POLITECNICO DI TORINO
Repository ISTITUZIONALE

Online maximum k-coverage

Original
Online maximum k-coverage / G., A., Boria, N., A., G., G., L., V. T., P.. - In: DISCRETE APPLIED MATHEMATICS. - ISSN 0166-218X. - 160:(2012), pp. 1901-1913. [10.1016/j.dam.2012.04.005]

Availability:
This version is available at: 11583/2500169 since:

Publisher:
Elsevier

Published
DOI:10.1016/j.dam.2012.04.005

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

12 October 2026

Online maximum k-coverage*

Giorgio Ausiello† Nicolas Boria‡ Aristotelis Giannakos‡
Giorgio Lucarelli‡ Vangelis Th. Paschos‡§

Abstract

We study an online model for the maximum k-vertex-coverage problem, where given a graph $G = (V, E)$ and an integer k, we ask for a subset $A \subseteq V$, such that $|A| = k$ and the number of edges covered by A is maximized. In our model, at each step i, a new vertex v_i is revealed, and we have to decide whether we will keep it or discard it. At any time of the process, only k vertices can be kept in memory; if at some point the current solution already contains k vertices, any inclusion of any new vertex in the solution must entail the irremediable deletion of one vertex of the current solution (a vertex not kept when revealed is irremediably deleted). We propose algorithms for several natural classes of graphs (mainly regular and bipartite), improving on an easy $\frac{1}{2}$-competitive ratio. We next settle a set-version of the problem, called maximum k-(set)-coverage problem. For this problem we present an algorithm that improves upon former results for the same model for small and moderate values of k.

1 Introduction

In the *maximum k-vertex-coverage* (MkVC) problem we are given a graph $G = (V, E)$ ($|V| = n$, $|E| = m$) and an integer k, we ask for a subset $A \subseteq V$, such that $|A| = k$ and the number of edges covered by A is maximized. The MkVC problem is NP-hard, since otherwise the optimal solution for the vertex cover problem could be found in polynomial time: for each k, $1 \leq k \leq n$, run the algorithm for the MkVC 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 vertex v_i with its adjacent edges is revealed, and we have to decide whether we will include v_i in the solution or discard it. At any time of the process, only k vertices can be kept in memory, so if at some point the

*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, CNRS UMR 7243 and Université Paris-Dauphine

§Institut Universitaire de France

current solution already contains k vertices, any inclusion of any new vertex in the solution must be compensated with the irremediable deletion of one vertex of the current solution. Of course, a vertex that is not kept when it is revealed is also irremediably deleted. To our knowledge, no online model for the MkVC problem has been studied until now.

A generalization of the MkVC problem is the *maximum k-(set)-coverage* (MkC) problem, where given a universe of elements $E = \{e_1, e_2, \ldots, e_m\}$, a collection of subsets of E, $S = \{S_1, S_2, \ldots, S_n\}$, and an integer $k \leq n$, we ask for a subcollection $A = \{A_1, A_2, \ldots, A_{|A|}\} \subseteq S$, such that $|A| = k$ and the number of elements of E covered by A is maximized. The online model for the MkC problem is the same as for the MkVC.

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 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 analogous online model for WEIGHTED MkC problem, where at each step i a set $S_i \in S$ together with its elements is revealed and only k such sets can be kept in memory, has been studied in [7], 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 the classical offline setting, the MkC problem is known to be non approximable within a factor $1-\frac{1}{e}$ [2]. On the other hand, even for the weighted version of the problem, an approximation algorithm of ratio $1 - \left(1 - \frac{1}{k}\right)^k$ is known [5]. 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 (i.e., the hitting set version of MkC), also called *maximum coverage problem*, has been studied: given a universe of elements $E = \{e_1, e_2, \ldots, e_m\}$, a collection of subsets of E, $S = \{S_1, S_2, \ldots, S_n\}$, 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]. Several improvements for some restricted cases of the MkVC problem are presented in [3, 4].

In this paper we study the online model described above for both the MkVC and the MkC problems. In Section 2, we prove several negative results on the competitiveness of any algorithm for the model handled for both problems. In

Section 3, we present algorithms for regular graphs, regular bipartite graphs, trees and chains, achieving non-trivial competitive ratios, improving upon an easy $\frac{1}{2}$ competitiveness result holding for any graph. Finally, in Section 4 the k-(set)-coverage problem is handled. For this problem, we present an algorithm that improves upon former results for the same model for small and moderate values of k.

The following notations will be used in the sequel. They are based upon the definition of the MkVC problem and are easily extendable to the MkC problem.

For any $A \subseteq V$, we denote by $E(A)$ the set of edges covered by A and by $m(A) = |E(A)|$ the number of these edges. Let $SOL = m(A)$ be the number of edges covered by our algorithms. Moreover, we denote by $A^* \subseteq V$ an optimal subset of vertices and by $OPT = m(A^*)$ the number of edges covered by an optimal solution. The maximum degree (or the degree when it is regular) of the input-graph $G = (V, E)$ is denoted by Δ. Dealing with MkC, Δ denotes the cardinality of a set that contains a maximum number of elements, that is, $\Delta = \max\{|S_i| : 1 \leq i \leq n\}$. For a subset $A \subseteq V$ and a vertex $v_i \in A$, we call *public* the edges of v_i that are shared by another vertex in A and *private* the edges of v_i that are covered just by v_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.

