UNIVERSITE PARIS DAUPHINE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision

CNRS UMR 7024

CAHIER DU LAMSADE

235

Avril 2006

On-line models for set-covering: the power of greediness

Giorgio Ausiello, Aristotelis Giannakos, Vangelis Th. Paschos

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

On-line models for set-covering: the power of greediness*

Giorgio Ausiello[1] Aristotelis Giannakos[2] Vangelis Th. Paschos[2]

[1] Dipartimento di Informatica e Sistemistica, Università degli Studi di Roma "La Sapienza"
ausiello@dis.uniroma1.it

[2] LAMSADE, CNRS UMR 7024 and Université Paris-Dauphine
{giannako,paschos}@lamsade.dauphine.fr

April 10, 2006

Abstract

We study an on-line model for set-covering implying that elements of the ground set of size n arrive one-by-one and with any such element σ_i, arrive also the names of the sets containing it in the final instance. Any new element has to be processed irrevocably before the arrival of the next element. We study limits on the competitiveness of several greedy rules solving several alternatives of this basic model. For any of them we give lower and upper bounds for the competitive ratio achieved. We finally deal with the maximum budget saving problem. Here, an initial budget is allotted that is destined to cover the cost of an algorithm for solving set-covering and the objective is to maximize the savings on the initial budget.

1 Introduction

Let C be a ground set of n elements and $\mathcal{S}$ a family of m subsets of C such that $\cup_{S\in\mathcal{S}}S = C$. The set-covering problem consists of finding a family $\mathcal{S}' \subseteq \mathcal{S}$, of minimum cardinality, such that $\cup_{S\in\mathcal{S}'}S = C$. In what follows, for an element $\sigma_i \in C$, we set $F_i = \{S_i^j \in \mathcal{S} : \sigma_i \in S_i^j\}$ and $f_i = |F_i|$; also, we set $f = \max\{f_i : i = 1, \ldots, n\}$.

The set-covering problem has been extensively studied over the past decades. It has been shown to be **NP**-hard in Karp's seminal paper ([6]) and $O(\log n)$-approximable for both weighted and unweighted cases (see [2], for the former and [5, 7, 10], for the latter; see also [8] for a comprehensive survey on the subject). This approximation ratio is the best achievable, unless $\mathbf{P} = \mathbf{NP}$ ([9]).

In *on-line* computation, one can assume that the instance is not known in advance but it is revealed step-by-step. Upon arrival of new data, one has to decide irrevocably which of these data are to be taken in the solution under construction. The fact that the instance is not known in advance, gives rise to several on-line models specified by the ways in which the final instance is revealed, or by the amount of information that is achieved by the on-line algorithm at each step. In any of these models, one has to devise algorithms, called on-line algorithms, constructing feasible solutions whose values are as close as possible to optimal off-line values, i.e., to values of optimal solutions assuming that the final instance is completely known in advance. The closeness of an on-line solution to an optimal off-line one is measured by the so-called

*A preliminary version of this paper, entitled *Greedy algorithms for on-line set-covering and related problems* has been included in the Proceedings of the 12th Computing: The Australasian Theory Symposium (CATS'06), J. Gudmundson (ed.), Australian Computer Science Communications 28(4), pp. 145–151, 2006

competitive ratio $m(x,y)/\operatorname{opt}(x)$, where x is an instance of the problem dealt, y the solution computed by the on-line algorithm dealt, $m(x,y)$ its value and $\operatorname{opt}(x)$ the value of an optimal off-line solution. This measure for on-line computation has been introduced in [11].

Informally, the basic on-line set-covering model adopted here is the following: elements of a ground set of size n arrive one-by-one and with any such element σ_i (arriving during step i), arrive also something about some of the sets containing σ_i. This "something" can be either the names of sets containing σ_i or more information related to their cardinalities, or their covering potential, etc. At each step, σ_i has to be processed immediately, i.e., it has either to be covered by some of the sets whose names has arrived with, or to be left uncovered risking so infeasibility of the final solution.

We first assume that together with an element the names of the sets containing it in the final instance are revealed. We show that if no further information is given, then the competitive ratio of any algorithm is $\Omega(n)$.

Next, we handle several algorithms dealing with the on-line model just sketched. The first one, called `TAKE-ALL`, takes at each step in the solution all the sets containing the element just revealed, if it is still uncovered. We show that this algorithm has tight competitive ratio $O(f)$, where f is the maximum number of sets in $\mathcal{S}$ that contain a ground element. The second algorithm, called `TAKE-AT-RANDOM`, is a randomized algorithm that at each step picks a set at random among the ones whose names are revealed an take it in the solution, if it is still uncovered. For this algorithm we provide an upper bound of $O(n)$ for its competitiveness, as well as an asymptotically matching lower bound.

