CAHIER DU LAMSADE

299

Septembre 2010

Online maximum k -coverage

GIORGIO AUSIELLO, NICOLAS BORJA,
ARISTOTELIS GIANNAKOS, GIORGIO LUCARELLI,
VANGELIS TH. PASCHOS

Online maximum k -coverage*

Giorgio Ausiello[†] Nicolas Boria[‡] Aristotelis Giannakos[‡]

Giorgio Lucarelli[‡] Vangelis Th. Paschos[‡]

September 17, 2010

Abstract

We study an online model for the maximum k -coverage problem, where given a universe of elements $E = \{e_1, e_2, \dots, e_m\}$, a collection of subsets of E , $S = \{S_1, S_2, \dots, S_n\}$, and an integer k , we ask for a subcollection $A \subseteq S$, such that $|A| = k$ and the number of elements of E covered by A is maximized. In our model, at each step i , a new set S_i is revealed, and we have to decide whether we will keep it or discard it. At any time of the process, only k sets can be kept in memory; if at some point the current solution already contains k sets, any inclusion of any new set in the solution must entail the irremediable deletion of one set of the current solution (a set not kept when revealed is irremediably deleted). We first propose an algorithm that improves upon former results for the same model. We next settle a graph-version of the problem, called maximum k -vertex coverage problem. Here also we propose non-trivial improvements of the competitive ratio for natural classes of graphs (mainly regular and bipartite).

1 Introduction

In the *maximum k -coverage* (MkC) problem given a universe of elements $E = \{e_1, e_2, \dots, e_m\}$, a collection of subsets of E , $S = \{S_1, S_2, \dots, S_n\}$, and an integer k , we ask for a subcollection $A \subseteq S$, such that $|A| = k$ and the number of elements of E covered by A is maximized. The MkC problem

*Research supported by the French Agency for Research under the DEFIS program TODO, ANR-09-EMER-010

[†]`ausiello@dis.uniroma1.it`, Dipartimento di Informatica e Sistemistica, Università degli Studi di Roma “La Sapienza”

[‡]`{boria,giannako,lucarelli,paschos}@lamsade.dauphine.fr`, LAMSADE, Université Paris-Dauphine and CNRS FRE 3234

is NP-hard, since otherwise the optimal solution for the set cover problem could be found in polynomial time: for each k , $1 \leq k \leq n$, run the algorithm for the MkC problem and stop when all elements are covered.

In this paper we consider the following online model for this problem: at each step i , a new set S_i is revealed, and we have to decide whether we will keep it or discard it. At any time of the process, only k sets can be kept in memory, so if at some point the current solution already contains k sets, any inclusion of any new set in the solution must be compensated with the irremediable deletion of one set of the current solution. Of course, a set that is not kept when it is revealed is also irremediably deleted. Finally, the total number n of sets that will be revealed is unknown, apart from $n \geq k$.

If we consider that the universe of elements corresponds to the edges of a graph and the subsets of E to vertices of the graph, we get the *maximum k -vertex coverage* ($MkVC$) problem. Hence, in the $MkVC$ problem given a simple graph $G = (V, E)$, where $V = \{v_1, v_2, \dots, v_n\}$ and $E = \{e_1, e_2, \dots, e_m\}$ are the sets of vertices and edges, respectively, and an integer k we ask for a subset $A \subseteq V$, such that $|A| = k$ and the number of elements of E covered by A is maximized.

Clearly, the $MkVC$ problem is a special case of the MkC problem where: (i) each element belongs to exactly two sets and (ii) the intersection of any two sets of S has size at most one, since multiple edges are not permitted.

The online model we study for the $MkVC$ problem is the same as for the MkC , apart from the fact that all the vertices of the graph are known a priori. What is unknown is the neighborhoods of the vertices. So, at each step i the edges that are adjacent to the vertex v_i are revealed. Note that since the vertices are known, their number n is also known for the $MkVC$ coverage problem.

The weighted generalization of the MkC problem, denoted by **WEIGHTED MkC** , has been also studied in the literature. In this problem, each element $e_i \in E$ has a non-negative weight $w(e_i)$, and the goal is to maximize the total weight of the elements covered by k sets.

The MkC problem is known to be non approximable within a factor $1 - \frac{1}{e}$ [6]. On the other hand, even for the weighted version of the problem, an approximation algorithm of ratio $1 - (1 - \frac{1}{k})^k$ is known [10]. This ratio tends to $1 - \frac{1}{e}$ as k increases, closing in this way the approximability question for the problem.

In [1] the inverse problem, also called *maximum coverage problem*, has been studied: given a universe of elements $E = \{e_1, e_2, \dots, e_m\}$, a collection of subsets of E , $S = \{S_1, S_2, \dots, S_n\}$, a non-negative weight $w(S_i)$ for each

$S_i \in S$, and an integer k , a set $B \subseteq E$ is sought, such that $|B| = k$ and the total weight of the sets in S that intersect with B is maximized. It is easy to see that this version is equivalent to the WEIGHTED MkC modulo the interchange of the roles between set-system and ground set. An algorithm of approximation ratio $1 - \left(1 - \frac{1}{p}\right)^p$ is presented in [1] for this problem, where p is the cardinality of the largest set in S . In the case where each set has cardinality equal to two then this problem coincides with the $MkVC$ problem; hence a $\frac{3}{4}$ approximation ratio is implied by the algorithm in [1]. Moreover, several improvements for some restricted cases of the $MkVC$ problem are presented in [7, 9].

The same online model for WEIGHTED MkC problem has also been studied in [12], where an algorithm of competitive ratio $\frac{1}{4}$ is given. The authors in their so called *set-streaming model* assume that the set of elements is known a priori. Nevertheless, they do not use this information in the proposed algorithm.

