



# Online Multiset Submodular Cover

Magnús M. Halldórsson<sup>1</sup> · Dror Rawitz<sup>2</sup>

Received: 29 June 2023 / Accepted: 23 April 2024 / Published online: 8 May 2024  
© The Author(s) 2024

## Abstract

We study the ONLINE MULTISSET SUBMODULAR COVER problem (OMSC), where we are given a universe  $U$  of elements and a collection of subsets  $\mathcal{S} \subseteq 2^U$ . Each element  $u_j \in U$  is associated with a nonnegative, nondecreasing, submodular polynomially computable set function  $f_j$ . Initially, the elements are uncovered, and therefore we pay a penalty per each unit of uncovered element. Subsets with various coverage and cost arrive online. Upon arrival of a new subset, the online algorithm must decide how many copies of the arriving subset to add to the solution. This decision is irrevocable, in the sense that the algorithm will not be able to add more copies of this subset in the future. On the other hand, the algorithm can drop copies of a subset, but such copies cannot be retrieved later. The goal is to minimize the total cost of subsets taken plus penalties for uncovered elements. We present an  $O(\sqrt{\rho_{\max}})$ -competitive algorithm for OMSC that does not dismiss subset copies that were taken into the solution, but relies on prior knowledge of the value of  $\rho_{\max}$ , where  $\rho_{\max}$  is the maximum ratio, over all subsets, between the penalties covered by a subset and its cost. We provide an  $O(\log(\rho_{\max})\sqrt{\rho_{\max}})$ -competitive algorithm for OMSC that does not rely on advance knowledge of  $\rho_{\max}$  but uses dismissals of previously taken subsets. Finally, for the capacitated versions of the ONLINE MULTISSET MULTICOVER problem, we obtain an  $O(\sqrt{\rho'_{\max}})$ -competitive algorithm when  $\rho'_{\max}$  is known and an  $O(\log(\rho'_{\max})\sqrt{\rho'_{\max}})$ -competitive algorithm when  $\rho'_{\max}$  is unknown, where  $\rho'_{\max}$  is the maximum ratio over all subset incarnations between the penalties covered by this incarnation and its cost.

**Keywords** Competitive analysis · Online multiset multicover · Capacitated covering · Team formation · Submodular coverage

---

✉ Dror Rawitz  
dror.rawitz@biu.ac.il

Magnús M. Halldórsson  
mmh@ru.is

<sup>1</sup> School of Computer Science, Reykjavik University, 101 Reykjavik, Iceland

<sup>2</sup> Faculty of Engineering, Bar Ilan University, 52900 Ramat Gan, Israel

## 1 Introduction

We study the MULTISSET MULTICOVER problem, in which we are given a universe  $U$  with  $n$  elements, where element  $u_j$  has a *covering requirement*  $b_j \in \mathbb{N}$  and a *penalty*  $p_j \in \mathbb{R}_+$ . There is a collection  $\mathcal{S}$  of  $m$  subsets, where the *coverage* of subset  $S_i$  of element  $u_j$  is  $a_{ij} \in \mathbb{N}$  and its *cost* is  $c_i$ . Without loss of generality, we assume that  $a_{ij} \leq b_j$ , for every  $i$  and  $j$ . (Note that a subset  $S_i$  is actually a multiset if  $a_{ij} > 1$ , for some element  $j$ , but we misuse the term “subset” for simplicity.) A solution is a multicollection  $\mathcal{C}$  of the subsets, where multiple copies of subsets are allowed. The cost of a solution is the cost of selected subsets plus the penalties for unsatisfied covering requirements. One may formulate MULTISSET MULTICOVER using the following integer linear program:

$$\begin{aligned} \min \quad & \sum_{i=1}^m c_i x_i + \sum_{j=1}^n p_j z_j \\ \text{s.t.} \quad & \sum_{i=1}^m a_{ij} x_i + z_j \geq b_j \quad \forall u_j \in U \\ & x_i \in \mathbb{N} \quad \forall S_i \in \mathcal{S} \\ & z_j \in \mathbb{N} \quad \forall u_j \in U \end{aligned} \tag{MM}$$

where the value of  $x_i$  represents the number of copies of subset  $S_i$  taken into the solution  $\mathcal{C}$ , and  $z_j$  is the amount of unsatisfied coverage by  $\mathcal{C}$  of element  $u_j$  (for which we pay a penalty per unit). Notice that, given a solution  $\mathcal{C}$  one can easily obtain the corresponding pair  $(x, z)$ . We define:

$$\text{COST}(\mathcal{C}) \triangleq \sum_i c_i x_i + \sum_j p_j z_j.$$

The above formulation is *unconstrained* in the sense that each subset can be used any number of times. In the *constrained* variant a subset can be used at most once, i.e.,  $x_i \in \{0, 1\}$ , for every  $i$ .

We study (generalizations of) the following online version of the MULTISSET MULTICOVER problem, called ONLINE MULTISSET MULTICOVER (OMM), that was introduced by Fraigniaud et al. [15]. Initially, the elements are uncovered—and hence incur a penalty per each unit of uncovered element. That is, initially we pay  $\sum_j p_j b_j$ . Subsets with various coverage and cost arrive online. In each time step, a new subset arrives, and the online algorithm must decide how many copies of the arriving subset to add to the solution. This decision is irrevocable, in the sense that the algorithm will not be able to add more copies of this subset in the future. On the other hand, the algorithm is allowed to drop copies of a subset, but such copies cannot be retrieved later. The goal is to minimize the total cost of subsets taken plus penalties for uncovered elements. As usual with online algorithms we measure an online algorithm by its *competitive ratio*: in this case, the maximum ratio of cost incurred by the algorithm to the best possible cost for any given instance. We consider expected cost if the algorithm is randomized.

OMM is an abstraction of the following *team formation* scenario (corresponding to a binary matrix  $A$  and binary requirements  $b$ ). Suppose we are embarking on a new project that is composed of  $n$  tasks. A task  $j$  can be satisfied by outsourcing at cost  $p_j$ , or by hiring an employee who possesses the ability or skill to perform task  $j$ . The goal is to minimize the project cost under the following procedure: We interview candidates one by one. After each interview we know the skills and the hiring cost of the candidate and must decide irrevocably whether to hire the candidate or not. In the more general setting, a task may require more effort, and this is captured by element requirements. A candidate  $i$  may be able to contribute more than one unit of effort to a task  $j$  and this is captured by the coverage  $a_{ij}$ . Finally, the unconstrained case is more appropriate when a candidate corresponds to a technology that may be purchased more than once.

As shown in [15], even randomized algorithms cannot have competitive ratio better than  $\Omega(\sqrt{\rho_{\max}})$ , where  $\rho_{\max}$  is the maximum ratio, over all subsets, between the penalties covered by a subset and its cost (see Sect. 2 for an exact definition). The lower bound holds even in the binary case and even if  $\rho_{\max}$  is known and one is allowed to drop previously selected subsets. On the other hand, they presented a deterministic algorithm with a competitive ratio of  $O(\sqrt{\rho_{\max}})$  that does not dismiss subsets that were taken into the solution. This algorithm works for both the constrained and unconstrained cases. However, this algorithm relies on prior knowledge of the value of  $\rho_{\max}$ . It was also shown that without dismissals and without such knowledge only the trivial  $O(\rho_{\max})$  bound is possible. It remained an open question whether there is an  $o(\rho_{\max})$ -competitive algorithm that has no knowledge of  $\rho_{\max}$  but allows dismissals.

We note that in online algorithms the initial solution is typically empty and its value increases with time. In contrast, the algorithm from [15] is unique in the sense that a solution get progressively *cheaper* with time. This is a natural dimension that is worth exploring more widely. We contribute to this by treating (just about) the most general forms of covering problems known. We consider two extensions of OMM in this paper.

