The Matroid Intersection Cover Problem

Sungjin Im[a], Benjamin Moseley[b], Kirk Pruhs[c]

[a] Computer Science and Engineering University of California, Merced
[b] Tepper School of Business Carnegie Mellon University
[c] Computer Science Department University of Pittsburgh

Abstract

We consider the matroid intersection cover problem. This is a special case of set cover where the sets are derived from the intersection of matroids. We introduce a technique for computing matroid intersection covers. We give polynomial-time algorithms to compute partition decompositions for matroids that commonly arise in combinatorial optimization problems. We then give a polynomial-time algorithm for computing matroid intersection covers given the partition decompositions of the matroids. Combining these algorithms, we obtain an $O(1)$-approximation algorithm when each of the $O(1)$ matroids is of a standard type.

Keywords: Set Cover, Matroid, Approximation, Matroid Intersection

1. Introduction

In the canonical set cover problem the input is a set system $M = (X, \mathcal{I})$, where X is a collection of n elements, and $\mathcal{I} \subseteq 2^X$ is a collection of subsets of X. A sub-collection $\mathcal{S} \subseteq \mathcal{I}$ is said to be a cover of $Y \subseteq X$ if each element of Y is in at least one set in $\mathcal{S}$. A feasible solution is a sub-collection $\mathcal{S} \subseteq \mathcal{I}$ that covers X. The objective is to minimize $|\mathcal{S}|$. The optimal objective value is called the **cover number** of M, which we denote as $\alpha(\mathbf{M})$. In this paper we are interested in the following special case of the set cover problem, which we call the matroid intersection cover problem:

> **Matroid Intersection Cover Problem:** The input is a collection of k matroids $M_1 = (X, \mathcal{I}_1), M_2 = (X, \mathcal{I}_2), \dots M_k = (X, \mathcal{I}_k)$. This input represents the set system $M = (X, \cap_{i=1}^k \mathcal{I}_i)$. A feasible solution $\mathcal{S} \subseteq \cap_{i=1}^k \mathcal{I}_i$ is a collection of sets that covers X. Each set in $\mathcal{S}$ must be independent in every matroid. The objective is to minimize $|\mathcal{S}|$.

This problem can alternatively be viewed as the Matroid Coloring Problem, where one can color elements with the same color if they are independent in all the given matroids and the goal is to color all elements with the minimum number of colors.

In this paper we consider the approximation ratio achievable by polynomial-time algorithms for natural matroid intersection cover problems in which the number k of matroids is small. In particular, we are interested in determining whether one can achieve $O(1)$-approximation when k is a constant, which would be better than what one can achieve for general set cover instances, assuming $P \neq NP$. Before stating our results, let us give some background, and discuss the relationship of matroid intersection cover with other matroid problems.

1.1. Background

Matroid intersection cover is a natural extension of the matroid intersection problem, as well as a generalization of the matroid cover problem (also known as the matroid partitioning problem) [11, 25]. The matroid cover problem is exactly our problem, matroid intersection cover, when $k = 1$.

The maximum coverage version of matroid intersection has been more widely studied in the algorithms community beginning with the work of Nemhauser and Wolsey [20]. In this problem, the goal is to find a single set of maximum size that is independent in each of the k matroids. The natural greedy algorithm has approximation ratio k [24]. An approximation ratio of $k - 1 + \epsilon$ is achievable using local search [18]. No polynomial-time algorithm can have approximation ratio $O(\frac{k}{\log k})$ unless $P = NP$ [20]. The natural repeated application of these maximum coverage algorithms to matroid intersection cover would yield approximation ratios that are $O(k \log n)$. Extensions to this maximum coverage problem to other submodular maximum coverage problems are considered in [24, 18].

The natural greedy algorithm will compute the maximum profit independent set in a matroid. In fact, this is exactly equivalent to a hereditary set system being a matroid [28, 22]. Edmonds showed that computing the minimum cover of a single matroid can be reduced in polynomial-time to computing the maximum independent set in the intersection of two matroids, which in turn can be computed in polynomial time [11, 25]. One can compute in polynomial-time the optimal matroid intersection cover for two partition matroids, since this is just the problem of edge coloring a bipartite graph [25]. The cover number of the intersection of two arbitrary matroids is at most twice the larger of the cover numbers of the two matroids [2, Theorem 8.9]; This proof is *non-constructive*. Matroid cover intersection is related to Rota's basis conjecture [12], which corresponds to the exact (non-approximate) version of our problem when one matroid corresponds to a partition matroid and the other is a linear matroid. It is known that matroid intersection cover is NP-Hard for $k \geq 3$. Indeed, it is known that it is NP-hard to determine if a 3-partite hypergraph of degree 3 is 3 edge colorable [21]. This corresponds to the matroid intersection cover problem when $k = 3$, the matroids are three partition matroids and the goal is to determine if the cover number is 3. As an immediate consequence, it is also NP-hard to approximate the cover number for three partition matroids within a factor of $4/3 - \epsilon$. It is also known that a super polynomial number of independence queries are required when $k = 2$ for two arbitrary matroids when independence of a set is given via an oracle [5].