In [4], an algorithm is presented about another online model for the MkC problem: the sets are known in advance while the elements arrive online. The list of sets that contain each element become known when this element arrives.

A similar setting for the online set cover problem has been also studied in [2, 5]. The difference is that not all elements will finally arrive, without knowing a priori which of them will do. Finally, in the model studied in [3] for the set cover problem, when an element arrives some information about the list of sets that contain it appears too (e.g., the maximum cardinality set that contains it or the set from its list that covers the larger number of not yet released elements, ...). We also mention here, another variant of the online set cover problem analyzed in [8].

In this paper we study the online model described above for both the MkC and the $MkVC$ problems. In Section 2 we first prove that we cannot have an algorithm for the online MkC problem of competitive ratio better than $\frac{1}{2}$. Then, we present an algorithm of ratio $\frac{1}{3}$, improving in this way the previously known $\frac{1}{4}$ ratio. In Section 3 we deal with the online $MkVC$ problem and we present algorithms for regular graphs, regular bipartite graphs, trees and chains.

We conclude this section by giving the notation that will be used in the sequel. It is based on the definition of the MkC problem and is easily extendable to the $MkVC$ problem.

For any $A \subseteq S$, we denote by $m(A)$ the number of these elements, counting each of them once. Let $SOL = m(A)$ be the number of elements covered

by our algorithms. Moreover, we denote by $A^* \subseteq S$ an optimal subcollection and by $OPT = m(A^*)$ the number of elements covered by an optimal solution.

The cardinality of the set that contains the maximum number of elements is denoted by Δ , that is $\Delta = \max\{|S_i| : 1 \leq i \leq n\}$. Dealing with MkVC, Δ denotes the maximum degree (or the degree when it is regular) of the input-graph $G(V, E)$, where $|V| = n$ and $|E| = m$.

For a subcollection $A \subseteq S$ and a set $S_i \in A$, we call *public* the elements of S_i that are shared by more than one sets in A and *private* the elements of S_i that are covered just by S_i in A .

Finally, as it is common in the online setting, the quality of an algorithm is measured by means of the so-called *competitive ratio* representing the ratio of the value of the solution computed by the algorithm over the optimal value of the whole instance, i.e., the value of an optimal (offline) solution of the final instance.

2 Maximum k -coverage

In this section we deal with the online maximum k -coverage problem. We start with a negative result for the restrictive case where swaps are not allowed (replacement of a set that belongs to the current solution by the newly revealed set is not permitted).

Proposition 1. *Any deterministic online algorithm that does not allow swaps cannot achieve a competitive ratio better than $O\left(\frac{1}{m^{1/(k+1)}}\right)$ for the MkC problem.*

Proof. We assume that sets S_i , $i = 1, \dots, k$, are disjoint and that $k \ll m$. Consider the following scenario. In step i , $1 \leq i \leq k$, a set S_i of cardinality $|S_i| = m^{i/(k+1)}$ is released. If the algorithm rejects S_i , then $m - \sum_{j=1}^i m^{j/(k+1)}$ sets, each of cardinality one, are released. If the algorithm selects S_i , then set S_{i+1} , with $|S_{i+1}| = m^{i/(k+1)}$ is released. If after the k -th set the algorithm has selected all the k released sets then a new set S_{k+1} of cardinality $|S_{k+1}| = m$ is released.

If after the step i the algorithm has rejected S_i then only sets of cardinality one are released. Hence, the algorithm covers $k - (i - 1) + \sum_{j=1}^{i-1} m^{j/(k+1)}$ elements, while the optimal solution covers $k - i + \sum_{j=1}^i m^{j/(k+1)}$ elements.

Thus:

$$\begin{aligned} \frac{SOL}{OPT} &= \frac{k - (i - 1) + \sum_{j=1}^{i-1} m^{i/(k+1)}}{k - i + \sum_{j=1}^i m^{i/(k+1)}} = \frac{k - (i - 1) + \frac{m^{i/(k+1)} - m^{1/(k+1)}}{m^{1/(k+1)} - 1}}{k - i + \frac{m^{(i+1)/(k+1)} - m^{1/(k+1)}}{m^{1/(k+1)} - 1}} \\ &= O\left(\frac{1}{m^{1/(k+1)}}\right) \end{aligned}$$

If the algorithm has selected all the k first released sets, then it covers exactly $\sum_{j=1}^k m^{i/(k+1)}$ elements, while the optimal solution (that includes S_{k+1} which is never selected by the online algorithm) covers m elements. Hence:

$$\frac{SOL}{OPT} = \frac{\sum_{j=1}^k m^{i/(k+1)}}{m} = \frac{\frac{m^{(k+1)/(k+1)} - m^{1/(k+1)}}{m^{1/(k+1)} - 1}}{m} = O\left(\frac{1}{m^{1/(k+1)}}\right)$$

that concludes the proof. $\square$

The next result is also a negative result but fits the model handled in the paper (i.e., where swaps are allowed). Recall that for the offline version of the MkC problem a $1 - \frac{1}{e} \simeq 0.63$ -inapproximability result is known [6].

