reachable from a in $G_i[F_j]$ and $\mathsf{partition}_i[F_j]$ is same as the set of strongly connected components in $G_i[\mathcal{B}_j]$. Using this characterization, we can compute $\mathsf{prime\text{-}sets}_i[F_j]$ and $\mathsf{partition}_i[F_j]$ in NC, given the graph $G_i[\mathcal{B}_j]$. For more details see Section 6 of the full version. Also, using the weighted decision oracle we can compute $G_i[\mathcal{B}_j]$ in NC. Thus, given $\mathcal{B}_j$, $\mathsf{prime\text{-}sets}_i[F_j]$ and $\mathsf{partition}_i[F_j]$ are computable in NC. Here, we would like to mention that constructing directed graphs using base exchange property is a well known technique in matroid literature and has been used in various contexts. For example, one can see the augmenting path based algorithm for matroid intersection in [31, Section 41.2], and some other context in [31, Section 40.3]. The definition of $G_i[\mathcal{B}_j]$ is very close to the definition used in the second example.

At very high level, this part of our algorithm is doing exactly the opposite of the idea used to construct isolating weight assignment family in [16, 22, 34]. They start from a face of the polytope and iteratively move to the subfaces of smaller dimensions until a corner point is reached. On the other hand, we are starting from a corner point of the face and iteratively reaching bigger faces until we cover the whole face.

Now we give a very brief overview of the correctness of our algorithm. For all $j \in \{0, 1, \ldots, \ell\}$, since F_j is a subface of $F_{\mathbf{w}}$, $\mathsf{tight\text{-}sets}_i[F_{\mathbf{w}}]$ is a sublattice of $\mathsf{tight\text{-}sets}_i[F_j]$ for all $i \in [2]$. Therefore $\mathsf{partition}_i[F_j]$ is a refinement of $\mathsf{partition}_i[F_{\mathbf{w}}]$, that is for all $S \in \mathsf{partition}_i[F_{\mathbf{w}}]$, the sets in $\mathsf{partition}_i[F_j]$ having nonempty intersection with S create a partition of S. Let $\mathcal{W}_{ij}^{(S)}$ denote the family of sets in $\mathsf{partition}_i[F_j]$ which have nonempty intersection with $S \in \mathsf{partition}_i[F_{\mathbf{w}}]$. As we move from $(j-1)$th iteration to jth iteration, our algorithm satisfies the following property: either the size of the smallest sets in $\mathcal{W}_{ij}^{(S)}$ satisfying the equation 4 becomes double, or if no such set exists in $\mathcal{W}_{ij}^{(S)}$, it becomes equal to $\{S\}$. Thus, after ℓth iteration, $\mathsf{partition}_i[F_\ell]$ becomes equal to $\mathsf{partition}_i[F_{\mathbf{w}}]$ for all $i \in [2]$. This leads us to prove that $F_\ell = F_{\mathbf{w}}$. Therefore, $\mathsf{prime\text{-}sets}_i[F_\ell]$ is also same as $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}}]$. In Algorithm 1, we describe all the steps to compute $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}}]$ and $\mathsf{partition}_i[F_{\mathbf{w}}]$ for all $i \in [2]$. For the detail analysis of the correctness and time complexity of our algorithm see Section 7 of the full version. From the above discussion, we can conclude that

▶ **Theorem 5.** *Let $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ be two matroids with $\mathcal{B}_1$ and $\mathcal{B}_2$ be the family of bases, respectively. Let $\mathbf{w}$ be a weight assignment on E with polynomially bounded weights, and $F_{\mathbf{w}}$ be the maximizing face of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ with respect to $\mathbf{w}$. Then, given M_1, M_2 and $\mathbf{w}$ as inputs, Algorithm 1 computes $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}}]$ and $\mathsf{partition}_i[F_{\mathbf{w}}]$ for all $i \in [2]$ in randomized NC, provided that the algorithm has an oracle access to* weighted-decision-MI. *Furthermore, for all positive integer c, the success probability of the algorithm can be made at least $1 - \frac{1}{m^c}$, where $m = |E|$.*

Algorithm 1 Computing prime sets and partitions corresponding to a max-weight face.

