Online Matroid Intersection: Submodular Water-Filling and Matroidal Welfare Maximization

Daniel Hathcock* Billy Jin† Kalen Patton‡ Sherry Sarkar* Michael Zlatin*

Abstract

We study two problems in online matroid intersection. First, we consider the problem of maximizing the size of a common independent set between a general matroid and a partition matroid whose parts arrive online. This captures the classic online bipartite matching problem when both matroids are partition matroids. Our main result is a $(1 - \frac{1}{e})$-competitive algorithm for the fractional version of this problem. This applies even for the poly-matroid setting, where the rank function of the offline matroid is replaced with a general monotone submodular function. The key new ingredient for this result is the construction of a "water level" vector for poly-matroids, which allows us to generalize the classic water-filling algorithm for online bipartite matching. This construction reveals connections to submodular utility allocation markets and principal partition sequences of matroids.

Our second result concerns the Online Submodular Welfare Maximization (OSWM) problem, in which items arriving online are allocated among a set of agents with the goal of maximizing their overall utility. If the utility function of each agent is a monotone, submodular function over the set of available items, then a simple greedy algorithm achieves a competitive ratio of $\frac{1}{2}$. Kapralov, Post, and Vondrák showed that in this case, no polynomial time algorithm achieves a competitive ratio of $\frac{1}{2} + \varepsilon$ for any $\varepsilon > 0$ unless NP = RP (SODA, 2013). We extend the RANKING algorithm of Karp, Vazirani, and Vazirani (STOC, 1990) to achieve an optimal $(1 - \frac{1}{e})$-competitive algorithm for OSWM in the case that the utility function of each agent is the rank function of a matroid.

*Carnegie Mellon University
†Cornell University
‡Georgia Institute of Technology

1 Introduction

In *Online Matroid Intersection*, the goal is to maximize the size of a common independent set between two matroids, when the elements are initially unknown and arrive in some online fashion. We focus on the case where one of the matroids is a partition matroid. In particular, suppose $\mathcal{M}$ is an arbitrary matroid and $\mathcal{Q}$ is a partition matroid with parts $Q_1, \ldots, Q_n$, both defined over (an initially unknown) ground set E. We have access to an independence oracle for $\mathcal{M}$ restricted to the elements which have been revealed so far; in other words, $\mathcal{M}$ is known offline. Parts from the partition matroid arrive online. When a part Q_j arrives, the elements in Q_j are revealed, and we immediately and irrevocably choose at most one element from Q_j. The goal is to maximize the cardinality of the set of chosen elements, subject to the set being independent in both matroids.

The online matroid intersection problem with part arrival is a natural generalization of the classic vertex-arrival online bipartite matching problem, which was introduced in a seminal work by Karp, Vazirani, and Vazirani [KVV90]. In online bipartite matching, we are given one side of a bipartite graph (the *offline* vertices) in advance, while the vertices on the other side arrive online. When an online vertex arrives, all of its incident edges are revealed, and the algorithm chooses at most one of the edges. The goal is to maximize the number of edges chosen, subject to the edges being a matching in the graph. In the language of online matroid intersection, online bipartite matching is the setting where $\mathcal{M}$ is the partition matroid over the edges whose parts are the neighborhoods of the vertices on the offline side, and $\mathcal{Q}$ is the partition matroid corresponding to the neighborhoods of the vertices arriving online.

In addition to generalizing vertex-arrival online bipartite matching, online matroid intersection captures a richer class of resource allocation problems with combinatorial constraints.

- **Restricted Matching Problems.** Bipartite matching models fundamental resource allocation problems, with one such example being matching tasks to servers. What if we have additional limitations on a cluster of servers on the same server rack, due to cooling or bandwidth constraints? We can model this problem by taking $\mathcal{M}$ to be a laminar matroid instead of partition matroid.

- **Coflows.** Say we have a computing resource which may process some tasks in parallel. For example, perhaps a single server rack is made up of different servers, each of which is equipped to handle only certain types of tasks. In this case, tasks which may be processed together on a single resource form a transversal matroid. Tasks processed in parallel are called *coflows*, inspired by applications to MapReduce [CS12]. Coflows governed by general matroid constraints have been studied [JKR17, IMPP19] in an offline setting. In an online formulation of this problem, we have Δ resources and n tasks arriving online which we must irrevocably assign to computing resources; the goal is to accommodate as many tasks as possible.

- **Matroid Coloring.** Matroid coloring [Edm65, Knu73] can be written as a matroid intersection problem, and hence, we can model the problem of coloring any matroid online. Say we have Δ colors and elements of a matroid $\mathcal{N}$ arrive one by one. We may color elements as they arrive; however, each color must be independent in $\mathcal{N}$. The objective is to color as many elements as possible. This may be modelled when $\mathcal{M}$ is the product of Δ copies of $\mathcal{N}$, and $\mathcal{Q}$ dictates that every element can be at most one color.

A related problem to online matroid intersection is *Online Submodular Welfare Maximization*. In this problem, m items arrive online to be allocated to n agents who each have utility functions

$f_i : 2^{[m]} \to \mathbb{R}_{\geq 0}$. Each utility function f_i is a monotone, submodular function on $[m]$. The goal is to maximize the total welfare of the agents, i.e., the sum of the utilities. In the offline setting, a $1 - 1/e$ approximation algorithm can be achieved using an algorithm for Monotone Submodular Maximization subject to a matroid constraint [CCPV11]. Surprisingly however, Kapralov, Post, and Vondrák [KPV13] show that achieving an competitive ratio greater than $1/2$ in polynomial time (achieved trivially with the Greedy algorithm) is impossible unless NP = RP.

We show that this barrier can be circumvented if the class of monotone submodular functions is restricted to those that arise from the rank function of a matroid. In particular, we assume that each utility function f_i for $i \in [n]$ is the rank function of matroid $\mathcal{M}_i$ over the ground set $[m]$ of items. In this setting, we may assume that each agent i only receives an independent set of $\mathcal{M}_i$, and thus we seek an allocation of maximum cardinality in the intersection of $\mathcal{M} = \mathcal{M}_1 \times \ldots \times \mathcal{M}_n$ and $\mathcal{Q}$. Because of this, unlike general monotone submodular valuations, the optimal *offline* allocation can be computed exactly for matroidal utilities using an algorithm for matroid intersection. Hence, this offline problem appears as one of the few tractable cases of the pricing and allocation problems in VCG auctions with combinatorial utilities.

The class of matroid rank valuations form a broad class which captures any utility function in which the marginal benefit to each agent of receiving an additional item is binary, and also satisfies diminishing marginal returns. More formally, the set of matroid rank functions is equivalent to the class of submodular utility functions with $\{0, 1\}$-marginal gains. When $\mathcal{M}_i$ is a uniform matroid for all i, we recover the well-studied online bipartite b-matching problem, but it also captures more general laminar constraints on agents. These more general constraints appear frequently in resource allocation problems to ensure a feasible allocation of tasks to servers. When each $\mathcal{M}_i$ is a transversal matroid, each agent utility from a set of items is the size of the maximum matching among this set. This can model the setting in which each agent represents a group of individuals whom, upon receiving a set of items, optimally match the items received to people in their group. Finally, note that each agent i may have a different type of matroid $\mathcal{M}_i$. This allows these different types of matroid valuations to be mixed and matched in any way which arises in the application of interest.

1.1 Our Results and Techniques

Fractional Online Poly-Matroid Intersection Our first result concerns the fractional version of online matroid intersection, where the goal of the algorithm is to maintain a fractional independent set in the intersection of both matroids whose size is as large as possible. We give a $(1 - 1/e)$-competitive algorithm for this problem. In fact, our result applies to the more general setting of fractional online *poly-matroid* intersection, in which the matroid $\mathcal{M}$ is replaced with the poly-matroid $\mathcal{P}_f = \{x \in \mathbb{R}^E_{\geq 0} : \forall S \subseteq E,\ x(S) \leq f(S)\}$ of some monotone submodular function f. We note that the $(1 - 1/e)$ competitive ratio is tight, because there is a matching upper bound even for fractional online bipartite matching [Fei19].

Theorem 1.1. *There exists a deterministic polynomial time $(1 - 1/e)$-competitive algorithm for fractional Online Poly-Matroid Intersection.*

Our algorithm for this setting generalizes the water-filling algorithm, a $(1 - 1/e)$-competitive algorithm for online bipartite matching. In online matching, the water-filling algorithm works by keeping track of a "water level" for each offline vertex, which is the fractional amount it has been filled so far. Upon the arrival of each online vertex, the algorithm continuously sends water to the

its neighbors with the lowest water level, until either all neighbors have been filled or the arriving vertex has been depleted.

One key challenge in obtaining a water-filling algorithm for online poly-matroid intersection is defining what "water levels" should be for an arbitrary poly-matroid $\mathcal{P}_f$. Intuitively, given a fractional allocation $\{x_e\}_{e \in E}$, the water level w_e of element e should represent how "filled" it is under the allocation x. For instance, in online bipartite matching, the water level of an edge (i, j) (where i is offline and j is online), is the fraction of vertex i's capacity used up by x. In the poly-matroid setting, edges become elements $e \in E$, but there is no longer a notion of unit-capacity vertices. Instead, each e is involved in many submodular constraints of the form $x(S) \leq f(S)$ for each $S \ni e$, and it is not immediately clear how these constraints should be aggregated into a water level w_e.

Nevertheless, we show that there exist natural values for these "submodular water levels", and we present three different ways of describing them.

1. **A combinatorial description.** Water levels can be viewed as a measure of density. We describe an iterative process in Algorithm 1 which finds a densest set under x (i.e. the set with the least multiplicative slack), contracts this set, and repeats until all elements are contracted. The density at which an element gets contracted is the water level of that element.

2. **A submodular optimization problem.** The level sets $\{e \in E : w_e \geq t\}$ for all $t \geq 0$ can be characterized by solutions to the submodular minimization problem $\min_{S \subseteq E}\{t \cdot f(S) - x(S)\}$. This program demonstrates that the level sets of w correspond to a weighted version of the *principal partition sequence* studied in [Nar91, Fuj08] for poly-matroids.

3. **A convex program.** The concept of water levels also arise from market equilibria. Jain and Vazirani [JV10] introduced a notion of *submodular utility allocation (SUA) markets* as a generalization of linear Fisher markets [NRTV07, Chapter 5]. In a submodular utility allocation market, we have some items A, each with weight m_a. We would like to fractionally select a set of items, with the goal of maximizing the Nash social welfare of the items; however, the set of items picked must be feasible in a poly-matroid defined by a sub-modular function f. It turns out the water level vector is precisely the optimal solution of an SUA market.