First, we consider OMM with submodular coverage. Assume that each element  $u_j$  is associated with a nonnegative, nondecreasing, submodular polynomially computable set function  $f_j$ , whose domain consists of multi-collection of subsets and its range is  $[0, b_j]$ . Recall that a function  $g$  is nondecreasing if  $g(\mathcal{A}) \leq g(\mathcal{B})$ , for all  $\mathcal{A} \subseteq \mathcal{B}$ , and  $g$  is submodular if  $g(\mathcal{A}) + g(\mathcal{B}) \geq g(\mathcal{A} \cup \mathcal{B}) + g(\mathcal{A} \cap \mathcal{B})$ , for every two subcollections  $\mathcal{A}$  and  $\mathcal{B}$ . Assume that  $f_j(\emptyset) = 0$ , for every  $j$ . Also, note that the domain of  $f_j$  is revealed in an online manner. Given a multicollection  $\mathcal{C}$  of  $\mathcal{S}$ , let  $x_i$  be the multiplicity of  $S_i$  in  $\mathcal{C}$  (as defined above). Also, let  $z_j = b_j - f_j(\mathcal{C})$ . Going back to the team formation scenario, a submodular function captures the case where several employees that work together on a task need to spend some effort on maintaining team work, and thus their combined productivity is less than the sum of their productivities. We refer to this problem as **ONLINE MULTISUBSET SUBMODULAR COVER** (abbreviated **OMSC**).

In another extension of OMM each subset  $S_i$  has a capacity  $\text{cap}(S_i)$  that limits the amount of coverage of a single copy of  $S_i$ . More specifically, when a subset  $S_i$  arrives, it can potentially cover  $a_{ij}$  of element  $j$ , for every  $j$ , but it can only cover at most  $\text{cap}(S_i)$  out of this potential coverage. In this case a solution can be described as a

pair  $(\mathcal{C}, y)$ , where  $\mathcal{C}$  is a multicollection and  $y$  satisfies the following conditions: (i)  $y_{ij} \leq a_{ij}$ , for every  $i$  and  $j$ , (ii)  $\sum_i y_{ij} \leq b_j$ , for every  $j$ , and (iii)  $\sum_j y_{ij} \leq \text{cap}(S_i)$ , for every  $i$ . We refer to  $y$  as an *incarnation function*, since it determines the actual coverage of each subset  $S_i$ . This capacitated version of OMM has two flavors. In the *soft capacities* case (unconstrained) one is allowed to use multiple copies of any subset  $S_i$ , provided that one pays for each such copy. On the other hand, in the *hard capacities* case (constrained) one is allowed to use each subset at most once.

Capacities may describe the setting where a candidate has the potential to work on many tasks, but has a limited workload capacity. For example, there may be two tasks that require six hours a day, each. That is,  $b_1 = b_2 = 6$ . Assume that there is a candidate who can work eight hours per day, but at most five hours per task. In addition, this person may be assigned to both tasks, as long as each assignment is for at most five hours. In this case,  $a_{11} = a_{12} = 5$  and  $\text{cap}(S_1) = 8$ . If this person is assigned to work five hours on task 1 and three on task 2, then we are using the incarnation  $y_{11} = 5$  and  $y_{12} = 3$ .

## 1.1 Our Contribution

We present an  $O(\sqrt{\rho_{\max}})$ -competitive algorithm for OMSC that does not dismiss subsets that were taken into the solution, but relies on prior knowledge of the value of  $\rho_{\max}$ . It works for both the constrained and unconstrained cases. This competitive ratio is tight for the setting where  $\rho_{\max}$  is known, since there is an  $\Omega(\sqrt{\rho_{\max}})$  lower bound on the competitive ratio of randomized algorithms even for the special case of OMM with  $b = 1$  [15] and even if dismissals are possible. Our algorithm is an extension of the one for OMM from [15], and it is based on two upper bounds, one for the total cost of the subsets and one for the penalties. The next result required a parameterized version of the former bound.

Our main result is an  $O(\log(\rho_{\max})\sqrt{\rho_{\max}})$ -competitive algorithm for OMSC that does not rely on advance knowledge of  $\rho_{\max}$ . This result is new also for OMM. In contrast to the first algorithm, this one uses dismissals of previously taken subsets. Our algorithm does not use standard online methods, like Doubling or Classify & Select. Instead, it executes the above mentioned algorithm, for every power of 2 in parallel. We show that it is sufficient to maintain  $O(\log(\rho_{\max}))$  executions at any given time. As a result, throughout the execution covers may be dropped completely, while others emerge. This algorithm also works for both the constrained and unconstrained cases. Specifically, in the constrained case, a subset used by several solutions is taken by the algorithm only once.

Finally, for the capacitated versions of OMM, we obtain an  $O(\sqrt{\rho'_{\max}})$ -competitive algorithm when  $\rho'_{\max}$  is known and an  $O(\log(\rho'_{\max})\sqrt{\rho'_{\max}})$ -competitive algorithm when  $\rho'_{\max}$  is unknown, where  $\rho'_{\max}$  is the maximum ratio over all subset incarnations between the penalties covered by this incarnation and its cost. For soft capacities this is a straightforward corollary from the OMM results. However, in the hard capacities case, we show a lower bound of  $\Omega(\rho_{\max})$  if element reassignments are disallowed. If one may reassign elements, then we show that OMM with hard capacities can be

reduced to OMSC. Submodularity is shown using a reduction to submodularity of *partial maximum weight matching functions* [3].

### 1.2 Related Work

In the most basic covering problem, SET COVER, the input consists of a universe  $U$  of  $n$  elements, a collection  $\mathcal{S} \subseteq 2^U$  of subsets of  $U$ , and a non-negative cost function  $c : \mathcal{S} \rightarrow \mathbb{N}$ . The instance is called *unweighted*, if  $c(S) = 1$ , for every  $S \in \mathcal{S}$ . A sub-collection  $\mathcal{C} \subseteq \mathcal{S}$  is a *set cover* if  $\cup_{S \in \mathcal{C}} S = U$ . The goal is to find a minimum weight set cover, i.e., a sub-collection  $\mathcal{C}$  that minimizes  $c(\mathcal{C}) \triangleq \sum_{S \in \mathcal{C}} c(S)$ . There are several  $\Omega(\log n)$  lower bounds [2, 14, 26, 28] on the approximation ratio of SET COVER. Dinur and Steurer [12] gave an optimal inapproximability result by showing that SET COVER is not approximable within  $(1 - o(1)) \ln n$ , unless  $P=NP$ . On the other hand, it is well known that the approximation ratio of the greedy algorithm is bounded by  $H_n$ , where  $H_n$  is the  $n$ th Harmonic number [9, 22, 25]. SET COVER can be approximated within  $\Delta \triangleq \max_u |\{S : u \in S\}|$  [5, 21]. On the other hand, it is NP-hard to approximate SET COVER within a factor of  $\Delta - 1 - \varepsilon$ , for any  $\varepsilon > 0$ , assuming  $\Delta > 2$  is a constant [10]. The case where  $\Delta = 2$  (i.e., VERTEX COVER) is NP-hard to approximate within a factor of  $10\sqrt{5} - 21 \approx 1.36$  [11].

