Dynamic Matroids: Base Packing and Covering

Tijn de Vos[1]
TU Graz, Austria

Mara Grilnberger
Department of Computer Science, University of Salzburg, Austria

Abstract

In this paper, we consider dynamic matroids, where elements can be inserted to or deleted from the ground set over time. The independent sets change to reflect the current ground set. As matroids are central to the study of many combinatorial optimization problems, it is a natural next step to also consider them in a dynamic setting. The study of dynamic matroids has the potential to generalize several dynamic graph problems, including, but not limited to, arboricity and maximum bipartite matching. We contribute by providing efficient algorithms for some fundamental matroid questions.

In particular, we study the most basic question of maintaining a base dynamically, providing an essential building block for future algorithms. We further utilize this result and consider the elementary problems of base packing and base covering. We provide a deterministic algorithm that maintains a $(1\pm\varepsilon)$-approximation of the base packing number Φ in $O(\Phi\cdot\text{poly}(\log n,\varepsilon^{-1}))$ queries per update. Similarly, we provide a deterministic algorithm that maintains a $(1\pm\varepsilon)$-approximation of the base covering number β in $O(\beta\cdot\text{poly}(\log n,\varepsilon^{-1}))$ queries per update. Moreover, we give an algorithm that maintains a $(1\pm\varepsilon)$-approximation of the base covering number β in $O(\text{poly}(\log n,\varepsilon^{-1}))$ queries per update against an oblivious adversary.

These results are obtained by exploring the relationship between *base collections*, a generalization of tree-packings, and base packing and covering respectively. We provide structural theorems to formalize these connections, and show how they lead to simple dynamic algorithms.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Dynamic graph algorithms

Keywords and phrases matroid, minimum weight base, base packing, base covering, dynamic

Digital Object Identifier 10.4230/LIPIcs.ESA.2026.57

Related Version *Full Version*: `https://arxiv.org/abs/2511.15460`

Funding *Tijn de Vos*: This research was funded in whole or in part by the Austrian Science Fund (FWF) `https://doi.org/10.55776/P36280`. For open access purposes, the author has applied a CC BY public copyright license to any author-accepted manuscript version arising from this submission. *Mara Grilnberger*: This publication has been supported by the EXDIGIT (Excellence in Digital Sciences and Interdisciplinary Technologies) project, funded by Land Salzburg under grant number 20204-WISS/263/6-6022. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 947702).

Acknowledgements The authors would like to thank Aleksander Christiansen for the preliminary discussions that were the inspiration for this research.

[1] The author ordering was randomized using `https://www.aeaweb.org/journals/policies/random-author-order/`.

1 Introduction

Matroids generalize different mathematical concepts such as graphs and vector spaces. They have applications in combinatorial optimization, geometry, topology, network theory, and coding theory [52, 34, 63, 55, 27]. In particular, matroid problems are often seen as "the problems where greedy algorithms are effective" (see, e.g., [56]). Although there are dynamic algorithms for specific matroids, the work on dynamic algorithms for general matroids is limited. Concurrent work by Chandrasekaran, Chekuri, and Zhu [10] also initiated the somewhat related study of *online* matroids, where the matroid is slowly revealed over time. With this paper, we would like to develop the study of dynamic matroids by providing efficient algorithms for some fundamental matroid questions. In particular, we study the most basic question of maintaining a base dynamically, providing essential building blocks for future algorithms. Let us start with some definitions.

Matroids. Formally, a matroid is defined as a tuple $\mathcal{M} = (E, \mathcal{I})$, where E is a finite *ground set* of elements and $\mathcal{I} \subseteq \mathcal{P}(E)$ a family of *independent sets*, such that the following three properties hold: 1) Non trivial: $\emptyset \in \mathcal{I}$. 2) Downward closure: if $A \in \mathcal{I}$ and $A' \subseteq A$, then $A' \in \mathcal{I}$. 3) Exchange property: if $A, B \in \mathcal{I}$ and $|A| > |B|$, then there is an element $e \in A \setminus B$ such that $B \cup \{e\} \in \mathcal{I}$.

We denote $n := |E|$ to be the size of the ground set. The *rank* of a set $A \subseteq E$ is defined as the size of the largest independent set it contains:

$$\mathrm{rk}(A) := \max_{\substack{A' \in \mathcal{I} \text{ s.t.} \\ A' \subseteq A}} |A'|.$$

We say that $B \subseteq E$ is a *base* of $\mathcal{M}$ if it is a maximal independent set, i.e., $\mathrm{rk}(B) = |B| = \mathrm{rk}(E)$. Given a weight function on the elements, a *minimum weight base* is a base of the smallest total weight. We define a *base packing* as a maximum set of disjoint bases. We define a *base covering* as a minimum set of bases (not necessarily disjoint) such that each element is contained in at least one base. Since $\mathcal{I}$ can be as large as $2^{|E|}$, it is often not given explicitly, but implicitly via oracle access. In this paper, we use a *rank oracle*, which provides $\mathrm{rk}(A)$ upon a query $A \subseteq E$.

Matroid Problems. Two classic matroid problems are matroid union and matroid intersection. They reduce to each other in polynomial time, see, e.g., [19, 45]. These problems generalize many graph problems, such as packing disjoint spanning trees, computing the arboricity, bipartite matching, see, e.g., [56]. Base packing and base covering can be seen as instances of matroid union. In this paper, we investigate these two fundamental cases in the dynamic setting.

One well-studied example of a matroid is the *graphic matroid*, where E is some set of edges and $A \subseteq E$ is independent if and only if it is acyclic. Base packing corresponds to packing disjoint spanning trees and the base covering number corresponds to the arboricity of the graph. Throughout, we generalize results for the graphic matroid and we will state the algorithms for graphic matroids as a comparison.

Algorithms for base packing and covering have been studied for a long time, see, e.g., [44, 14, 38, 39, 11, 5]. The state of the art is given by Quanrud [54]. They provide exact algorithms that use $\tilde{O}(n + k \cdot \mathrm{rk}(E)^2)$ independence-queries[2], where k is the packing/covering number respectively. They also provide $(1+\varepsilon)$-approximations in $\tilde{O}(n/\varepsilon)$ independence-queries.

[2] For simplicity, we use the notation $\tilde{O}(f) := O(f \operatorname{poly} \log f)$.

Dynamic Matroids. For dynamic matroids, we consider element insertions and deletions to the ground set.[3] When an element e is deleted, the independent sets $\mathcal{I}$ simply restrict to the sets without e: $\{I \in \mathcal{I} : e \notin I\}$. In other words, $\mathcal{M}$ is restricted to $E \setminus \{e\}$. When an element e is inserted, the adversary also decides on a collection $\mathcal{I}_e$ such that $(E \cup \{e\}, \mathcal{I} \cup \mathcal{I}_e)$ is a matroid and $e \in I$ for every $I \in \mathcal{I}_e$. This means that inserting and then deleting the same element results in the same matroid. Note that, the same is not true, if the operations are reversed. If an element is first deleted and then reinserted, the resulting matroid may differ from the previous version. The algorithm receives the element updates and can query the (new) independence sets.

An alternative definition of dynamic matroids could be as follows. One could allow for updates to $\mathcal{I}$ that do not stem from an element deletion or insertion. However, if we can completely change $\mathcal{I}$ in a single update, then we could go from any matroid on n elements to any other matroid on n elements. This means that any lower bound for a static algorithm carries over to *each* update of the dynamic algorithm. Hence, recomputing from scratch is the best one could do. On the other extreme, we could only allow adding or deleting *a single set* I from $\mathcal{I}$. We note that, in general, such a change to $\mathcal{I}$ will no longer guarantee that it is a matroid: consider adding I to $\mathcal{I}$. By the downward closure property, all subsets of I also have to be in $\mathcal{I}$. We consider the most restrictive (and hence most general) version of this, where we only add I if $I \setminus \{e\}$ is already in $\mathcal{I}$ for some element e. The alternative view is that the element e is added to the ground set[4], and the independent sets are updated accordingly.

We distinguish two types of adversaries: an *oblivious adversary* fixes the updates beforehand, independent of the random choices in the algorithm, while an *adaptive adversary* determines the next update depending on the current state of the algorithm. In this paper, our algorithms are either deterministic, hence hold against an adaptive adversary, or are randomized and hold against an oblivious adversary.

We note that dynamic matroids have been studied in the realm of submodular function maximization over dynamic matroids, see, e.g., [48, 13, 4].

Scope. We approach this model by first investigating the classical matroid problem of a minimum weight base and then considering the fundamental problems of base packing and covering. Although the former has been studied in [6] (see the discussion below Proposition 1), there are no results regarding the latter two. However, in the special case of the graphic matroid, all three problems have been well studied.

Minimum Weight Base. First, we give our result for maintaining a minimum weight base.

▶ **Proposition 1.** *There exists a deterministic algorithm that, given a dynamic matroid $\mathcal{M}$ with weight function $w\colon E \to [1, \dots, W]$, maintains a minimum weight base, where each update uses $O(\log n)$ rank-queries.*

Blikstad, Mukhopadhyay, Nanongkai, and Tu [6] provide an algorithm for maintaining a minimum weight basis under deletions. Each update requires $\tilde{O}(\sqrt{\mathrm{rk}(E)})$ worst-case rank-queries. However, using a "dynamic rank oracle", they also ensure $\tilde{O}(\sqrt{\mathrm{rk}(E)})$ worst-case update *time*, as it can only answer queries, where the answer can be computed efficiently on a concrete matroid. The algorithm is based on the MST algorithm with $\tilde{O}(\sqrt{|V|})$ update

[3] The *decremental* version (deletions only) of this has been studied in [6], and we argue below that this *fully dynamic* version is the natural and most general definition.

[4] If e was already part of another independent set $e \in I' \in \mathcal{I}$, we can model this by deleting and inserting e.

time by Frederickson [25], combined with the sparsification technique of Eppstein, Galil, Italiano, and Nissenzweig [23]. It seems likely that such a result can be extended to the fully dynamic setting. Blikstad et al. [6] use this as a subroutine and the authors seem to have optimized their result for their application in solving matroid union. The difference to our result, Proposition 1, is that we have a much lower query time, but do not give guarantees on the update time, since the computation time for answering a query depends on the concrete matroid.

Both the algorithm [6] and our algorithm from Proposition 1 use a rank oracle. Sometimes, algorithms are developed using an *independence oracle*, which only provides whether $A \in \mathcal{I}$. Clearly, the rank oracle is stronger than the independence oracle. Statically, it is even known that it is *strictly* stronger: computing a minimum weight base can be done with n simultaneous rank-queries, while this needs $\tilde{\Omega}(n^{1/3})$ rounds of simultaneous independence-queries [41]. It would be interesting to see if there exists a decremental algorithm for maintaining a minimum weight base using a sublinear number of independence queries.

Minimum Weight Base in the Graphic Matroid. The dynamic minimum spanning tree (MST) problem is one of the most studied problems in dynamic graph algorithms, see, e.g., [25, 24, 23, 26, 1, 32, 35, 50, 65]. The special case of an unweighted graph is the dynamic spanning tree problem (see, e.g., [33, 31, 58, 53, 37, 30, 64, 49, 36]), which also has close ties to dynamic connectivity. Let us highlight the state of the art: Holm, de Lichtenberg, and Thorup [35] maintain an MST deterministically with $O(\log^4 |E|)$ amortized update time. Nanongkai, Saranurak, and Wulff-Nilsen [50] provide a Las Vegas algorithm with $|V|^{o(1)}$ worst-case update time.