Proposition 2. *Any deterministic online algorithm cannot achieve a competitive ratio better than $\frac{k}{2k-1} \simeq \frac{1}{2}$ for the MkC problem.*

Proof. Assume that $2k - 1$ disjoint sets, $S_1, S_2, \dots, S_{2k-1}$, each of cardinality M , are released and the algorithm selects $k' \leq k$ of them. Wlog, let $S_1, S_2, \dots, S_{k'}$ be the sets selected by the algorithm. Next the set S_{2k} of cardinality $k \cdot M$ is released, which covers all the elements in $S_1 \cup S_2 \cup \dots \cup S_{k'}$ plus $(k - k')M$ new elements. If the algorithm selects S_{2k} , then $SOL = k \cdot M$; otherwise $SOL = k' \cdot M \leq k \cdot M$. The optimal solution consists of the sets $S_{k+1}, S_{k+2}, \dots, S_{2k-1}, S_{2k}$, and hence is of cardinality $(k - 1) \cdot M + k \cdot M = (2k - 1)M$. In all:

$$\frac{SOL}{OPT} \leq \frac{k \cdot M}{(2k - 1)M} = \frac{k}{2k - 1}$$

and the proof is completed. $\square$

Consider now ALGORITHM MkC for the online MkC problem. It initializes by selecting the first k released sets. Then, considering that the current solution A_j covers $m(A_j)$ elements, the algorithm replaces a set $Q \in A_j$ by the new released set P , only if the number of elements covered is increased by at least $\frac{m(A_j)}{x}$, where x is a parameter of the algorithm that will be specified later. We prove that ALGORITHM MkC achieves competitive ratio asymptotically equal to $\frac{1}{3}$ thus improving the $\frac{1}{4}$ -competitive result of [12].

ALGORITHM $MkC(x)$

- 1: $A_1 = \{\text{the first } k \text{ released sets}\};$
 - 2: **for** each released set P **do**
 - 3: Find the set $Q \in A_j$ that covers privately the smallest number of elements in A_j ;
 - 4: **if** $m(A_j \cup \{P\} \setminus \{Q\}) > m(A_j) + \frac{m(A_j)}{x}$ **then**
 - 5: $A_{j+1} = A_j \cup \{P\} \setminus \{Q\};$
-

Theorem 1. ALGORITHM $MkC(x)$ achieves a competitive ratio which is asymptotic to $\frac{1}{3}$.

Proof. Let A_z be the final solution obtained by the algorithm, i.e., $SOL = m(A_z)$. Fix, also, an optimal solution A^* .

For A^* , we consider the following two types of events: (a) ALGORITHM MkC does not select a set of A^* , and (b) ALGORITHM MkC deletes a set of A^* . Clearly, at most k such events may happen in total. We assume that the event i , $i \leq k$, happens when the current solution of the algorithm is A_{j_i} , $1 \leq j_i \leq z$. Note that several consecutive events can happen with respect to the same current solution, that is several consecutive j_i 's as well as the induced A_{j_i} 's can be equal.

To make clear our notation as well as the execution of ALGORITHM MkC , consider the example in Figure 1. There, after the release of the k first sets we get A_1 as the current solution. Assume that $Q \in A_1$ is the set of A_1 with the minimum number of private elements and let q be the private elements of Q in A_1 . Then, the set P which adds to the solution more than $\frac{m(A_1)}{x} + q$ more elements is released. The algorithm selects this set and removes Q . If $Q \in A^*$, then we have the first event, that is, $j_1 = 1$ and $A_{j_1} = A_1$. Next, the set P' which contains, say, $\frac{m(A_2)}{x} - 1$ private elements is released. Hence, the algorithm does not select it. If $P' \in A^*$ then we have the second event, that is, $j_2 = 2$ and $A_{j_2} = A_2$.

Figure 1: An example of the execution of ALGORITHM $MkC(x)$.

We will now try to provide an upper bound to the value $m(A^*) = OPT$

as function of the states A_{j_i} .

Consider the first event that happens in j_1 . If this event is of type (a) then the set of A^* which is not selected is of size smaller than $m(A_{j_1}) + \frac{m(A_{j_1})}{x}$, otherwise it should be selected. If the event is of type (b) then in the worst case the removed set has cardinality $m(A_{j_1})$.

Consider, now, the i -th event that happens in j_i . If this event is of type (a) then the set of A^* which is not selected is of size smaller than $m(A_{j_i}) + \frac{m(A_{j_i})}{x}$, otherwise it should be selected. But, some of the elements of A_{j_i} might also appear in $A_{j_{i-1}}$, and hence they are already taken into account for OPT . Thus, the i -th event of type (a) adds at most:

$$m(A_{j_i} \setminus A_{j_{i-1}}) + \frac{m(A_{j_i})}{x}$$

new elements to A^* . In a similar way, the i -th event of type (b) adds at most $m(A_{j_i} \setminus A_{j_{i-1}})$ new elements to $m(A^*)$.

We observe that in all cases the events of type (a) dominate the events of type (b). By summing up all elements of A^* as described above, we have:

$$OPT \leq m(A_{j_1}) + \sum_{i=2}^k m(A_{j_i} \setminus A_{j_{i-1}}) + \sum_{i=1}^k \frac{m(A_{j_i})}{x}$$

Since $m(A_{j_i} \setminus A_{j_{i-1}}) = m(A_{j_i}) - m(A_{j_{i-1}} \cap A_{j_i})$, we get:

$$OPT \leq m(A_{j_k}) + \sum_{i=2}^k m(A_{j_{i-1}} \cap A_{j_i}) + \sum_{i=1}^k \frac{m(A_{j_i})}{x}$$

Claim 1. $m(A_{j_{i-1}} \cap A_{j_i}) \leq \frac{|A_{j_{i-1}} \cap A_{j_i}|}{k} m(A_{j_{i-1}})$.