In the SET MULTI-COVER problem each element  $u_j$  is associated with a covering requirement  $b_j \in \mathbb{N}$ . A feasible solution is required to cover an element  $u_j$  using  $b_j$  sets, i.e., it is a sub-collection  $\mathcal{C} \subseteq \mathcal{S}$  such that  $|\{S : u \in S\} \cap \mathcal{C}| \geq b_j$ , for every  $u_j \in U$ . As in SET COVER, the goal in SET MULTI-COVER is to minimize  $c(\mathcal{C})$ . There are two versions on SET MULTI-COVER. In the *unconstrained* version  $\mathcal{C}$  is a multi-set that may contain any number of copies of a subset  $S$ , and in the *constrained* version each set may be picked at most once. We revert to SET COVER by setting  $b_j = 1$ , for every  $u_j$ , therefore the hardness results of SET COVER apply to both versions. On the positive side, both versions of SET MULTI-COVER can be approximated with  $O(\log n)$  using the natural greedy algorithm [29]. Constrained SET MULTI-COVER admits a  $\Delta$ -approximation algorithm [20]. In the unweighted case there is a deterministic  $(\Delta - b_{\min} + 1)$ -approximation algorithm [31] and a randomized  $(\Delta - b_{\min} + 1)(1 - o(1))$ -approximation algorithm [27], where  $b_{\min} = \min_j b_j$ . Finally, it is known that a  $(\Delta - b_{\min} + 1)$ -approximation algorithm for the weighted case can be obtained using primal-dual or local ratio.

MULTI-SET MULTI-COVER (MM) is an extension of SET MULTI-COVER in which the input also contains a *coverage*  $a_{ij} \in \mathbb{N}$ , for every  $S_i \in \mathcal{S}$  and  $u_j \in S_i$ . Hence,  $S_i$  can be viewed as a multi-set having  $a_{ij}$  copies of  $u_j$ , for each  $u_j$ . As in SET MULTI-COVER, the MM problem has two variants, *unconstrained* and *constrained*. Even the special case of constrained MM with one element is already NP-hard, since it is equivalent to MINIMUM KNAPSACK. Unconstrained MM can be approximated within an  $O(\log n)$  ratio by greedy [29]. Dobson [13] showed that greedy is an  $H(\max_i \sum_j a_{ij})$ -approximation algorithm for constrained MM. Carr et al. [7] gave an LP-based  $p$ -approximation algorithm, where  $p \triangleq \max_j |\{S_i : a_{ij} > 0\}|$ . Kolliopoulos and Young [23] obtained an  $O(\log n)$ -approximation algorithm for constrained MM that uses LP-rounding.

Wolsey [30] studied SUBMODULAR SET COVER, where the goal is to find a minimum cost collection  $\mathcal{C} \subseteq \mathcal{S}$  which covers all elements in  $U$ . Wolsey showed that the approximation ratio of the greedy algorithm is  $1 + O(\log f_{\max})$ , where  $f_{\max} = \max_{S \in \mathcal{S}} f(\{S\})$ . As mentioned in [8], CAPACITATED SET COVER can be seen a special case of SUBMODULAR SET COVER by defining  $f(\mathcal{C})$  as the maximum number of elements that a subcollection  $\mathcal{C}$  can cover.

Guha et al. [18] presented a primal-dual 2-approximation algorithm for VERTEX COVER with soft capacities (a local ratio interpretation was given in [4]) and a 3-approximation algorithm for capacitated VERTEX COVER with edge requirements. Both algorithms can be extended to SET COVER and the resulting approximation ratios are  $\Delta$  and  $\Delta + 1$ , respectively. Gandhi et al. [17] presented a 2-approximation algorithm for VERTEX COVER with soft capacities using an LP-rounding method called *dependent rounding*. Chuzhoy and Naor [8] presented a 3-approximation algorithm for unweighted VERTEX COVER with hard capacities, which is based on randomized rounding with alterations. They also proved that the weighted version of this problem is as hard to approximate as set cover. Gandhi et al. [16] improved the approximation ratio for unweighted VERTEX COVER with hard capacities to 2.

In the standard approach to online covering, it is the elements that arrive overtime as opposed to the subsets. Alon et al. [1] defined ONLINE SET COVER as follows (see also [6]). A collection of sets is known in advance. Elements arrive online, and the algorithm is required to maintain a cover of the elements that arrived: if the arriving element is not already covered, then some set from the given collection must be added to the solution. They presented an  $O(\log n \log m)$ -competitive algorithm, which is the best possible unless  $\text{NP} \subseteq \text{BPP}$  [24]. Observe that our problem has the complementary view of what is known in advance and what arrives online. Recently, Gupta and Levin [19] considered a problem called ONLINE SUBMODULAR COVER, which extends ONLINE SET COVER. In this problem the subsets are known and the submodular functions (elements) arrive in an online manner. They presented a polylogarithmic competitive algorithm whose ratio matches the one from [1] in the special case of ONLINE SET COVER.

### 1.3 Paper Organization

The remainder of this paper is organized as follows. In Sect. 2 we introduce some notation. In Sect. 3 we present the  $O(\sqrt{\rho_{\max}})$ -competitive algorithm for OMSC which relies on prior knowledge of the value of  $\rho_{\max}$ . Section 4 contains our  $O(\log(\rho_{\max})\sqrt{\rho_{\max}})$ -competitive algorithm for OMSC that does not rely on advance knowledge of  $\rho_{\max}$ . Section 5 studies the capacitated versions of OMM. We conclude in Sect. 6.

## 2 Preliminaries

When considering multicollections of subsets one should define set operations, such as union or intersection, on multicollections. If one implicitly assumes that each copy

of a subset  $S_i$  have a unique serial number, then multicollections become collections and set operations are properly defined. Let  $\mathcal{A}$  and  $\mathcal{B}$  be multi-collections. We assume that the operation  $\mathcal{A} \cup \mathcal{B}$  is done on two multicollections whose subset serial numbers may or may not match. Moreover, we assume that the disjoint union operation  $\mathcal{A} \uplus \mathcal{B}$  is done on two multicollections whose subset serial numbers do not match. Hence, the outcome of  $\mathcal{A} \uplus \mathcal{B}$  can be represented by the vector  $x^{\mathcal{A}} + x^{\mathcal{B}}$ , where  $x^{\mathcal{A}}$  and  $x^{\mathcal{B}}$  are the corresponding vectors that indicate the number of copies of each subset in the multicollections.

As mentioned earlier,  $g$  is submodular if  $g(\mathcal{A}) + g(\mathcal{B}) \geq g(\mathcal{A} \cup \mathcal{B}) + g(\mathcal{A} \cap \mathcal{B})$ , for every two multicollections  $\mathcal{A}$  and  $\mathcal{B}$ . An equivalent definition is that  $g$  is submodular if  $g(\mathcal{A} \cup \{S\}) - g(\mathcal{A}) \geq g(\mathcal{B} \cup \{S\}) - g(\mathcal{B})$ , for every  $\mathcal{A} \subseteq \mathcal{B}$  and  $S \in \mathcal{S}$ .

Given a multicollection  $\mathcal{C}$ , let  $F(\mathcal{C})$  be the weighted coverage (or, penalty savings) of  $\mathcal{C}$ , namely  $F(\mathcal{C}) = \sum_j p_j f_j(\mathcal{C})$ . Observe that  $F$  is submodular, since the class of submodular functions is closed under non-negative linear combinations.  $F$  is monotone as well.

For a subset  $S_i$  let  $\kappa(i)$  be the weighted coverage of a single copy of the subset  $S_i$  by itself, i.e.,

$$\kappa(i) \triangleq F(\{S_i\}) = \sum_{j \in S_i} p_j f_j(\{S_i\}) .$$

$\kappa(i)$  is the potential savings in penalties of a single copy of subset  $S_i$ . Observe that since the  $f_j$ s are submodular,  $\kappa(i)$  is the maximum amount of penalties that can be saved by a single copy of  $S_i$ . More formally, for any multicollection  $\mathcal{C}$ , we have that

$$\kappa(i) = \sum_j p_j f_j(\{S_i\}) \geq \sum_j p_j \left( f_j(\mathcal{C} \uplus \{S_i\}) - \sum_j f_j(\mathcal{C}) \right) = F(\mathcal{C} \uplus \{S_i\}) - F(\mathcal{C}) .$$

In addition, define  $\rho(i) \triangleq \kappa(i)/c_i$ , for every  $i$ , and  $\rho_{\max} \triangleq \max_i \rho(i)$ . Hence,  $\rho(i)$  is the ratio between the maximum potential savings and the cost of the subset  $S_i$ , namely it is the *cost effectiveness* of subset  $S_i$ . Observe that if  $\kappa(i) \leq c_i$  for some  $i$ , then the subset is redundant. It follows that we may assume, without loss of generality, that  $\kappa(i) > c_i$  for every  $i$ , which means that  $\rho(i) > 1$ , for every  $i$ , and that  $\rho_{\max} > 1$ .