Packing and Covering. The other two problems we treat in this paper are *base packing* and *base covering*. In *base packing* the goal is to pack as many disjoint bases in $\mathcal{M}$ as possible. Formally, we define the *(fractional) packing number* $\Phi_{\mathcal{M}}$ of a matroid $\mathcal{M}$ as follows

$$\Phi_{\mathcal{M}} := \min_{\substack{A \subseteq E \text{ s.t.} \\ \mathrm{rk}(\overline{A}) < \mathrm{rk}(E)}} \frac{|A|}{\mathrm{rk}(E) - \mathrm{rk}(\overline{A})}.$$

This is also known as matroid strength. The integral packing number is $\lfloor \Phi_{\mathcal{M}} \rfloor$. Edmonds [20] showed that $\lfloor \Phi_{\mathcal{M}} \rfloor$ equals the number of disjoint bases that can be packed in $\mathcal{M}$.

Dual to base packing, in *base covering* the goal is to cover $\mathcal{M}$ by as few bases as possible. Formally, we define the *(fractional) covering number* $\beta_{\mathcal{M}}$ of a matroid $\mathcal{M}$ as follows

$$\beta_{\mathcal{M}} := \max_{\substack{A \subseteq E \text{ s.t.} \\ A \neq \emptyset}} \frac{|A|}{\mathrm{rk}(A)}.$$

This is also known as matroid density. We define the integral covering number as $\lceil \beta_{\mathcal{M}} \rceil$. Edmonds [21] showed that $\lceil \beta_{\mathcal{M}} \rceil$ equals the minimum number of bases necessary to cover $\mathcal{M}$. We omit the subscript if the matroid is clear from the context.

Computing exact packing and covering is a hard question. Even in certain specific matroids, like the graphic matroid (see the paragraph below) this is relatively slow. In this paper, we aim for efficient algorithms with $\operatorname{poly}\log n$ queries per update. In particular, we give the first dynamic algorithms for approximating the fractional packing number and the fractional covering number.

▶ **Theorem 2.** *Let $\varepsilon \in (0, 1/2)$ be a parameter and let $\mathcal{M}$ be a dynamic matroid that at any moment contains at most n elements. If Φ is upper bounded by $\Phi_{\max}$, we can deterministically maintain a $(1 \pm \varepsilon)$-approximation of the fractional packing number with $O(\Phi_{\max}^2 \cdot \varepsilon^{-4} \cdot \log^3 n)$ worst-case rank-queries per update or $O(\Phi_{\max} \cdot \varepsilon^{-4} \cdot \log^3 n)$ amortized rank-queries per update.*

Since we give the first dynamic algorithm, our result can only be compared to using deterministic static algorithms to recompute the fractional packing number from scratch after every update. Using the state of the art by Chekuri and Quanrud [11], we get $\tilde{O}(n\Phi_{\max}/\varepsilon^2)$ queries per update.[5] Hence, our algorithm provides an exponential improvement in terms of n.

▶ **Theorem 3.** *Let $\varepsilon \in (0, 1/2)$ be a parameter and let $\mathcal{M}$ be a dynamic matroid that at any moment contains at most n elements. If β is upper bounded by $\beta_{\max}$, we can deterministically maintain a $(1 \pm \varepsilon)$-approximation of the fractional covering number with $O(\beta_{\max}^2 \cdot \varepsilon^{-4} \cdot \log^3 n)$ worst-case rank-queries per update.*

Again, there are no preexisting dynamic algorithms to compare our result to. Even static deterministic base covering is not as well studied. Quanrud [54] conjectures that the techniques from Chekuri and Quanrud [11] extend to deterministic approximate base covering in $\tilde{O}(n\beta_{\max}/\varepsilon^2)$ queries, which would transfer to $\tilde{O}(n\beta_{\max}/\varepsilon^2)$ queries per update.

Using randomness, we obtain an algorithm independent of the covering number against an oblivious adversary.

▶ **Theorem 4.** *Let $\varepsilon \in (0, 1/2)$ be a parameter and let $\mathcal{M}$ be a dynamic matroid that at any moment contains at most n elements. There exists a fully dynamic algorithm that maintains a $(1 \pm \varepsilon)$-approximation of the fractional covering number with $O(\log^6 n/\varepsilon^8)$ worst-case rank-queries per update. The algorithm is correct with high probability against an oblivious adversary.*

Once again, we compare to recomputing from scratch after every update using a static algorithm and obtain an exponential improvement. The state of the art for computing the static fractional covering number is by Quanrud [54] and would result in $\tilde{O}(n + \mathrm{rk}(E)/\varepsilon^3)$ queries per update. The same technique cannot be applied in the packing case to remove the dependence on $\Phi_{\max}$, as we discuss in Appendix A.2.

Packing and Covering in the Graphic Matroid. Dynamic tree-packing has been implicitly studied by Thorup [59]. This paper uses dynamic tree-packing due to its relation to min-cut. However, it implies a $(1 \pm \varepsilon)$-approximation of the fractional tree packing number[6] $\Phi \leq \Phi_{\max}$ in $\tilde{O}(\Phi_{\max}^2 \log^6 |E|/\varepsilon^{-4})$ amortized update time.

De Vos and Christiansen [16] give fully dynamic algorithms for $(1 \pm \varepsilon)$-approximate arboricity with $O(\mathrm{poly}(\log |E|, \varepsilon^{-1}))$ update time against an adaptive adversary. Banerjee, Raman, and Saurabh [3] maintain the exact arboricity with $\tilde{O}(|E|)$ update time. As such, we believe that an algorithm with $\mathrm{poly}\log(|E|)$ queries per update for *exact* packing/covering for general matroids requires a breakthrough in techniques. We focus on obtaining $(1 \pm \varepsilon)$-approximate results with $\mathrm{poly}\log(|E|)$ queries per update.

[5] Chekuri and Quanrud [11] only require the weaker independence-queries. However, even using a rank-oracle there is no known algorithm with $o(n)$ queries. The same holds for the results by [11, 54] stated below.

[6] We realize that the naming conventions here overlap. Rather than disregarding the conventions completely, we write "tree-packing" for the collection of trees and "tree packing" when talking about the tree packing number.

Other Applications. The graphic matroid is one of the most studied matroids, where the above references show that dynamic base packing and covering has been studied directly. In other matroids, this is not the case. However, base packing and covering (and hence their dynamic versions) have other interesting applications. Here, we mention two: the linear matroid and the partition matroid, see e.g. [52] for the definitions and connections.

In the linear matroid, base packing corresponds to the ability to decompose a matrix into many full-rank disjoint column sets. Base covering corresponds to finding multiple sets that form a basis for the spanned vector space, which has applications in (network) coding, see, e.g., [18, 8]. In the partition matroid and generalizations thereof, base packing and covering correspond to multi-way assignment and various types of scheduling, see, e.g., [7, 42].

Moreover, base packing and covering are highly connected to Shannon switching games [57, 46].

1.1 Technical Overview

Our main technical tool are *base collections*, which we will introduce first. Then, we will sketch how we use them to obtain our packing and covering results, Theorems 2–4.

1.1.1 Base Collections

An important tool for dynamic arboricity is tree-packing. This concept has first appeared in the seminal works by Nash-Williams [51] and Tutte [62]. It has also been well studied in its relation to the minimum cut of the graph [28, 40, 61, 59, 60, 12, 15, 17, 16]. We generalize the concept of tree-packings to matroids, and call it a *base collection*, to avoid confusion with respect to packing disjoint bases.

A *base collection* $\mathcal{B}$ is a family of bases B for the matroid $\mathcal{M}$, allowing multiple occurrences of the same base. The *load* of an element $e \in E$ is defined as $L^{\mathcal{B}}(e) := |\{B \in \mathcal{B} : e \in B\}|$. The *relative load* is defined as $\ell^{\mathcal{B}}(e) = L^{\mathcal{B}}(e)/|\mathcal{B}|$. Whenever the base collection is clear from context, we omit the superscript.

Next, we want to define some "ideal" relative load. Hereto, we first prove a corollary using Edmond's theorem on base packing (Theorem 10). We show that the lowest possible maximum load in base collections relate to the base packing number.

▶ **Corollary 5.** *We have that* $\max_{\mathcal{B}} \frac{1}{\max_{e \in E} \ell^{\mathcal{B}}(e)} = \Phi_{\mathcal{M}}$.

However, for purpose of analysis, we would like a *specific* packing with this property. More precisely, we would like the *loads* corresponding to that packing. This we call the *ideal relative loads*, a generalization of the "ideal tree-packing" for the graphic matroid [59].

Ideal Relative Loads. We define the concept of ideal loads, which are hard to compute, but capture the structural properties of the graph well. First, we define the *restricted matroid* $\mathcal{M}|A$ as $\mathcal{M}|A := (A, \mathcal{I} \mid A)$, where $\mathcal{I}|A := \{X \in \mathcal{I} \mid X \subseteq A\}$. Then, we assign ideal relative loads $\ell^*(e)$ for all $e \in E$:

- Let $A_0 \subset E$ be a set such that $\frac{|A_0|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A_0})} = \Phi_{\mathcal{M}}$.
- For all $e \in A_0$, set $\ell^*(e) = 1/\Phi_{\mathcal{M}}$.
- Recurse on the matroid $\mathcal{M}|\overline{A_0}$.

We show that this is well-defined, meaning that the resulting ideal relative load values are independent of the concrete choices of the sets $A_0 \subseteq E$ if there are multiple such sets that give $\frac{|A_0|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A_0})} = \Phi_{\mathcal{M}}$, in the full version of the Paper.

Greedy Base Collections. In this paper, we consider base collections $\mathcal{B} = \{B_1, \dots, B_k\}$ built greedily as follows: the i-th base B_i is a minimum weight base where the weights for each element are given by the relative loads induced by the base collection up until this base $\{B_1, \dots, B_{i-1}\}$.

In the graphic matroid, it has been shown [59] that a greedy tree-packing $\mathcal{T}$ of size $\Theta(\Phi \log |V|/\varepsilon^2)$ approximates the ideal packing well, resulting in every element having a relative load that differs from the ideal relative load by a small additive error as follows

$$|\ell^{\mathcal{T}}(e) - \ell^*(e)| \leq \varepsilon/\Phi \tag{1}$$

for all $e \in E$. We show a stronger statement for general matroids, for which we introduce the parameter γ to provide a better trade-off between the error and the number of greedy bases. The parameter γ satisfies $\Phi \leq \gamma \leq \beta$, which implies that $\Phi \leq \beta$ – which we have not shown yet. When considering the integer equivalents, the intuition is that $\lfloor \Phi \rfloor$ is the number of disjoint bases that fit in the matroid, and $\lceil \beta \rceil$ is the number of bases needed to cover the matroid, hence clearly $\lfloor \Phi \rfloor \leq \lceil \beta \rceil$. The fact that also $\phi \leq \beta$ is somewhat more technical, but easy to see using the structural results of Section 1.1.2.

▶ **Lemma 6.** *Let $\gamma \in [\Phi, \beta]$ and let $\mathcal{B}$ be a greedy base collection with $|\mathcal{B}| \geq 3\gamma \log n/\varepsilon^2$. Then*

$$|\ell^{\mathcal{B}}(e) - \ell^*(e)| \leq \varepsilon \ell^*(e) \tag{2}$$

for all $e \in E$ with $\ell^*(e) \geq 1/\gamma$ and

$$|\ell^{\mathcal{B}}(e) - \ell^*(e)| \leq \varepsilon/\gamma \tag{3}$$

for all $e \in E$ with $\ell^*(e) \leq 1/\gamma$.

The improvement over [59] in the special case of the graphic matroid can be seen as follows: Equation (1) can be reformulated, by picking ε accordingly, to obtain $|\ell^{\mathcal{B}}(e) - \ell^*(e)| \leq \varepsilon/\gamma$. However, it needs $|\mathcal{B}| = \Theta\left(\frac{\gamma^2}{\Phi} \log n/\varepsilon^2\right)$ bases. That is, it has a *quadratic* dependence on γ, where we obtain a *linear* dependence.