Proof. Let Q_r , $1 \leq r \leq |A_{j_{i-1}} \cap A_{j_i}|$, be the r -th set that is removed from $A_{j_{i-1}}$ between the events $i-1$ and i , and q_r be the private part of Q_r just before it is removed. We will show that, for any p , $1 \leq p \leq |A_{j_{i-1}} \cap A_{j_i}|$,

it holds that $\sum_{r=1}^p q_r \leq \frac{p}{k} m(A_{j_{i-1}})$, and thus:

$$m(A_{j_{i-1}} \cap A_{j_i}) = \sum_{r=1}^{|A_{j_{i-1}} \cap A_{j_i}|} q_r \leq \frac{|A_{j_{i-1}} \cap A_{j_i}|}{k} m(A_{j_{i-1}})$$

Assume a contrario that for the first time after the removal of the set Q_p

it holds that $\sum_{r=1}^p q_r > \frac{p}{k} m(A_{j_{i-1}})$, hence, $\sum_{r=1}^{p-1} q_r \leq \frac{p-1}{k} m(A_{j_{i-1}})$. Clearly,

$q_p > \frac{m(A_{j_{i-1}})}{k}$. Moreover, following ALGORITHM MkC Q_p has the smallest private part between the sets belonging in the solution when Q_p is selected to be removed. Thus, the $k - p$ sets of $A_{j_{i-1}}$ which are still in the solution have private parts of size greater than $(k - p) \frac{m(A_{j_{i-1}})}{k}$ in total. Consequently:

$$\begin{aligned} m(A_{j_{i-1}}) &> \sum_{r=1}^p q_r + (k - p) \frac{m(A_{j_{i-1}})}{k} \\ &> \frac{p}{k} m(A_{j_{i-1}}) + (k - p) \frac{m(A_{j_{i-1}})}{k} = m(A_{j_{i-1}}) \end{aligned}$$

a contradiction. Therefore, there is no p such that $\sum_{r=1}^p q_r > \frac{p}{k} m(A_{j_{i-1}})$, and the claim is proved. $\square$

By definition, it holds that $j_k \leq z$ and hence $m(A_{j_k}) \leq m(A_z)$. Moreover, by ALGORITHM MkC, $m(A_{j_k}) \geq (1 + \frac{1}{x})^{j_k - j_i} m(A_{j_i})$. Thus, using Claim 1, we have:

$$\begin{aligned} OPT &\leq m(A_{j_k}) + \sum_{i=2}^k \frac{|A_{j_{i-1}} \setminus A_{j_i}|}{k} m(A_{j_{i-1}}) + \sum_{i=1}^k \frac{m(A_{j_i})}{x} \\ &\leq m(A_{j_k}) + \frac{1}{k} \sum_{i=2}^k (j_i - j_{i-1}) \frac{m(A_{j_k})}{(1 + \frac{1}{x})^{j_k - j_{i-1}}} + \frac{1}{x} \sum_{i=1}^k m(A_{j_k}) \\ &\leq m(A_z) \left(1 + \frac{1}{k} \sum_{i=2}^k \frac{j_i - j_{i-1}}{(1 + \frac{1}{x})^{j_k - j_{i-1}}} + \frac{k}{x} \right) \end{aligned} \quad (1)$$

Let $d_i = j_i - j_{i-1}$ and $\beta = (1 + \frac{1}{x})^{-1}$. Then, the sum in the righthand side of (1) becomes $\sum_{i=2}^k d_i \cdot \beta^{\sum_{j=i}^k d_j}$.

Consider, now, the function:

$$f_k(\beta, d) = \sum_{i=2}^k d_i \cdot \beta^{\sum_{j=i}^k d_j} = d_k \cdot \beta^{d_k} + \sum_{i=2}^{k-1} d_i \cdot \beta^{\sum_{j=i}^k d_j} = \beta^{d_k} (d_k + f_{k-1}(\beta, d))$$

where:

$$\frac{\partial f_k}{\partial d_k} = \ln \beta \cdot \beta^{d_k} (d_k + f_{k-1}(\beta, d)) + \beta^{d_k} = \beta^{d_k} (\ln \beta (d_k + f_{k-1}(\beta, d)) + 1)$$

Since $\ln \beta < 0$, the global maximum is attained for $d_k + f_{k-1}(\beta, d) = \frac{-1}{\ln \beta}$. Thus:

$$\begin{aligned} f_k(\beta, d) &\leq \beta^{\frac{-1}{\ln \beta} - f_{k-1}(\beta, d)} \cdot \frac{-1}{\ln \beta} = -\frac{e^{\left(\frac{-1}{\ln \beta} - f_{k-1}(\beta, d)\right) \ln \beta}}{\ln \beta} \\ &= -\frac{e^{-f_{k-1}(\beta, d) \ln \beta}}{e \cdot \ln \beta} \end{aligned} \quad (2)$$

Claim 2. For any $k \geq 2$, it holds that $-f_k(\beta, d) \ln \beta \leq 1$.

Proof. We will prove it by induction to k . For $k = 2$, by definition it holds that $f_2(\beta, d) = d_2 \beta^{d_2}$. Hence:

$$-f_2(\beta, d) \ln \beta = -d_2 \beta^{d_2} \ln \beta = \frac{\ln \beta^{-d_2}}{\beta^{-d_2}} \leq 1$$

For $k > 2$, by (2), the inductive hypothesis and $\beta \leq 1$ we have:

$$-f_k(\beta, d) \ln \beta \leq \frac{e^{-f_{k-1}(\beta, d) \ln \beta}}{e} \leq \frac{e^1}{e} = 1$$

and the proof of the claim is completed. $\square$

We are well prepared now to conclude the proof of the theorem. Using Claim 2 and expression (2), we get:

$$f_k(\beta, d) \leq -\frac{e^{-f_{k-1}(\beta, d) \ln \beta}}{e \cdot \ln \beta} = -\frac{1}{\ln \beta} = -\frac{1}{\ln \left(1 + \frac{1}{x}\right)^{-1}} = \frac{1}{\ln \left(1 + \frac{1}{x}\right)}$$