Matroid intersection cover is related to scheduling coflows in a network. In coflow scheduling problems there are some constraints on the flows that may be simultaneously routed in the network. Often these constraints are matroid related. Chowdhury and Stoica [8] cited information gathering at a central location in the network as a common communication pattern. Motivated by this, Im et al. [13] considered the problem of scheduling flows with gammoidal (and more generally matroidal) constraints. The bulk of the algorithmic literature on co-flow scheduling has been on scheduling matchings in a bipartite graph [23, 15, 14, 3, 1]. Bipartite matchings can be expressed as the intersection of two partition matroids. The version of the problem with the makespan objective is a special case of matroid intersection cover problem where $k = 2$ and both matroids are partition matroids.

1.2. Our Results

Our motivation for considering approximation algorithms for matroid intersection cover was sparked by our interest in more general coflow scheduling problems and the observation that other natural algorithmic problems can be also modeled as matroid cover intersection problems. Some examples are given in Section 5.

We give a general algorithmic technique for computing matroid intersection covers. The first step of this algorithmic technique is to compute what we call a partition decomposition of each of the matroids. Roughly speaking a partition decomposition of a matroid M is a partition of the elements into sets of size $O(\alpha(M))$ such that any set, that contains no more than one element from any partition, can be covered by $b = O(1)$ independent sets. A partition decomposition generalizes the concept of a weak map into a partition matroid [22]. A weak map from a general matroid M to a partition matroid is a partition decomposition in which $b = 1$. Except for gammoids, all of our decompositions will be weak maps.

This paper gives polynomial-time algorithms to compute such partition decompositions for many of the types of matroids that commonly arise in combinatorial optimization problems. Namely, partition matroids, graphic matroids, laminar matroids, transversal matroids, and gammoids. We then give a polynomial-time algorithm for computing matroid intersection covers given the partition decompositions of the matroids. Combining these algorithms, we obtain an $O(1)$-approximation algorithm when each of the $O(1)$ matroids is of a standard type.

Formally, we define a partition decomposition as follows.

Definition 1. *A matroid $M = (X, \mathcal{I})$ is (b, c)-decomposable if X can be partitioned into sets $X_1, X_2, .., X_t$ such that:*

- *For all $i \in [t]$, it is the case that $|X_i| \leq c \cdot \alpha(M)$.*
- *Any set $Y = \{r_1, \ldots, r_t\}$, consisting of one representative element r_i from each X_i, can be covered by b sets from $\mathcal{I}$.*

A class of matroids is partition decomposable if there exists constants b and c such that every matroid in that class is (b, c)-decomposable.

Although several types of matroid decomposition exist [26], this type of matroid decomposition does not seem to have been previously considered. We believe this type of decomposition is of independent interest, as it seems likely it will be useful for related problems.

Section 3 is devoted to proving the following theorem:

Theorem 2. *There are polynomial-time algorithms to compute:*

- *A $(1, 2)$-decomposition of a graphic matroid given the underlying graph as input.*
- *A $(1, 3)$-decomposition of a laminar matroid given the laminar matroid as input.*
- *A $(1, 1)$-decomposition of a transversal matroid given the underlying bipartite graph as input.*
- *A $(18, 1)$-decomposition of a gammoid given the underlying directed graph as input.*

For graphic, transversal and laminar matroids, the construction of the partition can be constructed from the natural underlying structure of the matroid. For gammoids, the construction of the partition can be done by leveraging known results on unsplittably routing flow to a single source [16].

Section 4 will be devoted to proving the following theorem:

Theorem 3. *Consider k matroids $M_1, M_2, \ldots M_k$ defined over a common ground set X, where matroid M_i has cover number α_i. There is a polynomial-time algorithm that, given a (b_i, c_i)-decomposition of each matroid M_i, computes a cover of size at most $\prod_{i \in [k]} b_i \cdot \left(1 + \sum_{i \in [k]} (c_i \alpha_i - 1)\right)$, thus with cover number at most $\left(\prod_{i \in [k]} b_i\right) \cdot \left(\sum_{i \in [k]} c_i\right) \alpha^*$ where $\alpha^* = \max_{i \in [k]} \alpha_i$.*

As α^* is an obvious lower bound to the optimal cover number, an immediate consequence is that this algorithm has approximation ratio at most $\left(\prod_{i \in [k]} b_i\right) \cdot \left(\sum_{i \in [k]} c_i\right)$. Conceptually the algorithm is based on a natural greedy algorithm for hyperedge coloring k-partite hypergraphs, and the bound on the cover number follows directly from a simple analysis of the approximation ratio of this coloring algorithm.