Once a notion of water levels is defined, we follow the classic water-filling paradigm: when a part Q_j arrives, we incrementally increase the allocation x_e on those elements $e \in Q_j$ with smallest water level. However, while this intuitive notion of increasing water levels is straightforward to implement in the case of online bipartite matching, more care is required in the more general setting of online poly-matroid intersection. For example, when raising x_e for the $e \in Q_j$ of minimum water level, at what rate must each x_e be raised with respect to each other? In the general poly-matroid setting, these rates are not unique, so the algorithm as stated intuitively is not even well-defined.

To address this, we present a discretized version of the algorithm. At each arrival, we solve a convex program to determine the entire allocation x_e for each $e \in Q_j$ at once. We show that this allocation can be computed in polynomial time, and satisfies the key water-filling properties: (1) The set of elements which receive a nonzero allocation from Q_j all end up with minimum water level among elements in Q_j, and (2) If a part Q_j does not allocate a full unit of water, then all elements $e \in Q_j$ end up with water level 1.

Each of our three viewpoints on water levels sheds light on the desired properties which we will use in our analysis. We show four properties: monotonicity (Proposition 3.6), indication

of feasibility (Proposition 3.7), duality (Proposition 3.8), and locality (Proposition 3.9). These properties, as well as the water-filling property of our algorithm, allow us to leverage a primal-dual analysis to prove that the algorithm is $(1 - 1/e)$-competitive.

Matroidal Welfare Maximization There are several classes of utility functions (also called valuation functions) often considered in the welfare maximization literature (see [DFF21] for a discussion and comparison of some of these classes). The following are of interest specifically in *online* welfare maximization:

- *Matroid rank functions (MRF)*: each agent has a weight $a_i \in \mathbb{R}_{\geq 0}$ and matroid $\mathcal{M}_i$. The utility of $S \subseteq [m]$ is $a_i \cdot \text{rank}_{\mathcal{M}_i}(S)$.
- *Weighted matroid rank functions (WMRF)*: each agent has a weight vector $v^{(i)} \in \mathbb{R}_{\geq 0}^{[m]}$ on the items and a matroid $\mathcal{M}_i$. The utility of $S \subseteq [m]$ is $\max\{v^{(i)}(I) : I \subseteq S, I \text{ independent in } \mathcal{M}_i\}$.
- *Submodular functions*: each agent has a monotone submodular function f_i with $f_i(\varnothing) = 0$. The utility of $S \subseteq [m]$ is $f_i(S)$.

We have the relationship MRF $\subsetneq$ WMRF $\subsetneq$ Submodular between these classes. Kapralov, Post, and Vondrák [KPV13] addressed the Submodular class by showing that no polynomial time algorithm for online submodular welfare maximization can be better than $1/2$-competitive unless NP = RP. In the middle we have WMRF, which captures as a special case the edge-weighted online bipartite matching problem. The best known competitive ratio for edge-weighted online bipartite matching is 0.5368 [BC21], and there seem to be major technical barriers to reaching $1 - 1/e$ for this setting. The second main contribution of this paper is a tight $(1 - 1/e)$-competitive algorithm for online submodular utility allocation with MRF utility functions.

Theorem 1.2. *There exists a randomized polynomial time $(1-1/e)$-competitive algorithm for Online Submodular Welfare Maximization when the agents' utility functions are matroid rank functions.*

With this contribution, the MRF class of utility functions is, to the authors' knowledge, the broadest class known to admit a $1 - 1/e$ competitive ratio in polynomial time.

Our result captures the $(1 - 1/e)$-competitive ratios for online bipartite matching due to Karp, Vazirani, and Vazirani [KVV90], as well as the extension to vertex-weighted bipartite matching by Aggarwal, Goel, Karande, and Mehta [AGKM11], and b-matching setting by Albers and Schubert [AS21]. All of these cases can be modelled as online submodular welfare maximization in which each agent's utility function is the rank function of a *uniform matroid*[1]. We substantially generalize these settings by allowing the agents (i.e., the offline vertices) to have an arbitrary matroid constraint over their neighboring edges.

We apply the primal-dual framework introduced by Devanur, Jain, and Kleinberg [DJK13] to analyze a "Matroidal" RANKING algorithm. In this sense, our proof also unifies the analyses of the above special cases; specifically, we simplify the analysis in the case of b-matching by avoiding the need to use a configuration LP. Note that Albers and Schubert [AS21] observed that the standard b-matching LP could not be used to show that RANKING achieves a $1 - 1/e$ competitive ratio. We circumvent this issue by using the matroid-based LP. In the b-matching setting this is effectively equivalent to the b-matching LP where, for each offline vertex i, its capacity b_i is replaced with the degree of the vertex (i.e., if it has fewer than b_i incident edges).

[1] A matroid in which a set is independent if it has cardinality at most b.

1.2 Related Work

Online Matching There is an extensive line of work on online matching, starting with the work of Karp, Vazirani and Vazirani [KVV90], who gave a $(1 - 1/e)$-competitive algorithm for online bipartite matching in the adversarial order setting. The same competitive ratio was extended to the vertex-weighted setting in [AGKM11], and further to the vertex-weighted b-matching setting in [AS21]. Devanur, Jain, and Kleinberg [DJK13] showed how the results in [KVV90] and [AGKM11] could be derived using the online primal-dual framework, which unified and simplified the existing analyses. While the RANKING algorithm requires $O(n \log n)$ bits of randomness, Buchbinder, Naor, and Wajc [BNW23] provide a randomized rounding scheme requiring only $(1 \pm o(1)) \log \log n$ bits of randomness.

For edge-weighted online bipartite matching, [FHTZ20] were the first to break the $\frac{1}{2}$-competitive barrier. This has been subsequently refined in [SA21, GHH+21, BC21] to a 0.5368-competitive ratio. Online bipartite matching has also been studied in the vertex-arrival Bayesian setting, also known as the *Ride Hail* problem; [PPSW21] give a better-than-$1/2$ approximation for this setting.

Online Matroid Intersection Offline matroid intersection captures a broad class of combinatorial problems and has been well studied in both polyhedral theory and algorithms (for a survey on this topic, see [Sch03, Chapter 41]). In [GS17], Guruganesh and Singla study an online matroid intersection problem, with edges arriving in random order, beating the $1/2$-competitive ratio achieved by Greedy. This was very recently improved by Huang and Sellier [HS23] to $2/3 + \varepsilon$. In [BGH+23], the part-arrival online matroid intersection model is considered. They instead study the problem of *maintaining* a max cardinality independent set, while minimizing recourse.

Offline Submodular Welfare Maximization The offline variant of Submodular Welfare Maximization problem can be cast as a problem of maximizing a monotone submodular function subject to a (partition) matroid constraint. While a simple greedy algorithm achieves an approximation ratio of $\frac{1}{2}$, [CCPV11] gave an improved $(1 - 1/e)$-approximation using the Continuous Greedy algorithm followed by Pipage Rounding. There can be no $(1 - 1/e + \varepsilon)$-approximation unless P=NP since this problem captures max-k-cover as a special case [Fei98].

When the monotone submodular function is the rank function of a matroid, the welfare maximizing offline allocation can be computed optimally. These allocations have been well-studied in the context of designing socially optimal allocations which satisfy certain desirable fairness constraints [VZ23, DFSH23]. For example, in [BCIZ21] it is shown that an optimal allocation which is envy-free up to one item (EFX) exists and can be computed efficiently. In [BEF21] they study the case of valuations which have binary marginals, but which are not necessarily submodular [BEF21]. They also give truthfulness guarantees for private valuations, which has been further studied in [BV22].

Principal Partition The *principal partition* of a matroid is related to our definition of water levels. It is precisely the nested sets found in Algorithm 1, when x is the all ones vector. There have been several works studying principal partitions, including generalizations to arbitrary vectors $x \in \mathbb{R}^E$ and extensions from rank functions to sub-modular functions [Nar91, Fuj08]. We note that, although the objects studied in [Nar91] and [Fuj08] are closely related to our water levels, our work is distinct in (1) we shift perspective from the family of nested sets in [Fuj08] to the properties of a single vector w, and (2) we study properties of how w changes dynamically with the weights given by

x, such as monotonicity (Proposition 3.6), duality (Proposition 3.8), and locality (Proposition 3.9). These properties are clearly visible with our new perspective. In addition, we make the novel connection between principal partitions and water-filling in online bipartite matching.

The principal partition has been used for constant competitive algorithms in the matroid secretary problem under random assignment [Sot13]. Huang and Sellier [HS23] also use the principal partition to get an improved approximation ratio of $2/3 - \varepsilon$ for online matroid intersection with random order arrivals. Chandrasekaran and Wang [CW23] use the principal partition sequence for improved approximations in the submodular k-partition problem.

1.3 Organization of the Paper

First, in Section 2, we define the problem and introduce some preliminaries. Next, in Section 3, we develop a theory of water levels and provide three equivalent formulations. In Section 4, we give a water-filling algorithm for fractional online poly-matroid intersection and prove that it achieves a competitive ratio of $1 - 1/e$. In Section 5, we give a $(1 - 1/e)$-competitive algorithm for (integral) Online Submodular Welfare Maximization with matroidal utilities. We conclude with some future directions in Section 6

2 Problem Definition and Preliminaries

We start by defining the setting and notation for fractional *Online Poly-Matroid Intersection*. Note that this generalizes the problem of Online Matroid Intersection with (offline) matroid $\mathcal{M}$ by choosing the rank function of $\mathcal{M}$ to be the submodular function defining the poly-matroid.

Formally, we have an (offline) monotone submodular[2] function f over ground set E with $f(\varnothing) = 0$ and $f(\{e\}) > 0$ for all $e \in E$[3]. We also have an (online) partition matroid $\mathcal{Q}$ over E in which at most one element can be chosen from each of the parts $Q_1, \ldots, Q_m$ (i.e., the rank of each part in $\mathcal{Q}$ is 1). The ground set E is initially unknown. Parts arrive online one-by-one, each Q_j upon its arrival revealing its contained elements. We have offline access to an evaluation oracle for f that may be called on any subset of elements revealed so far.