Using (1), the ratio of the algorithm becomes:

$$\frac{SOL}{OPT} \geq \frac{m(A_z)}{m(A_z) \left(1 + \frac{1}{k} \cdot \frac{1}{\ln \left(1 + \frac{1}{x}\right)} + \frac{k}{x}\right)} = \frac{1}{1 + \frac{1}{\ln \left(1 + \frac{1}{x}\right)^k} + \frac{k}{x}} \quad (3)$$

Some simple algebra derives that the denominator of (3) is minimized for $x = k$. In this case we get:

$$\frac{SOL}{OPT} \geq \frac{1}{1 + \frac{1}{\ln \left(1 + \frac{1}{k}\right)^k} + \frac{k}{k}} \simeq \frac{1}{3}$$

that concludes the proof of the theorem. $\square$

It is hopefully clear from the proof of Theorem 1 just above, that it works also for the WEIGHTED MkC problem, up to the assumption that $m(\cdot)$ in ALGORITHM MkC and in the proof of Theorem 1 is the total weight of the elements and not their number. So, the following result immediately holds.

Theorem 2. *The WEIGHTED MkC problem can be solved within an asymptotic $\frac{1}{3}$ -competitive ratio.*

We conclude the section by providing an (unfortunately non tight yet) upper bound the competitiveness of ALGORITHM MkC .

Figure 2: An example in which the algorithm achieves $\frac{1}{2}$ -competitive ratio.

Proposition 3. *The competitive ratio of ALGORITHM MkC is bounded above by $\frac{1}{2}$.*

Proof. Consider the instance shown in Figure 2. There, ALGORITHM MkC rejects all sets that appear after A_1 because they do not increase the solution by at least $\frac{A_1}{k}$. Hence, the solution obtained covers $m(A_1)$ elements, while the optimal solution A^* contains the k sets that are released after A_1 , and thus covers $m(A_1) + k \cdot \left(\frac{m(A_1)}{k} - 1\right)$ elements. Therefore,

$$\frac{SOL}{OPT} = \frac{m(A_1)}{m(A_1) + k \cdot \left(\frac{m(A_1)}{k} - 1\right)} = \frac{1}{2 - \frac{k}{m(A_1)}} \simeq \frac{1}{2}$$

completing so the proof. $\square$

Let us note that we feel that a more involved technical analysis of ALGORITHM MkC could improve its competitiveness bringing it closer to $\frac{1}{2}$.

3 Maximum k -vertex coverage

In this section we deal with the online maximum k -vertex coverage problem. Note that there exists a trivial $\frac{1}{2}$ -competitive ratio for this problem. In fact, consider selecting k vertices of largest degrees. In an optimal solution all the edges are, at best, covered once, while in the solution created by this greedy algorithm, all the edges are, at worst, covered twice. Since the algorithm selects the largest degrees of the graph, the $\frac{1}{2}$ -competitive ratio is immediately concluded.

Let us note that by a similar proof to the one of Proposition 1, one can prove that for the case of $MkVC$, *any deterministic online algorithm that does not allow swaps cannot achieve a competitive ratio better than $O\left(\frac{1}{n^{1/(k+1)}}\right)$* : in i -th step a vertex of $n^{i/(k+1)}$ neighbors is released if the algorithm has selected all the sets that have been released in the previous steps; otherwise only vertices of degree one are released. The proof is completely similar to the proof of Proposition 1.

In next subsections we improve the $\frac{1}{2}$ -competitive ratio for several classes of graphs. But first, we give an easy upper bound for the number of elements covered by any solution that will be used later. Its proof is straightforward.

Proposition 4. $OPT \leq k\Delta$.

3.1 Regular graphs

The following preliminary result that will be used later holds for any algorithm for the $MkVC$ problem in regular graphs.

Proposition 5. *Any deterministic online algorithm achieves a $\frac{k}{n}$ -competitive ratio for the $MkVC$ problem on regular graphs.*

Proof. An optimal solution covers at most all the edges of the graph (recall that $|E| = m$), that is $OPT \leq m = \frac{n\Delta}{2}$. On the other hand, any solution covers $k\Delta$ edges, some of them eventually twice, i.e., at least $\frac{k\Delta}{2}$ edges, that is $SOL \geq \frac{k\Delta}{2}$. We so get $\frac{SOL}{OPT} \geq \frac{k}{n}$. $\square$

Let us note that the result of Proposition 5 for the $MkVC$ problem also holds for general graphs in the offline setting [9].

We now present an algorithm for the $MkVC$ problem in regular graphs (ALGORITHM $MkVC$ -R). This algorithm depends on a parameter x which, as in ALGORITHM MkC , indicates the improvement on the current solution that a new vertex should entail, in order to be selected for inclusion in the

solution. In other words, we replace a vertex of the current solution by the released one, only if the solution increases by at least $\lceil \frac{\Delta}{x} \rceil$ edges.

ALGORITHM $MkVC-R(x)$

```

1:  $A = \emptyset; B = \emptyset;$ 
2: for each released vertex  $v$  do
3:   if  $|A| < k$  then
4:      $A = A \cup \{v\};$ 
5:     if  $v$  improves the current solution by at least  $\lceil \frac{\Delta}{x} \rceil$  then
6:        $B = B \cup \{v\};$ 
7:   else if  $|B| < k$  and  $v$  improves the current solution by at least  $\lceil \frac{\Delta}{x} \rceil$  then
8:     Select a vertex  $u \in A \setminus B;$ 
9:      $A = A \cup \{v\} \setminus \{u\}; B = B \cup \{v\};$ 
10: return  $A;$ 

```

Theorem 3. ALGORITHM $MkVC-R$ achieves competitive ratio bounded below by $\frac{2n}{n + 2k + \sqrt{4k^2 + n^2}}$.