Input: Two matroids $M_1 = (E, \mathcal{I}_1)$, $M_2(E, \mathcal{I}_2)$, and a weight assignment $\mathbf{w} : E \to \mathbb{Z}_{\geq 0}$.
Output: prime-sets$_i[F_\mathbf{w}]$ and partition$_i[F_\mathbf{w}]$ for all $i \in [2]$, where $F_\mathbf{w}$ denotes the max-weight face.
Assumption: Oracle access to weighted-decision-MI.

```
1: Compute a base $B$ in $F_\mathbf{w}$.
2: $\mathcal{B}_0 \leftarrow \{B\}$.
3: for all $i \in [2]$ do in parallel
4:     Compute the graph $G_i[\mathcal{B}_0]$.
5:     Let $F_0$ be the minimal face containing $\mathcal{B}_0$.
6:     Compute prime-sets$_i[F_0]$, partition$_i[F_0]$ and $N_i^{F_0}$.
7: end for
8: for $j \leftarrow 1$ to $\lceil \log m \rceil$ do
9:     $\mathcal{B}_j \leftarrow \mathcal{B}_{j-1}$.
10:    for all $i \in [2]$ do in parallel
11:        for all $A \in$ partition$_i[F_{j-1}]$ do in parallel
12:            If exists, compute a base $B_{ij}^{(A)}$ in $F_\mathbf{w}$ such that
                   $|A \cap B_{ij}^{(A)}| \neq N_i^{F_{j-1}}(A)$.
13:            $\mathcal{B}_j \leftarrow \mathcal{B}_j \bigcup \left\{ B_{ij}^{(A)} \right\}$.
14:        end for
15:    end for
16:    for all $i \in [2]$ do in parallel
17:        Let $F_j$ be the minimal face containing $\mathcal{B}_j$.
18:        Compute the graph $G_i[\mathcal{B}_j]$.
19:        Compute prime-sets$_i[F_j]$, partition$_i[F_j]$ and $N_i^{F_j}$.
20:    end for
21: end for
22: return prime-sets$_i[F_\ell]$ and partition$_i[F_\ell]$ for $i \in [2]$ and $\ell = \lceil \log m \rceil$.
```

3.2 Proof idea of Theorem 3

In this section, we give a proof overview of Theorem 3, which states that there is a pseudo-deterministic NC algorithm for the matroid intersection search problem that uses the weighted-decision oracle. Since the weighted-decision for *linear* matroid intersection can be solved in RNC [28], we get a pseudo-deterministic NC algorithm for the search version of linear matroid intersection, that is, Theorem 2.

Suppose that $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ are two matroids with $\mathcal{B}_1$ and $\mathcal{B}_2$ as the family of bases, respectively. Let $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ be the common base polytope of M_1 and M_2. Let $\mathbf{w}_0$ be a weight assignment defined as $\mathbf{w}_0(a) = 1$ for all $a \in E$. Then the maximizing face of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ with respect to $\mathbf{w}_0$ is the polytope itself. Let $m = |E|$ and $\ell = \lceil \log m \rceil$. Now our idea is the following: We start from the weight assignment $\mathbf{w}_0$ and inductively construct a sequence of weight assignments

$$\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_\ell$$

such that for all $j \in \{0, 1, \ldots, \ell\}$, the weights in $\mathbf{w}_j$ are bounded by $O(m)$ and the length of the shortest cycle with respect to the face F_j is greater than 2^j where F_j denotes the maximizing face with respect to $\mathbf{w}_j$. Therefore, the face F_ℓ does not have any cycle, and from [22], it has a unique base. Now using the oracle access to weighted-decision-MI the base in F_ℓ can be computed in NC. Next we discuss how to construct $\mathbf{w}_j$ from $\mathbf{w}_{j-1}$.