Combining Theorem 2 and Theorem 3, we obtain the following corollaries:

Corollary 4. *For instances of the matroid intersection problem consisting of k matroids that are either graphic, laminar, or transversal matroids, there is a polynomial-time algorithm that has approximation ratio $O(k)$.*

Corollary 5. *For instances of the matroid intersection problem consisting of k matroids that are either graphic, laminar, or transversal matroids and ℓ matroids that are gammoids, there is a polynomial-time algorithm that has approximation ratio* $O((k+\ell)\cdot 18^{\ell})$.

In Section 5 we explain the approximation ratios that our technique yields for some particular matroid intersection cover problems.

2. Related Results

Set Cover is a canonical problem in the field of approximation algorithms [27, 29]. The greedy algorithm has an approximation ratio of $H_n \approx \ln n$ on n element instances [27, 29], and this is essentially optimal assuming $P \neq NP$ [10]. Polynomial-time algorithms with approximation ratio $o(\log n)$ are known for two types of special instances. The first type is where some parameter is known to be small. An example of this type is vertex cover, where no element can be in more than two sets. Many different 2-approximation algorithms are known for vertex cover [27, 29]. The second type is geometric based/inspired. An example of this type is covering points in the plane by a minimum number of discs, for which a polynomial time approximation scheme is known [19]. As another example, it is known that constant approximation is possible if the set system has bounded VC dimension [7]. VC dimension is at least arguably a geometrically inspired concept. One interesting aspect of the results in this paper is that they give $o(\log n)$ approximations for a different type of natural set cover instances. There are of course multitudinous papers dealing with generalizations and variations on set cover, e.g. weighted covers, capacitated covers, and multicovers.

Computing unsplittable flows in single-source networks is considered in [16, 17, 9, 4], with the main take-away message being that for many natural problems the optimal objective value does not degrade too much if flows are required to be routed unsplittably. Our partition decomposition of gammoids is based on techniques from [16].

Recently and independently of this work, [6] gave upper bounds on the cover number of the intersection of two matroids for common combinatorial matroids by giving partition decompositions. In particular, among other results, [6] also gave a $(1,1)$-decomposition of traversal matroids, and a $(1, 2-2/\alpha)$-decomposition for graphic matroids. The main result in [6] is a proof that gammoids are $(1, 2-2/\alpha)$-decomposable. The results in [6] are not explicitly algorithmic. Although presumably many of the existential proofs could be readily converted into efficient algorithms, this is not true for gammoids as [6] states finding an efficient algorithm to compute such a partition decomposition of a gammoid as an open problem.

3. Computing Partition Decompositions

In this section, we show how to compute $(O(1), O(1))$-decompositions for many types of matroids that commonly arise in the combinatorial optimization literature. This includes graphic matroids, transversal matroids, laminar matroids, and gammoids.

3.1. Graphic Matroids are $(1,2)$*-Decomposable*

We begin by considering the graphic matroid. Let M be a graphic matroid on a ground set X. In this case, X corresponds to edges of an undirected graph $G = (V, X)$. Independent sets of the matroid M correspond to acyclic subsets of edges. Let $\alpha := \alpha(M)$ be the value of the minimum cover. The goal of this section is to show the following theorem.

Theorem 6. *Graphic matroids are* $(1,2)$*-decomposable and this decomposition can be computed in polynomial time.*

To prove the theorem, we begin by defining the decomposition. Consider the graph G. Let v be the vertex in G that has the smallest degree that is non-zero. Let X_v be a set in the decomposition that contains all of the edges adjacent to v. Remove v and its adjacent edges from G. Then recurse on the resulting graph to obtain the next set in the partition. The procedure stops when there are no remaining edges. Notice that the sets X_v constructed are a partition of the ground set X and these correspond to the sets of the decomposition.

Our goal is to show that this is a valid decomposition. To do so, we first show that as edges are removed from the graph, the size of the optimal cover of the remaining edges never increases.

Lemma 7. *Consider any undirected graph* $G = (V, X)$ *and any vertex* $v \in V$*. Let* M *be the graphic matroid corresponding to* G*. Let* G' *be the graph obtained from* G *by deleting a vertex* v *and all of its adjacent edges and let* M' *be the corresponding graphic matroid for* G'*. Then we have* $\alpha(M') \leq \alpha(M)$.

Proof. Let $X_1^*, X_2^*, \ldots X_\alpha^*$ denote the optimal cover of M. Consider removing the edges adjacent to v from the sets X^*. The resulting sets are a cover of M' and are independent in M'. This is because each set X_i^* will correspond to an acyclic set of edges in G' since they were acyclic in G. □

We can use the previous lemma to show every part has size at most 2α.

Lemma 8. *Every partition* X_v *constructed has size at most* 2α.