The goal is to allocate values $(x_e)_{e \in Q_j}$ to the elements in Q_j immediately and irrevocably when Q_j arrives so as to maximize the final value of $\sum_{e \in E} x_e$. We must, however, allocate no more than 1 total value to elements in each Q_j, so $x(Q_j) := \sum_{e \in Q_j} x_e \leq 1$. Moreover, no set $S \subseteq E$ can have more total value than $f(S)$, so that $x(S) := \sum_{e \in S} x_e \leq f(S)$. Another way to say this is that the point x maintained must remain a feasible point in the intersection of the *poly-matroid* for f and the *matroid polytope* for $\mathcal{Q}$. These are defined respectively as:

$$\mathcal{P}_f := \left\{ x \in \mathbb{R}^E_{\geq 0} : x(S) \leq f(S) \text{ for every } S \subseteq E \right\}$$

and

$$\mathcal{Q} := \left\{ x \in \mathbb{R}^E_{\geq 0} : x(Q_j) \leq 1 \text{ for every } j = 1, \ldots, n \right\}.$$

Note that we refer to both the partition matroid and its polytope as $\mathcal{Q}$; it will be clear from context which we are referring to throughout the paper. We may now restate the constraints of the problem by saying that x must remain a point in $\mathcal{P}_f \cap \mathcal{Q}$.

[2] A function $f : 2^E \to \mathbb{R}_{\geq 0}$ is submodular if for all $A, B \subseteq E$, we have $f(A \cup B) \leq f(A) + f(B) - f(A \cap B)$. It is monotone if $f(A) \leq f(B)$ whenever $A \subseteq B$.

[3] This assumption is without loss of generality, since any e with $f(\{e\}) = 0$ can be removed.

Again, observe that this captures the fractional Online Matroid Intersection problem as a special case by using the rank function r of a matroid as the function f.

2.1 The Dual LP and the Lovász Extension

We analyze the algorithm for fractional online poly-matroid intersection via the primal-dual framework. The primal linear program (LP) describing Online Poly-matroid Intersection below, as well as its dual are:

$$\begin{array}{lll} \max & \sum_{e\in E} x_e & \\ \text{s.t.} & \sum_{e\in Q_j} x_e \le 1, & \forall j \in [n] \\ & \sum_{e\in S} x_e \le f(S), & \forall S \subseteq E \\ & x_e \ge 0, & \forall e \in E. \end{array} \qquad \begin{array}{lll} \min & \sum_{S\subseteq E} f(S)\cdot \alpha_S + \sum_{j=1}^{n} \beta_j & \\ \text{s.t.} & \sum_{S\ni e} \alpha_S + \beta_{j(e)} \ge 1, & \forall e \in E \\ & \alpha_S, \beta_j \ge 0, & \forall S \subseteq E,\, j \in [n]. \end{array}$$

In the dual, $j(e)$ denotes the index j for which $e \in Q_j$. The solution to the primal LP is the (fractional) optimal *offline* solution to a given instance, and we use OPT to denote its value. By strong duality, the optimal values of the primal and dual LPs are the same.

It will be useful for our analysis to re-write the dual objective in a different form. By a standard uncrossing argument, we may assume that the optimal dual α is supported on a nested family of sets. Thus, we can re-parameterize the $\alpha \in \mathbb{R}^{2^E}_{\ge 0}$ as $\gamma \in \mathbb{R}^{E}_{\ge 0}$ by $\gamma_e := \sum_{S\ni e} \alpha_S$. Notice that we can recover the part of the objective $\sum_{S\subseteq E} f(S)\cdot\alpha_S$ from γ as

$$\sum_{S\subseteq E} f(S)\cdot\alpha_S = \int_0^\infty f(\{e \in E : \gamma_e \ge t\})\, dt. \tag{1}$$

The right-hand-side integral is exactly the *Lovász extension* L_f of a submodular function f. This is a natural continuous extension of the submodular function f which has been studied in many contexts. Although equivalent formulations exist, it will be convenient for us to use the following definition.

Definition 2.1. *Let f be a monotone submodular function on E with $f(\emptyset) = 0$. The Lovász extension $L_f : \mathbb{R}^E_{\ge 0} \to \mathbb{R}$ of f is*

$$L_f(w) := \int_0^\infty f(\{e \in E : w_e \ge t\})\, dt.$$

With this definition, we can write our dual in terms of γ as follows.

$$\begin{array}{lll} \min & L_f(\gamma) + \sum_{j=1}^{n} \beta_j & \\ \text{s.t} & \gamma_e + \beta_{j(e)} \ge 1, & \text{for all } e \in E \\ & \gamma, \beta \ge 0. & \end{array}$$

3 Water Level Machinery

Our plan is to generalize the water-filling algorithm for online bipartite matching. We want to assign each element $e \in E$ a water level $w_e = w_e^{(x)}$ which depends on the current allocation x. When a part Q_j arrives, we continuously increase x_e for the element $e \in Q_j$ of lowest water level, until we either allocate a total of 1 unit of mass or no further increases to the elements in Q_j are feasible. In particular, x should remain a feasible point in the intersection of the poly-matroid (associated with monotone submodular function f with $f(\varnothing) = 0$) and the matroid polytope for partition matroid $\mathcal{Q}$.

To understand how we define this water level vector $w = w^{(x)} \in \mathbb{R}^E_{\geq 0}$ of allocation $x \in \mathcal{P}_f$, we first enumerate several properties that the water levels should satisfy in order for the water-filling algorithm to work as it does for online bipartite matching:

1. (Monotonicity) $w^{(x)}$ is coordinate-wise non-decreasing in x.
2. (Indication of Feasibility) $w^{(x)} \leq 1$ if and only if $x(S) \leq f(S)$ for all $S \subseteq E$.
3. (Locality) If $w_{e_1}^{(x)} \neq w_{e_2}^{(x)}$, then $\frac{\partial w_{e_2}^{(x)}}{\partial x_{e_1}} = 0$.
4. (Duality) $L_f(w^{(x)}) = \sum_{e \in E} x_e$.

Monotonicity and feasibility indication are natural properties which intuitively require that w_e is an indicator of how close x_e is to being part of a tight constraint. The need for locality and duality is less obvious, but they are important for the details of the primal-dual analysis of water-filling. Specifically, this is because the dual value γ_e of an element $e \in E$ will be defined as a function of the water level w_e. First, the locality property says that small changes to one element's allocation x_{e_1} should not affect any other element's water level w_{e_2} unless the two elements are already at the same water level. This is needed for the key water-filling property, that the "lowest water levels first" allocation strategy does not change the water level w_e (and thus the dual γ_e) for elements e at higher water levels. Secondly, the duality property is needed to relate increases in the dual objective term $L_f(\gamma)$ to increases to the primal objective $\sum_{e \in E} x_e$ as the algorithm progresses.

3.1 Definition of Water Levels and Equivalent Formulations

In order to define a water level vector that satisfies our desired properties, it will be convenient (and enlightening) to provide both algorithmic and static definitions, which we will prove are equivalent. By juggling three different definitions, we are able to provide succinct proofs of the four key properties of water levels.

First, let's consider a naive construction: For $x \in \mathbb{R}^E_{\geq 0}$, we could define $w_e^{(x)} = \max_{S \ni e} \frac{x(S)}{f(S)}$, i.e. the maximum density[4] of a poly-matroid constraint involving x_e. Such a definition clearly satisfies monotonicity and indication of feasibility. However, this definition does not capture the water levels from the classic bipartite matching setting (i.e. in a partition matroid), and we can see this already in the simple setting of $E = \{1, 2\}$ and $f(S) = |S|$. In such a setting, the poly-matroid effectively only has the constraints $0 \leq x_1, x_2 \leq 1$, so we intuitively should let $w_e = x_e$. However, notice that if $x_1 < x_2$, then we have $w_1 = \max_{S \ni 1} \frac{x(S)}{f(S)} = \frac{x_1 + x_2}{2}$. We see that this construction deviates from

[4] By convention, we consider the density of the empty set $\frac{x(\varnothing)}{f(\varnothing)}$ to be 0.

what we expect, and indeed we also find that our desired locality and duality properties are not satisfied. This problem arises because the heavier element x_2 influences the density of the densest constraint on x_1, despite the two variables being functionally independent.

The critical insight is that we can prevent this undesirable behavior by iteratively contracting sets with larger density before assigning the values of w_e to sets with lower density. This process is formalized in Algorithm 1, which gives our first construction of the water level vector $w^{(x)}$ we will ultimately use.

Algorithm 1: `A Combinatorial Presentation of Water Levels`

input : A point $x \in \mathbb{R}^E_{\geq 0}$.

Initialize $\ell \leftarrow 0$, $S_0 \leftarrow \varnothing$.
while $S_\ell \neq E$ **do**
  Let $f_\ell : E \setminus S_\ell \to \mathbb{R}_{\geq 0}$ be the contracted function $f_\ell(T) = f(T \cup S_\ell) - f(S_\ell)$.
  Let $T_{\ell+1}$ be the unique maximal set[5] in $\arg\max_{T \subseteq E \setminus S_\ell} \frac{x(T)}{f_\ell(T)}$, i.e. the largest densest set.
  Let $t_{\ell+1} = \frac{x(T_{\ell+1})}{f_\ell(T_{\ell+1})}$ be the density of $T_{\ell+1}$.
  Set $w_e \leftarrow t_{\ell+1}$ for all $e \in T_{\ell+1}$.
  Update $S_{\ell+1} \leftarrow S_\ell \cup T_{\ell+1}$.
  $\ell \leftarrow \ell + 1$.
return w.

The algorithm gives us a pictorial view of how we construct water levels, but it does not yet make obvious that our desired properties should hold. For that, we will also introduce the following more compact definition.

Definition 3.1 (Water levels)**.** *Define for a given $y \in \mathbb{R}^E_{\geq 0}$,*

$$S^*_f(y) := \textit{unique inclusion-wise maximal set in } \arg\min_{S \subseteq E}(f(S) - y(S)).$$

Then, the water levels *with respect to an allocation x in $\mathcal{P}_f$ are defined as*

$$w^{(x)}_e := \sup\left\{t > 0 : e \in S^*_f\left(\frac{x}{t}\right)\right\},$$

where we adopt the convention $\sup \varnothing = 0$.

This definition may seem mysterious at first, but it is in fact equivalent to the output of Algorithm 1. For readability, we delay this proof until Section 3.3.

Theorem 3.2. *For any monotone submodular function $f : 2^E \to \mathbb{R}_{\geq 0}$ with $f(\varnothing) = 0$ and $x \in \mathbb{R}^E_{\geq 0}$, the output of Algorithm 1 and the vector $w = w^{(x)}$ defined in Definition 3.1 are equal.*

Defining w with the sets $S^*_f(y)$ allows us to prove many nice properties using the following lemma.

Lemma 3.3. *The set $S^*_f(y)$ is non-decreasing in y, i.e. for any $y, z \in \mathbb{R}^E_{\geq 0}$ with $y \leq z$ coordinate-wise, we have $S^*_f(y) \subseteq S^*_f(z)$.*

[5]Such a set is unique due to the sub-modularity of f.