Intuitively, the cheapest possible way to cover the elements is by subsets with maximum cost effectiveness. Ignoring the sets and simply paying the penalties (i.e., the solution  $x = 0$  and  $z = b$ ) gives a  $\rho_{\max}$ -approximate solution.

In the capacitated version of OMM each subset  $S_i$  has a capacity  $\text{cap}(S_i)$ . This means that a single copy of  $S_i$  may cover a total of at most  $\text{cap}(S_i)$  units of the covering requirements of elements contained in  $S_i$ . Given  $S_i \in \mathcal{C}$ , we refer to  $a_i$  as the *potential coverage vector* of  $S_i$ . A vector  $a'_i \in \mathbb{R}_+^n$  is called an *incarnation* of  $S_i$  if (i)  $a'_{ij} \leq a_{ij}$ , for every  $j$ , (ii)  $\sum_j a'_{ij} \leq \text{cap}(S_i)$ , and (iii)  $\sum_i a'_{ij} \leq b_j$ . We use  $a'_i \leq a_i$  to denote that  $a'_i$  is an incarnation of  $S_i$ . In the capacitated case it is not enough to provide a multicollection  $\mathcal{C}$  as a solution, but rather one needs to assign an incarnation to every subset in  $\mathcal{C}$ . Hence, a solution can be described as a pair  $(\mathcal{C}, y)$ , where  $\mathcal{C}$  is a multicollection and  $y(S_i)$  is an incarnation of  $S_i$ , i.e.,  $y(S_i) \leq a_i$ . In the *soft capacities* case, one may use multiple incarnations of a subset  $S_i$ , provided that



one pays  $c_i$  for each one. In the *hard capacities* case, one may use only one incarnation of  $S_i$ . In the capacitated case, the maximum weighted coverage of a subset  $S_i$  depend on the incarnation of  $S_i$ . Hence, we define  $\kappa'(i) = \max_{a'_i \leq a_i} \sum_j p_j a'_{ij}$ . Also, define  $\rho'(i) \triangleq \kappa'(i)/c_i$ , for every  $i$ , and  $\rho'_{\max} \triangleq \max_i \rho'(i)$ .

### 3 Competitive Submodular Coverage

In this section we provide a deterministic online algorithm for OMSC that relies on the advance knowledge of  $\rho_{\max}$ . The section mainly focuses on unconstrained OMSC, but a similar approach would work for constrained OMSC.

Our algorithm is called THRESHOLD (Algorithm 1) and it generates a monotonically growing multicollection  $\mathcal{C}$  of subsets based on a simple deterministic threshold rule. (Initially,  $\mathcal{C} = \emptyset$ .) It is assumed that THRESHOLD has knowledge of an upper bound  $\sigma$  on  $\rho_{\max}$  (the maximum cost effectiveness over all subsets) and works as follows: Upon arrival of a new subset  $S_i$ , we add  $x_i$  copies of  $S_i$  to  $\mathcal{C}$ , where  $x_i$  is the largest integer satisfying

$$x_i \cdot c_i \leq (F(\mathcal{C} \uplus S_i^{x_i}) - F(\mathcal{C})) / \sqrt{\sigma}, \tag{1}$$

where  $S_i^q$  denotes the multicollection with  $q$  copies of  $S$ . Intuitively, we take the maximum possible number of copies of subset  $S_i$  that allows us to save a factor of at least  $\sqrt{\sigma}$  over the penalties avoided. Notice that  $x_i$  is well-defined because Inequality (1) is always satisfied by  $x_i = 0$ . Also, let  $z_j$  be the requirement of  $j$  that is uncovered by the current solution  $\mathcal{C}$ , namely let  $z_j = b_j - f_j(\mathcal{C})$ .

---

#### Algorithm 1: – THRESHOLD

---

- 1 Upon arrival of  $S_i$ , **begin**
  - 2     Let  $x_i$  be the largest integer satisfying (1).
  - 3      $\mathcal{C} \leftarrow \mathcal{C} \uplus S_i^{x_i}$
- 

We show that the competitive ratio of THRESHOLD is at most  $2\sqrt{\rho_{\max}}$ , if  $\sigma = \rho_{\max}$ . We assume that THRESHOLD is executed with a parameter  $\sigma$  such that  $\sigma \geq \rho_{\max}$ , since we shall need this extension in the next section.

Let  $\mathcal{C}_i$  be the solution that is constructed after the arrival of subset  $S_i$ , and let  $z_j^i$  be the corresponding value of  $z_j$ , for every  $i$ .

We first bound the cost  $\sum_i c_i x_i$  of the selected sets.

**Lemma 1** *Let  $\mathcal{C}$  be the solution computed by THRESHOLD with parameter  $\sigma$ , where  $\sigma \geq \rho_{\max}$ . Also, let  $\mathcal{C}^*$  be an optimal solution. Then,*

$$\sum_i c_i x_i \leq \frac{\rho_{\max}}{\sqrt{\sigma}} \sum_i c_i x_i^* + \frac{1}{\sqrt{\sigma}} \sum_j p_j (z_j^* - z_j).$$



**Proof** By condition (1), we have that

$$\begin{aligned} \sqrt{\sigma} \sum_i c_i x_i &\leq \sum_i (F(\mathcal{C}_{i-1} \uplus S_i^{x_i}) - F(\mathcal{C}_{i-1})) \\ &= \sum_i (F(\mathcal{C}_i) - F(\mathcal{C}_{i-1})) = F(\mathcal{C}) = \sum_j p_j (b_j - z_j). \end{aligned}$$

On the other hand, since  $\rho(i) = \kappa(i)/c_i$ , for every  $i$ , we have that

$$\begin{aligned} \sum_i c_i x_i^* &= \sum_i \frac{F(\{S_i\})}{\rho(i)} x_i^* \\ &\geq \frac{1}{\rho_{\max}} \sum_i x_i^* F(\{S_i\}) \\ &\geq \frac{1}{\rho_{\max}} \sum_i (F(\mathcal{C}_i^*) - F(\mathcal{C}_{i-1}^*)) \\ &= \frac{1}{\rho_{\max}} \sum_j p_j (b_j - z_j^*). \end{aligned}$$

It follows that

$$\begin{aligned} \sqrt{\sigma} \sum_i c_i x_i &\leq \sum_j p_j (b_j - z_j) = \sum_j p_j (b_j - z_j^*) + \sum_j p_j (z_j^* - z_j) \\ &\leq \rho_{\max} \sum_i c_i x_i^* + \sum_j p_j (z_j^* - z_j), \end{aligned}$$

as required. □

**Theorem 1** *Let  $\mathcal{C}$  be the solution computed by Algorithm THRESHOLD, and let  $\mathcal{C}^*$  be an optimal solution. Then,*

$$\text{COST}(\mathcal{C}) = \sum_i c_i x_i + \sum_j p_j z_j \leq 2 \left( \sqrt{\rho_{\max}} \sum_i c_i x_i^* + \sum_j p_j z_j^* \right).$$

**Proof** Due to Lemma 1, with  $\sigma = \rho_{\max}$ , we have that

$$\sum_i c_i x_i \leq \sqrt{\rho_{\max}} \sum_i c_i x_i^* + \frac{1}{\sqrt{\rho_{\max}}} \sum_j p_j (z_j^* - z_j).$$