In what follows, for reasons of papers length, some proofs are omitted. They can be found in appendix.

2 Negative results

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

Proposition 1. *Any deterministic online algorithm that does not allow swaps cannot achieve a competitive ratio better than:*

- $O\left(\frac{1}{(n-1)^{1/(k+1)}}\right)$ *for the* MkVC *problem;*
- $O\left(\frac{1}{m^{1/(k+1)}}\right)$ *for the* MkC *problem.*

The next negative result for the MkVC problem fits the model addressed in the paper (swaps are allowed).

Proposition 2. *Any deterministic online algorithm cannot achieve a competitive ratio better than* $\frac{2k}{3k-2} \simeq \frac{2}{3}$ *for the* MkVC *problem.*

Proof. Assume that $2k-1$ vertices, $v_1^1, v_2^1, \ldots, v_{2k-1}^1$, of degree one and $2k-1$ vertices, $v_1^2, v_2^2, \ldots, v_{2k-1}^2$, of degree two are released, such that $(v_i^1, v_i^2) \in E$, $1 \leq i \leq 2k-1$, and that the algorithm selects $k' \leq k$ of them. Wlog, let $v_1^2, v_2^2, \ldots, v_{k'}^2$ be the vertices selected by the algorithm. Next the vertex v_3 of degree k' is released, where $(v_i^2, v_3) \in E$, $1 \leq i \leq k'$. The solution of the algorithm at this time is $2k'$, while the inclusion or not of v_3 does not play any role for this value. Finally, $2k-1-k'$ vertices, $v_{k'+1}^3, v_{k'+2}^3, \ldots, v_{2k-1}^3$, of degree one are released, such that $(v_i^2, v_i^3) \in E$, $k'+1 \leq i \leq 2k-1$. In this last phase, the algorithm can increase its solution by at most $k-k'$ more edges. Hence, the final solution of the algorithm is at most $k+k'$. The optimum solution consists of the vertices $v_{k+1}^2, v_{k+2}^2, \ldots, v_{2k-1}^2, v_3$, and hence is of cardinality $2(k-1)+k'$. In all, $\frac{SOL}{OPT} = \frac{k+k'}{2(k-1)+k'} \leq \frac{2k}{3k-2}$. □

An analogous result can be proved for the MkC problem. Recall that for the offline version of the MkC problem an $1 - \frac{1}{e} \simeq 0.63$-inapproximability result is known [2].

Proposition 3. *Any deterministic online algorithm cannot achieve a competitive ratio better than* $\frac{k+2\sqrt{k}+1}{2k+2\sqrt{k}+1} \simeq \frac{1}{2}$ *for the* MkC *problem even in the case where all sets have the same cardinality.*

3 Maximum k-vertex-coverage

In this section we deal with the online maximum k-vertex-coverage problem. Note, first, that there exists an easy $\frac{1}{2}$-competitive ratio for this problem. In fact, consider selecting k vertices of largest degrees. In an optimum 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.

Proposition 4. *There is a* $\frac{1}{2}$*-competitive ratio for the online* MkVC *problem.*

In the next sections 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 5. $OPT \leq k\Delta$.

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

Let us note that the result of Proposition 6 for the MkVC problem also holds for general graphs in the offline setting [4].

We now present an algorithm for the MkVC problem in regular graphs. Our algorithm depends on a parameter x which 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 increases the edges in B by at least $\lceil \frac{\Delta}{x} \rceil$ **then**
6: $B = B \cup \{v\}$;
7: **else if** $|B| < k$ **and** v increases the edges in B 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;

As we will see in what follows, the best value for x is $x = \frac{n+2k+\sqrt{4k^2+n^2}}{2n}$, leading to the following theorem.

Theorem 1. ALGORITHM MkVC-R *achieves* 0.55*-competitive ratio.*

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 one endpoint in B and the other in $V \setminus B$, and by y_2 the number of edges with both endpoints in B. By definition,

$$SOL \geq y_1 + y_2 = b\Delta - y_2 = \frac{b\Delta - y_1}{2} + y_1 = \frac{b\Delta + y_1}{2} \tag{1}$$

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 $y_1 \geq (n-b)\left(\Delta - \lceil \frac{\Delta}{x} \rceil + 1\right)$, and considering also (1) we get:

$$SOL \geq (n-b)\left(\Delta - \left\lceil \frac{\Delta}{x} \right\rceil + 1\right) + y_2 \tag{2}$$

$$SOL \geq \frac{b\Delta + (n-b)\left(\Delta - \lceil \frac{\Delta}{x} \rceil + 1\right)}{2} \tag{3}$$