In other words, Thorup [59] showed that the values are an *additive approximation*, since the error is independent of $\ell^*(e)$. We show a *multiplicative approximation*, since the error depends linearly on $\ell^*(e)$. For our application, the latter leads to stronger results. To be precise, Lemma 6 shows that a collection of at least $3\beta \log n/\varepsilon^2$ bases gives a $(1 \pm \varepsilon)$-approximation of the covering number (using Theorem 9). A generalization of Thorup's version to matroids would give $|\mathcal{B}| = \Theta\left(\frac{\beta^2}{\Phi} \log n/\varepsilon^2\right)$.

The proof of Lemma 6 utilizes a technique by Young [66]. We consider a distribution of bases, such that picking the bases for a base collection randomly from this distribution would result in the relative loads being ideal in expectation. Then we analyze the number of times the relative load of an element does not approximate its ideal load well. We do this by replacing the randomly picked bases one after the other using a greedy approach to compute the new base. During this process, we consider pessimistic estimators for the number of violations at every step and show that they cannot be increased when a greedy base is added. This is also where the main difference to the graph case lies, as we need to consider the matroids restricted to elements or contracted to elements with certain ideal loads. The proof using the contraction is complicated by the fact that there is no notion of vertices in a general matroid. However, it still holds that any minimum weight base of $\mathcal{M}$ contains a minimum weight base for the matroid contracted to a subset.

To the best of our knowledge, the community was not aware of this property, given in Lemma 6, for the graphic matroid either. In particular, it means that we simplify the result of de Vos and Christiansen [16] for arboricity. They introduce an intricate procedure to artificially maintain $\Phi \approx \beta$ by adding virtual edges.

Dynamic Data Structure for Base Collections. Similar to the case of the graphic matroid, we show that we can maintain a greedy base collection efficiently under dynamic updates. The main building block for this is maintaining a dynamic minimum weight base, Proposition 1. We maintain this by exploiting their greedy properties, combined with a binary search, to design a simple, efficient algorithm. For details, see Section 3.

We then build a greedy base collection, by maintaining a minimum weight base for every base in the collection. We show how to handle the updates themselves and the recourse from changes in the weights due to the update. This gives the following result.

▶ **Lemma 7.** *There exists a deterministic algorithm that, given a matroid $\mathcal{M}$, maintains a greedy base collection $\mathcal{B}$ of size $|\mathcal{B}|$ with $O(|\mathcal{B}|^2 \log(|\mathcal{B}|n))$ worst-case rank-queries per update.*

1.1.2 Structural Results

Next, we consider how (ideal) base collections are related to packing and covering.

Note that, by the definition of ℓ^*, the maximum load in base collections is related to the base packing number.

▶ Remark 8. We have that $\frac{1}{\max_{e \in E} \ell^*(e)} = \Phi_{\mathcal{M}}$.

We also show that the minimum load in base collections is related to the base covering number. This generalizes the result of de Vos and Christiansen [16], who show the analogous result in the graphic matroid, where the covering number is called the arboricity. This insight is the base of the recent breakthrough by Cen et al. [9] that gives the first improvement on the running time of computing the arboricity in thirty years.

▶ **Theorem 9.** *We have that* $\frac{1}{\min_{e \in E} \ell^*(e)} = \max_{\substack{A \subseteq E \text{ s.t.} \\ A \neq \emptyset}} \frac{|A|}{\mathrm{rk}(A)}$.

The proof follows the same lines as [16], but has some more subtleties. To see this, we recall that the covering number is defined as

$$\beta_{\mathcal{M}} := \max_{\substack{A \subseteq E \text{ s.t.} \\ A \neq \emptyset}} \frac{|A|}{\mathrm{rk}(A)}.$$

For the special case of graphic matroids, the arboricity of a graph $G = (V, E)$ is defined as $\max_{\substack{S \subseteq V \text{ s.t.} \\ |S|>1}} \frac{|E(S)|}{|S|-1}$, where $E(S)$ is the set of edges of $G[S]$. The fact that the denominator of the equation changes from the size of a set to the rank of a set impacts the proof. The reason is that $|S \cup T| = |S| + |T|$ but we do not always have $\mathrm{rk}(A \cup B) \neq \mathrm{rk}(A) + \mathrm{rk}(B)$. We show that the *submodularity* of the rank function (see, e.g, [56]) suffices, i.e.,

$$\mathrm{rk}(A \cup B) + \mathrm{rk}(A \cap B) \leq \mathrm{rk}(A) + \mathrm{rk}(B).$$

1.1.3 Dynamic Packing and Covering

Using the results thus far, the worst-case result for dynamic base packing, Theorem 2, follows quite immediately: using Remark 8, we know that the packing number can be expressed in terms of the ideal relative loads. We can approximate this by a greedy base collection, see Lemma 6 and for more details refer to the proof of Theorem 2 in Appendix A.1. And finally, we know how to maintain this under dynamic updates using Lemma 7.

The amortized result in Theorem 2 uses the fact that an element is not contained in every base of the collection, and hence a better recourse argument is possible. This follows the same lines of argument as the case of the graphic matroid [16] but has some nuances. We want to highlight a special case: an insertion can *decrease* Φ (similarly, a deletion can *increase* Φ). Note that in graphs, this does not appear: such updates change the graph from disconnected to connected and the packing number of a disconnected graph is 0. In a matroid, the addition of this element needs to increase the overall rank, the packing number is not 0 and e is part of *any* base in $\mathcal{M}$ after the update. This breaks some of the argumentation for graphic matroids. In Appendix A.1 we show how to handle such technicalities efficiently.

The deterministic result for dynamic base covering, Theorem 3, is obtained in a similar manner. The number of worst-case rank-queries depends on the value known to upper bound the covering number at any point in time. To remove this dependency we make use of a sampling technique.

Sampling. We use *uniform sampling*, where every element is sampled with equal probability. This a standard technique in designing (graph) algorithms (see e.g. [47]). The approach is similar to the sampling in multi-graphs when maintaining the arboricity [16]. Although the sampling itself is the same as for multi-graphs – it is uniform sampling over the elements – the analysis is more involved for matroids.

Suppose we know (an approximation of) β. The idea is to sample with $p = \frac{\log n}{\varepsilon^2 \beta}$, such that the covering number in the resulting sampled matroid will be $\Theta(\log n/\varepsilon^2)$ with high probability. A constant approximation of β suffices, which we can get from the approximation before the update. In total, we maintain $\log n$ copies of the algorithm: one for each possible estimate of β. The $\log n/\varepsilon^2$ term here stems from the use of Chernoff bounds.

Given this uniform sampling probability p, we consider the expressions $\frac{|S|}{\mathrm{rk}(S)}$ for each set S. Consider a set S that maximizes this – the case of sets with smaller values is omitted in this overview for brevity, see Theorem 4 for details. We show that their value will remain between $(1-\varepsilon)p\beta$ and $(1+\varepsilon)p\beta$ with high probability, where β is the covering number in the original matroid. Obviously, the size of the set, $|S|$, behaves accordingly. The hard part is showing that the rank of the set, $\mathrm{rk}(S)$, behaves well under sampling, i.e., does not change by more than a factor $1 \pm \varepsilon$ with our choice of p. For graphic matroids, the expression simplifies by $\mathrm{rk}(S) = \#\text{vertices in } S$, which is clearly not affected by sampling edges. Hence these complications only occur for general matroids.

To investigate this, we first fix a rank r and consider all sets S of $\mathrm{rk}(S) = r$. When inspecting $\frac{|S|}{\mathrm{rk}(S)}$, we remark for our applications that we are only interested in the *maximum* among such sets, which restricts the number of possible sets to n^r. This allows us to use a union bound to obtain our result. For more details, see Appendix A.1

For the case of base packing a similar approach cannot be used. Again, the goal is to show how $\frac{|S|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{S})}$ behaves under uniform sampling. As before $|S|$ can be bounded by a Chernoff bound to be $(1 \pm \varepsilon)p|S|$. However, with the techniques known to us, it seems hard to show that $\mathrm{rk}(E) - \mathrm{rk}(\overline{S})$ changes by at most a factor $(1 \pm \varepsilon)$ *with high probability*. It remains an open question to obtain a dynamic base packing algorithm that updates independent of the packing number.

For graphic matroids, [16] show a trick against *adaptive adversaries*, where each vertex has ownership over some of the edges. Whenever one of its edges is affected by an update, it resamples all its edges. It turns out that this is strong enough to obtain results against adaptive adversaries. For general matroids, it is unclear how to bucket the elements in a way such that resampling one bucket every update protects against an adaptive adversary. It remains an open question how to design efficient algorithms in this case.

Independent Work

The concurrent work of Arkhipov and Kolmogorov [2] also obtains two of our results: Lemma 6 for the special case that $\gamma = \beta$ and Theorem 9. They use these results to show that they can maintain an approximation to the density of a graph by packing pseudoforests.

1.2 Organization

First, we show the structural results on base packing, base covering, and their connection to base collections in Section 2. Then, in Section 3, we show how to maintain a minimum weight base dynamically. Finally, in Appendix A, we show that dynamic greedy base collections can be used to obtain the dynamic packing and covering results. In the full version of the paper, we show that greedy base collections approximate the ideal base collections and how to maintain them under updates.

2 Packing, Covering, and Base Collections

In this section, we provide some structural results regarding base packing and the relationship to base collections. For the corresponding results for base covering we refer to the full version of the paper. These results are the foundation for the algorithms of Appendix A.

2.1 Base Packing

First, we examine how the largest possible relative load of any element in all possible base collections relates to the packing number. Recall that the following theorem gives the integral packing number of a matroid.

▶ **Theorem 10** ([20]). *A matroid $\mathcal{M}$ can pack k disjoint bases if and only if for every subset $A \subseteq E$ we have $|A| \geq k(\mathrm{rk}(E) - \mathrm{rk}(\overline{A}))$.*

To relate this to the graphic matroid, let $\mathcal{P}$ be a vertex partition of a graph $G = (V, E)$. Then the value $\frac{|A|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A})}$ corresponds to $\frac{|E(G/\mathcal{P})|}{|\mathcal{P}|-1}$, called the *partition value* of $\mathcal{P}$.

Next, we use this theorem to show a duality between this value and base collection.

▶ **Corollary 5.** *We have that* $\max_{\mathcal{B}} \frac{1}{\max_{e \in E} \ell^{\mathcal{B}}(e)} = \Phi_{\mathcal{M}}$.

Proof. We prove the two inequalities "$\leq$" and "$\geq$", starting with the former, which follows directly without using Theorem 10.

"$\leq$". Let $A^* \subseteq E$ be such that $\frac{|A^*|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A^*})} = \min_{\substack{A \subseteq E \text{ s.t.} \\ \mathrm{rk}(\overline{A}) < \mathrm{rk}(E)}} \frac{|A|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A})}$. Now we note that any base B needs to contain $\mathrm{rk}(E) - \mathrm{rk}(\overline{A^*})$ elements from A^*. So for any base collection $\mathcal{B}$, we have $\sum_{e \in A^*} \ell^{\mathcal{B}}(e) \geq \mathrm{rk}(E) - \mathrm{rk}(\overline{A^*})$. This means that on average for $e \in A^*$ we have $\ell^{\mathcal{B}}(e) \geq \frac{\mathrm{rk}(E)-\mathrm{rk}(\overline{A^*})}{|A^*|}$. In particular $\max_{e \in E} \ell^{\mathcal{B}}(e) \geq \max_{e \in A^*} \ell^{\mathcal{B}}(e) \geq \frac{\mathrm{rk}(E)-\mathrm{rk}(\overline{A^*})}{|A^*|}$. Equivalently, we get that for any base collection $\mathcal{B}$

$$\frac{1}{\max_{e\in E}\ell^{\mathcal{B}}(e)} \leq \frac{|A^*|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A^*})}.$$

Now we see that

$$\max_{\mathcal{B}} \frac{1}{\max_{e\in E}\ell^{\mathcal{B}}(e)} \leq \frac{|A^*|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A^*})} = \min_{\substack{A\subseteq E \text{ s.t.}\\ \mathrm{rk}(\overline{A})<\mathrm{rk}(E)}} \frac{|A|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A})}.$$

"$\geq$". W.l.o.g., we assume that $\min_{\substack{A\subseteq E \text{ s.t.}\\ \mathrm{rk}(\overline{A})<\mathrm{rk}(E)}} \frac{|A|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A})}$ is an integer. This can be achieved by copying each element in the matroid $\mathrm{rk}(E)!$ times, since $\mathrm{rk}(E)-\mathrm{rk}(\overline{A^*}) \in \{1,2,\dots \mathrm{rk}(E)\}$. This has no impact on the analysis as it blows up the values on the left and right equally.