We then assume that together with the names of the sets covering a revealed element, an information concerning their covering potential is also communicated. We show that, in this case, the competitive ratio of any algorithm that, for any arriving uncovered element, takes in the cover at least one set containing it is bounded below by $\Omega(\sqrt{n})$. Assuming that the covering potential information given with any element is the name of the larger set containing it (in the final instance) we show that the competitive ratio of the greedy algorithm, called `TAKE-LARGEST`, consisting of taking this set in the solution is $O(n)$ and that it is tight.

We next address the following question: "what an on-line algorithm must know about sets in order to guarantee an upper competitive bound of $O(\sqrt{n})$?" We show that such a bound can be attained at least by an algorithm, called `TAKE-LARGEST-ON-FUTURE-ITEMS`, which at any step i takes some set containing σ_i covering the most of the ground set-elements that have not been yet covered (clearly, any uncovered element is yet unrevealed). This assumption can be seen as the on-line counterpart of the natural greedy (off-line) set-covering algorithm, called `GREEDY` in what follows. We recall that this algorithm takes in the solution a set covering the most of the still uncovered elements.

Finally, we address a budget variant of set-covering. We assume that two algorithms collaborate to solve the problem. The application-cost of the former is just the cardinality of the solution it finally computes, while, for the latter, its application cost is the cardinality of its solutions augmented by an overhead due, for example, to the fact that it is allowed to wait before making its decisions. We can see the application cost of the former algorithm as a kind of budget allotted to them that is not allowed to be overcome. The objective is to perform the maximum possible saving upon the initial budget. We show that there exists a natural algorithm-cost model such that `GREEDY` is asymptotically optimal for maximum budget saving when the budget allotted is the application cost of `TAKE-LARGEST-ON-FUTURE-ITEMS`.

The remainder of this paper is organized as follows. In Section 2, we describe the main result presented in the paper more in detail and we compare our approach to other approaches introduced in the literature. Section 3 is devoted to a general competitiveness lower bound for simple on line algorithms. In Section 4, we present two simple algorithms, `TAKE-ALL` and

TAKE-AT-RANDOM and we discuss their performances. In Sections 5 and 6 the power of look ahead is discussed, first by showing that any algorithm that knows the covering potential of the sets which contain a revealed (not covered) item cannot attain a competitive ratio better than $\Omega(\sqrt{n/2})$ and then by exhibiting an algorithm whose competitive ratio matches such bound. Section 7 is devoted to the budget version of the problem. Finally, in Section 8, conclusive issues are discussed.

2 Description of the main results and related work

In [1], the following on-line set-covering model has been studied. We suppose that we are given an instance $(\mathcal{S}, C)$ that it is known in advance, but it is possible that only a part of it, i.e., a sub-instance $(\mathcal{S}_p, C_p)$ of $(\mathcal{S}, C)$ will finally arrive; this sub-instance is not known in advance. A picturesque way to apprehend the model is to think of the elements of C as lights initially switched off. Elements switch on (get activated) one-by-one. Any time an element c gets activated, the algorithm has to decide which among the sets of $\mathcal{S}$ containing c has to be included in the solution under construction (since we assume that $(\mathcal{S}, C)$ is known in advance, all these sets are also known). In other words, the algorithm has to keep an online cover for the activated elements. The algorithm proposed for this model achieves competitive ratio $O(\log n \log m)$ (even if less than n elements of C will be finally switched on and less than m subsets of $\mathcal{S}$ include these elements).

The on-line model dealt here is inspired (yet quite different since it does not allow that, once arrived, an element can disappear) from the one in [1]. Moreover, and most important, the instance $(\mathcal{S}, C)$ is not known in advance. In our model, we are given an arrival sequence $\Sigma = (\sigma_1, \ldots, \sigma_n)$ of the elements of C (i.e., elements of C are switched on following the order $\sigma_1, \ldots, \sigma_n$), the objective is to find, for any $i \in \{1, \ldots, n\}$, a family $\mathcal{S}'_i \subseteq \mathcal{S}$ such that $\{\sigma_1, \ldots, \sigma_i\} \subseteq \cup_{S \in \mathcal{S}'_i} S$. Recall that for any σ_i, $i = 1, \ldots, n$, we denote by $F_i = \{S_i^j : S_i^j \in \mathcal{S}, \sigma_i \in S_i^j\}$ the sets of $\mathcal{S}$ containing σ_i, by f_i the cardinality of F_i, usually called frequency, and we set $f = \max_{\sigma_i \in \Sigma}\{f_i\}$. We denote by $\bar{S}_i^j$ the subset of the elements of $S_i^j \in F_i$ still remaining uncovered and by δ_i^j the cardinality of $\bar{S}_i^j$.