For instance, we immediately obtain the following "level set" characterization of w.

Corollary 3.4. *We have* $\{e \in E : w_e^{(x)} \geq t\} = S_f^*(\frac{x}{t})$.

Proof. From Lemma 3.3, the sets $S_f^*(\frac{x}{t})$ are nested and decreasing in t, which together with the definition of $w_e^{(x)}$ in Definition 3.1 immediately gives the statement of the Corollary. □

Proof of Lemma 3.3. Let $S_y = S_f^*(y)$ and $S_z = S_f^*(z)$. Then we have

$$\begin{aligned} f(S_y \cap S_z) - y(S_y \cap S_z) + f(S_y \cup S_z) - z(S_y \cup S_z) &\leq f(S_y) - y(S_y \cap S_z) + f(S_z) - z(S_y \cup S_z) \\ &\leq f(S_y) - y(S_y) + f(S_z) - z(S_z). \end{aligned}$$

The first inequality follows from the submodularity of f. The second inequality follows from the modularity of y and z, and the fact that $z(S_y \setminus S_z) \geq y(S_y \setminus S_z)$. By minimality of $f(S_y) - y(S_y)$ and $f(S_z) - z(S_z)$, we must have $f(S_y) - y(S_y) = f(S_y \cap S_z) - y(S_y \cap S_z)$ and $f(S_z) - z(S_z) = f(S_y \cup S_z) - z(S_y \cup S_z)$. Therefore, by maximality of S_z, we have $S_y \cup S_z = S_z$. □

Finally, we also find an unexpected connection to market equilibria. Jain and Vazirani [JV10] introduced a notion of *submodular utility allocation markets* which can be described with the following convex program.

$$\begin{aligned} \max_u \quad & \textstyle\sum_{e \in E} m_e \log u_e \\ & \textstyle\sum_{e \in S} u_e \leq f(S), \quad \forall S \subseteq E \qquad (\alpha_S) \\ & u_e \geq 0. \end{aligned} \tag{SUA}$$

It turns out the water levels of an allocation x can be computed from the optimal utilities of an SUA market where each element e has weight $m_e := x_e$. Algorithm 1 also gives us optimal duals to (SUA), which we also prove in Section 3.3.

Theorem 3.5. *Consider the vector w, the nested sets $S_1 \subset \cdots \subset S_L$, and the levels $t_1, \ldots, t_L$ generated by Algorithm 1. Then $t_1 > \cdots > t_L \geq 0$. Moreover, if we define*

$$\begin{aligned} \alpha_{S_L} &:= t_L \\ \alpha_{S_\ell} &:= t_\ell - t_{\ell+1} \quad \ell = 1, \ldots, L-1 \end{aligned}$$

and $\alpha_S = 0$ for all other $S \subseteq E$, then, $u_e = \frac{x_e}{w_e}$ is an optimal primal solution to (SUA) and α_S is an optimal dual solution.

3.2 Key Properties of Water Levels

Armed with the characterizations of water levels, we show they satisfy the desired properties.

Proposition 3.6 (Monotonicity)**.** *The vector $w^{(x)}$ is coordinate-wise non-decreasing in x.*

Proof. This follows immediately from Lemma 3.3, since the level sets $\{e \in E : w_e^{(x)} \geq t\} = S_f^*(\frac{x}{t})$ for each fixed t only increase as x increases. □

Proposition 3.7 (Indication of Feasibility)**.** *$w^{(x)} \leq 1$ if and only if $x(S) \leq f(S)$ for all $S \subseteq E$.*

Proof. This follows from Algorithm 1. If $w^{(x)} \leq 1$, then in particular $t_1 = \max_{S \subseteq E} \frac{x(S)}{f(S)} \leq 1$, which means $x(S) \leq f(S)$ for all $S \subseteq E$. Conversely, if $x(S) \leq f(S)$ for all $S \subseteq E$ then clearly $t_1 \leq 1$. Moreover, the densities t_ℓ are decreasing by Theorem 3.5, implying $t_\ell \leq 1$ for all ℓ. Thus $w^{(x)} \leq 1$. □

Proposition 3.8 (Duality)**.** *For any $x \in \mathbb{R}^E_{\geq 0}$, we have $L_f(w^{(x)}) = \sum_{e \in E} x_e$.*

Proof. For this, we refer to the convex program formulation of water levels from Theorem 3.5. Taking the optimal primal/dual pair u_e, α_S from Theorem 3.5, the complementary slackness conditions of (SUA) give $\alpha_S \sum_{e \in S} u_e = \alpha_S f(S)$ for each $S \subseteq E$. Summing over those S with $\alpha_S > 0$, we get

$$\sum_{\ell=1}^{L} \alpha_{S_\ell} f(S_\ell) \overset{(a)}{=} \sum_{\ell=1}^{L} \alpha_{S_\ell} \sum_{e \in S_\ell} u_e \overset{(b)}{=} \sum_{\ell=1}^{L} \alpha_{S_\ell} \sum_{e \in S_\ell} \frac{x_e}{w_e} = \sum_{e \in E} \left(\sum_{\ell : S_\ell \ni e} \alpha_{S_\ell} \right) \cdot \frac{x_e}{w_e} \overset{(c)}{=} \sum_{e \in E} x_e.$$

Here, (a) is using $\alpha_S \sum_{e \in S} u_e = \alpha_S f(S)$ for each $S \subseteq E$, (b) is because $u_e = \frac{x_e}{w_e}$ by Theorem 3.5, and (c) is because $\sum_{\ell : S_\ell \ni e} \alpha_{S_\ell} = w_e$, using the fact that the sets S_ℓ are nested and the definition of α_{S_ℓ} in Theorem 3.5. Finally, the LHS is equal to $L_f(w^{(x)})$ by (1) and using $\sum_{\ell : S_\ell \ni e} \alpha_{S_\ell} = w_e$. □

Proposition 3.9 (Upwards Locality[6])**.** *Let $w^{(x)}, w^{(y)}$ be the water level vectors of x, y respectively, where $x \leq y$. Let $\tau = \max\{w_e^{(y)} : y_e \neq x_e\}$ be the largest water level on an element where x and y differ. Then for all $e \in E$ with $w_e^{(y)} > \tau$, we must have $w_e^{(x)} = w_e^{(y)}$.*

Proof. It suffices to show that $\{e \in E : w_e^{(x)} \geq t\} = \{e \in E : w_e^{(y)} \geq t\}$ for all $t > \tau$. Let $A = \{e \in E : x_e = y_e\}$. By Proposition 3.6 and by choice of τ, we have

$$\{e \in E : w_e^{(x)} \geq t\} \subseteq \{e \in E : w_e^{(y)} \geq t\} \subseteq A.$$

However, since x and y are equal on A, we have from Definition 2.1 that

$$\{e \in E : w_e^{(x)} \geq t\} = \{e \in E : w_e^{(y)} \geq t\} = \text{unique maximal set in } \arg\min_{S \subseteq A} (f(S) - x(S)). \qquad \square$$

3.3 Proofs of Equivalence of Water Level Definitions

In this sub-section, we prove the combinatorial decomposition in Algorithm 1 and the SUA market formulation both produce the water levels vector defined in Definition 3.1.

Theorem 3.2. *For any monotone submodular function $f : 2^E \to \mathbb{R}_{\geq 0}$ with $f(\varnothing) = 0$ and $x \in \mathbb{R}^E_{\geq 0}$, the output of Algorithm 1 and the vector $w = w^{(x)}$ defined in Definition 3.1 are equal.*

Proof. Let $\widehat{w}$ be the output of Algorithm 1, and let w be the vector defined in Definition 3.1. To prove $\widehat{w} = w$, we will show that all level sets of $\widehat{w}$ and w are equal. That is, for all $t \geq 0$, we will show

$$\{e \in E : \widehat{w}_e \geq t\} = \{e \in E : w_e \geq t\}.$$

[6]For convenience, we only prove this weaker formulation of locality. Although the version put forth at the beginning of Section 3 is true, it requires a more careful analysis to prove and is not needed for our applications.

Consider the sets $S_0, \dots, S_L$ and levels $t_1, \dots, t_L$ found in Algorithm 2. For completeness, define $t_0 := +\infty$. Notice that by construction of the algorithm, we have for $\ell \in \{0, \dots, L-1\}$ and $t \in (t_{\ell+1}, t_\ell]$,

$$\{e \in E : \widehat{w}_e \geq t\} = S_\ell.$$

Therefore, using Corollary 3.4, it suffices to show that $S_\ell = S_f^*(x/t) = \{e \in E : w_e \geq t\}$ for all such ℓ and t. We do this by proving the following claim through induction on ℓ.

Claim 3.10. *For each $\ell \in \{1, \dots, L\}$ we have*

- $S_f^*(x/t) = S_{\ell-1}$ *for* $t \in (t_\ell, t_{\ell-1})$.
- $S_f^*(x/t_\ell) = S_\ell$

For $\ell = 0$ and $t \in (t_1, \infty)$, notice that every non-empty set $S \subseteq E$ must have density $\frac{x(S)}{f(S)} \leq t_1 < t$. Therefore, we have $f(S) - x(S)/t > 0$, so $S_f^*(t) = \varnothing$. Moreover, when $t = t_1$, then $f(S) - x(S)/t_1 \geq 0$ for all S, and S_1 is the maximal set with $f(S) - x(S)/t_1 = 0$, so we have $S_f^*(x/t_1) = S_1$.