Proof. Consider any set X_v and let $G' = (V', X')$ be the graph that remains in the algorithm just before X_v is constructed. We know that the edges of G' can be partitioned into at most α sets of acyclic edges by Lemma 7. Notice that any acyclic subgraph of G' has at most $|V'|-1$ edges. This implies that G' has at most $\alpha(|V'|-1)$ edges and the aggregate degrees of the vertices is at most $2\alpha(|V'|-1)$. By definition of the algorithm v is the minimum degree vertex in V' and therefore has at most 2α adjacent edges. Hence $|X_v| \leq 2\alpha$. □

To complete the proof of the theorem, it will be shown that any set S containing up to one element from each partition X_v is independent in the original matroid M. This is established in the following lemma.

Lemma 9. *Let $X_{v_1}, X_{v_2}, \ldots X_{v_\ell}$ be the partition of X constructed by the algorithm. For any set S such that $|S \cap X_{v_i}| \leq 1$ it is the case that S is an acyclic set of edges in G.*

Proof. For the sake of contradiction, suppose S contains a cycle C for some S such that $|S \cap X_{v_i}| \leq 1$. Assume wlog that $X_{v_1}, X_{v_2}, \ldots X_{v_\ell}$ were constructed in this order. Notice the vertices of the cycle will be the vertices v_i where $|X_{v_i} \cap S| = 1$. If edge (u, v) is part of C, then either $(u, v) \in X_u$ or $(u, v) \in X_v$. Since C can collect at most one edge from each partition, this implies that either $(v_{i(1)}, v_{i(2)}) \in X_{v_{i(1)}}$, $(v_{i(2)}, v_{i(3)}) \in X_{v_{i(2)}}$, ..., $(v_{i(|C|)}, v_{i(1)}) \in X_{v_{i(|C|)}}$ or $(v_{i(1)}, v_{i(2)}) \in X_{v_{i(2)}}$, $(v_{i(2)}, v_{i(3)}) \in X_{v_{i(3)}}$, ..., $(v_{i(|C|)}, v_{i(1)}) \in X_{v_{i(1)}}$ for some $1 \leq i(1) < i(2) < ... < i(|C|) \leq \ell$. For the first case, we have a contradiction since $X_{v_{i(1)}}$ was constructed before $X_{v_{i(|C|)}}$, meaning that $X_{v_{i(|C|)}}$ has no edges adjacent to $v_{i(1)}$. For the second case, we obtain a similar contradiction. □

Theorem 6 follows immediately from Lemmas 8 and 9.

3.2. Transversal Matroids are (1, 1)*-Decomposable*

This section shows that transversal matroids are (1, 1)-decomposable. Let M be a transversal matroid defined over a ground set X. In this case, X corresponds to one side of a bipartite graph $G = (X \cup Y, E)$. A subset of X is independent if there exists a matching of G that that matches all vertices in X. A partition matroid is a special case of a transversal matroid in which there is a partition $X_1, \ldots X_k$ of the ground set X and a bound u_i for each X_i. A collection of elements is then independent if it does not contain more than u_i elements from any X_i.

Theorem 10. *Transversal matroids are* (1, 1)*-decomposable and this decomposition can be computed in polynomial time.*

Proof. Adopting the above notation, let $\alpha := \alpha(M)$ be the minimum cover of X using sets independent in M. Let $X_1^*, X_2^*, \ldots X_\alpha^*$ denote a minimum cover of all of the elements in X such that each of these sets is independent in M. Assume wlog that the sets are disjoint. For each set X_i^* let E_i^* be a matching saturating X_i^* in G. Let $E^* = E_1^* \cup E_2^* \cup \ldots E_\alpha^*$ be the subset of E edges used in the matchings. Let the graph G^* be the graph induced by E^*. The partition will be induced by the vertices in Y and their neighbors in G^* that are in the ground set X. In particular, let X_v be all elements in X that are adjacent to $v \in Y$ in G^*. This completes the definition. Notice that these sets are computable in polynomial time since the minimum cover of M and E^* are computable.

We need to show two properties to complete the proof. The first property is that $|X_v| \leq \alpha$. This follows because the construction of the graph G^* ensures that every vertex has degrees at most α since E^* consists of α matchings. The second property that needs to be shown is that every set S is independent in M if $|S \cap X_v| \leq 1$ for all $v \in Y$. Indeed, consider any such set S. The definition of S implies that each vertex in S is adjacent to a unique vertex in G^*. Thus, S must correspond to a subset of X that can be saturated in a matching of G and therefore S is independent in M. □

3.3. Laminar Matroids are (1, 3)*-Decomposable*

In this section we show that laminar matroids are (1, 3)-decomposable. Let M be a laminar matroid on a ground set X. Let $\mathcal{F}$ be the laminar family of subsets of X associated with the matroid – the family is said to be laminar if for any $A, B \in \mathcal{F}$, we have $A \cap B = \emptyset$, $A \subseteq B$, or $B \subseteq A$. Further each set $A \in \mathcal{F}$ is associated with a positive integer $b(A)$, which is called the *capacity* of the set. A set $I \subset X$ is independent in M if $|I \cap A| \leq b(A)$ for all $A \in \mathcal{F}$.

The goal of this section is to show the following theorem.