Now suppose that $\min_{\substack{A\subseteq E \text{ s.t.}\\ \mathrm{rk}(\overline{A})<\mathrm{rk}(E)}} \frac{|A|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A})} = k$, by Theorem 10 this means that we can pack k disjoint bases. Let $\mathcal{B}$ be the base collection consisting of these k bases. Then for each element $e \in B \in \mathcal{B}$, we have $\ell^{\mathcal{B}}(e) = 1/k$. For other elements, we have $\ell^{\mathcal{B}}(e) = 0$. Hence we have $\max_{e\in E}\ell^{\mathcal{B}}(e) = 1/k$. Equivalently

$$\frac{1}{\max_{e\in E}\ell^{\mathcal{B}}(e)} = k = \min_{\substack{A\subseteq E \text{ s.t.}\\ \mathrm{rk}(\overline{A})<\mathrm{rk}(E)}} \frac{|A|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A})}.$$

Hence clearly

$$\max_{\mathcal{B}} \frac{1}{\max_{e\in E}\ell^{\mathcal{B}}(e)} \geq k = \min_{\substack{A\subseteq E \text{ s.t.}\\ \mathrm{rk}(\overline{A})<\mathrm{rk}(E)}} \frac{|A|}{\mathrm{rk}(E)-\mathrm{rk}(\overline{A})}. \quad \blacktriangleleft$$

However, for purpose of analysis, we would like a *specific* packing with this property. More precisely, we would like the *loads* corresponding to that packing. Hence, we use the ideal relative loads.

2.2 Base Covering

As shown in [21], the number of bases that are needed to cover all elements of e are given by

$$\left\lceil \max_{\substack{A\subseteq E \text{ s.t.}\\ A\neq\emptyset}} \frac{|A|}{\mathrm{rk}(A)} \right\rceil.$$

To obtain a similar result as for base packing, we relate the optimal base collection to the covering number. This generalizes the arboricity result of de Vos and Christiansen [16, Theorem 25], which is the statement in the special case of graphic matroids.

▶ **Theorem 9.** *We have that* $\frac{1}{\min_{e\in E}\ell^*(e)} = \max_{\substack{A\subseteq E \text{ s.t.}\\ A\neq\emptyset}} \frac{|A|}{\mathrm{rk}(A)}$.

Proof. Let $E_i := E \setminus \bigcup_{j<i} A_j$ and let $\mathcal{M}_i = \mathcal{M}|E_i$ for all recursion levels denoted by the indices $i \in I$. First, we note that $1/\min_e \ell^*(e) = \max_{i\in I} \Phi_{\mathcal{M}_i}$ over all recursion levels i. Let j be the last iteration in which $\Phi_{\mathcal{M}_i}$ is maximized. Since the Φ-values are non-decreasing (see the full version for the proof) this also has to be the last level of the recursion and therefore $E_{j+1} = E_j \setminus A_j = \emptyset$ and $\mathrm{rk}(E_j) = \mathrm{rk}(A_j)$. We get

$$1/\min_e \ell^*(e) = \frac{|A_j|}{\mathrm{rk}(E_j)-\mathrm{rk}(E_j\setminus A_j)} = \frac{|A_j|}{\mathrm{rk}(A_j)} \leq \max_{\substack{A\subseteq E \text{ s.t.}\\ A\neq\emptyset}} \frac{|Y|}{\mathrm{rk}(Y)}.$$

Now, we go on to show $1/\min_e \ell^*(e) \geq \frac{|Y|}{\operatorname{rk}(Y)}$ for any subset $Y \subseteq E$. Note that the recursively defined sets A_i for $i \in I$ are disjoint and that $\bigcup_{i\in I} A_i = E$. Hence, we get

$$|Y| = \sum_{i\in I} |A_i \cap Y|.$$

Next, we want to bound $|Y|$ by $\sum_{i\in I} \Phi_{\mathcal{M}_i} \operatorname{rk}(A_i \cap Y)$ where it remains to show the inequality $|A_i \cap Y| \leq \Phi_{\mathcal{M}_i} \operatorname{rk}(A_i \cap Y)$ for any $i \in I$. Assume towards a contradiction that there is a level $k \in I$ for which the inequality does not hold. In the following let $\overline{A_i} := E_i \setminus A_i$ for each $i \in I$. Now, consider the value

$$\Phi' = \frac{|A_k \setminus Y|}{\operatorname{rk}(E_k) - \operatorname{rk}(\overline{A_k} \cup (A_k \cap Y))} = \frac{|A_k| - |A_k \cap Y|}{\operatorname{rk}(E_k) - \operatorname{rk}(\overline{A_k} \cup (A_k \cap Y))}.$$

Since the rank function is submodular we get

$$\Phi' \leq \frac{|A_k| - |A_k \cap Y|}{\operatorname{rk}(E_k) - \operatorname{rk}(\overline{A_k}) - \operatorname{rk}(A_k \cap Y)}.$$

By our assumption that $|A_k \cap Y| > \Phi_{\mathcal{M}_k} \operatorname{rk}(A_k \cap Y)$ and the definition of $\Phi_{\mathcal{M}_k}$ we further get

$$\Phi' < \frac{|A_k| - \Phi_{\mathcal{M}_k} \operatorname{rk}(A_k \cap Y)}{\operatorname{rk}(E_k) - \operatorname{rk}(\overline{A_k}) - \operatorname{rk}(A_k \cap Y)} = \frac{\Phi_{\mathcal{M}_k}(\operatorname{rk}(E_k) - \operatorname{rk}(\overline{A_k})) - \Phi_{\mathcal{M}_k} \operatorname{rk}(A_k \cap Y)}{\operatorname{rk}(E_k) - \operatorname{rk}(\overline{A_k}) - \operatorname{rk}(A_k \cap Y)}.$$

This contradicts the choice of A_k, as picking $A_k \setminus Y$ in recursion level k would result in a smaller Φ-value. Next, we show that $\sum_{i\in I} \operatorname{rk}(A_i \cap Y) \leq \operatorname{rk}(Y)$. Note that for every $i \in I$ with $i \neq 0$ we have $E_i = \overline{A_{i-1}}$. Hence, for the last level of recursion $i_{\max}$, we get

$$\sum_{i\in I} \operatorname{rk}(E_i \cap Y) - \operatorname{rk}(\overline{A_i} \cap Y) = \left(\sum_{i\in I, i<i_{\max}} \operatorname{rk}(E_i \cap Y) - \operatorname{rk}(E_{i+1} \cap Y) \right) + \operatorname{rk}(E_{i_{\max}} \cap Y) - \operatorname{rk}(\emptyset) \leq \operatorname{rk}(Y)$$

Now it remains to show that $\operatorname{rk}(A_i \cap Y) \leq \operatorname{rk}(E_i \cap Y) - \operatorname{rk}(\overline{A_i} \cap Y)$ for all $i \in I$. In the following, consider an arbitrary $i \in I$. Note that if $\Phi_{\mathcal{M}_i}$ is an integer then

$$\operatorname{rk}(A_i \cap Y) \leq \frac{|A_i \cap Y|}{\Phi_{\mathcal{M}_i}} \tag{4}$$

since the matroid $\mathcal{M}_i$ contains $\Phi_{\mathcal{M}_i}$ disjoint bases. Hence, every base of $\mathcal{M}_i|(A_i \cap Y)$ contains at most $|A_i \cap Y|/\Phi_{\mathcal{M}_i}$ many elements. Next, we want to show

$$\frac{|A_i \cap Y|}{\operatorname{rk}(E_i \cap Y) - \operatorname{rk}(\overline{A_i} \cap Y)} \leq \Phi_{\mathcal{M}_i}. \tag{5}$$

Assume the inequality did not hold. Similar to the approach above, consider the value

$$\Phi'_{\mathcal{M}_i} = \frac{|A_i \setminus Y|}{\operatorname{rk}(E_i) - \operatorname{rk}(\overline{A_i} \cup (A_i \cap Y))} = \frac{\Phi_{\mathcal{M}_i}(\operatorname{rk}(E_i) - \operatorname{rk}(\overline{A_i})) - |A_i \cap Y|}{\operatorname{rk}(E_i) - \operatorname{rk}(\overline{A_i} \cup (A_i \cap Y))}.$$

By the assumption we get

$$\Phi'_{\mathcal{M}_i} < \frac{\Phi_{\mathcal{M}_i}(\operatorname{rk}(E_i) - \operatorname{rk}(\overline{A_i}) - \operatorname{rk}(E_i \cap Y) - \operatorname{rk}(\overline{A_i} \cap Y))}{\operatorname{rk}(E_i) - \operatorname{rk}(\overline{A_i} \cup (A_i \cap Y))}.$$

Now, in order to get $\Phi'_{\mathcal{M}_i} < \Phi_{\mathcal{M}_i}$ we need

$$\mathrm{rk}(\overline{A_i} \cup (A_i \cap Y)) \leq \mathrm{rk}(\overline{A_i}) + \mathrm{rk}(E_i \cap Y) - \mathrm{rk}(\overline{A_i} \cap Y).$$

This inequality holds since we have

$$\begin{aligned}\mathrm{rk}(\overline{A_i} \cup (A_i \cap Y)) + \mathrm{rk}(\overline{A_i} \cap Y) &= \mathrm{rk}(\overline{A_i} \cup (E_i \cap Y)) + \mathrm{rk}(\overline{A_i} \cap (E_i \cap Y)) \\ &\leq \mathrm{rk}(\overline{A_i}) + \mathrm{rk}(E_i \cap Y)\end{aligned}$$

where the last inequality is due to the submodularity of the rank function. Now we have that (4) and (5) hold for any $i \in I$. We can again assume w.l.o.g. that $\Phi_{\mathcal{M}_i}$ is an integer using the same technique as described in the proof of Corollary 5, as the rank remains unchanged. Putting (4) and (5) together, we get the desired property which directly gives the following lower bound on the rank of Y

$$\sum_{i \in I} \mathrm{rk}(A_i \cap Y) \leq \mathrm{rk}(Y).$$

Finally, we get:

$$\begin{aligned}\frac{|Y|}{\mathrm{rk}(Y)} &\leq \frac{\sum_{i \in I} \Phi_{\mathcal{M}_i} \mathrm{rk}(A_i \cap Y)}{\sum_{i \in I} \mathrm{rk}(A_i \cap Y)} \\ &\leq \max_{i \in I} \Phi_{\mathcal{M}_i} \\ &= \frac{1}{\min_e \ell^*(e)}.\end{aligned}$$

◀

3 Maintaining a Dynamic Minimum Weight Base

In this section, we give an algorithm to maintain a minimum weight base.