When σ_i switches on, something about sets in F_i, $j = 1, \ldots, f_i$ is revealed. We first assume that this "something" is the names of the sets covering σ_i, i.e., the names in F_i. We show that, if no further information is supplied, then no on-line algorithm can achieve competitive ratio better than $O(n)$.

We then study the competitiveness of two algorithms dealing with this model, namely TAKE-ALL and TAKE-AT-RANDOM.

The former, any time some σ_i arrives, if it is still uncovered, i.e., if none of the sets taken in the solution up-to-date does belong to F_i, then takes all the sets in F_i in the solution under construction. We show that the competitive ratio of TAKE-ALL is bounded above by f, the maximum frequency of the final instance, and that this ratio is tight and can be even exponential with n.

In the case of TAKE-AT-RANDOM, any time some σ_i arrives, if it is still uncovered, the algorithm picks at random a set in F_i and puts it in the solution. We show that its competitive ratio is bounded above by $O(n)$. We also prove that its expected competitive ratio cannot be better than $O(n^{1-(1/\epsilon)})$, for any $\epsilon > 0$. We so have a lower bound that asymptotically matches its competitive ratio.

We next assume that together with F_i, some more information is revealed about the covering potential of some of the sets in F_i. We show that under this assumption, no on-line algorithm that, upon the arrival of an uncovered element, processes it by adding to the cover at least one set containing it can guarantee competitive ratio better than $\sqrt{n/2}$.

Furthermore, we show that if together with F_i the name of some set $\hat{S}_i \in \text{argmax}\{|S_i^j|, j = 1, \ldots, f_i\}$ is revealed, then the greedy rule (called TAKE-LARGEST, in the sequel) that if σ_i is still uncovered, takes $\hat{S}_i$ in the solution, achieves competitive ratio $O(n)$.

The two results mentioned just above show that there is an important gap between the general lower bound of $\sqrt{n/2}$ and the ratio achieved by TAKE-LARGEST. Hence, a question rises naturally: "what an on-line algorithm must know in order to achieve a ratio closer to this bound, say $O(\sqrt{n})$, respecting always the given on-line model? We show that there exists at least an implementation doing this, if it owns some look-ahead informations.

We show that if together with σ_i and F_i, the name of some set $\tilde{S}_i \in \text{argmax}\{\delta_i^j, j = 1, \ldots, f_i\}$, i.e., the name of a set in F_i covering the most of the still unrevealed elements is revealed, then the greedy rule (TAKE-LARGEST-ON-FUTURE-ITEMS) that adds $\tilde{S}_i$ in the cover, if σ_i remains still uncovered, achieves tight competitive ratio $O(\sqrt{n})$. Let us note that TAKE-LARGEST-ON-FUTURE-ITEMS is a kind of on-line analogue of GREEDY. Hence, analysis of its competitiveness is interesting by its own.

Note also that a basic and very interesting feature of the introduced models is their small memory requirements, since the only information needed is the binary encoding of the names of the sets. This is a major difference between our approach and the one of [1]. There, anytime an element gets activated, the algorithm needs to compute the value of a potential function using an updated weight parameter for each element and then chooses covering sets in a suitable way so that this potential be non-increasing; the greedy online algorithm in our model needs only a constant number of memory places, making it more appropriate for handling very large instances with very few hardware ressources. For instance, the rules used here use at most $O(m)$ space.

We are so faced to the power (rather the weakness) of greediness. Let us recall that the on-line models that we consider assume no knowledge of the topology of the final instance $(\mathcal{S}, C)$ and immediate processing of any arriving element σ_i. Obviously, the rules that we consider are the only to feasibly solve the problem in such situations. Furthermore they are very efficient in time and in memory requirements, hence well adapted to face really on-line practical situations. On the other hand, since no a priori knowledge of instance's topology is admitted, no algorithm can do complex preliminary calculations (as the potential calculations in [1]) in order to judiciously choose the set to be included in the cover under construction.

In many real-life problems, it is meaningful to relax the main specification of the online setting, that is, to keep a solution for any partially revealed instance, in order to achieve a better solution quality. In this sense, a possible relaxation is to consider that several algorithms collaborate in order to return the final solution. The costs of using these algorithms can be different the ones from the others, depending upon the sizes of the solutions computed, the time overheads they take in order to produce them, etc. Moreover, we can assume that an initial common budget is allotted to all these algorithms and that this budget is large enough to allow use of at least one of the algorithms at hand to solve the problem without exceeding it. A nice objective could be in this case, to use these algorithms in such a way that a maximum of the initial budget is saved.