Using the upper bound for the optimum provided by Proposition 5 and expressions (2) and (3), respectively, we get the following ratios:

$$\begin{aligned} \frac{SOL}{OPT} &\geq \frac{(n-b)\left(\Delta - \left\lceil \frac{\Delta}{x} \right\rceil + 1\right) + y_2}{k\Delta} \\ &\geq \frac{(n-b)(x-1)}{kx} = \frac{n(x-1)-b(x-1)}{kx} \end{aligned} \tag{4}$$

$$\begin{aligned} \frac{SOL}{OPT} &\geq \frac{\frac{b\Delta + (n-b)\left(\Delta - \left\lceil \frac{\Delta}{x} \right\rceil + 1\right)}{2}}{k\Delta} \\ &\geq \frac{bx + (n-b)(x-1)}{2kx} \geq \frac{n(x-1)+b}{2kx} \end{aligned} \tag{5}$$

Observe that the righthand side of (4) decreases with b while that of (5) 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:

$$\frac{SOL}{OPT} \geq \frac{n(x-1) + \frac{n(x-1)}{2x-1}}{2kx} = \frac{n(x-1)}{k(2x-1)} \tag{6}$$

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

$$\frac{SOL}{OPT} \geq \frac{k\left\lceil \frac{\Delta}{x} \right\rceil}{k\Delta} \geq \frac{1}{x} \tag{7}$$

Note that (6) increases with x while (7) 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:

$$\frac{SOL}{OPT} \geq \frac{2n}{n+2k+\sqrt{4k^2+n^2}} \tag{8}$$

If $k < 0.55n$, the ratio of (8) 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 6 that holds for any algorithm, for $k > 0.55n$, gives $\frac{SOL}{OPT} \geq \frac{k}{n} \geq \frac{0.55n}{n} = 0.55$. □

Let us note that, as it can be easily derived from (8), *when $k = o(n)$ the competitive ratio of* ALGORITHM MkVC-R *is asymptotical to 1.*

3.2 Regular bipartite graphs

A better ratio can be achieved if we further restrict 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 [6]). In what follows in this section, we consider that the number of vertices, n, is known a priori.

Our ALGORITHM MkVC-B initializes its solution with the first k released vertices. At this point, a maximum independent set B, of size $b \leq k$, in the graph induced by these k vertices is found. The vertices of this independent set will surely appear in the final solution. For the remaining $k-b$ vertices we check if they cover at least $\frac{\frac{n\Delta}{2}-b\Delta}{\left\lceil \frac{n-b}{k-b} \right\rceil}$ 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.

ALGORITHM MkVC-B

1: $A = \{$the first k 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** $m(A) \geq b\Delta + \frac{\frac{n\Delta}{2}-b\Delta}{\left\lceil \frac{n-b}{k-b} \right\rceil}$ **then**
6: **return** A;
7: **else**
8: $A = B$;
9: **else**
10: $A = A \cup \{v\}$
11: **return** A;

Theorem 2. ALGORITHM MkVC-B *achieves a* 0.6075*-competitive ratio.*

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 $\left\lceil \frac{n-b}{k-b} \right\rceil$ batches. Hence, in average, each batch covers $\frac{\frac{n\Delta}{2}-b\Delta}{\left\lceil \frac{n-b}{k-b} \right\rceil}$ of those edges; so there exists a batch that covers at least $\frac{\frac{n\Delta}{2}-b\Delta}{\left\lceil \frac{n-b}{k-b} \right\rceil}$ of them. Therefore, the algorithm covers in total at least $b\Delta + \frac{\frac{n\Delta}{2}-b\Delta}{\left\lceil \frac{n-b}{k-b} \right\rceil}$ edges. Using Proposition 5, we get $\frac{SOL}{OPT} \geq \frac{b\Delta+\frac{\frac{n\Delta}{2}-b\Delta}{\left\lceil \frac{n-b}{k-b} \right\rceil}}{k\Delta} = \frac{b+\frac{\frac{n}{2}-b}{\left\lceil \frac{n-b}{k-b} \right\rceil}}{k}$ and since this quantity increases with b it holds that:

$$\frac{SOL}{OPT} \geq \frac{\frac{k}{2} + \frac{\frac{n}{2}-\frac{k}{2}}{\left\lceil \frac{n-\frac{k}{2}}{k-\frac{k}{2}} \right\rceil}}{k} = \frac{k + \frac{n-k}{\left\lceil \frac{2n-k}{k} \right\rceil}}{2k} \tag{9}$$

If $k \leq 0.6075n$, then (9) leads to $\frac{SOL}{OPT} \geq 0.6075$. Otherwise, using Proposition 6 we get the same ratio and the theorem is concluded. □

Note that by (9), ALGORITHM MkVC-B *achieves a competitive ratio asymptotical to* $\frac{3}{4}$ *when* $k = o(n)$.

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

Note that, if the number of vertices of degree greater than 1 is $r < k$ then our algorithm finds an optimum 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 7 becomes $\left(1 - \frac{k-1}{k\Delta}\right)$. This ratio is better than the ratio proved for regular bipartite graphs in Theorem 2 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. The main idea of the algorithm is to partition the solution, A, into two disjoint parts, whose size is dynamically adjusted: the set B of vertices that contribute two edges in $E(A)$ and the set C of vertices that contribute one edge in $E(A)$. Thus, $A = B \cup C$ and $B \cap C = \emptyset$.