Minimum Weight Base. Given a universe E, and weight function $w\colon E \to [1, \ldots, W]$, a *minimum weight base* is a base B of minimum total weight $w(B) = \sum_{e \in B} w(e)$. Note that if the weights are unique, the minimum weight base is unique. For simplicity, we would like the weights to be unique without increasing the maximum weight. The minimum weight base of a matroid can be computed using a simple greedy approach as shown in [29, 22]. Here, the exact weights of the elements are never taken into account. It suffices to consider the order on the elements implied by their weights. Thus we can essentially re-weight the elements at each update, assigning each element a unique weight in $[1, \ldots, n]$ respecting, firstly, the weights given by w, and secondly, in case of equal weight, the lexicographic order of the unique ids.[7] For a *dynamic minimum weight base*, the goal is to maintain a minimum weight base under element insertions and deletions. More generally, we can also allow for weight increases or decreases, but this can also be modeled by deleting the element and inserting it again with the new weight.

By inverting the order, we an algorithm for the minimum weight base problem also solves the *maximum* weight base problem. This is equivalent to the maximum independent set problem, as bases are maximal by definition. However, note that this does *not* correspond to the maximum independent set problem in graphs: independent sets in graphs, i.e., a set of vertices where no two are adjacent, in fact do not form a matroid. The nomenclature was developed independently – no pun intended.

[7] Note that after each update, many weights can change. However, this does not require additional queries, so does not affect our query complexity.

Preliminaries. The *span* of a set $A \subseteq E$ is denoted by $\text{span}(A)$ and is defined as the set $\{e \in E | \, \text{rk}(A) = \text{rk}(A + e)\}$. We say that the set A spans element $e \in E$ or that A spans a set $B \subseteq E$ if $e \in \text{span}(A)$ or $B \subseteq \text{span}(A)$, respectively. The set A always spans itself as well as all other elements that can be added to A without increasing the size of the maximal independent subset.

A *circuit* $C \subseteq E$ is an inclusion-wise minimal dependent set of elements, i.e., $C \notin \mathcal{I}$ and for every $x \in C$, $C \setminus \{x\} \in \mathcal{I}$. The unique circuit in $S + e$ for $S \in \mathcal{I}$ and $S + e \notin \mathcal{I}$ is denoted by $C(S + e)$. We have the following lemma concerning circuits.

▶ **Lemma 11** ([56]). *Let C and C' be circuits. If $x \in C \cap C'$ and $y \in C \setminus C'$, then there exists a circuit in $(C \cup C') \setminus \{x\}$ containing y.*

Algorithm. We base our algorithm on two fundamental properties of minimum spanning trees (see e.g., [43]), which also hold for minimum weight bases in matroids.

- The *cycle property*: For any cycle, the edge with the largest weight cannot be in the MST.
- The *cut property*: For any cut, the edge with the smallest weight that crosses the cut is in the MST.

In the proof below, we implicitly use and prove these properties for matroids. Note that in this context, a "cut" is a minimal set of elements that reduces the rank.

▶ **Proposition 1.** *There exists a deterministic algorithm that, given a dynamic matroid $\mathcal{M}$ with weight function $w \colon E \to [1, \ldots, W]$, maintains a minimum weight base, where each update uses $O(\log n)$ rank-queries.*

Proof. Without loss of generality, we assume that the weights are unique. We can do tie-breaking of elements with equal weight by lexicographic order.

We introduce the following notation: the sets $B_{\leq t} := \{f \in B : w(f) \leq t\}$ and similarly $E_{\leq t} := \{f \in E : w(f) \leq t\}$. Let us first consider element insertion. If the insertion increases the rank of the matroid, we can simply add the new element to B to get a minimum weight base for the new matroid. Otherwise, suppose e is inserted, finding the circuit in $B + e$ can be costly. However, we just need to find the element f of highest weight on this circuit (possibly e itself if it was inserted with a high weight). Formally, we do this as follows.

- Find min t s.t. $\text{rk}(B_{\leq t}) = \text{rk}(B_{\leq t} + e)$.
- Output the unique element f with weight t.

Note that now f is the element of highest weight on the circuit of $B + e$. Next, we show that we can use this method to maintain a minimum base correctly over a sequence of insertions that do not increase the rank of the matroid. We assume that before the insertion B was a base with minimum weight. Note that all subsets of E that were previously independent, are still independent after the addition of e to the matroid. The algorithm outputs the element f. In case (a) $w(f) > w(e)$ the maintained base B' is set to $B - f + e$, otherwise in case (b) $w(f) < w(e)$ $B' = B$ remains unchanged. Now assume towards a contradiction that after the insertion there is another base B'' such that $w(B'') < w(B')$. Note that $e \in B''$, otherwise B would not have been a minimum weight base. Further, there is an element e' in $C(B + e) - e$ with $B'' - e + e' \in \mathcal{I}$. To see this, let E_c be the set of elements of $c - B''$ for a circuit c. If adding an element of $E_{C(B+e)}$ to B'' results in a circuit containing e, we are done. Otherwise, we know that $e \notin C(B'' + e'')$ for any $e'' \in E_{C(B+e)}$. Considering such an element e_1 from $E_{C(B+e)}$, by Lemma 11 we know that there is a circuit $c_1 \subseteq (C(B'' + e_1) + C(B + e)) - e_1$ with $e \in c_1$. If $|E_{c_1}| = 1$ we are done, otherwise (since $c_1 \subseteq B'' + E_{C(B+e)} - e_1$) we can continue by removing the next element $e_2 \in c_1 \setminus B''$ in

the same way to find a circuit $c_2 \subseteq (C(B'' + e_2) + c_1) - e_2 \subseteq B'' + E_{c_1} - e_2$ with $e \in c_2$. We can continue in the same way, further restricting the number of elements in the next circuit that are not from B'' until we get a circuit $c' \subseteq B'' + e_j$ with $e \in c'$. Hence, for the element $e_j \in C(B + e) - e$ we have $B'' - e + e_j$ is a base even before the insertion of e. Since $e_j \in C(B + e)$, we have $w(e_j) \leq w(f)$. Now, in case (a) where the maintained base was set to $B - f + e$ we get

$$\begin{aligned} w(B'' - e + e_j) &= w(B'') - w(e) + w(e_j) < w(B - f + e) - w(e) + w(e_j) \\ &= w(B) - w(f) + w(e_j) \leq w(B). \end{aligned}$$

In case (b) where B remained unchanged we have

$$w(B'' - e + e_j) = w(B'') - w(e) + w(e_j) < w(B),$$

where the last inequality is due to $w(e_j) \leq w(f) < w(e)$. In either case, this contradicts the assumption that B was a base of minimum weight before the insertion.

Next, we consider the deletion of an element e, which restricts the matroid to $E - e$. If the deletion decreases the rank of the matroid, the element cannot be replaced and $B \setminus \{e\}$ gives the new minimum weight base. Otherwise, we need to find an element f of minimum weight such that $B \cup \{f\}$ contains a circuit in the matroid before the deletion. Similar to an insertion, the approach is as follows.

- Find min t s.t. $\mathrm{rk}(B - e + E_{\leq t}) = \mathrm{rk}(B - e) + 1$.
- Output the unique element f with weight t.

Again, assume that before the insertion B was a base with minimum weight and also that $e \in B$ otherwise B stays a minimum weight base. Then f is the element of smallest weight that is not already spanned by $B - e$ (if no such element exists, e was part of all bases before the deletion and $B - e$ is still the minimum weight base). It remains to argue that $B' = B - e + f$ has the smallest total weight after the deletion. Assume there was another base B'' for the restricted matroid with $w(B'') < w(B - e + f)$. Then $B'' + e$ contains a cycle C and there is an element $e' \in C$ that is not in B. Note that $\{b \in B | w(b) < w(e)\} = \{b \in B'' | w(b) < w(e)\}$ and therefore $w(e') > w(e)$. Now we consider the base for the original matroid $B'' + e - e'$. But then B would not have been a minimum weight base before the deletion, since

$$w(B''+e-e') = w(B'')+w(e)-w(e') < w(B')+w(e)-w(e') \leq w(B')+w(e)-w(f) = w(B)$$

where the last inequality follows from $w(f) \leq w(e')$, which remains to be argued. We do so by showing that e' is also an element along with f that is not spanned by $B - e$. Assume towards a contradiction that $B - e$ spans e'. Then there is a circuit $C = C(B - e + e')$ where e' has the highest weight out of all elements on the circuit. Since the circuit c also exists in the matroid after the deletion of e, B'' cannot be a minimum weight base after the deletion, as it contains e'. This is because there is an element of smaller weight on C that can be exchanged for e' in B''. Similar to the argument in the case of an insertion, we consider the circuit $C(B'' + d)$ for some $d \in C$. If e' is on that circuit we have found the element to exchange e' with. Otherwise there is a circuit C' in $(C + C(B'' + d)) - d$ that contains e'. If C' contains only one element not from B'', we are done. Otherwise we can find a new circuit containing e' with less elements from $C - B''$ as described above until there is only one element left. Hence, e' is not spanned by $B - e$ and B' is the new minimum weight base.

For the number of queries, note that we can find the minimum with a binary search over the space of all adjusted weights $[1, \ldots, n]$. ◀

We use Proposition 1 to maintain a greedy base collection under updates. The details can be found in the full version. Then we use dynamic base collections to obtain the dynamic packing and covering results in Appendix A.

References

1 David Alberts and Monika Rauch Henzinger. Average-case analysis of dynamic graph algorithms. *Algorithmica*, 20(1):31–60, 1998. Announced at SODA'95. doi:10.1007/PL00009186.

2 Pavel Arkhipov and Vladimir Kolmogorov. Greedy matroid base packings with applications to dynamic graph density and orientations. *arXiv preprint arXiv:2511.13205*, 2025. doi:10.48550/arXiv.2511.13205.

3 Niranka Banerjee, Venkatesh Raman, and Saket Saurabh. Fully dynamic arboricity maintenance. *Theor. Comput. Sci.*, 822:1–14, 2020. Announced at COCOON 2019. doi:10.1016/J.TCS.2020.04.010.

4 Kiarash Banihashem, Leyla Biabani, Samira Goudarzi, MohammadTaghi Hajiaghayi, Peyman Jabbarzade, and Morteza Monemizadeh. Dynamic algorithms for matroid submodular maximization. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 3485–3533. SIAM, 2024. doi:10.1137/1.9781611977912.125.

5 Joakim Blikstad. Breaking o(nr) for matroid intersection. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 31:1–31:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ICALP.2021.31.

6 Joakim Blikstad, Sagnik Mukhopadhyay, Danupon Nanongkai, and Ta-Wei Tu. Fast algorithms via dynamic-oracle matroids. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1229–1242. ACM, 2023. doi:10.1145/3564246.3585219.

7 Rainer E Burkard and En-Yu Yao. Constrained partitioning problems. *Discrete applied mathematics*, 28(1):21–34, 1990. doi:10.1016/0166-218X(90)90091-P.

8 Peter G Casazza, Gitta Kutyniok, and Friedrich Philipp. Introduction to finite frame theory. *Finite frames: theory and applications*, pages 1–53, 2013.

9 Ruoxu Cen, Henry L. Fleischmann, George Z. Li, Jason Li, and Debmalya Panigrahi. Fast algorithms for graph arboricity and related problems. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025*, 2025. arXiv:2507.15598.

10 Karthekeyan Chandrasekaran, Chandra Chekuri, and Weihao Zhu. Online disjoint spanning trees and polymatroid bases. In Keren Censor-Hillel, Fabrizio Grandoni, Joël Ouaknine, and Gabriele Puppis, editors, *52nd International Colloquium on Automata, Languages, and Programming, ICALP 2025, July 8-11, 2025, Aarhus, Denmark*, volume 334 of *LIPIcs*, pages 44:1–44:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.ICALP.2025.44.