Now, assume that that for some ℓ, we have $S_\ell = S_f^*(\frac{x}{t_\ell})$. Consider $t \in [t_{\ell+1}, t_\ell)$. Notice that by Corollary 3.4 and our inductive hypothesis, we know that

$$S_f^*\left(\frac{x}{t}\right) \supseteq S_f^*\left(\frac{x}{t_\ell}\right) = S_\ell.$$

Therefore, we can write[7]

$$\begin{aligned} S_f^*\left(\frac{x}{t}\right) &= \arg\min_{S \supseteq S_\ell} (f(S) - x(S)/t), \\ &= S_\ell \cup \left[\arg\min_{T \subseteq E \setminus S_\ell} (f(S_\ell \cup T) - x(S_\ell \cup T)/t)\right], \\ &= S_\ell \cup \left[\arg\min_{T \subseteq E \setminus S_\ell} (f_\ell(T) - x(T)/t)\right], \end{aligned}$$

where $f_\ell(T) := f(S_\ell \cup T) - f(S_\ell)$ as in Algorithm 1. Again, we notice that if $t \in (t_{\ell+1}, t_\ell)$, then $f_\ell(T) - x(T)/t > 0$ for any non-empty T, so $S_f^*\left(\frac{x}{t}\right) = S_\ell \cup \varnothing = S_\ell$. On the other hand, if $t = t_\ell$, then $f_\ell(T) - x(T)/t_{\ell+1} \geq 0$ for all T, and $T_{\ell+1}$ from Algorithm 1 is the unique maximal set with $f_\ell(T) - x(T)/t_{\ell+1} = 0$. Therefore, $S_f^*\left(\frac{x}{t_{\ell+1}}\right) = S_\ell \cup T_{\ell+1} = S_{\ell+1}$, as desired. □

Theorem 3.5. *Consider the vector w, the nested sets $S_1 \subset \cdots \subset S_L$, and the levels $t_1, \dots, t_L$ generated by Algorithm 1. Then $t_1 > \cdots > t_L \geq 0$. Moreover, if we define*

$$\begin{aligned} \alpha_{S_L} &:= t_L \\ \alpha_{S_\ell} &:= t_\ell - t_{\ell+1} \quad \ell = 1, \dots, L-1 \end{aligned}$$

and $\alpha_S = 0$ for all other $S \subseteq E$, then, $u_e = \frac{x_e}{w_e}$ is an optimal primal solution to (SUA) and α_S is an optimal dual solution.

Algorithm 2: `An Alternate Combinatorial Presentation of Water Levels`

input : A point $x \in \mathbb{R}^n_{\geq 0}$.

1 Initialize $t = 0$, all elements are considered "unfrozen"

2 **while** *there exists an unfrozen element* **do**

3 Raise t until the vector

$$x^{(t)} = \begin{cases} t \cdot x_e & \text{if } e \text{ is unfrozen} \\ t_{\text{frozen}}(e) \cdot x_e & \text{if } e \text{ is frozen} \end{cases}$$

has a tight set including at least one unfrozen element.

4 Freeze all the elements in the (unique) largest such tight set S_t of $t \cdot x$.

5 We set all newly frozen elements in S to have $t_{\text{frozen}}(e) := t$.

6 Set $w_e = \frac{1}{t}$ for all $e \in S_t$.

output: A vector w.

Proof. We begin with an rephrasing of Algorithm 1. Instead of contracting elements as we did in Algorithm 1, we "freeze" elements in Algorithm 2. Scaling x until some set is saturated is equivalent to measuring the multiplicative slack of sets; therefore, the densities in Line 4 of Algorithm 1 are precisely the time steps at which we freeze a new set of elements in Line 4 of Algorithm 2.

We use the KKT conditions to show u_e and α_S are optimal. Denote the Lagrange multipliers for the constraints $u_e \geq 0$ by μ_e. We will set $\mu_e = 0$ if $x_e > 0$ and $\mu_e = w_e$ otherwise. The KKT conditions are as follows:

- *Primal Feasibility:* $\sum_{e \in S} u_e \leq f(S)$ for all $S \subseteq E$.
- *Dual Feasibility:* $\alpha_S \geq 0$ for all $S \subseteq E$.
- *Stationarity Conditions:* For all $e \in E$,

$$\frac{x_e}{u_e} = \sum_{S \ni e} \alpha_S - \mu_e.$$

- *Complementary Slackness:* $\alpha_S > 0$ implies $\sum_{e \in S} u_e = f(S)$ and $\mu_e \cdot u_e = 0$.

Primal feasibility follows from the fact that Algorithm 2 maintains feasibility of $t \cdot x$ on unfrozen elements. Dual feasibility also easily follows since t only rises. If $x_e > 0$, then $\mu_e = 0$ and

$$\frac{x_e}{u_e} = w_e = \sum_{S \ni e} \alpha_S$$

as desired. Otherwise, if $x_e = 0$, then since $\mu_e = w_e$, the stationary condition still holds. Lastly, we check complementary slackness. We have a positive α_S precisely on the sets $E_1, \ldots, E_L$, and so it suffices to check these sets are tight. Indeed, by definition of Algorithm 2, these sets are tight. Lastly, we have that if $x_e > 0$, then $\mu_e = 0$ and that finishes our complementary slackness conditions. □

[7] For conciseness, we simply use arg min to denote the unique maximal minimizing set.

4 A Fractional Algorithm for Online (Poly-)Matroid Intersection

We now formally present the water-filling algorithm in Algorithm 3. The program which computes the allocation at each time step is given in Line 3. Despite having an exponential number of constraints, this optimization problem is polynomial-time solvable up to any desired accuracy because (i) the objective is concave and (ii) the constraints admit a polynomial-time separation oracle.[8]

Algorithm 3: `Submodular Water-Filling Algorithm`

input : A monotone submodular function f, and parts $Q_1, \ldots, Q_m$ from a partition matroid arriving online, both over ground set E.

output: A fractional allocation $x \in \mathcal{P}_f \cap \mathcal{Q}$, with $\sum_e x_e$ at least $(1 - 1/e) \cdot \mathsf{OPT}$.

1 **while** *some part Q_j arrives:* **do**

2 Let $E_j := Q_1 \cup \cdots \cup Q_j$ denote the elements revealed so far.

3 Choose allocation $(x_e : e \in Q_j)$ by solving the following convex program:

$$\begin{aligned} \max_{x,u} \quad & \sum_{e \in Q_j} x_e(1 - \log x_e) + \sum_{e \in E_j} x_e \log u_e \\ \text{s.t.} \quad & \textstyle\sum_{e \in Q_j} x_e \leq 1 \\ & u(S) \leq f(S) \qquad \forall\, S \subseteq E_j \\ & x_e, u_e \geq 0. \end{aligned}$$

Note that the variables are $(x_e : e \in Q_j)$ and $(u_e : e \in E_j)$. The x_e's for the parts that have previously arrived are fixed to their existing values.

4 **return** the resulting allocation x

If our definition of water levels behaves in the desired way, then the allocation x_e for $e \in Q_j$ should satisfy the following property at the end of Q_j's allocation:

Water-Filling Property. Consider any part Q_j, and let w denote the water level vector at the end of Q_j's allocation. Then

(1) For each $e \in Q_j$ with $x_e > 0$, we have $w_e = \min_{e' \in Q_j} w_{e'}$.

(2) If $\sum_{e \in Q_j} x_e < 1$, then $w_e = 1$ for all $e \in Q_j$.

Lemma 4.1. *The vector x returned by Algorithm 3 is a feasible allocation, and it satisfies the water-filling property above.*

Proof. Consider any j, and let x be the resulting vector after Q_j's allocation; we will show that it is feasible and that it satisfies the water-filling property. To prove this, we use the KKT conditions of the convex program solved in Line 3 of Algorithm 3. Introduce dual variable λ for the constraint $\sum_{e \in Q_j} x_e \leq 1$, variables α_S for the constraint $u(S) \leq f(S)$, and μ_e, θ_e for the non-negativity

[8] The constraints $u(S) \leq f(S)$ can be separated by finding a subset $S \subseteq E$ which minimizes $f(S) - u(S)$, which is a submodular minimization problem. The other constraints are clearly easily separated.

constraints of x_e and u_e, respectively. Then, the Lagrangian of the convex program is:

$$\mathcal{L}(x, u; \lambda, \alpha, \mu, \theta) = \sum_{e \in Q_j} x_e(1 - \log x_e) + \sum_{e \in E_j} x_e \log u_e$$
$$+ \lambda \left(1 - \sum_{e \in Q_j} x_e\right) + \sum_{S \subseteq E_j} \alpha_S(f(S) - u(S)) + \sum_{e \in Q_j} \mu_e x_e + \sum_{e \in E_j} \theta_e u_e.$$

The KKT conditions imply that at optimality, the primal/dual solutions satisfy

1. (Stationarity)
   - For all $e \in Q_j$, $\frac{\partial \mathcal{L}}{\partial x_e} = 0$: $-\log(x_e) + \log(u_e) - \lambda + \mu_e = 0$.
   - For all $e \in E_j$, $\frac{\partial \mathcal{L}}{\partial u_e} = 0$: $\frac{x_e}{u_e} - \sum(\alpha_S : e \in S) + \theta_e = 0$.
2. (Complementary Slackness)
   - For all $e \in Q_j$, $x_e \mu_e = 0$.
   - For all $e \in E_j$, $u_e \theta_e = 0$.
   - For all $S \subseteq E_j$, $\alpha_S(f(S) - u(S)) = 0$.
   - $\lambda(1 - \sum_{e \in Q_j} x_e) = 0$.
3. (Feasibility) x, u are feasible to the convex program, and the dual variables λ, α, μ, and θ are non-negative.

Another fact we will use is that the water levels corresponding to x are exactly $w_e = \frac{x_e}{u_e}$ for all $e \in E$, by Theorem 3.5.

With these tools in hand, we first show that x is a feasible allocation. Let x_0 and w_0 denote the allocation and water level vector, respectively, before part Q_j arrived. Recall from Proposition 3.7 that an allocation is feasible if and only if all water levels are at most 1. Inductively, we may assume that x_0 is feasible, so $w_0 \leq 1$. To show that x remains feasible, it suffices to show that $w_e \leq 1$ for all $e \in E$. The first stationarity condition implies that for all $e \in Q_j$, we have

$$w_e = \frac{x_e}{u_e} = e^{-\lambda + \mu_e}. \tag{2}$$

If $x_e > 0$, then $\mu_e = 0$ from complementary slackness, which implies $w_e = e^{-\lambda} \leq 1$. Furthermore, Proposition 3.9 (locality) implies that for any element e whose allocation did not change during Q_j's arrival, its water level w_e is at most the largest water level of an element whose allocation did change. Thus, $w_e \leq 1$ for all $e \in E$, so x is feasible.

We now show that the water-filling property holds at the end of Q_j's allocation.

(1) Consider any $e \in Q_j$ with $x_e > 0$. From (2), we have

$$w_e = \frac{x_e}{u_e} = e^{-\lambda + \mu_e} \geq e^{-\lambda},$$

where the final inequality is because $\mu_e \geq 0$. If $x_e > 0$, we know from complementary slackness that $\mu_e = 0$, implying $w_e = e^{-\lambda}$. Thus we have shown $w_e \geq e^{-\lambda}$ for all $e \in Q_j$, with equality if $x_e > 0$. This proves part (1) of the water-filling property.