ALGORITHM MkVC-C

1: $A = \emptyset$; $B = \emptyset$; $C = \emptyset$; In any step $A \equiv B \cup C$;
2: **for** each released vertex v **do**
3: **if** $|B| \leq k$ **and** v adds two new edges to the solution **then**
4: **if** $|A| = k$ **then**
5: Delete an arbitrary vertex from C;
6: $B = B \cup \{v\}$;
7: **else if** $|A| < k$ **and** v adds one new edge to the solution **then**
8: $C = C \cup \{v\}$;
9: **if** the inclusion of v in A has as a result three consecutive vertices to appear in A **then**
10: Move v from C to B; Remove the middle vertex from A;
11: **return** A;

Proposition 8. *For the* MkVC *problem in chains,* ALGORITHM MkVC-C *returns the (offline) optimum, if* $k < \left\lceil \frac{n}{3} \right\rceil$ *or* $k \geq \left\lceil \frac{2n}{3} \right\rceil$*, and achieves a* 0.75*-competitive ratio, if* $\left\lceil \frac{n}{3} \right\rceil \leq k < \left\lceil \frac{2n}{3} \right\rceil$.

4 Maximum k-(set)-coverage

In this section we present ALGORITHM MkC for the online maximum k-(set)-coverage 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 $\frac{m(A_j)}{k}$. We prove that ALGORITHM MkC achieves competitive ratio strictly greater than $\frac{1}{4}$ but that tends to $\frac{1}{4}$ as k increases. Recall that the algorithm presented in [7] achieves also an $\frac{1}{4}$-competitive ratio. However, our analysis is tight and gives better results for moderately large values of k.

ALGORITHM MkC

1: $j = 1$; $A_j = \{$the first k 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** $m(A_j \setminus \{Q\} \cup \{P\}) > m(A_j) + \frac{m(A_j)}{k}$ **then**
5: $j = j + 1$; $A_j = A_{j-1} \setminus \{Q\} \cup \{P\}$;

To analyze ALGORITHM MkC, let A_z be the final solution obtained, i.e., $SOL = m(A_z)$. Fix, also, an optimum solution A^*.

We consider the following two types of events that may happen during the execution of the algorithm upon arrival of a set P: (a) $P \in A^*$ and ALGORITHM MkC does not select it, and (b) $P \notin A^*$ and ALGORITHM MkC discards $Q \in A^*$ in order to insert P into its current solution. Clearly, at most k such events may happen in total. However, not all events happen in a different current solution; let $\ell \leq k$ be the number of the different current solutions when events happen. Let A_{j_i}, $1 \leq i \leq \ell$, $1 \leq j_i \leq z$, be the i-th of these current solutions, and k_i, $1 \leq i \leq \ell$, be the number of events occurred in A_{j_i}.

We will now provide an upper bound to the value $OPT = m(A^*)$ as function of the states A_{j_i}, $1 \leq i \leq \ell$. Consider that the s-th, $1 \leq s \leq k$, event happens in j_i. Let P_s be the new set that arrives at j_i and $Q_s \in A_{j_i}$ be the set that covers privately the smallest number of elements in A_{j_i}. Let, also, $\widetilde{Q}_s \subseteq Q_s$ be the set of private elements of Q_s in A_{j_i}.

If the event is of type (a) then $P_s \in A^*$ is not selected and it covers a subset of the elements in $E(A_{j_i} \setminus \{Q_s\})$ plus its private elements, $\widetilde{P}_s \subseteq P_s$, in $E(A_{j_i} \setminus \{Q_s\} \cup \{P_s\})$. Note that it is $m(\widetilde{P}_s) \leq m(\widetilde{Q}_s) + \frac{m(A_{j_i})}{k}$, otherwise P_s should be selected by the algorithm. Moreover, $m(\widetilde{Q}_s) \leq \frac{m(A_{j_i})}{k}$, since Q_s has the smallest private part in A_{j_i}, and hence $m(\widetilde{P}_s) \leq \frac{2m(A_{j_i})}{k}$. If the event is of type (b) then $Q_s \in A^*$ is removed and the elements covered by Q_s are a subset of $E(A_{j_i})$. In all, in the worst case we have:

$$E(A^*) \subseteq \bigcup_{i=1}^{\ell} E(A_{j_i}) \cup \bigcup_{s=1}^{k} \widetilde{P}_s \subseteq E(A_{j_\ell}) \cup \bigcup_{i=2}^{\ell} \left[E(A_{j_{i-1}}) \setminus E(A_{j_i})\right] \cup \bigcup_{s=1}^{k} \widetilde{P}_s$$

Therefore, for the value of the optimum solution A^* we have:

$$OPT \leq m(A_{j_\ell}) + \sum_{i=2}^{\ell} m(E(A_{j_{i-1}}) \setminus E(A_{j_i})) + \sum_{i=1}^{\ell} \left(k_i \frac{2m(A_{j_i})}{k} \right)$$

Claim 1. $m(E(A_{j_{i-1}}) \setminus E(A_{j_i})) \leq \frac{|A_{j_{i-1}} \setminus A_{j_i}|}{k} m(A_{j_{i-1}})$, $2 \leq i \leq \ell$.