Theorem 11. *Laminar matroids are* (1, 3)*-decomposable and this decomposition can be computed in polynomial time.*

We say that a set $A \in \mathcal{F}$ is a leaf set of the family $\mathcal{F}$ if there is no $B \in \mathcal{F}$ such that $B \subset A$. For any non-leaf set $A \in \mathcal{F}$, we can assume wlog $b(A) \geq 2$; otherwise, we can remove any subsets of A from $\mathcal{F}$ as they are redundant. If there is an element x in X that doesn't belong to any leaf set, we create a singleton set $\{x\}$ with capacity 1 and add it to the family without loss of generality. We also assume that $|X| > 3\alpha$ since if $|X| \leq 3\alpha$, then the set X itself is an obvious (1, 3)-decomposition of M.

We now describe how we obtain a (1, 3)-decomposition of M. Let $\alpha := \alpha(M)$ be the min cover size of M.

1. For each leaf set $A \in \mathcal{F}$ if $|A| \leq \alpha$, set $b(A)$ to 1. Otherwise, partition A into disjoint subsets $A_1, A_2, ..., A_k$ so that $|A_1| = |A_2| = ... = |A_{k-1}| = \alpha$ and $|A_k| \leq \alpha$. Let $b(A_1) = b(A_2) = ... = b(A_k) = 1$. Add these leaf sets to $\mathcal{F}$.

2. Linearly order elements $x_1, x_2, ..., x_n$ so that for any $1 \leq u < v < w \leq n$, if x_u, $x_w \in A$, where $A \in \mathcal{F}$, then $x_v \in A$.

3. From left to right, we repeat the following: collect the smallest possible collection of leaf sets and take the union of them, so that the union has least 2α elements. Let the resulting unions, $X_1, ..., X_t$ be our decomposition of X.

The following observation is immediate from the algorithm definition since after partitioning leaf sets, every leaf set has cardinality at most α.

Observation 12. *For all $i \in \{1, 2, ..., t\}$, $2\alpha \leq |X_i| < 3\alpha$.*

To show $X_1, X_2, ..., X_t$ is a valid (1, 3)-decomposition of M, it suffices to show the following.

Lemma 13. *For any $I \subseteq X$ such that $|I \cap X_i| \leq 1$ for all $i \in [t]$, we have $|I \cap A| \leq b(A)$ for all $A \in \mathcal{F}$.*

Proof. For the sake of contradiction suppose $|I \cap A| > b(A)$ for some $A \in \mathcal{F}$ that we didn't create. Let $X_{j(1)}$, $X_{j(2)}$, ..., $X_{j(|I \cap A|)}$ be the sets that include an element from $I \cap A$ – note that the number of such sets is exactly $|I \cap A|$ since I can have at most one element from each partition. We consider three cases and show a contradiction for each case.

Case (i) $b(A) \geq 3$. Because of the second and third steps, $X_{j(2)}$, ..., $X_{j(|I \cap A|-1)}$ must be subsets of A. Thus, $|A| \geq \sum_{i \in [|I \cap A|-1] \setminus \{1\}} |X_{j(i)}| \geq 2\alpha \cdot (b(A) - 1)$ by Observation 12 and

due to the assumption that $|I \cap A| > b(A)$. However, we have $|A| \le \alpha \cdot b(A)$ since otherwise the min cover size of M would be greater than α. Combining these two inequalities gives us $b(A) \le 2$, which is a contradiction.
Case (ii) $b(A) = 2$. As before, we have $X_{j(2)} \subseteq A$. Further, $A \cap X_{j(1)} \neq \emptyset$. By Observation 12, we have $|A| > 2\alpha$. This is a contradiction since $b(A) = 2$.
Case (iii) $b(A) = 1$. In this case, $A \subseteq X_{t'}$ for some $t' \in [t]$. This is because A must be a leaf set that we didn't create. Since $|I \cap X_{t'}| \le 1$, we have $|I \cap A| \le 1$, which means I didn't violate A's capacity. □

Thus, we have shown Theorem 11.

3.4. Gammoids Are (18, 1)-Decomposable

This section shows that gammoid matroids are $(18, 1)$-decomposable. Gammoids are defined as follows. Let M be a gammoid defined on the ground set X of n elements. The matroid is associated with an directed graph $G = (V, E)$ with a sink $t \in V$. The ground set of X is a collection of sources, $\{s_1, \dots s_n\} \subseteq V$. A subset $S \subseteq X$ is independent in the matroid if there is routing of a unit of flow from each of the sources in S to the given sink t so that no more than one unit of flow is routed over any edge.