For all $j \in \{0, 1, \dots, \ell\}$, let $\mathcal{C}_{F_j}$ denotes the set of all cycles with respect to the face F_j. From the induction hypothesis, for some j, all the cycles in $\mathcal{C}_{F_j}$ have length greater than 2^j. Then from [22], there are at most m^4 many cycles of length at most 2^{j+1}. Let $\mathcal{W}$ be a polynomially large family of weight assignments with polynomially bounded weights such that one of the weight assignments in $\mathcal{W}$ gives nonzero circulation to all the cycles in $\mathcal{C}_{F_j}$ of length at most 2^{j+1}. There are well known NC constructions of such a family $\mathcal{W}$ (see e.g., [16, Lemma 2.3]). For each $\mathbf{w} \in \mathcal{W}$ we do the following in parallel: combine $\mathbf{w}_j$ and $\mathbf{w}$ in decreasing order of precedence. Let $\mathbf{w}'$ be the combined weight and $F_{\mathbf{w}'}$ is the maximizing face with respect to it. Now using our RNC algorithm discussed in the previous section, compute $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}'}]$ and $\mathsf{partition}_i[F_{\mathbf{w}'}]$ for all $i \in [2]$. Now, construct the bipartite graph $\mathcal{G}[F_{\mathbf{w}'}]$ from $\mathsf{partition}_1[F_{\mathbf{w}'}]$ and $\mathsf{partition}_2[F_{\mathbf{w}'}]$ as defined in the description of [22]. The length of the shortest cycles in $\mathcal{G}[F_{\mathbf{w}'}]$ can be computed in NC. Thus, in NC, we can compute the lexicographically smallest weight assignment $\mathbf{w} \in \mathcal{W}$ such that the length of the shortest cycles in $\mathcal{G}[F_{\mathbf{w}'}]$ is greater than 2^{j+1}.

Algorithm 2 Pseudo-deterministic NC algorithm for computing a common base of two matroids.

Input: Two matroids $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$.
Output: A common base of M_1 and M_2, if exists.
Assumption: Oracle access to weighted-decision-MI.

```
 1: $\mathbf{w}_0 \leftarrow \mathbf{1}$.
 2: for $j \leftarrow 1$ to $\lceil \log m \rceil$ do
 3:     Compute a family of weight assignments $\mathcal{W}$ as promised by Lemma 4.
 4:     for all $\mathbf{w} \in \mathcal{W}$ do in parallel
 5:         Combine $\mathbf{w}_{j-1}$ and $\mathbf{w}$ with descending order in precedence.
 6:         For a $\mathbf{w} \in \mathcal{W}$, let $\mathbf{w}'$ be the combined weight.
 7:         Let $F_{\mathbf{w}'}$ be the maximizing face of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$ with respect to $\mathbf{w}'$.
 8:         For all $i \in [2]$, compute $\mathsf{prime\text{-}sets}_i[F_{\mathbf{w}'}]$ and $\mathsf{partition}_i[F_{\mathbf{w}'}]$ using Algorithm 1.
 9:         Let $\mathcal{G}[F_{\mathbf{w}'}]$ be the bipartite graph induced by $\mathsf{partition}_1[F_{\mathbf{w}'}]$ and $\mathsf{partition}_1[F_{\mathbf{w}'}]$.
10:         Compute the length of shortest cycle of $\mathcal{G}[F_{\mathbf{w}'}]$.
11:     end for
12:     Take some fixed ordering on $\mathcal{W}$, like lexicographic ordering.
13:     Take the smallest $\mathbf{w}$ such that the length of the shortest cycle in $\mathcal{G}[F_{\mathbf{w}'}] > 2^j$.
14:     $\mathbf{w}_j \leftarrow \sum_{i=1}^{2} \sum_{S \in \mathsf{prime\text{-}sets}_i[F_{\mathbf{w}'}]} \mathbf{1}_S$.
15: end for
16: Compute the unique common base maximizing $\mathbf{w}_{\lceil \log m \rceil}$ and output.
```