Using Claim 1 and since $m(A_{j_\ell}) > m(A_{j_i})$, $1 \leq i \leq \ell - 1$, and $\sum_{i=1}^{\ell} k_i = k$ we get:

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

By definition, it holds that $j_\ell \leq z$ and hence $m(A_{j_\ell}) \leq m(A_z) = SOL$. Moreover, by ALGORITHM MkC, $m(A_{j_\ell}) \geq \left(1 + \frac{1}{k}\right)^{j_\ell - j_i} m(A_{j_i})$. Thus, we have:

$$\begin{aligned} \frac{SOL}{OPT} &\geq \frac{1}{1 + \frac{1}{k} \sum_{i=2}^{\ell} \frac{j_i - j_{i-1} + 2}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} + \frac{2(k-\ell+1)}{k}} \\ &= \frac{1}{3 + \frac{1}{k} \sum_{i=2}^{\ell} \frac{j_i - j_{i-1} + 2}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} - \frac{2(\ell-1)}{k}} \end{aligned} \tag{10}$$

Claim 2. *For any $\ell \geq 2$, it holds that $\sum_{i=2}^{\ell} \frac{j_i - j_{i-1} + 2}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} \leq \frac{g(\ell)}{\ln\left(1+\frac{1}{k}\right)}$, where $g(\ell) = \frac{\left(1+\frac{1}{k}\right)^2}{e} \cdot e^{g(l-1)}$ and $g(2) = \frac{\left(1+\frac{1}{k}\right)^2}{e}$.*

Using Claim 2 and (10), we get $\frac{SOL}{OPT} \geq \frac{1}{3 + \frac{1}{k}\left[\frac{g(\ell)}{\ln\left(1+\frac{1}{k}\right)} - 2(\ell-1)\right]}$, where $g(\ell) = \frac{\left(1+\frac{1}{k}\right)^2}{e} \cdot e^{g(l-1)}$ and $g(2) = \frac{\left(1+\frac{1}{k}\right)^2}{e}$. This quantity is minimized for an $\ell = o(k)$. The ratio r achieved by ALGORITHM Mkc for different values of k is shown in Table 1.

k	2	3	5	10	30	50	100	300	500	1000
r	0.333	0.324	0.314	0.300	0.282	0.275	0.268	0.261	0.258	0.256

Table 1: Approximation ratio of ALGORITHM Mkc

To see that the ratio achieved by ALGORITHM Mkc is always greater than $\frac{1}{4}$, consider the following expression for the ratio, slightly less fine than (10):

$$\frac{SOL}{OPT} \geq \frac{1}{3 + \frac{1}{k} \sum_{i=2}^{\ell} \frac{j_i - j_{i-1}}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} + \frac{1}{k} \sum_{i=2}^{\ell} \left(\frac{2}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} - 2 \right)}$$

Note first that if $\ell = 1$ then both sums on the denominator are zero and hence we have a $\frac{1}{3}$-competitive ratio. If $\ell \geq 2$ we have the following analysis. For the first sum, by a similar analysis as in Claim 2 we can prove that $\sum_{i=2}^{\ell} \frac{j_i - j_{i-1}}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} \leq \frac{g(\ell)}{\ln\left(1+\frac{1}{k}\right)}$, where $g(\ell) = \frac{1}{e} \cdot e^{g(l-1)}$ and $g(2) = \frac{1}{e}$. It is easy to see by a simple

induction that $g(\ell) \leq 1$ for any $\ell \geq 2$ and hence $\sum_{i=2}^{\ell} \frac{j_i - j_{i-1}}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} \leq \frac{1}{\ln\left(1+\frac{1}{k}\right)} \leq k$. For the second sum, we have:

$$\sum_{i=2}^{\ell} \left(\frac{2}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} - 2 \right) \leq \sum_{i=2}^{2} \left(\frac{2}{\left(1+\frac{1}{k}\right)} - 2 \right) = \frac{2k}{k+1} - 2 = -\frac{2}{k+1}$$

Therefore, using these bounds to the ratio we get:

$$\frac{SOL}{OPT} \geq \frac{1}{4 - \frac{2}{k(k+1)}} = \frac{1}{4} + \frac{1}{4} \frac{1}{2k(k+1)-1}$$

It is hopefully clear from the previous discussion, that the analysis of ALGORITHM MkC works also for the WEIGHTED MkC problem, up to the assumption that $m(\cdot)$ in ALGORITHM MkC is the total weight of the elements and not their number.

We conclude this section by providing a tight example for the ratio achieved by ALGORITHM MkC. The idea of the example is strongly based upon the proof given above, which indicates the "critical" values of ℓ and j_i, $1 \leq i \leq \ell$. For simplicity, we will consider the case where $k = 3$, but it is easy to extend our example for any k, by appropriately choosing the values of ℓ and j_i.