Proof. Note that $B \subseteq A$ consists of the vertices that improve the solution by at least $\lceil \frac{\Delta}{x} \rceil$; b denotes the number of these vertices, i.e., $b = |B|$. We denote by y_1 the number of edges with their one endpoint in B and the other in $V \setminus B$, and by y_2 the number of edges with their both endpoints in B . By definition,

$$SOL \geq y_1 + y_2 = b\Delta - y_2 = \frac{b\Delta - y_1}{2} + y_1 = \frac{b\Delta + y_1}{2} \quad (4)$$

We shall handle two cases, depending on the value of b with respect to k .

If $b < k$ then each vertex $v \in V \setminus B$ is not selected by ALGORITHM $MkVC-R(x)$ to be in B because it is adjacent to at most $\lceil \frac{\Delta}{x} \rceil - 1$ vertices of $V \setminus B$. Thus, there are at least $\Delta - \lceil \frac{\Delta}{x} \rceil + 1$ edges that connect v with vertices in B . Summing up for all the vertices in $V \setminus B$, it holds that $y_1 \geq (n - b)(\Delta - \lceil \frac{\Delta}{x} \rceil + 1)$, and considering also (4) we get:

$$SOL \geq (n - b) \left(\Delta - \left\lceil \frac{\Delta}{x} \right\rceil + 1 \right) + y_2 \quad (5)$$

$$SOL \geq \frac{b\Delta + (n - b) \left(\Delta - \left\lceil \frac{\Delta}{x} \right\rceil + 1 \right)}{2} \quad (6)$$

Using the upper bound for the optimum provided by Proposition 4 and expressions (5) and (6), respectively, we get the following ratios:

$$\begin{aligned}\frac{SOL}{OPT} &\geq \frac{(n-b)(\Delta - \lceil \frac{\Delta}{x} \rceil + 1) + y_2}{k\Delta} \\ &\geq \frac{(n-b)(x-1)}{kx} = \frac{n(x-1) - b(x-1)}{kx}\end{aligned}\quad (7)$$

$$\begin{aligned}\frac{SOL}{OPT} &\geq \frac{\frac{b\Delta + (n-b)(\Delta - \lceil \frac{\Delta}{x} \rceil + 1)}{2}}{k\Delta} \\ &\geq \frac{bx + (n-b)(x-1)}{2kx} \geq \frac{n(x-1) + b}{2kx}\end{aligned}\quad (8)$$

Observe that the righthand side of (7) decreases with b while that of (8) increases; thus, the worst case occurs when righthand sides of them are equal, that is:

$$\frac{n(x-1) - b(x-1)}{kx} = \frac{n(x-1) + b}{2kx} \Leftrightarrow b = \frac{n(x-1)}{2x-1}$$

and hence:

$$\frac{SOL}{OPT} \geq \frac{n(x-1) + \frac{n(x-1)}{2x-1}}{2kx} = \frac{n(x-1)}{k(2x-1)}\quad (9)$$

If $b = k$, then trivially holds that:

$$\frac{SOL}{OPT} \geq \frac{k \lceil \frac{\Delta}{x} \rceil}{k\Delta} \geq \frac{1}{x}\quad (10)$$

Note that (9) increases with x while (10) decreases; therefore, for the worst case we have:

$$\frac{n(x-1)}{k(2x-1)} = \frac{1}{x} \Leftrightarrow x = \frac{n + 2k + \sqrt{4k^2 + n^2}}{2n}$$

In all, it holds that:

$$\frac{SOL}{OPT} \geq \frac{2n}{n + 2k + \sqrt{4k^2 + n^2}}\quad (11)$$

as claimed. $\square$

Let us note that, as it can be easily derived from Theorem 3, when $k = o(n)$ the competitive ratio of ALGORITHM *mkvc-R* is asymptotical to 1.

Putting together Proposition 5 and Theorem 3, the following theorem holds.

Theorem 4. *The $MkVC$ problem can be solved within an 0.55-competitive ratio in regular graphs.*

Proof. If $k < 0.55n$, the ratio of ALGORITHM $MkVC$ -R (expression 11) leads to:

$$\frac{SOL}{OPT} \geq \frac{2n}{n + 2(0.55n) + \sqrt{4(0.55n)^2 + n^2}} = \frac{2}{2.11 + \sqrt{2.21}} = 0.55$$

On the other hand, the ratio provided in Proposition 5, for $k > 0.55n$, gives $\frac{SOL}{OPT} \geq \frac{k}{n} \geq \frac{0.55n}{n} = 0.55$.

Putting all the above together, the 0.55-competitive ratio claimed is concluded. $\square$

ALGORITHM $MkVC$ -R(x) can be also used for the online MkC problem when all sets have the same cardinality. Nevertheless, this algorithm cannot give a competitive ratio better than $O\left(\frac{1}{\sqrt{k}}\right)$ for this case, which is much worst than the ratio achieved in Section 2. For completeness, its analysis is given in appendix.

3.2 Regular bipartite graphs

A better ratio can be achieved if we further restrict ourselves in regular bipartite graphs. A key-point of such improvement is that the maximum independent set can be found in polynomial time in bipartite graphs (see for example [11]).

Our ALGORITHM $MkVC$ -B initializes its solution with the first k released vertices. At this point, a maximum independent set B , of size b , in the graph induced by these k vertices is found. The vertices of this independent set will surely appear to the final solution. For the rest $k - b$ vertices we check if they cover at least $\frac{\frac{n\Delta}{2} - b\Delta}{\lceil \frac{n-b}{k-b} \rceil}$ edges different from those covered by the independent set B . If yes, we return the solution consisting of the b vertices of the independent set and these $k - b$ vertices. Otherwise, we wait for the next $k - b$ vertices and we repeat the check. In ALGORITHM $MkVC$ -B, $G[A]$ denotes the subgraph of G induced by the vertex-subset A .