Next we show how to compute $\mathbf{w}_{j+1}$ from $\mathbf{w}'$ such that weights in $\mathbf{w}_{j+1}$ are bounded by $O(m)$. Define $\mathbf{w}_{j+1}$ as the following:

$$\mathbf{w}_{j+1} = \sum_{i=1}^{2} \sum_{S \in \mathsf{prime\text{-}sets}_i[F_{\mathbf{w}'}]} \mathbf{1}_S,$$

where $\mathbf{1}_S \in \mathbb{R}^E$ denotes the indicator vector for the set S. From the defnition, it is clear that weights are bounded by $2m$, and can be computed in NC from $\mathsf{prime\text{-}sets}_1[F_{\mathbf{w}'}]$ and $\mathsf{prime\text{-}sets}_2[F_{\mathbf{w}'}]$. Using the description of $P(\mathcal{B}_1 \cap \mathcal{B}_2)$, we can show that every point $\mathbf{x}$ in the maximizing face F_{j+1} must satisfy $\mathbf{x}(S) = r_i(S)$ for all $i \in [2]$, $S \in \textsf{prime-sets}_i[F_{\mathbf{w}'}]$. This implies that $\textsf{prime-sets}_i[F_{\mathbf{w}'}]$ is a subset of $\textsf{tight-sets}_i[F_{j+1}]$. Thus $\textsf{tight-sets}_i[F_{\mathbf{w}'}]$ is a subset of $\textsf{tight-sets}_i[F_{j+1}]$ since all the sets in a lattice family can be written as a union of its prime sets. This helps us to show that $F_{\mathbf{w}'}$ is same as F_{j+1}. Also, one can verify that each step of our algorithm as has a unique answer, therefore it is pseudo-deterministic. In Algorithm 2, we describe the steps of our pseudo-deterministic NC reduction from search-MI to weighted-decision-MI. For the proof of correctness and time complexity analysis of our algorithm, see Section 8 of the full version.

References

1 Manindra Agrawal, Thanh Minh Hoang, and Thomas Thierauf. The polynomially bounded perfect matching problem is in NC^2. In *24th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 4393 of *Lecture Notes in Computer Science*, pages 489–499. Springer Berlin Heidelberg, 2007. doi:10.1007/978-3-540-70918-3_42.

2 Nima Anari and Vijay V. Vazirani. Matching is as easy as the decision problem, in the NC model. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 54:1–54:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ITCS.2020.54.

3 Stuart J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Information Processing Letters*, 18(3):147–150, 1984.

4 Allan Borodin, Stephen Cook, and Nicholas Pippenger. Parallel computation for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58(1-3):113–136, July 1984.

5 Allan Borodin, Joachim von zur Gathen, and John Hopcroft. Fast parallel matrix and GCD computations. *Information and Control*, 52(3):241–256, 1982.

6 Kevin Buchin, David Eppstein, Maarten Löffler, Martin Nöllenburg, and Rodrigo I. Silveira. Adjacency-preserving spatial treemaps. *J. Comput. Geom.*, 7(1):100–122, 2016. doi:10.20382/jocg.v7i1a6.

7 Laszlo Csanky. Fast parallel matrix inversion algorithms. *SIAM Journal on Computing*, 5(4):618–623, 1976. doi:10.1137/0205040.

8 Marek Cygan, Harold N. Gabow, and Piotr Sankowski. Algorithmic applications of baur-strassen's theorem: Shortest cycles, diameter and matchings. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 531–540. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.72.

9 Marek Cygan, Harold N. Gabow, and Piotr Sankowski. Algorithmic applications of baur-strassen's theorem: Shortest cycles, diameter, and matchings. *J. ACM*, 62(4), 2015. doi:10.1145/2736283.

10 Elias Dahlhaus and Marek Karpinski. Matching and multidimensional matching in chordal and strongly chordal graphs. *Discrete Applied Mathematics*, 84(1–3):79–91, 1998.

11 Samir Datta, Raghav Kulkarni, and Sambuddha Roy. Deterministically isolating a perfect matching in bipartite planar graphs. *Theory of Computing Systems*, 47:737–757, 2010. doi:10.1007/s00224-009-9204-8.

12 Richard A. Demillo and Richard J. Lipton. A probabilistic remark on algebraic program testing. *Information Processing Letters*, 7(4):193–195, 1978.

13 Jack Edmonds. Submodular functions, matroids, and certain polyhedra. In *Combinatorial Structures and Their Applications, Gordon and Breach, New York*, pages 69–87, 1970.