The next step is to bound the total sum of penalties that  $\mathcal{C}$  pays and  $\mathcal{C}^*$  does not pay, namely to bound

$$\sum_j p_j \max \{ z_j - z_j^*, 0 \}.$$

Define

$$\Delta_i \triangleq \max \{x_i^* - x_i, 0\} .$$

If  $\Delta_i = 0$ , for all  $i$ , then  $\mathcal{C}^* \subseteq \mathcal{C}$  and  $z_j \leq z_j^*$ , for every  $j$ , and we are done. Otherwise, let  $i$  be an index such that  $\Delta_i > 0$ . Due to condition (1) in the  $i$ th step, we have that

$$c_i x_i \leq (F(\mathcal{C}_i) - F(\mathcal{C}_{i-1})) / \sqrt{\rho_{\max}} ,$$

where  $\mathcal{C}_i = \mathcal{C}_{i-1} \uplus S_i^{x_i}$ , while

$$c_i x_i^* > \left( F \left( \mathcal{C}_{i-1} \uplus S_i^{x_i^*} \right) - F(\mathcal{C}_{i-1}) \right) / \sqrt{\rho_{\max}} .$$

Hence,

$$\begin{aligned} c_i \Delta_i &= c_i x_i^* - c_i x_i > \left( F \left( \mathcal{C}_i \uplus S_i^{\Delta_i} \right) - F(\mathcal{C}_i) \right) / \sqrt{\rho_{\max}} \\ &\geq \left( F \left( \mathcal{C} \uplus S_i^{\Delta_i} \right) - F(\mathcal{C}) \right) / \sqrt{\rho_{\max}} , \end{aligned}$$

where the second inequality is due to the submodularity of  $F$ . It follows that

$$\begin{aligned} \sqrt{\rho_{\max}} \sum_i c_i \Delta_i &> \sum_i \left( F \left( \mathcal{C} \uplus S_i^{\Delta_i} \right) - F(\mathcal{C}) \right) \\ &\geq F \left( \mathcal{C} \uplus \bigoplus_i S_i^{\Delta_i} \right) - F(\mathcal{C}) \\ &\geq F(\mathcal{C}^*) - F(\mathcal{C}) \\ &\geq \sum_j p_j \max \{z_j - z_j^*, 0\} , \end{aligned}$$

where the second inequality is due to the submodularity of  $F$ , and the third is due to the monotonicity of  $F$ . Since  $x^* \geq \Delta$ , we have that

$$\sum_j p_j \max \{z_j - z_j^*, 0\} < \sqrt{\rho_{\max}} \sum_i c_i \Delta_i \leq \sqrt{\rho_{\max}} \sum_i c_i x_i^* .$$

Putting it all together, we have that

$$\begin{aligned} &\sum_i c_i x_i + \sum_j p_j z_j \\ &\leq \sum_i c_i x_i + \sum_j p_j z_j^* + \sum_j p_j \max \{z_j - z_j^*, 0\} \\ &\leq \sqrt{\rho_{\max}} \sum_i c_i x_i^* + \frac{1}{\sqrt{\rho_{\max}}} \sum_j p_j (z_j^* - z_j) + \sum_j p_j z_j^* + \sum_j p_j \max \{z_j - z_j^*, 0\} \end{aligned}$$

$$\begin{aligned}
 &= \sqrt{\rho_{\max}} \sum_i c_i x_i^* + \frac{1}{\sqrt{\rho_{\max}}} \sum_j p_j \max \{z_j^* - z_j, 0\} - \frac{1}{\sqrt{\rho_{\max}}} \sum_j p_j \max \{z_j - z_j^*, 0\} \\
 &\quad + \sum_j p_j z_j^* + \sum_j p_j \max \{z_j - z_j^*, 0\} \\
 &\leq \sqrt{\rho_{\max}} \sum_i c_i x_i^* + \left(1 + \frac{1}{\sqrt{\rho_{\max}}}\right) \sum_j p_j z_j^* - \left(1 - \frac{1}{\sqrt{\rho_{\max}}}\right) \sum_j p_j \max \{z_j - z_j^*, 0\} \\
 &\leq (2\sqrt{\rho_{\max}} - 1) \sum_i c_i x_i^* + \left(1 + \frac{1}{\sqrt{\rho_{\max}}}\right) \sum_j p_j z_j^*,
 \end{aligned}$$

as required. □

Theorem 1 leads us to an upper bound on the competitive ratio of Algorithm THRESHOLD.

**Corollary 2** *Algorithm THRESHOLD is  $2\sqrt{\rho_{\max}}$ -competitive for unconstrained OMSC, assuming prior knowledge of  $\rho_{\max}$ .*

The same approach would work for the constrained variant of OMSC. In this case the value of  $x_i$  in the Condition described by (1) is also bounded by 1.

**Corollary 3** *Algorithm THRESHOLD is  $2\sqrt{\rho_{\max}}$ -competitive for constrained OMSC, assuming prior knowledge of  $\rho_{\max}$ .*

### 4 Algorithm Without Prior Knowledge of $\rho_{\max}$

In this section we provide a deterministic online algorithm for OMSC that does not rely on advance knowledge of  $\rho_{\max}$ . This algorithm is based on Algorithm THRESHOLD, but as opposed to algorithm THRESHOLD, it uses dismissals. Recall that it was shown that dismissals are necessary in order to obtain an  $o(\rho_{\max})$ -competitive ratio [15] even for OMM. The competitive ratio of the algorithm is  $O(\log(\rho_{\max})\sqrt{\rho_{\max}})$ .

Algorithm MULTI-THRESHOLD maintains a variable  $\bar{\rho}$  that holds, at any given moment,  $\rho_{\max}$  of the instance that it has seen so far. In addition, it computes in parallel, using THRESHOLD, a solution  $\mathcal{C}^k$ , with the parameter  $\sigma = 2^k$ , for all  $k$  such that  $2^k \in [\bar{\rho}, \bar{\rho}^2]$  (initially,  $\bar{\rho} = 1$ ). Each invocation of THRESHOLD is run independently. That is, each copy of THRESHOLD makes its decisions without taking into account the decisions of other copies. Let  $(x^k, z^k)$  be the pair of variable vectors that correspond to  $\mathcal{C}^k$ .

Upon arrival of subset  $S_i$ , the Algorithm MULTI-THRESHOLD updates  $\bar{\rho}$ , if need be. If  $\bar{\rho}$  is updated, then the algorithm terminates the invocations of THRESHOLD for all parameters that are smaller than  $\bar{\rho}$ . That includes dismissing all subsets that were taken by those invocations. The algorithm then executes a step of THRESHOLD for each power of 2 in the interval  $[\bar{\rho}, \bar{\rho}^2]$ .

Consider such an execution of THRESHOLD with parameter  $\sigma = 2^4 = 16$ . As long as  $\bar{\rho} < \sqrt{16} = 4$ , then this execution does nothing. Hence, it is safe to activate it only upon arrival of a subset  $S_i$  such that  $\rho(i) \geq 4$ .

---

**Algorithm 2:** MULTI- THRESHOLD

---

```

1 Upon arrival of  $S_i$ , begin
2   if  $\rho(i) > \bar{\rho}$  then
3     Abort THRESHOLD for every  $\sigma = 2^k \in [\bar{\rho}, \rho(i))$ 
4     Initialize THRESHOLD for every  $\sigma = 2^k \in [\bar{\rho}^2, \rho(i)^2)$ 
5     Update  $\bar{\rho}$ :  $\bar{\rho} \leftarrow \rho(i)$ 
6   Execute THRESHOLD on  $S_i$  for every  $\sigma = 2^k \in [\bar{\rho}, \bar{\rho}^2)$ 

```

---

The next lemma generalizes this argument in order to justify the fact that MULTI-THRESHOLD executes THRESHOLD only on parameters in the interval  $[\bar{\rho}, \bar{\rho}^2]$ . In other words, we show that one may assume that MULTI- THRESHOLD actually executes THRESHOLD with parameter  $2^k$ , for all  $k \geq \ell$ , where  $\ell = \min \{k : 2^k \geq \bar{\rho}\}$ , on the whole instance.