For $k = 3$, one can see that the ratio of ALGORITHM MkC is minimized when $\ell = 2$ and $j_2 - j_1 = 1$. Hence, consider the scenario shown in Figure 1. Let

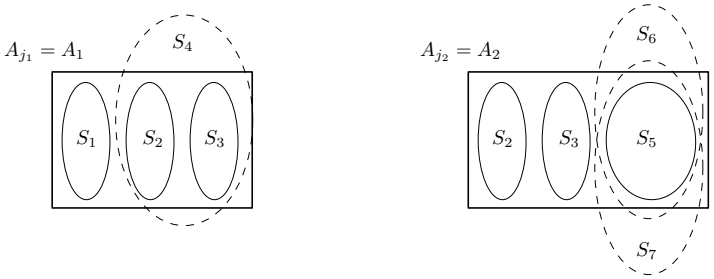

Figure 1: A tight example for ALGORITHM MkC when $k = 3$.

$A_1 = \{S_1, S_2, S_3\}$ be the solution after the first three sets have been released. These sets are disjoint and each one covers c elements. Next, the set S_4 appears, which covers $2c - \epsilon$ new elements plus all elements in S_2 and S_3. The algorithm does not select S_4, since it may choose S_1 as a candidate for swapping; in this case the new solution would cover $m(\{S_2, S_3, S_4\}) = 4c - \epsilon$ elements which is smaller than $m(A_1) + \frac{m(A_1)}{k} = 4c$. Then, the set S_5 is released, that is disjoint with the previous sets and covers $2c$ elements. Thus, the algorithm replaces S_1 by S_5, and the new solution is $A_2 = \{S_2, S_3, S_5\}$. Finally, S_6 and S_7 are released, each covering the elements in S_5 plus $\frac{8c}{3} - \epsilon$ new elements. ALGORITHM MkC does not select none of them, since they do not satisfy the algorithm's criterion.

So, the final solution, A_2, covers $m(S_2) + m(S_3) + m(S_5) = 4c$ elements. The optimal solution consists of sets S_4, S_6 and S_7 and covers $OPT = (4c - \epsilon) +$

$(2c + \frac{8c}{3} - \epsilon) + (\frac{8c}{3} - \epsilon) = \frac{34c}{3} - \epsilon$ elements. Therefore, the ratio achieved by ALGORITHM MkC is $\frac{SOL}{OPT} = \frac{4c}{\frac{34c}{3}-\epsilon} \simeq \frac{12}{34} = 0.353$.

Note that the gap between this ratio and the ratio 0.324 given in Table 1 is due to the fact that the elements of S_1 do not appear to the optimal solution. Indeed, if S_1 was included to the optimal, then $OPT = \frac{37c}{3} - \epsilon$ and $\frac{SOL}{OPT} = \frac{4c}{\frac{37c}{3}-\epsilon} \simeq \frac{12}{37} = 0.324$. This gap decreases as $k \to \infty$.

5 Conclusions

There exist several interesting questions arising from the results presented in this paper. The first of them is to improve the easy $\frac{1}{2}$-competitive ratio for MkVC in general graphs and the (less easy) worst-case $\frac{1}{4}$-competitive ratio in set systems. Another open question is to provide tighter upper bounds for the on-line model handled in regular graphs. We still do not see how one can improve the analysis of ALGORITHM MkC in the case of equal cardinalities, or how to tighten the upper bound of Proposition 3 in Section 2, in order to match (or to get closer to) the competitive ratio of ALGORITHM MkC. Let us note that an algorithm in the spirit of ALGORITHM MkVC-R of Section 3.1 for the case of equal-cardinality sets, only achieves ratio $\frac{1}{\sqrt{k}}$.

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] U. Feige. A threshold of $\ln n$ for approximating set cover. *J. of the ACM*, 45:634–652, 1998.

[3] U. Feige and M. Langberg. Approximation algorithms for maximization problems arising in graph partitioning. *J. of Algorithms*, 41:174–211, 2001.

[4] Q. Han, Y. Ye, H. Zhang, and J. Zhang. On approximation of max-vertex-cover. *European Journal of Operational Research*, 143:342–355, 2002.

[5] D. S. Hochbaum and A. Pathria. Analysis of the greedy approach in problems of maximum k-coverage. *Naval Research Logistics*, 45:615–627, 1998.

[6] V. Th. Paschos. A survey of approximately optimal solutions to some covering and packing problems. *ACM Comp. Surveys*, 29:171–209, 1997.

[7] 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 Appendix

A.1 Proof of Proposition 1

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

Proof. For the MkVC problem, let $k \ll n$ and consider the following scenario. In step i, $1 \leq i \leq k$, the central vertex v_i of a star with $d(v_i) = (n-1)^{i/(k+1)}$ is released. If the algorithm rejects v_i, then the remaining $\sum_{j=1}^{i}(n-1)^{j/(k+1)}$ vertices of the i stars plus $n - i - \sum_{j=1}^{i}(n-1)^{j/(k+1)}$ singleton vertices are released. If the algorithm selects v_i, then vertex v_{i+1}, with $d(v_{i+1}) = (n-1)^{(i+1)/(k+1)}$ is released. If after the k-th vertex the algorithm has selected all the k released vertices then a new vertex v_{k+1} with degree $d(v_{k+1}) = n-1$ is released; finally the remaining vertices of the stars and $n-k-\sum_{j=1}^{k}(n-1)^{j/(k+1)}$ singleton vertices are released.