11 Chandra Chekuri and Kent Quanrud. Near-linear time approximation schemes for some implicit fractional packing problems. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 801–820. SIAM, 2017. doi:10.1137/1.9781611974782.51.

12 Chandra Chekuri, Kent Quanrud, and Chao Xu. LP relaxation and tree packing for minimum k-cuts. In Jeremy T. Fineman and Michael Mitzenmacher, editors, *2nd Symposium on Simplicity in Algorithms, SOSA 2019, January 8-9, 2019, San Diego, CA, USA*, volume 69 of *OASIcs*, pages 7:1–7:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/OASIcs.SOSA.2019.7.

13 Xi Chen and Binghui Peng. On the complexity of dynamic submodular maximization. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 1685–1698. ACM, 2022. doi:10.1145/3519935.3519951.

14 William H. Cunningham. Improved bounds for matroid partition and intersection algorithms. *SIAM J. Comput.*, 15(4):948–957, 1986. doi:10.1137/0215066.

15 Mohit Daga, Monika Henzinger, Danupon Nanongkai, and Thatchaphol Saranurak. Distributed edge connectivity in sublinear time. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 343–354. ACM, 2019. doi:10.1145/3313276.3316346.

16 Tijn de Vos and Aleksander B. G. Christiansen. Tree-packing revisited: Faster fully dynamic min-cut and arboricity. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 700–749. SIAM, 2025. doi:10.1137/1.9781611978322.21.

17 Michal Dory, Yuval Efron, Sagnik Mukhopadhyay, and Danupon Nanongkai. Distributed weighted min-cut in nearly-optimal time. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1144–1153. ACM, 2021. doi:10.1145/3406325.3451020.

18 Randall Dougherty, Chris Freiling, and Kenneth Zeger. Networks, matroids, and non-shannon information inequalities. *IEEE Transactions on Information Theory*, 53(6):1949–1969, 2007. doi:10.1109/TIT.2007.896862.

19 J Edmonds. Submodular functions, matroids, and certain polyhedra. In *Proc. Calgary Int. Conference on Combinatorial Structures and Their Applications, 1970*. Gordon and Breach, 1970.

20 Jack Edmonds. Lehman's switching game and a theorem of tutte and nash-williams. *J. Res. Nat. Bur. Standards Sect. B*, 69:73–77, 1965.

21 Jack Edmonds. Minimum partition of a matroid into independent subsets. *J. Res. Nat. Bur. Standards Sect. B*, 69:67–72, 1965.

22 Jack Edmonds. Matroids and the greedy algorithm. *Math. Program.*, 1(1):127–136, December 1971. doi:10.1007/BF01584082.

23 David Eppstein, Zvi Galil, Giuseppe F. Italiano, and Amnon Nissenzweig. Sparsification - a technique for speeding up dynamic graph algorithms. *J. ACM*, 44(5):669–696, 1997. Announced at FOCS'92. doi:10.1145/265910.265914.

24 David Eppstein, Giuseppe F. Italiano, Roberto Tamassia, Robert Endre Tarjan, Jeffery R. Westbrook, and Moti Yung. Maintenance of a minimum spanning forest in a dynamic plane graph. *J. Algorithms*, 13(1):33–54, 1992. doi:10.1016/0196-6774(92)90004-V.

25 Greg N. Frederickson. Data structures for on-line updating of minimum spanning trees, with applications. *SIAM J. Comput.*, 14(4):781–798, 1985. doi:10.1137/0214055.

26 Greg N. Frederickson. Ambivalent data structures for dynamic 2-edge-connectivity and k smallest spanning trees. *SIAM J. Comput.*, 26(2):484–538, 1997. Announced at FOCS'91. doi:10.1137/S0097539792226825.

27 Takaaki Fujita. Matroid, ideal, ultrafilter, tangle, and so on: Reconsideration of obstruction to linear decomposition. *International Journal of Mathematics Trends and Technology (IJMTT)*, 70(7):18–29, 2024. doi:10.14445/22315373/IJMTT-V70I7P104.

28 HN Gabow. A matroid approach to finding edge connectivity and packing arborescences. *Journal of Computer and System Sciences*, 2(50):259–273, 1995. Announced at STOC 1991.

29 David Gale. Optimal assignments in an ordered set: An application of matroid theory. *Journal of Combinatorial Theory*, 4(2):176–180, 1968. doi:10.1016/S0021-9800(68)80039-0.

30 David Gibb, Bruce M. Kapron, Valerie King, and Nolan Thorn. Dynamic graph connectivity with improved worst case update time and sublinear space. *CoRR*, abs/1509.06464, 2015. arXiv:1509.06464.

31 Monika Rauch Henzinger and Valerie King. Randomized fully dynamic graph algorithms with polylogarithmic time per operation. *J. ACM*, 46(4):502–516, 1999. doi:10.1145/320211.320215.

32 Monika Rauch Henzinger and Valerie King. Maintaining minimum spanning forests in dynamic graphs. *SIAM J. Comput.*, 31(2):364–374, 2001. doi:10.1137/S0097539797327209.

33 Monika Rauch Henzinger and Mikkel Thorup. Sampling to provide or to bound: With applications to fully dynamic graph algorithms. *Random Struct. Algorithms*, 11(4):369–379, 1997. doi:10.1002/(SICI)1098-2418(199712)11:4\%3C369::AID-RSA5\%3E3.0.CO;2-X.

34 Petr Hlinený and Geoff Whittle. Matroid tree-width. *Eur. J. Comb.*, 27(7):1117–1128, 2006. doi:10.1016/J.EJC.2006.06.005.

35 Jacob Holm, Kristian de Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *J. ACM*, 48(4):723–760, 2001. Announced at STOC'98. doi:10.1145/502090.502095.

36 Shang-En Huang, Dawei Huang, Tsvi Kopelowitz, Seth Pettie, and Mikkel Thorup. Fully dynamic connectivity in o(log n(loglog n)^2) amortized expected time. *TheoretiCS*, 2, 2023. Announced at SODA'17. doi:10.46298/THEORETICS.23.6.

37 Bruce M. Kapron, Valerie King, and Ben Mountjoy. Dynamic graph connectivity in polylogarithmic worst case time. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1131–1142. SIAM, 2013. doi:10.1137/1.9781611973105.81.

38 David R. Karger. Random sampling in matroids, with applications to graph connectivity and minimum spanning trees. In *34th Annual Symposium on Foundations of Computer Science, Palo Alto, California, USA, 3-5 November 1993*, pages 84–93. IEEE Computer Society, 1993. doi:10.1109/SFCS.1993.366879.

39 David R. Karger. Random sampling and greedy sparsification for matroid optimization problems. *Math. Program.*, 82:41–81, 1998. doi:10.1007/BF01585865.

40 David R Karger. Minimum cuts in near-linear time. *Journal of the ACM (JACM)*, 47(1):46–76, 2000. Announced at STOC 1996. doi:10.1145/331605.331608.

41 Richard M. Karp, Eli Upfal, and Avi Wigderson. The complexity of parallel search. *J. Comput. Syst. Sci.*, 36(2):225–253, 1988. doi:10.1016/0022-0000(88)90027-X.

42 Yasushi Kawase, Kei Kimura, Kazuhisa Makino, and Hanna Sumita. Optimal matroid partitioning problems. *Algorithmica*, 83(6):1653–1676, 2021. doi:10.1007/S00453-021-00797-9.

43 Jon M. Kleinberg and Éva Tardos. *Algorithm design*. Addison-Wesley, 2006.

44 Donald Ervin Knuth. *Matroid partitioning*. Computer Science Department, Stanford University, 1973.

45 Eugene L Lawler. Optimal matroid intersections. In *Proc. Calgary Int. Conference on Combinatorial Structures and Their Applications, 1970*. Gordon and Breach, 1970.

46 Alfred Lehman. A solution of the shannon switching game. *Journal of the Society for Industrial and Applied Mathematics*, 12(4):687–725, 1964.

47 Andrew McGregor, David Tench, Sofya Vorotnikova, and Hoa T. Vu. Densest subgraph in dynamic graph streams. In Giuseppe F. Italiano, Giovanni Pighizzini, and Donald Sannella, editors, *Mathematical Foundations of Computer Science 2015 - 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part II*, volume 9235 of *Lecture Notes in Computer Science*, pages 472–482. Springer, 2015. doi:10.1007/978-3-662-48054-0_39.

48 Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Deletion-robust submodular maximization: Data summarization with "the right to be forgotten". In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2449–2458. PMLR, 2017. URL: http://proceedings.mlr.press/v70/mirzasoleiman17a.html.

49 Danupon Nanongkai and Thatchaphol Saranurak. Dynamic spanning forest with worst-case update time: adaptive, las vegas, and o(n$^{1/2 - \varepsilon}$)-time. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1122–1129. ACM, 2017. doi:10.1145/3055399.3055447.

50 Danupon Nanongkai, Thatchaphol Saranurak, and Christian Wulff-Nilsen. Dynamic minimum spanning forest with subpolynomial worst-case update time. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 950–961. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.92.

51 C St JA Nash-Williams. Edge-disjoint spanning trees of finite graphs. *Journal of the London Mathematical Society*, 1(1):445–450, 1961.

52 James G. Oxley. *Matroid theory*. Oxford University Press, 1992.

53 Mihai Pătraşcu and Mikkel Thorup. Planning for fast connectivity updates. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 263–271. IEEE Computer Society, 2007. doi:10.1109/FOCS.2007.54.

54 Kent Quanrud. Faster exact and approximation algorithms for packing and covering matroids via push-relabel. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 2305–2336. SIAM, 2024. doi:10.1137/1.9781611977912.82.

55 András Recski. *Matroid theory and its applications in electric network theory and in statics*, volume 6. Springer Science & Business Media, 2013.

56 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*. Springer, 2003.

57 Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948. doi:10.1002/J.1538-7305.1948.TB01338.X.

58 Mikkel Thorup. Near-optimal fully-dynamic graph connectivity. In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 343–350. ACM, 2000. doi:10.1145/335305.335345.

59 Mikkel Thorup. Fully-dynamic min-cut. *Comb.*, 27(1):91–127, 2007. Announced at STOC 2001. doi:10.1007/S00493-007-0045-2.

60 Mikkel Thorup. Minimum k-way cuts via deterministic greedy tree packing. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 159–166. ACM, 2008. doi:10.1145/1374376.1374402.

61 Mikkel Thorup and David R. Karger. Dynamic graph algorithms with applications. In Magnús M. Halldórsson, editor, *Algorithm Theory - SWAT 2000, 7th Scandinavian Workshop on Algorithm Theory, Bergen, Norway, July 5-7, 2000, Proceedings*, volume 1851 of *Lecture Notes in Computer Science*, pages 1–9. Springer, 2000. doi:10.1007/3-540-44985-X_1.

62 William Thomas Tutte. On the problem of decomposing a graph into n connected factors. *Journal of the London Mathematical Society*, 1(1):221–230, 1961.

63 Dominic JA Welsh. *Matroid theory*. Courier Corporation, 2010.

64 Christian Wulff-Nilsen. Faster deterministic fully-dynamic graph connectivity. In *Encyclopedia of Algorithms*, pages 738–741. Springer, 2016. Announced at SODA'13. doi:10.1007/978-1-4939-2864-4_569.

65 Christian Wulff-Nilsen. Fully-dynamic minimum spanning forest with improved worst-case update time. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1130–1143. ACM, 2017. doi:10.1145/3055399.3055415.

66 Neal E. Young. Randomized rounding without solving the linear program. In Kenneth L. Clarkson, editor, *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, 22-24 January 1995. San Francisco, California, USA*, pages 170–178. ACM/SIAM, 1995. URL: http://dl.acm.org/citation.cfm?id=313651.313689.