(2) Suppose $\sum_{e \in Q_j} x_e < 1$. By complementary slackness, this implies $\lambda = 0$. Hence, by (2), we have for all $e \in Q_j$

$$w_e = \frac{x_e}{u_e} = e^{\mu_e} \geq 1.$$

However, we already showed that $w_e \leq 1$ for all $e \in E$. Thus, $w_e = 1$ for all $e \in Q_j$. □

4.1 Analysis

Now that we have shown that our algorithm maintains the water-filling property, we proceed with a primal-dual analysis to prove Theorem 1.1. Recall the dual program from Section 2.1:

$$\begin{aligned} \min \quad & L_f(\gamma) + \sum_{j=1}^{n} \beta_j \\ \text{s.t} \quad & \gamma_e + \beta_{j(e)} \geq 1, \qquad \text{for all } e \in E \\ & \gamma, \beta \geq 0. \end{aligned}$$

where L_f is the Lovász extension of f, and $j(e)$ is defined such that $e \in Q_{j(e)}$. We will construct a set of dual variables based on the primal allocation of Algorithm 3. Specifically, after each arrival of Q_j and allocation of primal values, we update the dual variables by recomputing the water levels $w = w^{(x)}$, then assigning:

$$\begin{aligned} \gamma_e &:= G(w_e) \quad \text{for all } e \in E, \\ \beta_j &:= \left(1 - g\left(\min_{e \in Q_j} w_e\right)\right) \sum_{e \in Q_j} x_e. \end{aligned}$$

where $g(x) := e^{x-1}$, and $G(x) := \int_0^x g(t)\,dt = e^{x-1} - e^{-1}$. Note that after Q_j's allocation, the values of γ_e are updated for all $e \in E$ revealed thus-far, but the only β variable that is updated is β_j. Observe that since the algorithm only increases the primal allocation x component-wise, then by Proposition 3.6, the dual variables also only increase as the algorithm progresses.

To show a $1 - 1/e$ competitive ratio, we need to show approximate feasibility of the dual, and that the dual increase is at most the primal increase.

4.1.1 Approximate Feasibility

Focus on a particular element $e \in E$. Let $j = j(e)$, so $e \in Q_j$. We simply check two cases: if $w_e = 1$, then

$$\gamma_e + \beta_j \geq \gamma_e = G(1) = 1 - 1/e.$$

Otherwise if $w_e < 1$, then by part (2) of the water-filling property (Lemma 4.1) we must have $\sum_{e' \in Q_j} x_{e'} = 1$. Thus, we have

$$\gamma_e + \beta_j = G(w_e) + 1 - g\left(\min_{e' \in Q_j} w_{e'}\right) \geq G(w_e) + 1 - g(w_e) \geq 1 - 1/e,$$

since $G(w_e) = g(w_e) - 1/e$.

4.1.2 ΔPrimal ≥ ΔDual

In order to lower bound the value of the primal solution, we will show that at each arrival, it increases at least as much as the value of the dual solution. When Q_j arrives, the change in the primal is simply $\Delta\text{Primal} = \sum_{e \in Q_j} x_e$. Let $w^{(\text{init})}$ denote the water levels induced by x before the arrival of Q_j, and $w^{(\text{fin})}$ the water levels induced by x after the arrival of Q_j and allocation of primal values. We also denote $q_j^* := \min_{e \in Q_j} w_e^{(\text{fin})}$. Then we may write the change in dual as:

$$\begin{aligned}\Delta\text{Dual} &= \Delta L_f(G(w)) + \left(1 - g\left(q_j^*\right)\right) \cdot \sum_{e \in Q_j} x_e \\ &= \Delta L_f(G(w)) + \Delta\text{Primal} - g\left(q_j^*\right) \cdot \sum_{e \in Q_j} x_e,\end{aligned}$$

where $\Delta L_f(G(w)) = L_f(G(w^{(\text{fin})})) - L_f(G(w^{(\text{init})}))$. Hence, it suffices to show that

$$\Delta L_f(G(w)) \leq g\left(q_j^*\right) \cdot \sum_{e \in Q_j} x_e.$$

Using the integral definition of the Lovász extension, we have

$$\begin{aligned}\Delta L_f(G(w)) &= L_f(G(w^{(\text{fin})})) - L_f(G(w^{(\text{init})})) \\ &= \int_0^\infty \left[f(\{e : G(w_e^{(\text{fin})}) \geq t\}) - f(\{e : G(w_e^{(\text{init})}) \geq t\})\right] dt.\end{aligned}$$

Notice that $w_e^{(\text{fin})} = w_e^{(\text{init})}$ for all e such that $w_e^{(\text{fin})} > \min_{e \in Q_j} w_e^{(\text{fin})} = q_j^*$, by Proposition 3.9. Thus, we can truncate this integral at $G(q_j^*)$ and perform the substitution $t = G(u)$ to get

$$\begin{aligned}\Delta L_f(G(w)) &= \int_0^{G(q_j^*)} \left[f(\{e : G(w^{(\text{fin})}) \geq t\}) - f(\{e : G(w^{(\text{init})}) \geq t\})\right] dt \\ &= \int_0^{q_j^*} \left[f(\{e : w^{(\text{fin})} \geq u\}) - f(\{e : w^{(\text{init})} \geq u\})\right] g(u) du \\ &\leq g(q_j^*) \int_0^{q_j^*} \left[f(\{e : w^{(\text{fin})} \geq u\}) - f(\{e : w^{(\text{init})} \geq u\})\right] du \\ &\leq g(q_j^*) \cdot \Delta L_f(w)\end{aligned}$$

Finally, we will use the duality property of water levels from Proposition 3.8: $L_f(w) = \sum_{e \in E} x_e$. This means $\Delta L_f(w) = \sum_{e \in Q_j} x_e$, which gives

$$\Delta L_f(G(w)) \leq g(q_j^*) \sum_{e \in Q_j} x_e$$

as desired.

4.1.3 Proof of the Main Theorem

The above discussion immediately gives the proof of the main theorem:

Proof of Theorem 1.1. Let x be the primal allocation given by Algorithm 3, and γ, β the associated dual assignment defined above. Since we showed that at each step of the algorithm, $\Delta\text{Primal} \geq \Delta\text{Dual}$, we have that $\sum_e x_e \geq L_f(\gamma) + \sum_j \beta_j$. By approximate feasibility, we have that $\gamma' := \frac{e}{e-1} \cdot \gamma$ and $\beta' := \frac{e}{e-1} \cdot \beta$ together form a feasible dual solution. Finally, positive homogeneity[9] of the Lovász extension and duality together give

$$\begin{aligned}\sum_e x_e &\geq L_f(\gamma) + \sum_j \beta_j \\ &= \left(1 - \frac{1}{e}\right) \cdot \left(L_f(\gamma') + \sum_j \beta'_j\right) \\ &\geq \left(1 - \frac{1}{e}\right) \cdot \mathsf{OPT}_{\text{Dual}} = \left(1 - \frac{1}{e}\right) \cdot \mathsf{OPT}.\end{aligned}$$

□

5 Online Submodular Welfare Maximization for Matroidal Utilities

In this section, we give a $(1 - 1/e)$-competitive algorithm for the Online Submodular Welfare Maximization problem where the utility function of each agent is the rank function of a matroid.

Formally, there are n agents, and m items. The items arrive one at a time online. Each agent has an associated utility function $f_i : 2^{[m]} \to \mathbb{Z}_{\geq 0}$ which is the rank function of a matroid $\mathcal{M}_i$ on ground set $[m]$. In each time step, we must irrevocably assign the arriving item to some agent. Suppose items $U_i \subseteq [m]$ have been assigned to agents $i \in [n]$. Then the *welfare* of this allocation is

$$\sum_{i \in [n]} f_i(U_i).$$

The goal is to assign items to maximize welfare, as compared to the optimal offline allocation. We work in the value oracle model, where we can query the value $f_i(S)$ for any $i \in [n]$ and $S \subseteq [m]$ in constant time.

Our algorithm extends to the setting where each agent has a non-negative weight a_i. In this case, the utility function becomes $f_i := a_i \cdot \text{rank}_{\mathcal{M}_i}$.

5.1 The Matroidal Ranking Algorithm

First, some notation. For an allocation of items to agents, we will denote by U_i the set of items assigned to agent i. Given such an allocation, we say that an item j is *available* to agent i if $f_i(U_i + j) > f_i(U_i)$.

The algorithm proceeds as follows. Independently for each agent i, select w_i uniformly at random from $[0, 1]$. Let the *priority* of agent i be defined as $a_i \cdot (1 - g(w_i))$, where $g(z) := e^{z-1}$. When an item arrives, consider the set of agents to whom this item is available, and assign the item to the highest priority agent among these. See Algorithm 4 for a formal description.

Remark 5.1 (Perusal perspective)**.** *We note that the Matroidal Ranking Algorithm yields the same allocation as the following procedure. In order of decreasing priority, each agent "peruses"*

[9] $L_f(\lambda x) = \lambda L_f(x)$ for $\lambda \geq 0$. This follows from the definition of Lovász extension.

the full set of items in their arrival order, and greedily picks any item which increases their utility. While this perusal perspective cannot be implemented online, it yields an identical allocation as the Matroidal Ranking Algorithm, and will be useful for the analysis.

Algorithm 4: `Matroidal Ranking Algorithm`

input : An instance of Matroid-OSWM with agents $i \in [n]$, items $[m]$, and utility functions $f_i : 2^{[m]} \to \mathbb{Z}_{\geq 0}$.

output: An allocation of items $U_i \subseteq [m]$ to each agent i with welfare at least $(1 - \frac{1}{e})$ times the welfare of the optimal offline allocation.

1. Select a value $w_i \in [0, 1]$ uniformly and independently for each agent, and set the priority of agent i to be $a_i(1 - g(w_i))$.
2. When an item j arrives, assign it to the highest priority agent to whom it is available.
3. **return** the resulting allocation U_i.