We note that this definition of gammoid is equivalent to another definition of gammoid based on vertex-disjoint paths. To see the equivalence, we assume wlog that every source has no incoming edges and has only one out-going edge by adding dummy edges: if s_i is a source vertex, we can add an edge (s_i', s_i) and consider s_i' as the new corresponding source. We first consider the reduction from vertex-disjoint-path gammoids to edge-disjoint-path gammoids, where we split each vertex v into v_{in} and v_{out} and connect to v_{in} all vertices adjacent to v and v_{out} to all vertices adjacent from v, and connect v_{in} to v_{out}. It is easy to see that each path before the splitting has a uniquely corresponding path after the splitting, and two paths before the splitting are vertex-disjoint if and only if the corresponding paths are edge-disjoint after the splitting. The other direction can be shown by replacing each vertex v with two sets of vertices $S(v)^+$ and $S(v)^-$ where $|S(v)^+| = \delta^+(v)$ and $|S(v)^-| = \delta^-(v)$. There is a complete bipartite graph from vertices in $S(v)^-$ to vertices in $S(v)^+$. Each edge (v, u) is replaced with an edge going from a unique vertex in $S^+(v)$ to a unique vertex in $S^-(u)$.

Theorem 14. *There is a polynomial time algorithm to compute* $(18, 1)$*-decomposition of a gammoid* M *given the representation* (G, t, X) *as input.*

Proof. Lemma 6.8 in [16] shows how to compute in polynomial time a collection $\{T_1, \dots T_h\}$ of reverse arborescences rooted at vertices $\{\ell_1, \dots \ell_h\}$, and a partition $X = \{X_1, \dots X_h\}$ of the sources with the following properties:

- Each X_i has size at most $\alpha(M)$.
- For all i, all sources in X_i are connected to an edge in T_i.
- No edge in G is contained in more than two T_i.
- There is a feasible fractional routing F of one unit of flow from each ℓ_i to t such that the flow through any edge is at most 7.

The collection X will be our partition decomposition.

Now consider a collection S of sources consisting of at most one arbitrary source s_i from each X_i. There must then be a fractional routing of one unit of flow from each source in S to t such that the flow through any edge is at most 9. One way to achieve this is to route flow from each s_i to ℓ_i within T_i and then from ℓ_i to the sink t following F. Each edge can have most two units routed through it due to the initial routing within T_i, and at most 7 units routed through it due to F.

Lemma 5.5 in [16] shows how to transform a fractional routing with maximum flow c on any edge into a collection of $2c$ integral routings that collectively route all the flow, and such that no edge has more than one unit of flow routed through it in each of the $2c$ routings. (Lemma 5.5 in [16] is only stated for undirected graphs, but it is noted later, right before Theorem 6.14, that it still holds for directed graphs.) Applying the lemma to our fractional flow, we get a decomposition of S into 18 independent sets. □

We note that using techniques from [16] one can also show that gammoids that arise from undirected graphs are $(6, 1)$-decomposable.

4. Covering Partitioned Matroids

The purpose of this section is to prove Theorem 3. Let P_i be a (b_i, c_i)-decomposition of matroid M_i. Consider the k-partite k-uniform hypergraph $H = (V_1, \dots V_k, E)$ where there is one vertex in each V_i corresponding to each partition in P_i, and one hyperedge e corresponding to each element e in the ground set X. Hyperedge e will be incident to the vertex in V_i corresponding to the partition to which e belongs in P_i. Define a proper coloring of the hyperedges to be an assignment of colors to the hyperedges such that no pair of hyperedges incident on a common vertex are assigned the same color. Note for all $i \in [k]$ that by the definition of (b_i, c_i)-decomposability the hyperedges colored a particular color in a proper coloring of H can be decomposed into b_i independent sets in M_i. And thus for all $i \in [k]$ the color classes in a proper coloring of H correspond to a cover of X by sets that each be decomposed into b_i independent sets in M_i.

Our algorithm will first color the hyperedges in H in a greedy fashion. That is, the hyperedges are considered in arbitrary order, and hyperedge e is assigned the first color not already assigned to any edge that is incident on a common vertex with e. The number of colors used by this greedy algorithm will be at most one more than the maximum number of hyperedges that can be incident on a common vertex with a particular hyperedge e. This is at most $\sum_{i \in [k]} (c_i \alpha_i - 1)$. Thus this greedy algorithm produces a hyperedge coloring that can be interpreted as a cover C of X of at most $1 + \sum_{i \in [k]} (c_i \alpha_i - 1)$ sets.

Then for each $S \in C$ and each matroid M_i, our algorithm uses any polynomial time algorithm for matroid covering to partition each set S into a collection $\mathcal{D}_i^S = \{D_i^S(1), \ldots, D_i^S(b_j)\}$ parts such that each part $D_i^S(j)$ is independent in M_i. Now let V be the collection k dimensional vectors where component i is an integer in the range $[1, b_i]$. So the cardinality of V is $\prod_{i=1}^k b_i$. Then for all $v \in V$, let S^v be the collection of $s \in S$ such that for all $i \in [k]$ it is the case that $s \in D_i^S(v(i))$, where $v(i)$ is component i of v. In other words, S^v is the collection of all elements of S that are in part $v(i)$ in $\mathcal{D}_i^S$. Our final cover will be the collection of all such S^v, namely:

$$\{S^v \mid S \in C \text{ and } v \in V\}$$

5. Applications of Our Techniques

The following problems can all be modeled as matroid intersection cover problems. We also remark that minimizing makespan of a co-flow of jobs that arrive all at the same time is a special case of our problem when $k = 2$ and the input is two partition matroids.