A Dynamic Packing and Covering

In this section, we combine the structural results from Section 2 with the results on (dynamic) base collections (specifically Lemma 6 and Lemma 7) to obtain efficient algorithms for base packing and base covering. Some of the theorems follow directly by maintaining a greedy base collection of a large enough size. For other results, we use additional techniques specific to the dynamic challenges at hand.

A.1 Dynamic Matroid Packing

Next, we combine our results thusfar to obtain our dynamic base packing results. By Remark 8, Lemma 6, and Lemma 7, we obtain the worst-case result almost directly. To obtain better bounds with amortization, some more work is required.

▶ **Theorem 2.** *Let $\varepsilon \in (0, 1/2)$ be a parameter and let $\mathcal{M}$ be a dynamic matroid that at any moment contains at most n elements. If Φ is upper bounded by $\Phi_{\max}$, we can deterministically maintain a $(1 \pm \varepsilon)$-approximation of the fractional packing number with $O(\Phi_{\max}^2 \cdot \varepsilon^{-4} \cdot \log^3 n)$ worst-case rank-queries per update or $O(\Phi_{\max} \cdot \varepsilon^{-4} \cdot \log^3 n)$ amortized rank-queries per update.*

Proof. We maintain a base collection $\mathcal{B}$ of $\Theta\left(\frac{\Phi_{\max} \log n}{\varepsilon^2}\right)$ greedy bases and maintain the maximum $\ell^{\mathcal{B}}(e)$ using a max-heap directly giving the approximation.

First, we argue that the maintained value is indeed a $(1 \pm \varepsilon)$-approximation of the packing number. We want to show

$$(1-\varepsilon)\Phi \leq \frac{1}{\max_{e \in E} \ell^{\mathcal{B}}(e)} \leq (1+\varepsilon)\Phi.$$

We show the left hand side, the right hand side is then analogous.

From Lemma 6 we know that if we maintain a base collection of size $\Omega(\Phi \log n/\varepsilon'^2)$, $\ell^{\mathcal{B}}(e) \geq \ell^*(e) - \frac{\varepsilon'}{\Phi}$ for all $e \in E$ for an $\varepsilon' > 0$ to be defined later, this gives

$$\max_{e \in E} \ell^{\mathcal{B}}(e) \geq \max_{e \in E} \ell^*(e) - \frac{\varepsilon'}{\Phi}.$$

So for $\varepsilon' := \varepsilon/(1+\varepsilon)$ we get

$$\frac{1}{\max_{e \in E} \ell^{\mathcal{B}}(e)} \leq \frac{1}{\max_{e \in E} \ell^*(e) - \frac{\varepsilon}{(1+\varepsilon)\Phi}} = \frac{1}{\frac{1}{\Phi} - \frac{\varepsilon}{(1+\varepsilon)\Phi}} = (1+\varepsilon)\Phi,$$

using that $\max_{e \in E} \ell^*(e) = \Phi^{-1}$, see also Remark 8. Similarly,

$$\begin{aligned}\frac{1}{\max_{e \in E} \ell^{\mathcal{B}}(e)} &\geq \frac{1}{\max_{e \in E} \ell^*(e) + \frac{\varepsilon}{(1+\varepsilon)\Phi}} = \frac{1}{\frac{1}{\Phi} + \frac{\varepsilon}{(1+\varepsilon)\Phi}} = \left(\frac{1+\varepsilon}{1+\varepsilon+\varepsilon}\right)\Phi \\ &= \left(1 - \frac{\varepsilon}{1+2\varepsilon}\right)\Phi \geq (1-\varepsilon)\Phi,\end{aligned}$$

Next, we consider the number of queries per update. As updating $\mathcal{B}$ can require $|\mathcal{B}|^2$ updates to a greedy base in $\mathcal{B}$ maintaining the base collection takes $O\left(\frac{\Phi_{\max}^2 \log^3(n)}{\varepsilon^4}\right)$ queries for any update (worst-case), see Lemma 7.

For an amortized update time, we can reduce this using a technique by [16]. We note that any element e is in

$$L^{\mathcal{B}}(e) = \ell^{\mathcal{B}}(e)|\mathcal{B}| \leq \max_{e' \in E} \ell^{\mathcal{B}}(e')|\mathcal{B}| \leq \frac{|\mathcal{B}|}{(1+\varepsilon)\Phi} \leq \frac{|\mathcal{B}|}{\Phi}$$

bases. We want to exploit this fact, together with the fact that we only need to consider the first $\Theta(\Phi \log n/\varepsilon^2)$ bases of $\mathcal{B}$ when the base packing number is Φ by Lemma 6. To exploit this, we partition $\mathcal{B}$ into buckets, where the ith bucket contains 2^i bases, with $i = 0, \ldots, \Theta(\log |\mathcal{B}|)$. Intuitively, at any point in time when the packing number is Φ and $3\Phi \log n/\varepsilon^2 \in (2^i, 2^{i+1}]$, we can process an update e in the bases of the first $i+1$ buckets efficiently – which is we all we need to output our estimate of Φ. More formally, denote Φ for the packing number before the update and Φ' for the packing number after the update.

First, we consider an insertion such that $\Phi' \geq \Phi$. We have $\Phi' \leq \Phi + 1 \leq 2\Phi$. Note that after insertion, e will be contained in at most $2 \cdot 2^{i+2}/\Phi' = \Theta(\log n/\varepsilon^2)$ bases. When e is part of some base B_j, this can lead to at most $|\mathcal{B}|$ recourse in all of $\mathcal{B}$ (see Lemma 7 for the proof). So inserting $\mathcal{B}$ takes at most $|\mathcal{B}| + \Theta(|\mathcal{B}| \log n/\varepsilon^2) = \Theta(|\mathcal{B}| \log n/\varepsilon^2)$, where the first factor $\mathcal{B}$ is due to determining which bases to insert e in.

It remains to show how to make the changes to bases in the buckets $j > i$. That is where the amortization comes in: rather than performing these updates now. We initialize a priority queue of updates for each bucket. We claim that when a bucket j becomes relevant, i.e., $3\Phi \log n/\varepsilon^2 \in (2^j, 2^{j+1}]$, we can perform all updates from the queue in $\Theta(|\mathcal{B}| \log n/\varepsilon^2)$ queries per update. Hereto, we first perform all insertions from the queue, and then all deletions.

Rather than performing one insertion at a time, we consider each base of the matroid, and perform all necessary insertions, where we prioritize the elements with a smaller weight – this is easily done by maintaining the queue as a min-heap. This means that each inserted element will only be part of $O(\log n/\varepsilon^2)$ bases, since this process is equivalent to computing the greedy bases from scratch. Each such insertion leads to a recourse of $O(|\mathcal{B}|)$. Hence in total, we use $O(|\mathcal{B}| \log n/\varepsilon^2)$ queries per insertion.

The insertions now have ensured that the packing number in the matroid is actually high enough, so each element that is deleted, is part of $O(\log n/\varepsilon^2)$ bases.

Note that the case that the update is a deletion e such that $\Phi' \leq \Phi$ is analogous.

We are left with two cases: the update is an insertion and $\Phi' < \Phi$, or the update is a deletion and $\Phi' > \Phi$. Note that for tree-packing in graphs, this does not appear: such updates change the graph from disconnected to connected and the packing number of a disconnected graph is 0.

Consider an insertion e such that $\Phi' < \Phi$. The addition of this element needs to increase the overall rank of the matroid. In this case, e is part of *any* base in $\mathcal{M}$ after the insertion. To see this, consider the insertion of an element e and let E be the set of elements before the insertion with packing number Φ. In the following, we do not need to consider sets $S \subseteq E + e$ that do not contain e since in that case we have

$$\frac{|S|}{\mathrm{rk}(E+e) - \mathrm{rk}((E+e) \setminus S)} = \frac{|S|}{\mathrm{rk}(E+e) - \mathrm{rk}((E \setminus S) + e)} \geq \frac{|S|}{\mathrm{rk}(E) - \mathrm{rk}((E) \setminus S)}.$$

This is immediately obvious in the case where $\mathrm{rk}(E+e) = \mathrm{rk}(E)$. Otherwise $\mathrm{rk}(E+e) = \mathrm{rk}(E) + 1$ but in this case $\mathrm{rk}((E \setminus S) + e) = \mathrm{rk}(E \setminus S) + 1$ as well. Hence, these sets cannot be the reason the Φ-value decreases. Now, we want to show that if it decreases, the rank of the matroid must increase. To this end, assume that $\mathrm{rk}(E+e) = \mathrm{rk}(E)$ and consider any $S \subseteq E$ with $S \neq \emptyset$. Then we have

$$\frac{|S+e|}{\mathrm{rk}(E+e) - \mathrm{rk}((E+e) \setminus (S+e))} = \frac{|S|+1}{\mathrm{rk}(E) - \mathrm{rk}((E \setminus S))} > \Phi.$$

It remains to consider the set $\{e\}$. By the definition, this cannot be the set giving the new Φ' as $\mathrm{rk}((E+e)\setminus\{e\}) = \mathrm{rk}(E) = \mathrm{rk}(E+e)$. Hence, the rank of the matroid has to increase when Φ decreases after the insertion of an element e. Further, we have that in this case we also $\Phi' = 1$ and therefore e is in every base. This is because when $\mathrm{rk}(E+e) = \mathrm{rk}(E)+1$ we have

$$\frac{|\{e\}|}{\mathrm{rk}(E+e)-\mathrm{rk}(E)} = 1.$$

For any other set S containing e we have that

$$\frac{|S+e|}{\mathrm{rk}(E+e)-\mathrm{rk}((E+e)\setminus(S+e))} = \frac{|S|+1}{\mathrm{rk}(E)-\mathrm{rk}((E\setminus S))+1} \geq 1.$$

where the last inequality follows from $|S| \geq \mathrm{rk}(S)$ and the submodularity of the rank-function.

Let E' be the new ground set $E' = E+e$ and let $\mathcal{M}'$ be the matroid after the insertion. Assume that e is not in all bases after the insertion. Then there is a set B that is a base of the matroid before and after the insertion of e. Hence, the overall rank does not change i.e. $\mathrm{rk}_{\mathcal{M}}(E) = \mathrm{rk}_{\mathcal{M}'}(E')$. Now, consider the set A that gives $\Phi' = |A|/(\mathrm{rk}_{M'}(E') - \mathrm{rk}_{\mathcal{M}'}(E'\setminus A)$. The element $e \notin A$ since otherwise we would have $\mathrm{rk}_{\mathcal{M}'}(E'\setminus A) = \mathrm{rk}_{\mathcal{M}'}(E\setminus(A-e))$ and therefore the set $A-e$ would give a smaller Φ' compared to A. Hence, $\mathrm{rk}_{\mathcal{M}'}(E'\setminus A) \geq \mathrm{rk}_{\mathcal{M}'}(E\setminus A) = \mathrm{rk}_{\mathcal{M}}(E\setminus A)$ and we further get

$$\Phi > \frac{|A|}{\mathrm{rk}_{\mathcal{M}'}(E') - \mathrm{rk}_{\mathcal{M}'}(E'\setminus A)} \geq \frac{|A|}{\mathrm{rk}_{\mathcal{M}}(E) - \mathrm{rk}_{\mathcal{M}}(E\setminus A)}.$$

As this would give a smaller packing number than Φ for the matroid before the insertion, we get that e must be in all bases of the new matroid after the insertion. This update is performed easily, since adding to all bases $B \in \mathcal{B}$ can be done in $|\mathcal{B}|$ updates to the corresponding minimum weight base – where this update has no recourse. So it takes $O(|\mathcal{B}|\log n)$ rank-queries by Proposition 1.