5.2 Analysis

Consider the primal and dual problems below. The primal has variables x_{ij} representing to what extent item $j \in [m]$ is allocated to agent $i \in [n]$. Notice that in the primal, rather than directly optimizing for the welfare of the agents, we simply maximize the total quantity of items assigned, while the constraints enforce that each agent receives an independent set of items with respect to their matroid. Furthermore, there are constraints for each item enforcing that each is assigned at most once. Thus, an integer binary solution to the primal corresponds to a feasible allocation with objective value equal to the welfare of the allocation.

$$\begin{aligned} \max \quad & \sum_{i \in [n]} \left(a_i \cdot \textstyle\sum_{j \in [m]} x_{ij} \right) \\ \text{s.t.} \quad & x(S) \leq f_i(S), \qquad \forall i \in [n],\, S \subseteq [m] \\ & \textstyle\sum_{i \in [n]} x_{ij} \leq 1, \qquad \forall j \in [m] \\ & x \geq 0 \end{aligned}$$

$$\begin{aligned} \min \quad & \sum_{i \in [n]} \sum_{S \subseteq [m]} f_i(S)\, \alpha_{i,S} + \sum_{j \in [m]} \beta_j \\ \text{s.t.} \quad & \textstyle\sum_{S \ni j} \alpha_{i,S} + \beta_j \geq a_i, \qquad \forall i \in [n],\, j \in [m] \\ & \alpha, \beta \geq 0 \end{aligned}$$

Consider the primal solution $\overline{x}$ induced by the allocation at the end of the Matroidal Ranking algorithm. This $\overline{x}$ depends on the random values w_i which were chosen for each agent. We will construct a dual solution $(\overline{\alpha}, \overline{\beta})$ whose objective value is the same as that of $\overline{x}$, and which is approximately feasible in expectation. In particular, we will have

$$\mathbb{E}_{w \sim [0,1]^n} \left[\sum_{S \ni j} \overline{\alpha}_{i,S} + \overline{\beta}_j \right] \geq \left(\frac{e-1}{e} \right) \cdot a_i.$$

for each $(i, j) \in [n] \times [m]$. This implies that the scaled up solution $(\frac{e}{e-1})(\overline{\alpha}, \overline{\beta})$ is a feasible in expectation, and that $\overline{x}$ is $(\frac{e-1}{e})$-approximately optimal.

Dual Assignment We now define the dual solution $(\overline{\alpha}, \overline{\beta})$. For each agent $i \in [n]$, if U_i is the set of items assigned to agent i at the end of the algorithm, let $S_i := \text{span}_{\mathcal{M}_i}(U_i)$ be the *span* of U_i with respect to $\mathcal{M}_i$ (i.e. the largest set of items containing U_i whose rank is equal to $\text{rank}_{\mathcal{M}_i}(U_i)$). We assign $\overline{\alpha}_{i,S_i} := a_i \cdot g(w_i)$. For each item j, if $j \in U_i$ for some $i \in [n]$, we set $\overline{\beta}_j := a_i \cdot (1 - g(w_i))$. The remaining variables are set to zero.

Primal = Dual The dual solution described above has objective value equal to the primal solution returned by the algorithm. To see this, note that whenever an item is assigned to an agent i, the objective value of the dual increases by exactly a_i. Furthermore, since we only assign an item to an agent if it is available to them, the primal objective value increases by a_i as well.

Expected Approximate Feasibility We now show that the dual solution is approximately feasible in expectation. Fix a particular agent-item pair $(\widehat{i}, \widehat{j})$. We will focus on the dual constraint $\sum_{S \ni \widehat{j}} \alpha_{\widehat{i},S} + \beta_{\widehat{j}} \geq a_{\widehat{i}}$. First, condition on all random choices w_i for $i \neq \widehat{i}$. We denote these choices by $w_{-\widehat{i}}$. Note that, once $w_{-\widehat{i}}$ is fixed, the run of the algorithm is determined by the value of $w_{\widehat{i}}$. Hence, for any choice $w_A \in [0,1]$ we will denote a run of the algorithm in which $w_{\widehat{i}} = w_A$ as A.

For an agent $i \in [n]$ and run A of the algorithm with $w_{\widehat{i}} = w_A \in [0,1]$, let $U_i^{(A,t)} \subseteq [m]$ denote the set of items assigned to agent i at time step t of run A of the algorithm. Likewise, let $U_i^{(A)}$ denote the set of items assigned to agent i at the end of run A of the algorithm. We will also write $\text{span}(U_i^{(A,t)})$ to mean $\text{span}_{\mathcal{M}_i}(U_i^{(A,t)})$, the span in agent i's matroid of the set of items assigned to agent i (and similarly for $\text{span}(U_i^{(A)})$).

We now define the *critical threshold* w^* to be maximum value of $w_{\widehat{i}}$ such that $\widehat{j}$ is in the span of the items assigned to $\widehat{i}$ in the final allocation. Formally,

$$w^* := \sup \left\{ w_A \in [0,1] : \widehat{j} \in \text{span}\left(U_{\widehat{i}}^{(A)}\right) \right\}.$$

We define $\sup(\varnothing) = 0$ by convention.

The following key lemma characterizes several invariants that hold throughout the Matroidal Ranking algorithm. Specifically, it describes how the $\text{span}(U_i^{(A,t)})$ changes as w_A changes. This will allow us to lower bound the expected amount of dual value assigned during the procedure.

Lemma 5.2. *Fix $w_{-\widehat{i}}$, and values w_A and w_B in $[0,1]$ with $w_A < w_B$. Consider the two separate runs of the algorithm: A with $w_{\widehat{i}} = w_A$ and B with $w_{\widehat{i}} = w_B$. Then at each iteration t, we have*

(1) $\text{span}(U_{\widehat{i}}^{(A,t)}) \supseteq \text{span}(U_{\widehat{i}}^{(B,t)})$,

(2) $\text{span}(U_i^{(A,t)}) = \text{span}(U_i^{(B,t)})$ *for all* $i \in [n]$ *with* $w_i \leq w_A$,

(3) If $w_B = 1$, *then* $\text{span}(U_i^{(A,t)}) \subseteq \text{span}(U_i^{(B,t)})$ *for all* $i \neq \widehat{i}$.

Proof. Points (1) and (2) follow from the perusal perspective of Algorithm 4. In particular, for point (1), agent $\widehat{i}$ only peruses earlier in run A than in run B, so $\widehat{i}$ has more items to choose from in run A. For the tth item j, if $j \in U_{\widehat{i}}^{(B,t)}$ and it was not already spanned by $U_{\widehat{i}}^{(A,t-1)}$, then it would be chosen by $\widehat{i}$ in step t of run A. So $U_{\widehat{i}}^{(B,t)} \subseteq \text{span}(U_{\widehat{i}}^{(A,t)})$, which implies point (1).

For point (2), the perusal of all agents i with $w_i < w_A$ is identical in both runs A and B of the algorithm, so in particular, $U_i^{(A,t)} = U_i^{(B,t)}$, implying point (2).

We prove point (3) by induction. Let $w_B = 1$. Suppose for induction (3) holds at iteration t, and a new item j arrives. Consider some $i \neq \widehat{i}$. First, if $j \in \text{span}(U_i^{(B,t)})$ already at time t, then we have by induction

$$\text{span}(U_i^{(A,t+1)}) \subseteq \text{span}\left(\text{span}(U_i^{(A,t)}) \cup \{j\}\right) \subseteq \text{span}(U_i^{(B,t+1)})$$

as desired.

So suppose otherwise that $j \notin \text{span}(U_i^{(B,t)})$. This means that in run B, when item j arrives, it is available to agent i. For the invariant in point (3) to break, item j must be assigned to i in run A but not assigned to i in run B. If j is not assigned to i in run B, it must be assigned to some other i' with $w_{i'} < w_i$ (since when j arrives, it is available to agent i). In particular, since $w_{\widehat{i}} = 1$, we know $i' \neq i$ and we may apply induction to i'. This tells us that at time t (before j's arrival), $\text{span}(U_{i'}^{(A,t)}) \subseteq \text{span}(U_{i'}^{(B,t)})$, and therefore in run A, item j was also available to i'. Since $w_{i'} < w_i$, this contradicts that j is assigned to i in run A. □

This yields the following pair of corollaries.

Corollary 5.3. *If $w_{\widehat{i}} < w^*$ then $\widehat{j} \in \text{span}(U_{\widehat{i}})$*

Proof. This follows directly from the definition of w^*, and point (1) from Lemma 5.2. □

Corollary 5.4. *If $w^* < 1$, then item $\widehat{j}$ is always assigned to an agent with water level at most w^*.*

Proof. Observe first that for any $\varepsilon > 0$, if $w_{\widehat{i}} = w^* + \varepsilon \leq 1$ then item $\widehat{j}$ is assigned to some agent i with $w_i < w^* + \varepsilon$. This is because $w_{\widehat{i}} > w^*$ implies both that item $\widehat{j}$ is *available* to $\widehat{i}$ when it arrives, and that it is not assigned to $\widehat{i}$. Since this holds for every $\varepsilon > 0$ yet there are only finitely many agents, there is some such $i =: i^*$ with $w_{i^*} \leq w^*$, and some $\varepsilon^* > 0$ such that $\widehat{j}$ is assigned to i^* when $w_{\widehat{i}} = w^* + \varepsilon^*$.

Now we claim that for any value of $w_{\widehat{i}}$, item $\widehat{j}$ is always available to i^* when $\widehat{j}$ arrives. This implies the claim. First, we compare instance A of the algorithm with $w_{\widehat{i}} = w_A := w^* + \varepsilon^*$ to any instance B with $w_{\widehat{i}} = w_B > w^* + \varepsilon^*$. By Lemma 5.2(2) applied to i^*, we have $\text{span}(U_{i^*}^{(A,t)}) = \text{span}(U_{i^*}^{(B,t)})$ at the time t when $\widehat{j}$ arrives. So, in particular, $\widehat{j}$ is available to i^* in instance B, since it's available in instance A.

Since the above holds for any $w_B > w^* + \varepsilon$, it in particular holds for $w_B = 1$. Now we apply Lemma 5.2(3) to i^* on any instance A with $w_{\widehat{i}} = w_A < 1$ and instance B with $w_{\widehat{i}} = w_B = 1$. We have $\text{span}(U_{i^*}^{(A,t)}) \subseteq \text{span}(U_{i^*}^{(B,t)})$ at time t when $\widehat{j}$ arrives. Therefore, again, since i^* is available to $\widehat{j}$ in instance B, it is also available in instance A.

So in all cases, $\widehat{j}$ is available to i^* (with water level $w_{i^*} \leq w^*$) when it arrives. So $\widehat{j}$ is always assigned to an agent of water level at most w^*. □

This gives us all ingredients required for the final proof that the Matroidal Ranking algorithm achieves a $(1 - 1/e)$-competitive ratio.

Proof of Theorem 1.2. Let $\overline{x}$ be the primal solution given by the Matroidal Ranking algorithm, and $(\overline{\alpha}, \overline{\beta})$ the corresponding dual solution described above. We showed that the primal and dual objectives are equal: $\sum_i (a_i \cdot \sum_j \overline{x}_{ij}) = \sum_{i,S} f_i(S)\overline{\alpha}_{i,S} + \sum_j \overline{\beta}_j$. To argue that $\overline{x}$ is $(1 - 1/e)$-competitive in expectation with the offline optimal primal solution, it then suffices by duality to show that, in expectation, $\overline{\alpha}, \overline{\beta}$ are approximately feasible.