1. The input is a k-partite hypergraph G in which each hyperedge contains exactly one vertex from each of the k sets in the partition of the vertices. A feasible solution is a coloring of the hyperedges such that no two hyperedges that share a vertex receive the same color. The objective is to minimize the number of colors. As this problem plays a special role in our results, let us refer to this as the hypergraph coloring problem.

2. The input is a directed graph, where each vertex v has an associated bound b_v, each edge has a color, and each color c has an associated bound u_c. A feasible solution is a partition $E_1, \ldots, E_t$ of the edges such that:

   - Each E_i is the union of disjoint arborescences.
   - For all vertices v and for all E_i the out-degree of v in E_i is at most b_v.
   - For all colors c and for all E_i the number of edges colored c in E_i is at most u_c.

   The objective is to minimize t.

3. The input is $k = O(1)$ graphs/networks $G_i = (V_i, E_i)$, $i \in [k]$, a sink vertex t_i in each V_i, and a collection $S \subseteq \cap_{i=1}^k V_i$ of source vertices common to all networks (the rest of the networks are disjoint). Each source needs to be simultaneously connected by a dedicated path P_i in each G_i to each t_i for one consecutive unit of time. Multiple sources can be connected at the same time as long as their paths are disjoint. The problem is to minimize the time that it takes to achieve this.

4. The input is a layered graph G where the vertices are partitioned into k layers, and all edges are between vertices in adjacent layers. Further each layer has an associated laminar decomposition where each set s in the laminar decomposition has an associated bound b_s. A full path in G is a simple path from a vertex in the first layer to a vertex in the last layer. A layered covering of G is a collection $P_1, \ldots, P_t$ where each P_i consists of a disjoint collection of full paths such that:

   - Every full path is in some P_i, and
   - For every P_i, and for every layer and for every set s in the laminar decomposition of this layer, the number of paths in P_i that contain a vertex in s is at most b_s.

   The problem is to find the layered covering that minimizes t.

We now explain how to obtain bounds on the approximation ratios of the polynomial-time algorithm for these problems that one can derive from Corollary 5:

1. A feasible color class can be represented as the intersection of k partition matroids. Since partition matroids are $(1, 1)$-decomposable, applying Theorem 3 we obtain a polynomial-time k-approximation algorithm.

2. The feasible A_i can be represented by the intersection of a graphic matroid guaranteeing that the edges are acyclic (ignoring the directions of the edges), a partition matroid guaranteeing that in-degree of each vertex is at most one, a partition matroid bounding the out-degree of each vertex, and a partition matroid bounding the number of colors. Thus as there are three partition matroids, each of which are $(1, 1)$-decomposable, and one graphic matroid, which is $(1, 2)$-decomposable, applying Theorem 3 yields a polynomial-time 5-approximate algorithm.

3. The collection of sources that can be feasibly routed in a time step can be represented by the intersection of k gammoids, which we have shown are $(18, 1)$-decomposable. Thus applying Theorem 3 we obtain a $k18^k$-approximation algorithm.

4. The feasible P_i can be represented as the intersection of k laminar matroids over the full paths. Each partition corresponds to a laminar matroid. As we have shown that laminar matroids are $(1, 3)$-decomposable, applying Theorem 3 we obtain a polynomial time $3k$-approximation algorithm.

6. Conclusion and Open Problems

There are many natural interesting open questions that natural arise from the results in this paper, including:

- Is there an efficient algorithm to find a $(1, O(1))$ decomposition of a gammoid? If not, how about gammoids derived from undirected graphs?
- [6] conjectured that all matroids are $(1, 2)$-decomposable. Is this really the case? If not, how about $(1, O(1))$ or even $(O(1), O(1))$? If not, can the class of matroids that have such partition decompositions have nice characterizations?

- In the event that general matroids are not $(O(1), O(1))$-decomposable, is polynomial-time $O(1)$-approximation still possible for matroid intersection cover with $O(1)$ general matroids?
- Or even stronger, is $O(k)$-approximation possible with k general matroids?
- If the sets have weights, is $O(1)$-approximation possible for the problem of computing the minimum weight matroid intersection cover?

Acknowledgments: We thank James Oxley, Ron Aharoni, and Anupam Gupta for helpful discussions. S. Im was supported in part by NSF grants CCF-1409130, CCF-1617653, and CCF-1844939. B. Moseley was supported in part by a Google Research Award, an Infor Research Award, a Carnegie Bosch Junior Faculty Chair and NSF grants CCF-1824303, CCF-1845146, CCF-1733873 and CMMI-1938909. K. Pruhs was supported in part by NSF grants CCF-1421508, CCF-1535755, CCF-1907673, CCF-2036077 and an IBM Faculty Award.