If after the step i the algorithm has rejected v_i, then only vertices of degree at most one are released. Hence, the algorithm covers $k-(i-1)+\sum_{j=1}^{i-1}(n-1)^{j/(k+1)}$ edges, while the optimum solution covers $k-i+\sum_{j=1}^{i}(n-1)^{j/(k+1)}$ edges. Thus:

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

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

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

that concludes the proof.

For the MkC problem the proof is similar. In this case, in phase i, if all previously released sets are selected by the algorithm then a set of cardinality $m^{i/(k+1)}$ is released. □

A.2 Proof of Proposition 3

Proposition 3. *Any deterministic online algorithm cannot achieve a competitive ratio better than* $\frac{k+2\sqrt{k}+1}{2k+2\sqrt{k}+1} \simeq \frac{1}{2}$ *for the* MkC *problem even in the case where*

all sets have the same cardinality.

Proof. A *r-sunflower* is a set system of regular sets of size Δ with a common intersection of size r; the sets of a sunflower are called *petals*.

Consider the following scenario. The adversary starts by sending $\frac{\Delta(p-1)}{p}$-sunflower petals where $\frac{\Delta}{p} \in \mathbb{N}$, while the algorithm keeps k' of them; it continues so until the first time τ where there are $k - \left\lfloor \frac{k'}{p} \right\rfloor$ rejected sets. Notice that this will be always the case for some $\tau \leq \left\lceil \frac{k(2p-1)}{p} \right\rceil$.

Then the adversary starts sending disjoint sets, each one matching private parts of p petals in the solution, until the maximum number of private parts have been matched.

The solution of the algorithm will cover at most $\frac{(k'+p-1)}{p}\Delta$ elements, while the optimum will cover at least $\left\lfloor \frac{k'}{p} \right\rfloor \Delta$ elements by the matching sets plus $\left\lceil \frac{k-\frac{k'}{p}+p-1}{p} \right\rceil \Delta$ elements by rejected petals. Thus, the ratio is bounded above by $\frac{k'+p-1}{k+(p-1)\frac{k'}{p}+p-1}$ where $0 \leq k' \leq k$, which is less than or equal to the simplified expression $\frac{pk+p(p-1)}{(2p-1)k+p(p-1)}$. This expression is minimized when $p = \sqrt{k}+1$, that is $\frac{(\sqrt{k}+1)k+(\sqrt{k}+1)\sqrt{k}}{(2(\sqrt{k}+1)-1)k+(\sqrt{k}+1)\sqrt{k}} = \frac{k+2\sqrt{k}+1}{2k+2\sqrt{k}+1}$, which for k large enough tends asymptotically to $\frac{1}{2}$. □

A.3 Proof of Proposition 6

Proposition 6. *Any deterministic online algorithm achieves a $\frac{k}{n}$-competitive ratio for the* MkVC *problem on regular graphs.*

Proof. An optimum 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 possibly twice, i.e., at least $\frac{k\Delta}{2}$ edges, that is $SOL \geq \frac{k\Delta}{2}$. We so get $\frac{SOL}{OPT} \geq \frac{k}{n}$. □

A.4 Proof of Proposition 7

Proposition 7. *The* MkVC *problem can be solved within $\left(1 - \frac{k-1}{\Delta^*}\right)$-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 optimum 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. □

A.5 Proof of Proposition 8

Proposition 8. *For the* MkVC *problem in chains,* ALGORITHM MkVC-C *returns the (offline) optimum, if* $k < \lceil \frac{n}{3} \rceil$ *or* $k \geq \lceil \frac{2n}{3} \rceil$*, and achieves a* 0.75*-competitive ratio, if* $\lceil \frac{n}{3} \rceil \leq k < \lceil \frac{2n}{3} \rceil$.

Proof. Since there do not exist three consecutive vertices in the solution obtained by the algorithm, each vertex in B has at most one adjacent vertex in C, and hence $|B| \geq |C|$.

If $k < \lceil \frac{n}{3} \rceil$ then assume, for contradiction, that in the final solution it holds that $C \neq \emptyset$ and let $v \in C$. Then, $SOL = 2|B| + |C| \leq 2(k-1) + 1 = 2k - 1 < 2\lceil \frac{n}{3} \rceil - 1$. Thus, the non-covered edges of the input graph are at least $n - 2\lceil \frac{n}{3} \rceil + 2$. Since the number of connected components in the solution of the algorithm is at most $k - 1 < \lceil \frac{n}{3} \rceil - 1$, there are two adjacent edges not covered by the algorithm; let u the common vertex of these edges. But the algorithm in Lines 3–6 should have removed v and add u in A, a contradiction. Thus, $C = \emptyset$ which means that all the k vertices of the solution cover privately exactly two edges, as in the optimum.

If $k \geq \lceil \frac{2n}{3} \rceil$, then the algorithm returns a solution containing all the vertices of the graph. Indeed, since $|B| \geq |C|$ we have $SOL = 2|B| + |C| \geq 2\lceil \frac{n}{3} \rceil + \lfloor \frac{n}{3} \rfloor = n$. Hence, the optimum solution is obtained by the algorithm.