For any fixed agent-item pair $(\widehat{i}, \widehat{j})$, we condition on the values of $w_{-\widehat{i}}$, and let w^* be the critical threshold. Then Corollary 5.3 implies that

$$\mathbb{E}_{w_{\widehat{i}} \sim [0,1]} \left[\sum_{S \ni \widehat{j}} \overline{\alpha}_{\widehat{i},S} \,\middle|\, w_{-\widehat{i}} \right] \geq \int_0^{w^*} a_{\widehat{i}} \cdot g(z)\, dz = a_{\widehat{i}} \cdot \left(g(w^*) - \frac{1}{e} \right).$$

Similarly, Corollary 5.4 implies that

$$\mathbb{E}_{w_{\widehat{i}} \sim [0,1]} \left[\overline{\beta}_{\widehat{j}} \,\middle|\, w_{-\widehat{i}} \right] \geq \int_0^1 a_{\widehat{i}} \cdot (1 - g(w^*))\, dz = a_{\widehat{i}} \cdot (1 - g(w^*)).$$

(note if $w^* = 1$, then the RHS is 0, so the inequality still holds, despite Corollary 5.4 not applying). The sum is then $a_{\widehat{i}} \cdot (1 - 1/e)$, and since this does not depend on the conditional values of $w_{-\widehat{i}}$, we may drop the conditioning to get

$$\mathbb{E}_{w \sim [0,1]^n} \left[\sum_{S \ni \widehat{j}} \overline{\alpha}_{\widehat{i},S} + \overline{\beta}_{\widehat{j}} \right] \geq a_{\widehat{i}} \cdot \left(1 - \frac{1}{e} \right)$$

as desired. □

6 Future Directions

In this paper, we presented the problem of Online Matroid Intersection with part arrivals and showed two settings where a $(1 - 1/e)$-competitive algorithm can be achieved. There are several open questions.

- *Integral Online Matroid Intersection:* The most natural question that remains is to determine whether a $1 - 1/e$ (or even $1/2 + \epsilon$) competitive algorithm exists for general (integral) Online Matroid Intersection with part arrivals. We note that this setting does not obviously follow from either of our current approaches. The ranking algorithm used in Section 5 does not easily extend to general offline matroids $\mathcal{M}$, and it is known that a $(1 - 1/e)$-approximate fractional solution cannot be rounded online to an integral solution even for bipartite matching [DJ12].

- *Item-weighted Matroidal OSWM:* We may also consider the OSWM setting where valuations f_i are weighted matroid rank functions (WMRF) as defined in Section 1.1. Recall that this generalizes both the agent-weighted matroidal valuations considered in Section 5 and the edge-weighted online bipartite matching setting, in which $> 1/2$ competitive algorithms are known [FHTZ20, SA21, GHH+21, BC21]. Therefore, we suspect that a $1/2 + \epsilon$ competitive algorithm should exist in this setting as well.

References

[AGKM11] Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1253–1264. SIAM, 2011. 4, 5

[AS21] Susanne Albers and Sebastian Schubert. Optimal algorithms for online b-matching with variable vertex capacities. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 2:1–2:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. 4, 5

[BC21] Guy Blanc and Moses Charikar. Multiway online correlated selection. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 1277–1284. IEEE, 2021. 4, 5, 22

[BCIZ21] Nawal Benabbou, Mithun Chakraborty, Ayumi Igarashi, and Yair Zick. Finding fair and efficient allocations for matroid rank valuations. *ACM Transactions on Economics and Computation*, 9(4):1–41, 2021. 5

[BEF21] Moshe Babaioff, Tomer Ezra, and Uriel Feige. Fair and truthful mechanisms for dichotomous valuations. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 5119–5126. AAAI Press, 2021. 5

[BGH+23] Niv Buchbinder, Anupam Gupta, Daniel Hathcock, Anna R. Karlin, and Sherry Sarkar. Maintaining matroid intersections online. *CoRR*, abs/2309.10214, 2023. 5

[BNW23] Niv Buchbinder, Joseph (Seffi) Naor, and David Wajc. Lossless online rounding for online bipartite matching (despite its impossibility). In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 2030–2068. SIAM, 2023. 5

[BV22] Siddharth Barman and Paritosh Verma. Truthful and fair mechanisms for matroid-rank valuations. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022*, pages 4801–4808. AAAI Press, 2022. 5

[CCPV11] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011. 2, 5

[CS12] Mosharaf Chowdhury and Ion Stoica. Coflow: a networking abstraction for cluster applications. In Srikanth Kandula, Jitendra Padhye, Emin Gün Sirer, and Ramesh Govindan, editors, *11th ACM Workshop on Hot Topics in Networks, HotNets-XI, Redmond, WA, USA - October 29 - 30, 2012*, pages 31–36. ACM, 2012. 1

[CW23] Karthekeyan Chandrasekaran and Weihang Wang. Approximating submodular k-partition via principal partition sequence. In Nicole Megow and Adam D. Smith, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2023, September 11-13, 2023, Atlanta, Georgia, USA*, volume 275 of *LIPIcs*, pages 3:1–3:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. 6

[DFF21] Shahar Dobzinski, Uriel Feige, and Michal Feldman. Are gross substitutes a substitute for submodular valuations? In Péter Biró, Shuchi Chawla, and Federico Echenique, editors, *EC '21: The 22nd ACM Conference on Economics and Computation, Budapest, Hungary, July 18-23, 2021*, pages 390–408. ACM, 2021. 4

[DFSH23] Amitay Dror, Michal Feldman, and Erel Segal-Halevi. On fair division under heterogeneous matroid constraints. *Journal of Artificial Intelligence Research*, 76:567–611, 2023. 5

[DJ12] Nikhil R. Devanur and Kamal Jain. Online matching with concave returns. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 137–144. ACM, 2012. 22

[DJK13] Nikhil R. Devanur, Kamal Jain, and Robert D. Kleinberg. Randomized primal-dual analysis of ranking for online bipartite matching. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, page 101–107, USA, 2013. Society for Industrial and Applied Mathematics. 4, 5

[Edm65] Jack Edmonds. Minimum partition of a matroid into independent subsets. *J. Res. Nat. Bur. Standards Sect. B*, 69B:67–72, 1965. 1

[Fei98] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998. 5

[Fei19] Uriel Feige. *Tighter Bounds for Online Bipartite Matching*, pages 235–255. Springer Berlin Heidelberg, Berlin, Heidelberg, 2019. 2

[FHTZ20] Matthew Fahrbach, Zhiyi Huang, Runzhou Tao, and Morteza Zadimoghaddam. Edge-weighted online bipartite matching. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 412–423. IEEE, 2020. 5, 22

[Fuj08] Satoru Fujishige. Theory of principal partitions revisited. In William J. Cook, László Lovász, and Jens Vygen, editors, *Research Trends in Combinatorial Optimization, Bonn Workshop on Combinatorial Optimization, November 3-7, 2008, Bonn, Germany*, pages 127–162. Springer, 2008. 3, 5

[GHH+21] Ruiquan Gao, Zhongtian He, Zhiyi Huang, Zipei Nie, Bijun Yuan, and Yan Zhong. Improved online correlated selection. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 1265–1276. IEEE, 2021. 5, 22

[GS17] Guru Prashanth Guruganesh and Sahil Singla. Online matroid intersection: Beating half for random arrival. In Friedrich Eisenbrand and Jochen Koenemann, editors, *Integer Programming and Combinatorial Optimization*, pages 241–253, Cham, 2017. Springer International Publishing. 5

[HS23] Chien-Chung Huang and François Sellier. Robust sparsification for matroid intersection with applications. *CoRR*, abs/2310.16827, 2023. 5, 6

[IMPP19] Sungjin Im, Benjamin Moseley, Kirk Pruhs, and Manish Purohit. Matroid coflow scheduling. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 145:1–145:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. 1

[JKR17] Hamidreza Jahanjou, Erez Kantor, and Rajmohan Rajaraman. Asymptotically optimal approximation algorithms for coflow scheduling. In Christian Scheideler and Mohammad Taghi Hajiaghayi, editors, *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2017, Washington DC, USA, July 24-26, 2017*, pages 45–54. ACM, 2017. 1

[JV10] Kamal Jain and Vijay V. Vazirani. Eisenberg-gale markets: Algorithms and game-theoretic properties. *Games Econ. Behav.*, 70(1):84–106, 2010. 3, 10

[Knu73] Donald E. Knuth. Matroid partitioning. Technical report, Stanford University, Stanford, CA, USA, 1973. 1

[KPV13] Michael Kapralov, Ian Post, and Jan Vondrák. Online submodular welfare maximization: Greedy is optimal. 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 1216–1225. SIAM, 2013. 2, 4

[KVV90] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 352–358. ACM, 1990. 1, 4, 5

[Nar91] H. Narayanan. The principal lattice of partitions of a submodular function. *Linear Algebra and its Applications*, 144:179–216, 1991. 3, 5

[NRTV07] Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007. 3

[PPSW21] Christos H. Papadimitriou, Tristan Pollner, Amin Saberi, and David Wajc. Online stochastic max-weight bipartite matching: Beyond prophet inequalities. In Péter Biró, Shuchi Chawla, and Federico Echenique, editors, *EC '21: The 22nd ACM Conference on Economics and Computation, Budapest, Hungary, July 18-23, 2021*, pages 763–764. ACM, 2021. 5

[SA21] Yongho Shin and Hyung-Chan An. Making three out of two: Three-way online correlated selection. In Hee-Kap Ahn and Kunihiko Sadakane, editors, *32nd International Symposium on Algorithms and Computation, ISAAC 2021, December 6-8, 2021, Fukuoka, Japan*, volume 212 of *LIPIcs*, pages 49:1–49:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. 5, 22

[Sch03] Alexander Schrijver. *Combinatorial optimization. Polyhedra and efficiency. Vol. B*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003. Matroids, trees, stable sets, Chapters 39–69. 5

[Sot13] José A. Soto. Matroid secretary problem in the random-assignment model. *SIAM J. Comput.*, 42(1):178–211, 2013. 6

[VZ23] Vignesh Viswanathan and Yair Zick. A general framework for fair allocation under matroid rank valuations. In Kevin Leyton-Brown, Jason D. Hartline, and Larry Samuelson, editors, *Proceedings of the 24th ACM Conference on Economics and Computation, EC 2023, London, United Kingdom, July 9-12, 2023*, pages 1129–1152. ACM, 2023. 5