[1] Saksham Agarwal, Shijin Rajakrishnan, Akshay Narayan, Rachit Agarwal, David Shmoys, and Amin Vahdat. Sincronia: Near-optimal network design for coflows. In *Conference of the ACM Special Interest Group on Data Communication*, pages 16–29, 2018.

[2] Ron Aharoni and Eli Berger. The intersection of a matroid and a simplicial complex. *Transactions of the American Mathematical Society*, pages 4895–4917, 2006.

[3] Saba Ahmadi, Samir Khuller, Manish Purohit, and Sheng Yang. On scheduling coflows. In *Conference on Integer Programming and Combinatorial Optimization*, pages 13–24, 2017.

[4] Nikhil Bansal, Anupam Gupta, Ravishankar Krishnaswamy, Viswanath Nagarajan, Kirk Pruhs, and Cliff Stein. Multicast routing for energy minimization using speed scaling. In *Mediterranean Conference on Algorithms*, pages 37–51, 2012.

[5] Kristóf Bérczi and Tamás Schwarcz. Complexity of packing common bases in matroids. *CoRR*, abs/1903.03579, 2019.

[6] Kristóf Bérczi, Tamás Schwarcz, and Yutaro Yamaguchi. List colouring of two matroids through reduction to partition matroids. *CoRR*, abs/1911.10485, 2019.

[7] Hervé Brönnimann and Michael T. Goodrich. Almost optimal set covers in finite vc-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.

[8] Mosharaf Chowdhury and Ion Stoica. Coflow: A networking abstraction for cluster applications. In *ACM Workshop on Hot Topics in Networks*, pages 31–36. ACM, 2012.

[9] Yefim Dinitz, Naveen Garg, and Michel X. Goemans. On the single-source unsplittable flow problem. In *Symposium on Foundations of Computer Science*, pages 290–299, 1998.

[10] Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *Symposium on Theory of Computing*, pages 624–633, 2014.

[11] Jack Edmonds. Submodular functions, matroids, and certain polyhedra. In *Calgary International Conference on Combinatorial Structures and their Applications*, pages 69 – 92, 1969.

[12] Rosa Huang and Gian-Carlo Rota. On the relations of various conjectures on Latin squares and straightening coefficients. *Discrete Math.*, 128(1-3):225–236, 1994.

[13] Sungjin Im, Benjamin Moseley, Kirk Pruhs, and Manish Purohit. Matroid coflow scheduling. In *International Colloquium on Automata, Languages and Programming*, 2019.

[14] Samir Khuller, Jingling Li, Pascal Sturmfels, Kevin Sun, and Prayaag Venkat. Select and permute: An improved online framework for scheduling to minimize weighted completion time. In *Latin American Theoretical INformatics Symposium*, pages 669–682. Springer, 2018.

[15] Samir Khuller and Manish Purohit. Brief announcement: Improved approximation algorithms for scheduling co-flows. In *Symposium on Parallelism in Algorithms and Architectures*, pages 239–240. ACM, 2016.

[16] Jon M. Kleinberg. Single-source unsplittable flow. In *Symposium on Foundations of Computer Science*, pages 68–77, 1996.

[17] Stavros G. Kolliopoulos and Clifford Stein. Improved approximation algorithms for unsplittable flow problems. In *Symposium on Foundations of Computer Science*, pages 426–435, 1997.

[18] Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Math. Oper. Res.*, 35(4):795–806, November 2010.

[19] Nabil H. Mustafa and Saurabh Ray. PTAS for geometric hitting set problems via local search. In *ACM Symposium on Computational Geometry*, pages 17–22, 2009.

[20] G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Math. Oper. Res.*, 3(3):177–188, August 1978.

[21] Pawel Obszarski and Andrzej Jastrzbski. Edge-coloring of 3-uniform hypergraphs. *Discrete Applied Mathematics*, 217:48 – 52, 2017. Combinatorial Optimization: Theory, Computation, and Applications.

[22] James Oxley. *Matroid Theory*. Oxford graduate texts in mathematics. Oxford University Press, 2006.

[23] Zhen Qiu, Cliff Stein, and Yuan Zhong. Minimizing the total weighted completion time of coflows in datacenter networks. In *Symposium on Parallel Algorithms and Architectures*, pages 294–303. ACM, 2015.

[24] Joachim Reichel and Martin Skutella. Evolutionary algorithms and matroid optimization problems. *Algorithmica*, 57(1):187–206, May 2010.

[25] Alexander Schrijver. Matroid Intersection. In *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24, chapter 41. Springer Science & Business Media, 2003.

[26] K. Truemper. *Matroid Decomposition*. Elsevier Science, 2014.

[27] Vijay Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.

[28] D.J.A. Welsh. *Matroid theory*. L.M.S. monographs. Academic Press, 1976.

[29] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.