Theorem 5. *ALGORITHM $MkVC$ -B achieves a competitive ratio bounded*

above by $\frac{k + \frac{n-k}{\lceil \frac{2n-k}{k} \rceil}}{2k}$.

ALGORITHM $MkVC-B$

```

1:  $A = \{\text{the first } k \text{ released vertices}\};$ 
2: Find a maximum independent set  $B \subseteq A$  in  $G[A]$ ;  $b = |B|$ ;
3: for each released vertex  $v$  do
4:   if  $|A| = k$  then
5:     if  $m(A) \geq b\Delta + \frac{\frac{n\Delta}{2} - b\Delta}{\lceil \frac{n-b}{k-b} \rceil}$  then
6:       return  $A$ ;
7:     else
8:        $A = B$ ;
9:     else
10:       $A = A \cup \{v\}$ 
11: return  $A$ ;

```

Proof. Let us call *batch* the set of the $k - b$ vertices of $A \setminus B$ in Lines 5–10 of ALGORITHM $MkVC-B$.

The solution obtained by this algorithm contains a maximum independent set of size b . Since the input graph is bipartite, it holds that $b \geq \frac{k}{2}$.

The number of edges of the graph uncovered by the vertices of the maximum independent set is in total $\frac{n\Delta}{2} - b\Delta$. Any of these edges is covered by vertices belonging to at least one of the $\lceil \frac{n-b}{k-b} \rceil$ batches. Hence, in average, each batch covers $\frac{\frac{n\Delta}{2} - b\Delta}{\lceil \frac{n-b}{k-b} \rceil}$ of those edges; so there exists a batch that covers at least $\frac{\frac{n\Delta}{2} - b\Delta}{\lceil \frac{n-b}{k-b} \rceil}$ of them. Therefore, the algorithm covers in total at least $b\Delta + \frac{\frac{n\Delta}{2} - b\Delta}{\lceil \frac{n-b}{k-b} \rceil}$ edges.

Using Proposition 4, we get:

$$\frac{SOL}{OPT} \geq \frac{b\Delta + \frac{\frac{n\Delta}{2} - b\Delta}{\lceil \frac{n-b}{k-b} \rceil}}{k\Delta} = \frac{b + \frac{\frac{n}{2} - b}{\lceil \frac{n-b}{k-b} \rceil}}{k}$$

and since this quantity increases with b it holds that:

$$\frac{SOL}{OPT} \geq \frac{\frac{k}{2} + \frac{\frac{\frac{n}{2} - \frac{k}{2}}{\lceil \frac{n - \frac{k}{2}}{k - \frac{k}{2}} \rceil}}{k}}{k} = \frac{k + \frac{n-k}{\lceil \frac{2n-k}{k} \rceil}}{2k} \quad (12)$$

as claimed. $\square$

Note that ALGORITHM *MkVC-B* also achieves a competitive ratio asymptotical to $\frac{3}{4}$ when $k = o(n)$.

Putting together Theorem 5 and Proposition 5, the following theorem holds and concludes this section.

Theorem 6. *The *MkVC* problem can be solved within a 0.6075-competitive ratio in regular bipartite graphs.*

Proof. If $k \leq 0.6075n$, then expression (12) in the proof of Theorem 5 leads to $\frac{SOL}{OPT} \geq 0.6075$. Otherwise, using Proposition 5 we get the same ratio and the theorem is concluded. $\square$

3.3 Trees and chains

In this section we give algorithms that further improve the competitive ratios for the *MkVC* problem in trees and chains. Dealing with trees the following result holds.

Proposition 6. *The *MkVC* problem can be solved within $(1 - \frac{k-1}{\Delta^*})$ -competitive ratio in trees, where Δ^* is the sum of the k largest degrees in the tree. The ratio is tight.*

Proof. An upper bound for the optimal solution is $OPT \leq \Delta^*$, that is the case where k non-adjacent vertices of the largest degree are selected.

Consider the algorithm that selects the k vertices of the largest degrees. These k vertices cover Δ^* edges, some of them possibly twice. It is easy to see that the number of such edges is maximized when the subgraph induced by the k selected vertices is connected. In this case, there are $k - 1$ edges covered twice. Hence, the total number of covered edges is $\Delta^* - (k - 1)$, while at most Δ^* edges can be covered by any solution. $\square$

Note that, if the number of vertices of degree greater than 1 is $r < k$ then our algorithm finds an optimal solution using just r vertices, since the edges that are adjacent to the leaves are covered by their other endpoints.

Furthermore, in the case where all the internal vertices of the tree have the same degree Δ , the ratio provided by Proposition 6 becomes $(1 - \frac{k-1}{k\Delta})$. This ratio is better than the ratio proved for regular bipartite graphs in Theorem 6 for any $\Delta \geq 3$, but it is worse for $\Delta = 2$, i.e., in the case where the input graph is a chain. An improvement for the *MkVC* problem in chains follows.

Proposition 7. *ALGORITHM *MkVC-C* achieves a 0.75-competitive ratio for the *MkVC* problem in chains.*

ALGORITHM $MkVC-C$

```
1: Split the “memory” of the solution into two parts  $B$  and  $C$ , i.e.,  $A = B \cup C$ ;  
2: for each released vertex  $v$  do  
3:   if  $|B| \leq k$  and  $v$  adds two new edges to the solution then  
4:      $B = B \cup \{v\}$ ;  
5:   if  $|A| > k$  then  
6:     Delete an arbitrary vertex from  $C$ ;  
7:   else if  $|A| \leq k$  and  $v$  adds one new edge to the solution then  
8:      $C = C \cup \{v\}$ ;  
9: return  $A$ ;
```

Proof. Note, first, that if $k < \frac{n}{3}$ or $k > \frac{2n}{3}$, then the algorithm finds an optimal solution.