**Lemma 2** *Let  $k$  be such that  $2^k > \bar{\rho}^2$ , and assume that we execute THRESHOLD with parameter  $\sigma = 2^k$  starting at the arrival of the first subset. Then,  $C_k = \emptyset$ .*

**Proof** When THRESHOLD is run with parameter  $\sigma = 2^k$ , the algorithm adds a copy of a subset  $S_i$  to the solution only if (1) is satisfied with (at least)  $x_i^k = 1$ . If  $\sigma = 2^k > \bar{\rho}^2$ , then no set will be included. □

The multicollection which is computed by MULTI- THRESHOLD is:

$$C = \bigcup_{k:2^k \in [\bar{\rho}, \bar{\rho}^2]} C^k \quad \text{or} \quad x = \sum_{k:2^k \in [\bar{\rho}, \bar{\rho}^2]} x^k .$$

Let  $z_j = b_j - f_j(C)$ , for every  $j$ . (We note that we need not hold a separate set of copies of each subset for each THRESHOLD execution – it suffices to hold the maximum number of copies required.)

We observe that the penalties that are paid by the combined solution is smaller than the penalty that any invocation would have to pay.

**Observation 1**  $z \leq z^k$ , for every  $k$  such that  $2^k \in [\bar{\rho}, \bar{\rho}^2]$ .

Next, we bound the competitive ratio of MULTI- THRESHOLD. We first show that the cost of subsets (without penalties) taken by invocation with parameter  $\sigma = 2^k$ , for  $k > \ell$ , is bounded by the optimal cost.

**Lemma 3** *Let  $k > \ell$  such that  $2^k \leq \bar{\rho}^2$ . Then,  $\sum_i c_i x_i^k \leq \sqrt{\bar{\rho}} \cdot \text{COST}(C^*)$ .*

**Proof** Due to Lemma 2 one may assume that the invocation of THRESHOLD with parameter  $2^k$  is run on the whole instance. Define a new instance similar to the original instance but with coverage requirements  $b_j^k = f_j(C^k)$ , for every  $j$ . The solution  $C^k$  covers all of the requirements in the new instance by definition. Moreover, notice that THRESHOLD would compute  $C^k$  if executed on the new instance with parameter  $2^k$ . Let  $C'$  be an optimal solution for the new instance.

Lemma 1 applied to THRESHOLD with parameter  $2^k$  and the new instance implies that

$$\sum_i c_i x_i^k \leq \frac{\bar{\rho}}{\sqrt{2^k}} \sum_i c_i x'_i + \frac{1}{\sqrt{2^k}} \sum_j p_j z'_j .$$

Since  $2^k > \bar{\rho}$ , we have that

$$\sum_i c_i x_i^k \leq \sqrt{\bar{\rho}} \sum_i c_i x'_i + \frac{1}{\sqrt{\bar{\rho}}} \sum_j p_j z'_j \leq \sqrt{\bar{\rho}} \cdot \text{COST}(\mathcal{C}') .$$

The lemma follows, since  $\text{COST}(\mathcal{C}') \leq \text{COST}(\mathcal{C}^*)$ . □

We bound the total cost of the solution using Lemma 3 and the fact that the solution computed by invocation with parameter  $\sigma = 2^\ell$  is  $O(\sqrt{\rho_{\max}})$ -competitive.

**Theorem 4** *There exists an  $O(\log(\rho_{\max})\sqrt{\rho_{\max}})$ -competitive algorithm for unconstrained OMSC.*

**Proof** Consider the solution  $\mathcal{C}$  at termination of MULTI- THRESHOLD. Since  $\bar{\rho} = \rho_{\max}$  at termination, we have that  $x_i = \sum_{k:2^k \in [\rho_{\max}, \rho_{\max}^2]} x_i^k$ , for every  $i$ . By Lemma 2 we may assume that the invocation of THRESHOLD with parameter  $2^\ell$  run on the whole instance. Also, by Observation 1 it follows that  $z_j \leq z_j^\ell$ , for every  $j$ , and in particular for  $k = \ell$ . Hence,

$$\begin{aligned} \text{COST}(\mathcal{C}) &= \sum_i c_i x_i + \sum_j p_j z_j \\ &\leq \sum_i c_i \sum_{k:2^k \in [\rho_{\max}, \rho_{\max}^2]} x_i^k + \sum_j p_j z_j \\ &\leq \sum_i c_i \sum_{k \neq \ell: 2^k \in [\rho_{\max}, \rho_{\max}^2]} x_i^k + \sum_i c_i x_i^\ell + \sum_j p_j z_j^\ell \\ &\leq \log(\rho_{\max})\sqrt{\rho_{\max}} \cdot \text{COST}(\mathcal{C}^*) + 2\sqrt{2\rho_{\max}} \cdot \text{COST}(\mathcal{C}^*) , \end{aligned}$$

where the second inequality follows from Lemma 3 and Corollary 2. □

By replacing Corollary 2 with Corollary 3, we have that:

**Theorem 5** *There exists an  $O(\log(\rho_{\max})\sqrt{\rho_{\max}})$ -competitive algorithm for constrained OMSC.*

### 5 Capacitated Online Multiset Multicover

In this section we consider the capacitated versions of ONLINE MULTISSET MULTICOVER. We deal with soft capacities using a straightforward reduction to uncapacitated

OMM. As for hard capacities, we show a lower bound of  $\Omega(\rho_{\max})$ , if subset reassignments are not allowed. If reassignments are allowed, we show that the capacitated coverage is submodular, and this paves the way to a reduction to OMSC.

## 5.1 Soft Capacities

We first show that our results extend to soft capacities.

**Theorem 6** *There exists an  $O(\sqrt{\rho'_{\max}})$ -competitive algorithm for OMM with soft capacities, assuming prior knowledge of  $\rho'_{\max}$ .*

**Proof** Consider a subset  $S_i$ . One can simulate the arrival of  $S_i$  with the arrival of all its possible incarnations, namely with the arrival of  $\mathcal{S}_i = \{S'_i \subseteq S_i : |S'_i| = \text{cap}(S_i)\}$ . The theorem follows, since the maximum cost effectiveness is  $\rho'_{\max}$ .  $\square$

Theorems 4 and 6 lead to the following result:

**Theorem 7** *There exists an  $O(\log(\rho'_{\max})\sqrt{\rho'_{\max}})$ -competitive algorithm for OMM with soft capacities.*

## 5.2 Hard Capacities Without Reassignments

The following construction shows a lower bound of  $\Omega(\rho'_{\max})$ , for the case where elements cannot be reassigned to other subsets.