Consider a deletion e such that $\Phi' > \Phi$. As argued for the previous case, this means that e was part of *every* base in $\mathcal{M}$, so we can remove it easily from any $B \in \mathcal{B}$ n $|\mathcal{B}|$ updates to the corresponding minimum weight base – where this update has no recourse. So it takes $O(|\mathcal{B}|\log n)$ rank-queries by Proposition 1.

We obtain a total of $O\left(\frac{\Phi_{\max}\log^3(n)}{\varepsilon^4}\right)$ amortized rank-queries for any update. ◀

A.2 Dynamic Matroid Covering

Similar to the previous section, we now present our results for dynamic base covering.

A.2.1 Deterministic

First, we provide a deterministic result. We note that for base packing, we also have an algorithm with an amortized bound that depends linearly on Φ. For base covering, we could obtain a result in a similar manner that would depend on $\beta_{\max}^2/\Phi$. However, in this case, we cannot relate Φ and $\beta_{\max}$: Φ can be constant, even if $\beta_{\max}$ is polynomial.

▶ **Theorem 3.** *Let $\varepsilon \in (0, 1/2)$ be a parameter and let $\mathcal{M}$ be a dynamic matroid that at any moment contains at most n elements. If β is upper bounded by $\beta_{\max}$, we can deterministically maintain a $(1\pm\varepsilon)$-approximation of the fractional covering number with $O(\beta_{\max}^2 \cdot \varepsilon^{-4} \cdot \log^3 n)$ worst-case rank-queries per update.*

Proof. Analogously to the packing case, we maintain a base collection $\mathcal{B}$ of $\Theta\left(\frac{\beta_{\max}\log n}{\varepsilon^2}\right)$ greedy bases and maintain the minimum $\ell^{\mathcal{B}}(e)$ using a heap directly giving the approximation.

Again, we argue that the maintained value is a $(1\pm\varepsilon)$-approximation of the covering number. From Lemma 6 we get that

$$(1-\varepsilon)\frac{1}{\beta} \leq \min_{e\in E} \ell^B(e) \leq (1+\varepsilon)\frac{1}{\beta}.$$

By using $\varepsilon' := \frac{\varepsilon}{1+\varepsilon}$ to determine the size of the base collection we get, as for Theorem 2:

$$(1-\varepsilon)\beta \leq \frac{1}{\min_{e\in E}\ell^{\mathcal{B}}(e)} \leq (1+\varepsilon)\beta.$$

Maintaining the base collection takes $O\left(\frac{\beta_{\max}^2\log^3(n)}{\varepsilon^4}\right)$ queries for any update (worst-case), see Lemma 7. ◀

A.2.2 Oblivious adversary

Next, we show how uniform sampling can help us to obtain update times independent of β against oblivious adversaries. The approach uses a standard sampling technique (see, e.g., [47]) and is similar to the sampling to maintain arboricity in multigraphs, as shown in [16].

▶ **Theorem 4.** *Let $\varepsilon\in(0,1/2)$ be a parameter and let $\mathcal{M}$ be a dynamic matroid that at any moment contains at most n elements. There exists a fully dynamic algorithm that maintains a $(1\pm\varepsilon)$-approximation of the fractional covering number with $O(\log^6 n/\varepsilon^8)$ worst-case rank-queries per update. The algorithm is correct with high probability against an oblivious adversary.*

Proof. We create $O(\log n)$ matroids $\mathcal{M}_i$ by sampling each element with probability $p_i = \frac{24c\log n}{2^i\varepsilon^2}$ (for some constant $c\geq 2$ to be set later and i such that $p_i<1$) to get the respective universe E_i. $\mathcal{M}_i$ is then given by the restriction of the original matroid to the sampled elements $\mathcal{M}_i = \mathcal{M}|E_i$. Now we want to show that if $\beta\in[2^{i-1},2^{i+2})$ then $\frac{1}{p_i}\beta_i$ is a $(1\pm\varepsilon)$-approximation of β. We show this in two parts.

First, we argue that $\frac{1}{p_i}\beta_i \geq (1-\varepsilon)\beta$. Consider a set $S\subseteq E$ with $\beta = \frac{|S|}{\operatorname{rk}_{\mathcal{M}}(S)}$. Let $S_i = S\cap E_i$, then we get

$$\Pr\left(\frac{|S_i|}{p_i\operatorname{rk}_{\mathcal{M}_i}(S_i)} < (1-\varepsilon)\beta\right) = \Pr\left(|S_i| < (1-\varepsilon)p_i\beta\operatorname{rk}_{\mathcal{M}_i}(S_i)\right).$$

Since $\operatorname{rk}_{\mathcal{M}_i}(S_i)\leq\operatorname{rk}_{\mathcal{M}}(S) = \frac{|S|}{\beta}$ and $\mathbb{E}(|S_i|) = p_i|S|$ we get by a Chernoff bound

$$\begin{aligned}\Pr\left(|S_i| < (1-\varepsilon)p_i\beta\operatorname{rk}_{\mathcal{M}_i}(S_i)\right) &\leq \Pr\left(|S_i| < (1-\varepsilon)p_i\beta\operatorname{rk}_{\mathcal{M}}(S)\right) = \Pr\left(|S_i| < (1-\varepsilon)p_i|S|\right)\\ &\leq e^{-\varepsilon^2 p_i|S|/2} = e^{-\varepsilon^2 p_i\beta\operatorname{rk}_{\mathcal{M}}(S)/2} \leq e^{-\varepsilon^2 p_i 2^{i-1}\operatorname{rk}_{\mathcal{M}}(S)/2}\\ &\leq n^{-c\operatorname{rk}_{\mathcal{M}}(S)}.\end{aligned}$$

Hence

$$\Pr\left(\frac{|S_i|}{p_i\operatorname{rk}_{\mathcal{M}_i}(S_i)} < (1-\varepsilon)\beta\right) \leq n^{-c\operatorname{rk}_{\mathcal{M}}(S)}.$$

Now note that $\beta_i = \max_{\emptyset\neq S'\subseteq E_i}\frac{|S'|}{\operatorname{rk}_{\mathcal{M}_i}(S')} \geq \frac{|S_i|}{\operatorname{rk}_{\mathcal{M}_i}(S_i)}$, so w.h.p. $\frac{1}{p_i}\beta_i\geq(1-\varepsilon)\beta$.

Second, we need to show that with high probability the covering number is bounded as follows $\frac{1}{p_i}\beta_i \leq (1+\varepsilon)\beta$.

Let $S_i \subseteq E_i$ be any non-empty subset. We will show that w.h.p. $\frac{|S_i|}{p_i \operatorname{rk}_{\mathcal{M}_i}(S_i)} \leq (1+\varepsilon)\beta$. Hereto, we consider each possible rank $1 \leq r \leq \operatorname{rk}_{\mathcal{M}}(E_i) = \operatorname{rk}(\mathcal{M}|E_i)$ of a non-empty subset of E_i separately. Let S_i be the subset of E_i with rank r with maximum $|S_i|$. Note that for any other set $A \subseteq E_i$ with the same rank we have $\frac{|A|}{p_i \operatorname{rk}_{\mathcal{M}_i}(A)} \leq \frac{|S_i|}{p_i \operatorname{rk}_{\mathcal{M}_i}(S_i)}$.

Now we consider the set $S := \operatorname{span}_{\mathcal{M}}(S_i)$. For this set we have (a) $\operatorname{rk}_{\mathcal{M}}(S) = \operatorname{rk}_{\mathcal{M}}(S_i) = \operatorname{rk}_{\mathcal{M}_i}(S_i)$ since $S_i \subseteq E_i$ and (b) $S \cap E_i = S_i$ since no element of E_i can be added to S_i without an increase in the rank. Therefore we get

$$\begin{aligned}\Pr\left(\frac{|S_i|}{p_i \operatorname{rk}_{\mathcal{M}_i}(S_i)} > (1+\varepsilon)\beta\right) &= \Pr\left(\frac{|S \cap E_i|}{p_i \operatorname{rk}_{\mathcal{M}}(S)} > (1+\varepsilon)\beta\right) \\ &= \Pr\left(|S \cap E_i| > (1+\varepsilon)p_i\beta \operatorname{rk}_{\mathcal{M}}(S)\right).\end{aligned}$$

Further, since $\beta \operatorname{rk}_{\mathcal{M}}(S) \geq |S|$, using a Chernoff bound we get

$$\begin{aligned}\Pr\left(|S \cap E_i| > (1+\varepsilon)p_i\beta \operatorname{rk}_{\mathcal{M}}(S)\right) &\leq e^{-\varepsilon^2 p_i \beta \operatorname{rk}_{\mathcal{M}}(S)/3} = e^{-\varepsilon^2 24c\log(n)\beta \operatorname{rk}_{\mathcal{M}}(S)/3\varepsilon^2 2^i} \\ &\leq e^{-4c\log(n)\operatorname{rk}_{\mathcal{M}}(S)} \leq n^{-c(\operatorname{rk}_{\mathcal{M}}(S)+2)}\end{aligned}$$

where the second to last inequality follows since $\beta \geq 2^{i-1}$. Note that for such a set S we have that $S = \operatorname{span}_M(S)$. Also, among the sets of maximum size with rank r the set S is unique. If there was another set $S_j \neq S_i$ with $\operatorname{span}(S_j) = S$, then $S_j \subseteq S$ and we would have an element $e \in S_j \setminus S_i$ and $|S_i + e| > |S_i|$ while $\operatorname{rk}_{\mathcal{M}}(S_i + e) = \operatorname{rk}_M(S_i)$. This contradicts the choice of S_i. Hence, for any rank r it suffices to union bound over all sets S with $S = \operatorname{span}(S)$ and $\operatorname{rk}(S) = r$, there are at most n^r such sets. So the statements holds for all such S with probability n^{-2c}.

This allows us to then union bound over all possible ranks r , there are at most $\operatorname{rk}_{\mathcal{M}}(E)$ many and n updates, so the statement holds with probability $n^{-2c} \cdot n \cdot n = n^{2-2c} \leq n^{-c}$.

Hence, we get that $\beta_i \leq (1+\varepsilon)\beta p_i \leq 2^3 \frac{24c\log n}{\varepsilon^2} = O\left(\frac{\log n}{\varepsilon^2}\right)$ when $\beta \leq 2^{i+2}$. The algorithm maintains a greedy base collection of size $\Theta\left(\frac{\log^2 n}{\varepsilon^4}\right)$ for each of the $O(\log n)$ matroids $\mathcal{M}_i$ and one for $\mathcal{M}$ itself. Whenever $\beta \in [2^{i-1}, 2^{i+2})$ the collection corresponding to $\mathcal{M}_i$ gives the correct approximation. All other collections might give values that differ drastically from β and are simply disregarded. At any point in time to know which $\mathcal{M}_i$ gives the correct value it suffices to consider the estimate from the previous update. This is because if the old estimate $\frac{\beta_i}{p_i}$ was in $[2^i, 2^{i+1})$, then after the update we have that $\beta < (1+\varepsilon)\frac{\beta_i}{p_i} + 1 \leq 2^{i+2}$ and similarly $\beta \geq (1-\varepsilon)\frac{\beta_i}{p_i} - 1 \geq 2^{i-1}$. If the current estimate is in $\Theta(\log n/\varepsilon^2)$ the estimate on the original matroid is considered, otherwise the interval $[2^i, 2^{i+1})$ that contains the previous estimate determines the base collection $\mathcal{M}_i$ that is used for the estimate after the next update. Now it remains to analyze the runtime. For each matroid $\mathcal{M}_i$ we maintain a greedy base collection of size $O\left(\frac{\log^2 n}{\varepsilon^4}\right)$.

There are $O(\log n)$ matroids that could be affected by an update, so by Lemma 7, an update requires $O\left(\frac{\log^6(n)}{\varepsilon^8}\right)$ rank-queries in the worst-case. ◀