It is easy to see that $|B| \geq |C|$. Moreover, it holds that $SOL = 2|B| + |C|$, while $OPT \leq 2k = 2|B| + 2|C|$. Hence:

$$\frac{SOL}{OPT} \geq \frac{2|B| + |C|}{2|B| + 2|C|} \geq \frac{2|B| + |B|}{2|B| + 2|B|} = \frac{3}{4} = 0.75$$

and the proof is completed. $\square$

4 Conclusions

We have presented improved algorithms for a natural online model of the MkC problem and of the $MkVC$ problem in regular general and bipartite graphs. To our opinion, the main open problem with respect to the model adopted is the relaxation of the graph-regularity hypothesis and the extension of our results to general graphs.

References

- [1] A. Ageev and M. Sviridenko. Approximation algorithms for maximum coverage and max cut with given sizes of parts. In *7th Integer Programming and Combinatorial Optimization (IPCO'99)*, volume 1610 of *LNCS*, pages 17–30. Springer, 1999.
- [2] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor. The online set cover problem. In *35th Annual ACM Symposium on the Theory of Computing (STOC'03)*, pages 100–105. ACM, 2003.

- [3] G. Ausiello, N. Bourgeois, A. Giannakos, and V. Th. Paschos. Greedy algorithms for on-line set-covering. *Algorithmic Operations Research*, 4:36–48, 2009.
- [4] B. Awerbuch, Y. Azar, A. Fiat, and T. Leighton. Making commitments in the face of uncertainty: how to pick a winner almost every time (extended abstract). In *28th Annual ACM Symposium on Theory of Computing (STOC'96)*, pages 519–530. ACM, 1996.
- [5] N. Buchbinder and J. Naor. Online primal-dual algorithms for covering and packing problems. In *13th Annual European Symposium on Algorithms (ESA'05)*, volume 3669 of *LNCS*, pages 689–701. Springer, 2005.
- [6] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45:634–652, 1998.
- [7] U. Feige and M. Langberg. Approximation algorithms for maximization problems arising in graph partitioning. *Journal of Algorithms*, 41:174–211, 2001.
- [8] F. Grandoni, A. Gupta, S. Leonardi, P. Miettinen, P. Sankowski, and M. Singh. Set covering with our eyes closed. In *49th Annual IEEE Symposium on Foundations of Computer Science (FOCS'08)*, pages 347–356, 2008.
- [9] Q. Han, Y. Ye, H. Zhang, and J. Zhang. On approximation of max-vertex-cover. *European Journal of Operational Research*, 143:342–355, 2002.
- [10] D. S. Hochbaum and A. Pathria. Analysis of the greedy approach in problems of maximum k -coverage. *Naval Research Logistics*, 45:615–627, 1998.
- [11] V. Th. Paschos. A survey of approximately optimal solutions to some covering and packing problems. *ACM Computing Surveys*, 29:171–209, 1997.
- [12] B. Saha and L. Getoor. On maximum coverage in the streaming model & application to multi-topic blog-watch. In *9th SIAM International Conference on Data Mining (DM'09)*, pages 697–708. SIAM, 2009.

A Analysis of ALGORITHM $MkVC-R$ for the MkC problem

We analyze, here, ALGORITHM $MkVC-R(x)$ for the special case of MkC problem where every set contains exactly Δ elements.

Proposition 8. ALGORITHM $MkVC-R(x)$ provides an approximation ratio bounded by $\frac{2}{\sqrt{4k+1}+1} \sim \frac{1}{\sqrt{k}}$. This bound is asymptotically tight.

Proof. Let $b = |B|$. Denote by A the solution computed by ALGORITHM $MkVC-R$ and fix an optimal solution A^* . As in the proof of Theorem 4 we distinguish two cases, with respect to the values of b and k .

If $b < k$ then consider the following partition of the optimum: $A^* = (A^* \cap A) \cup (A^* \setminus A)$. The first part can be bounded easily by: $m(A^* \cap A) \leq m(A) = SOL$. For the second part, note that at most k sets contributing to it were not chosen by the algorithm. Thus, they represent individually an improvement of less than $\frac{\Delta}{x}$ with respect to A and, obviously, also with respect to $A^* \cap A$. So, the number of elements in the second part is at most $\frac{k\Delta}{x}$ and the competitive ratio of ALGORITHM $MkVC-R(x)$ can be written as:

$$\frac{SOL}{OPT} \geq \frac{SOL}{SOL + k\frac{\Delta}{x}} = \frac{1}{1 + \frac{k\Delta}{xSOL}}$$

and observing that $SOL \geq \Delta$ we obtain:

$$\frac{SOL}{OPT} \geq \frac{1}{1 + \frac{k}{x}} = \frac{x}{x+k} \quad (13)$$

If $b = k$ then as in the proof of Theorem 4 (expression (10)) we have:

$$\frac{SOL}{OPT} \geq \frac{1}{x} \quad (14)$$

Since (13) increasing with x while (14) is decreasing, some easy algebra derives that the optimal value for x is $\frac{\sqrt{4k+1}+1}{2}$ and, putting it to (13) or (14), the claimed bound follows.

For the tightness of the ratio, consider the following scenario. Assume, that initially k sets that cover exactly the same Δ elements are released; the algorithm selects all of them. Then, $k-1$ sets are released; each of them covers $\frac{\Delta}{x} - 1$ elements privately. Thus, the algorithm does not select any of them and hence $SOL = \Delta$. The optimal solution consists of one of the first sets plus the $k-1$ last sets, and hence, $OPT = \Delta + (k-1)(\frac{\Delta}{x} - 1)$. Therefore, $\frac{SOL}{OPT} \simeq \frac{x}{x+k}$ which is the same ratio as in case where $b < k$ (due to (13)). $\square$