If $\lceil \frac{n}{3} \rceil \leq k < \lceil \frac{2n}{3} \rceil$, then for the solution created by the algorithm we have $SOL = 2|B| + |C|$, while $OPT \leq 2k = 2|B| + 2|C|$. Since $|B| \geq |C|$, we get:

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

that completes the proof. □

A.6 Proof of Claim 1

Claim 1. $m(E(A_{j_{i-1}}) \setminus E(A_{j_i})) \leq \frac{|A_{j_{i-1}} \setminus A_{j_i}|}{k} m(A_{j_{i-1}})$, $2 \leq i \leq \ell$.

Proof. Let Q_r, $1 \leq r \leq |A_{j_{i-1}} \setminus A_{j_i}|$, be the r-th set that is removed from $A_{j_{i-1}}$ between the events $i-1$ and i, considering only the sets that exist in $A_{j_{i-1}}$. Let, also, $\widetilde{Q}_r$ be the private part of Q_r just before it is removed. We will show that, for any p, $1 \leq p \leq |A_{j_{i-1}} \setminus A_{j_i}|$, it holds that $\sum_{r=1}^{p} q_r \leq \frac{p}{k} m(A_{j_{i-1}})$, and thus:

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

Assume for a contradiction 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. □

A.7 Proof of Claim 2

Claim 2. *For any $\ell \geq 2$, it holds that $\sum_{i=2}^{\ell} \frac{j_i - j_{i-1} + 2}{\left(1+\frac{1}{k}\right)^{j_\ell - j_{i-1}}} \leq \frac{g(\ell)}{\ln\left(1+\frac{1}{k}\right)}$, where $g(\ell) = \frac{\left(1+\frac{1}{k}\right)^2}{e} \cdot e^{g(l-1)}$ and $g(2) = \frac{\left(1+\frac{1}{k}\right)^2}{e}$.*

Proof. Set $d_i = j_i - j_{i-1}$. Consider the function $f_\ell(d) = \sum_{i=2}^{\ell} \frac{d_i+2}{\left(1+\frac{1}{k}\right)^{\sum_{j=i}^{\ell} d_j}}$. We will prove the claim by induction to ℓ.

For $\ell = 2$ we have $f_2(d) = \sum_{i=2}^{2} \frac{d_i+2}{\left(1+\frac{1}{k}\right)^{\sum_{j=i}^{2} d_j}} = \frac{d_2+2}{\left(1+\frac{1}{k}\right)^{d_2}}$, where $\frac{\partial f_2(d)}{\partial d_2} = \frac{1-(d_2+2)\ln\left(1+\frac{1}{k}\right)}{\left(1+\frac{1}{k}\right)^{d_2}}$. The global maximum is attained for $d_2 + 2 = \frac{1}{\ln\left(1+\frac{1}{k}\right)}$. Thus: $f_2(d) \leq \frac{1}{\ln\left(1+\frac{1}{k}\right)} \cdot \frac{1}{\left(1+\frac{1}{k}\right)^{\frac{1}{\ln\left(1+\frac{1}{k}\right)}-2}} = \frac{1}{\ln\left(1+\frac{1}{k}\right)} \cdot \frac{\left(1+\frac{1}{k}\right)^2}{e}$.

Assume that the statement is true for $\ell - 1$.

For ℓ, we have:

$$\begin{aligned} f_\ell(d) &= \sum_{i=2}^{\ell} \frac{d_i+2}{\left(1+\frac{1}{k}\right)^{\sum_{j=i}^{\ell} d_j}} = \frac{d_\ell+2}{\left(1+\frac{1}{k}\right)^{d_\ell}} + \sum_{i=2}^{\ell-1} \frac{d_i+2}{\left(1+\frac{1}{k}\right)^{\sum_{j=i}^{\ell} d_j}} \\ &= \frac{d_\ell+2}{\left(1+\frac{1}{k}\right)^{d_\ell}} + \frac{1}{\left(1+\frac{1}{k}\right)^{d_\ell}} \cdot f_{\ell-1}(d) \end{aligned}$$

where $\frac{\partial f_\ell(d)}{\partial d_\ell} = \frac{1-(d_\ell+2+f_{\ell-1}(d))\ln\left(1+\frac{1}{k}\right)}{\left(1+\frac{1}{k}\right)^{d_\ell}}$. The global maximum is attained for

$d_\ell + 2 + f_{\ell-1}(d) = \frac{1}{\ln\left(1+\frac{1}{k}\right)}$. Thus:

$$\begin{aligned} f_\ell(d) &\leq \frac{1}{\ln\left(1+\frac{1}{k}\right)} \cdot \frac{1}{\left(1+\frac{1}{k}\right)^{\frac{1}{\ln\left(1+\frac{1}{k}\right)}-2-f_{\ell-1}(d)}} \\ &\leq \frac{1}{\ln\left(1+\frac{1}{k}\right)} \cdot \frac{\left(1+\frac{1}{k}\right)^2}{e} \cdot \left(1+\frac{1}{k}\right)^{\frac{g(\ell-1)}{\ln\left(1+\frac{1}{k}\right)}} \\ &= \frac{1}{\ln\left(1+\frac{1}{k}\right)} \cdot \frac{\left(1+\frac{1}{k}\right)^2}{e} \cdot e^{g(\ell-1)} \end{aligned}$$

and the claim follows. □