14 David Eppstein, Elena Mumford, Bettina Speckmann, and Kevin Verbeek. Area-universal and constrained rectangular layouts. *SIAM J. Comput.*, 41(3):537–564, 2012. doi:10.1137/110834032.

15 Stephen Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-NC. *SIAM Journal on Computing*, 0(0):STOC16–218–STOC16–235, 2019. `doi:10.1137/16M1097870`.

16 Stephen A. Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-nc. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA*, pages 754–763, 2016.

17 Eran Gat and Shafi Goldwasser. Probabilistic search algorithms with unique answers and their cryptographic applications. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:136, 2011. URL: `http://eccc.hpi-web.de/report/2011/136`.

18 Oded Goldreich, Shafi Goldwasser, and Dana Ron. On the possibilities and limitations of pseudodeterministic algorithms. In Robert D. Kleinberg, editor, *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pages 127–138. ACM, 2013. `doi:10.1145/2422436.2422453`.

19 Shafi Goldwasser and Ofer Grossman. Bipartite perfect matching in pseudo-deterministic NC. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 87:1–87:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. `doi:10.4230/LIPIcs.ICALP.2017.87`.

20 Dima Grigoriev and Marek Karpinski. The matching problem for bipartite graphs with polynomially bounded permanents is in NC (extended abstract). In *28th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 166–172, 1987. `doi:10.1109/SFCS.1987.56`.

21 Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Identity testing for constant-width, and any-order, read-once oblivious arithmetic branching programs. *Theory of Computing*, 13(2):1–21, 2017.

22 Rohit Gurjar and Thomas Thierauf. Linear matroid intersection is in quasi-nc. In *49th Annual ACM Symposium on Theory of Computing*, pages 821–830, 2017.

23 Robert W. Irving, Paul Leather, and Dan Gusfield. An efficient algorithm for the "optimal" stable marriage. *J. ACM*, 34(3):532–543, 1987. `doi:10.1145/28869.28871`.

24 Richard M. Karp, Eli Upfal, and Avi Wigderson. Constructing a perfect matching is in random NC. *Combinatorica*, 6(1):35–48, 1986.

25 Richard M. Karp, Eli Upfal, and Avi Wigderson. The complexity of parallel search. *Journal of Computer and System Sciences*, 36(2):225–253, 1988. `doi:10.1016/0022-0000(88)90027-X`.

26 László Lovász. On determinants, matchings, and random algorithms. In *FCT*, volume 79, pages 565–574, 1979.

27 Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7:105–113, 1987. `doi:10.1007/BF02579206`.

28 H. Narayanan, Huzur Saran, and Vijay V. Vazirani. Randomized parallel algorithms for matroid union and intersection, with applications to arboresences and edge-disjoint spanning trees. *SIAM J. Comput.*, 23(2):387–397, 1994. `doi:10.1137/S0097539791195245`.

29 Øystein Ore. Über höhere Kongruenzen. *Norsk Mat. Forenings Skrifter Ser. I*, 7(15):27, 1922.

30 Piotr Sankowski. NC algorithms for weighted planar perfect matching and related problems. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 97:1–97:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. `doi:10.4230/LIPIcs.ICALP.2018.97`.

31 Alexander Schrijver. *Combinatorial optimization : polyhedra and efficiency. Vol. B. , Matroids, trees, stable sets. chapters 39-69.* Algorithms and combinatorics. Springer-Verlag, Berlin, Heidelberg, New York, N.Y., et al., 2003. URL: `http://opac.inria.fr/record=b1124843`.

32 J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.

33 Ashok Subramanian. A polynomial bound on the number of light cycles in an undirected graph. *Information Processing Letters*, 53(4):173–176, 1995. `doi:10.1016/0020-0190(94)00202-A`.

34 Ola Svensson and Jakub Tarnawski. The matching problem in general graphs is in quasi-nc. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 696–707, 2017. doi:10.1109/FOCS.2017.70.

35 Raghunath Tewari and N. V. Vinodchandran. Green's theorem and isolation in planar graphs. *Information and Computation*, 215:1–7, 2012. doi:10.1016/j.ic.2012.03.002.

36 Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, EUROSAM '79, pages 216–226, 1979.