**Theorem 8** *The competitive ratio of any randomized online algorithm for OMM with hard capacities is  $\Omega(\rho_{\max})$  if reassignments are disallowed. This bound holds even if the algorithm is allowed to discard sets, and for inputs with unit penalties and costs, and subsets of size at most two.*

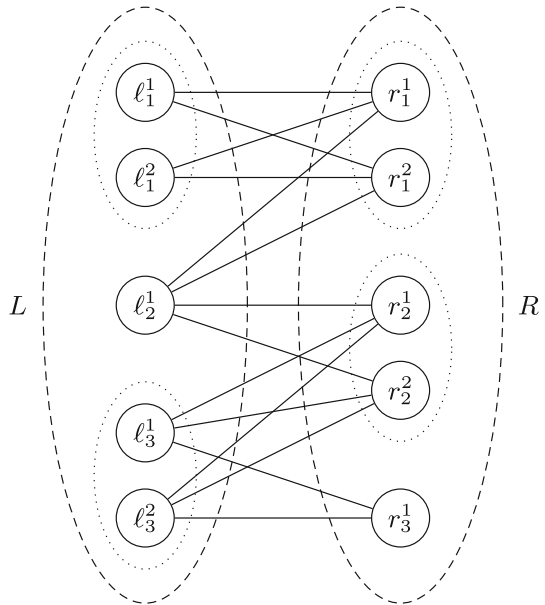
**Proof** Let ALG be a randomized algorithm. Consider an instance with two unit penalty elements  $u_1, u_2$ . The coverage requirements of both elements is  $\beta$ . The sequence consists of two unit cost subsets with capacity  $\beta$ . It starts with  $S_1 = \{u_1, u_2\}$ ,  $a_{11} = a_{12} = \beta$ . The second subset  $S_2$  has two options, each with probability 0.5: (i)  $S_2 = \{u_1\}$ ,  $a_{11} = \beta$ , and  $a_{12} = 0$ , and (ii)  $S'_2 = \{u_2\}$ ,  $a_{12} = \beta$ , and  $a_{11} = 0$ . Clearly, the optimal cost is 2. Also,  $\rho_{\max} = \rho'_{\max} = \beta$ .

Let  $A$  be the event that ALG uses at least  $\beta/2$  units of the capacity of  $S_1$  to cover  $u_1$ . If  $A$  occurs and the adversary presents the subset  $S_2$ , then at least  $\beta/2$  units of the cover requirements of  $u_2$  are unsatisfied, for a penalty of at least  $\beta/2$ . Similarly, if  $A$  does not occur and the adversary presents the subset  $S'_2$ , the same  $\beta/2$  penalty applies to  $u_1$ . In either case, with probability  $\frac{1}{2}$ , the cost is at least  $\beta/2$ . Hence, the expected cost of ALG is at least  $\beta/4$ , for a competitive ratio of at least  $\beta/8$ .  $\square$

## 5.3 Hard Capacities with Reassignments

Now consider the version of OMM, where reassignments are allowed. In this case, it is only natural to have the best possible assignment with respect to the current

**Fig. 1** A bipartite graph  $G = (L, R, E)$  for the instance  $U = \{1, 2, 3\}$ , where  $b = (2, 2, 1)$  and  $S = \{\{1\}, \{1, 2\}, \{2, 3\}\}$ , where  $\text{cap}(S_1) = \text{cap}(S_3) = 2$  and  $\text{cap}(S_2) = 1$



multicollection  $\mathcal{C}$  at any time. Formally, given  $\mathcal{C} \subseteq \mathcal{S}$ , let  $\tilde{y}$  be an incarnation function that maximizes the weighted coverage of  $\mathcal{C}$ , i.e.,  $\tilde{y}_{\mathcal{C}} = \text{argmax}_y F(y(\mathcal{C}))$ . Also, denote  $\tilde{F}(\mathcal{C}) = F(\tilde{y}_{\mathcal{C}})$ .

The next natural step is to modify Algorithm THRESHOLD by using  $\tilde{F}$  instead of  $F$ . Hence, we would like to prove that  $\tilde{F}$  is submodular. Before dealing with submodularity we reduce the problem of finding the best cover of a universe  $U$  using a capacitated collection  $\mathcal{C}$  of subsets to an instance of the MAXIMUM WEIGHTED MATCHING problem in a bipartite graph.

Given a multicollection  $\mathcal{S}$ , we define the following bipartite graph  $G = (L, R, E)$ , where

$$L = \{\ell_i^s : S_i \in \mathcal{C}, s \in \{1, \dots, \text{cap}(S_i)\}\}$$

$$R = \{r_j^t : u_j \in U, t \in \{1, \dots, b_j\}\}$$

$$E = \{(\ell_i^s, r_j^t) : u_j \in S_i, s \in \{1, \dots, \text{cap}(S_i)\}, t \in \{1, \dots, b_j\}\}.$$

The weight of an edge  $(\ell_i^s, r_j^t)$  is defined as  $p_j$ , i.e.,  $w(\ell_i^s, r_j^t) = p_j$ . A depiction is given in Fig. 1.

Given a multicollection  $\mathcal{C}$ , let  $L_{\mathcal{C}}$  be the vertices in  $L$  that correspond to subsets in  $\mathcal{C}$ , i.e.,  $L_{\mathcal{C}} = \{\ell_i^s \in L : S_i \in \mathcal{C}\}$ . The graph  $G[L_{\mathcal{C}} \times R]$  is an induced subgraph of  $G$  that contains that all element vertices on the right side, but only vertices that correspond to subsets in  $\mathcal{C}$  on the left side.

**Lemma 4** Consider an instance of capacitated MULTISSET MULTICOVER and a multicollection  $\mathcal{C} \subseteq \mathcal{S}$ . Then, an incarnation function  $y$  induces a fractional matching  $M$



in  $G[L_C \times R]$ , such that the weight of the matching is equal to the total penalty that is covered by  $y$ , and vice versa.

**Proof** Given an incarnation  $y$ , let  $y'$  be the following fractional matching:

$$y'(\ell_i^s, r_j^t) = \frac{y(S_i, u_j)}{\text{cap}(S_i) \cdot b_j} .$$

$y'$  is a fractional matching, since

$$\sum_{\ell_i^s: u_j \in S_i} y'(\ell_i^s, r_j^t) = \sum_{\ell_i^s: u_j \in S_i} \frac{y(S_i, u_j)}{\text{cap}(S_i) \cdot b_j} = \sum_{u_j \in S_i} b_j \cdot \frac{y(S_i, u_j)}{\text{cap}(S_i) \cdot b_j} = \sum_{u_j \in S_i} \frac{y(S_i, u_j)}{\text{cap}(S_i)} \leq 1 ,$$

for every  $r_j^t$ , and similarly

$$\begin{aligned} \sum_{r_j^t: u_j \in S_i} y'(\ell_i^s, r_j^t) &= \sum_{r_j^t: u_j \in S_i} \frac{y(S_i, u_j)}{\text{cap}(S_i) \cdot b_j} \\ &= \sum_{S_i: u_j \in S_i} \text{cap}(S_i) \cdot \frac{y(S_i, u_j)}{\text{cap}(S_i) \cdot b_j} = \sum_{S_i: u_j \in S_i} \frac{y(S_i, u_j)}{b_j} \leq 1 , \end{aligned}$$

for every  $\ell_i^s$ .

For the other direction, let  $y'$  be a fractional matching. Define the following incarnation function:

$$y(S_i, u_j) = \sum_s \sum_t y'(\ell_i^s, r_j^t) .$$

We have that

$$\sum_j y(S_i, u_j) = \sum_j \sum_s \sum_t y'(\ell_i^s, r_j^t) = \sum_s \sum_{\ell_i^s: u_j \in S_i} y'(\ell_i^s, r_j^t) \leq \sum_s 1 = \text{cap}(S_i) ,$$

and

$$\sum_i y(S_i, u_j) = \sum_i \sum_s \sum_t y'(\ell_i^s, r_j^t) = \sum_t \sum_{r_j^t: u_j \in S_i} y'(\ell_i^s, r_j^t) \leq \sum_t 1 = b_j ,$$

Finally, it is not hard to verify that the total sum of covered penalties is equal to the weight of the matching. □

Since the polytope of bipartite MAXIMUM WEIGHTED MATCHING is known to have integral optimal solutions, Lemma 4 implies that:

**Corollary 9** Consider an instance of capacitated MULTISSET MULTICOVER and a multicollection  $C \subseteq S$ . Then, there exists an optimal integral incarnation function  $y$ .

Now we move to proving that  $\tilde{F}$  is submodular. To do that we use a result by Bar-Noy and Rabanca [3] that showed submodularity of *partial maximum weight matching functions*.

**Definition 1** [3] Let  $G = (L, R, E)$  be a bipartite graph and let  $w : E \rightarrow \mathbb{R}_+$  be a non-negative weight function on the edges. The partial maximum weight matching function  $f : 2^A \rightarrow \mathbb{R}$  maps any subset  $U \subseteq L$  to the value of a maximum weight matching in  $G[U \cup R]$ .

**Lemma 5** [3] *Let  $f$  be a partial maximum weight matching function for a bipartite graph  $G$  with non-negative edge weights. Then,  $f$  is submodular.*

By Lemmas 4, 5, and Corollary 9, we have that:

**Lemma 6** *Given an instance of capacitated MULTISSET MULTICOVER,  $\tilde{F}$  is submodular.*

Since  $\tilde{F}$  is submodular one may use the algorithms from the previous sections.

**Theorem 10** *There exists an  $O(\sqrt{\rho'_{\max}})$ -competitive algorithm for OMM with hard capacities, assuming prior knowledge of  $\rho'_{\max}$ .*

**Proof** We use Algorithm THRESHOLD with  $\tilde{F}$  instead of  $F$ . We assume that Algorithm THRESHOLD uses the bipartite graph  $G$  in order to compute its solution. If this is done using an augmenting path algorithm, then  $z_j$  can only decrease, for every  $u_j$ , throughout execution.

By Lemma 6,  $\tilde{F}$  is submodular and  $\tilde{F}$  is also monotone. Therefore the analysis of the Algorithm THRESHOLD applies with  $\rho'_{\max}$  replacing  $\rho_{\max}$ . □

Theorems 4 and 10 lead to the following result:

**Theorem 11** *There exists an  $O(\log(\rho'_{\max})\sqrt{\rho'_{\max}})$ -competitive algorithm for OMM with hard capacities.*

## 6 Conclusion

It was shown in [15] that randomized algorithms cannot have competitive ratio better than  $\Omega(\sqrt{\rho_{\max}})$ , even for OMM. This lower bound holds even if  $\rho_{\max}$  is known, and even if one is allowed to drop previously selected subsets. We gave a matching upper bound for OMSC, but the corresponding algorithm is based on the prior knowledge of  $\rho_{\max}$ . The competitive ratio of our second algorithm is  $O(\log(\rho_{\max})\sqrt{\rho_{\max}})$ , and this leaves a multiplicative gap of  $O(\log(\rho_{\max}))$ . It remains an open question whether there is an  $O(\sqrt{\rho_{\max}})$ -competitive algorithm that has no knowledge of  $\rho_{\max}$ , even for OMM.

**Acknowledgements** Magnús M. Halldórsson was supported by the Icelandic Research Fund Grant 2310015. This research was done while Dror Rawitz was on sabbatical at Reykjavik University.

**Author Contributions** Equal contribution by the authors

**Funding** Open access funding provided by Bar-Ilan University.

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

1. Alon, N., Awerbuch, B., Azar, Y., Buchbinder, N., Naor, J.: The online set cover problem. *SIAM J. Comput.* **39**(2), 361–370 (2009)
2. Alon, N., Moshkovitz, D., Safra, S.: Algorithmic construction of sets for  $k$ -restrictions. *ACM Trans. Algorithms* **2**(2), 153–177 (2006)
3. Bar-Noy, A., Rabanca, G.: Tight approximation bounds for the seminar assignment problem. In: 14th International Workshop on Approximation and Online Algorithms. LNCS, vol. 10138, pp. 170–182 (2016)
4. Bar-Yehuda, R., Bendel, K., Freund, A., Rawitz, D.: Local ratio: a unified framework for approximation algorithms. *ACM Comput. Surv.* **36**(4), 422–463 (2004)
5. Bar-Yehuda, R., Even, S.: A linear time approximation algorithm for the weighted vertex cover problem. *J. Algorithms* **2**, 198–203 (1981)
6. Buchbinder, N., Naor, J.: Online primal-dual algorithms for covering and packing. *Math. Oper. Res.* **34**(2), 270–286 (2009)
7. Carr, R.D., Fleischer, L., Leung, V.J., Phillips, C.A.: Strengthening integrality gaps for capacitated network design and covering problems. In: 11th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 106–115 (2000)
8. Chuzhoy, J., Naor, J.: Covering problems with hard capacities. *SIAM J. Comput.* **36**(2), 498–515 (2006)
9. Chvátal, V.: A greedy heuristic for the set-covering problem. *Math. Oper. Res.* **4**(3), 233–235 (1979)
10. Dinur, I., Guruswami, V., Khot, S., Regev, O.: A new multilayered PCP and the hardness of hypergraph vertex cover. *SIAM J. Comput.* **34**(5), 1129–1146 (2005)
11. Dinur, I., Safra, S.: The importance of being biased. In: 34th Annual ACM Symposium on the Theory of Computing, pp. 33–42 (2002)
12. Dinur, I., Steurer, D.: Analytical approach to parallel repetition. In: 46th Annual ACM Symposium on the Theory of Computing, pp. 624–633 (2014)
13. Dobson, G.: Worst-case analysis of greedy heuristics for integer programming with nonnegative data. *Math. Oper. Res.* **7**(4), 515–531 (1982)
14. Feige, U.: A threshold of  $\ln n$  for approximating set cover. *J. ACM* **45**(4), 634–652 (1998)
15. Fraigniaud, P., Halldórsson, M.M., Patt-Shamir, B., Rawitz, D., Rosén, A.: Shrinking maxima, decreasing costs: new online packing and covering problems. *Algorithmica* **74**(4), 1205–1223 (2016)
16. Gandhi, R., Halperin, E., Khuller, S., Kortsarz, G., Srinivasan, A.: An improved approximation algorithm for vertex cover with hard capacities. *J. Comput. Syst. Sci.* **72**(1), 16–33 (2006)
17. Gandhi, R., Khuller, S., Parthasarathy, S., Srinivasan, A.: Dependent rounding in bipartite graphs. In: 43rd IEEE Symposium on Foundations of Computer Science, pp. 323–332 (2002)
18. Guha, S., Hassin, R., Khuller, S., Or, E.: Capacitated vertex covering. *J. Algorithms* **48**(1), 257–270 (2003)
19. Gupta, A., Levin, R.: The online submodular cover problem. In: 31st Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1525–1537 (2020)

20. Hall, N.G., Hochbaum, D.S.: A fast approximation algorithm for the multicovering problem. *Discrete Appl. Math.* **15**(1), 35–40 (1986)
21. Hochbaum, D.S.: Approximation algorithms for the set covering and vertex cover problems. *SIAM J. Comput.* **11**(3), 555–556 (1982)
22. Johnson, D.S.: Approximation algorithms for combinatorial problems. *J. Comput. Syst. Sci.* **9**, 256–278 (1974)
23. Kolliopoulos, S.G., Young, N.E.: Approximation algorithms for covering/packing integer programs. *J. Comput. Syst. Sci.* **71**(4), 495–505 (2005)
24. Korman, S. On the Use of Randomization in the Online Set Cover Problem. Master's Thesis, Weizmann Institute of Science, Rehovot, (2004)
25. Lovász, L.: On the ratio of optimal integral and fractional covers. *Discrete Math.* **13**, 383–390 (1975)
26. Lund, C., Yannakakis, M.: On the hardness of approximating minimization problems. *J. ACM* **41**(5), 960–981 (1994)
27. Peleg, D., Schechtman, G., Wool, A.: Randomized approximation of bounded multicovering problems. *Algorithmica* **18**(1), 44–66 (1997)
28. Raz, R., Safra, S.: A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In: 29th Annual ACM Symposium on the Theory of Computing, pp. 475–484 (1997)
29. Vazirani, V.V.: *Approximation Algorithms*. Springer, Berlin (2001)
30. Wolsey, L.A.: An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica* **2**, 385–393 (1982)
31. Wool, A. Approximating Bounded 0-1 Integer Linear Programs. Master's Thesis. The Weizmann Institute of Science (1992)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.