For the case of set-covering, the following budget-model, giving rise to what we call *maximum budget saving problem* is considered in Section 7. We assume that two algorithms collaborate to solve it: say TAKE-LARGEST-ON-FUTURE-ITEMS and the greedy (off-line) algorithm. The application-cost of the former is just the cardinality of the solution it finally computes, while, for the latter, its application cost is the cardinality of its solutions augmented by an overhead due, for example, to the fact that it is allowed to wait before making its decisions. For an instance x of set-covering, the initial budget considered is $B(x) = \sqrt{n}\,\text{opt}(x)$ (this is in order that at least TAKE-LARGEST-ON-FUTURE-ITEMS is able to compute a solution of x without exceeding the budget for any x). Denote by $c(x, y)$ the cost of using A in order to compute a cover y for x.

The objective is to maximize the quantity $B(x) - c(x, y)$ and, obviously, the maximum possible economy on x is $B(x) - \mathrm{opt}(x)$. We show in Section 7 that there exists a natural algorithm-cost model such that `GREEDY` is asymptotically optimal for maximum budget saving.

Before closing this section, let us quote another approach that could be considered to be at midway between semi-on-line and reoptimization approaches, developed in [3]. There, the problem tackled is the maintenance of approximation ratio achieved by an algorithm while the set-covering instance undergoes limited changes. More precisely, assume that a set-covering instance $(\mathcal{S}, C)$ and a solution $\mathcal{S}'$ for it are given. How many insertions of some of the ground elements in subsets that did not previously contain these elements produce an instance for which the solution $\mathcal{S}'$ of the initial instance guarantees the same approximation ratio in both of them? In [3] it is shown that if solution $\mathcal{S}'$ has been produced by application of the natural greedy algorithm achieving approximation ratio $O(\log n)$ ([2]), then after $O(\log n)$ such insertions initial solution $\mathcal{S}'$ still guarantees the same approximation ratio. In the same spirit lies also the similar set-covering model in [12].

3 The price of ignorance

In this section we consider the first version of the on-line model sketched in Section 2. Assume an arrival sequence $\Sigma = (\sigma_1, \ldots, \sigma_n)$ of the elements of C, and the objective is to find, for any $i \in \{1, \ldots, n\}$, a family $\mathcal{S}'_i \subseteq \mathcal{S}$ such that $\{\sigma_1, \ldots, \sigma_i\} \subseteq \cup_{S \in \mathcal{S}'_i} S$. Once an element σ_i, $i = 1, \ldots,$ switches on, only the encodings of the members of F_i are also revealed.

For this case we first prove that when no additional information is given, any rule that has to cover a new element without any look-ahead behaves rather badly. In this sense strategies that choose elements either randomly or based upon observations of the past (for instance take the subset that has appeared the most frequently or the most rarely until now) are highly inefficient.

Proposition 1. *If no information is given about the sets revealed with an arriving not covered ground element, then the competitive ratio of implementation of any greedy principle is* $\Omega(n)$

Proof. The adversary reveals a first uncovered element along with the names $S_1, \ldots, S_N$ of N sets covering it in the final instance. He then keeps revealing uncovered elements along with all sets from $S_1, \ldots, S_N$ not already taken into the cover, until the algorithm has taken all N sets into the cover.

Suppose w.l.o.g. that the algorithm has taken $S_1, \ldots, S_{l_1}, S_{l_1+1}, \ldots, S_{l_2}, S_{l_2+1}, \ldots, S_{l_3}$, and at the kth and final step, $S_{l_{k-1}+1}, \ldots, S_{l_k} = S_N$.

The adversary can give the following interpretation to the instance: $\mathcal{S} = \{S_1, \ldots, S_N\}$. There are $n = \log N + k$ ground elements, namely $\{\sigma_1, \ldots, \sigma_{\log N}, 1, \ldots, k\}$ (notice also that $k \leqslant N$). The set S_i taken at step j contains the elements σ_p, for all places p where the binary expression of i has an 1, plus elements in $\{1, \ldots, j\}$. The arrival sequence is $1, \ldots, k$.

In this interpretation, the set S_N, taken at the last step, is the ground set itself; thus, in such a setting, the competitive ratio of the algorithm would be N, i.e., $\Omega(n)$ ∎

It can be immediately seen that under the given model, any deterministic algorithm that takes a specific set containing a new (uncovered) element σ_i (for example, the set of F_i that comes first in lexicographic order) achieves competitive ratio $O(n)$. Indeed, it chooses at most n sets for an optimum greater than, or equal to 1.

4 Competitiveness of `TAKE-ALL` and `TAKE-AT-RANDOM`

4.1 Algorithm `TAKE-ALL`

Recall that algorithm `TAKE-ALL`, whenever a newly revealed element σ_i is not already covered by sets already taken in the solution due to former arrivals, it takes in the solution the whole

family F_i the names of the members of which have been revealed together with σ_i.

Proposition 2. *The competitive ratio of* `TAKE-ALL` *is bounded above by* f*. This ratio is tight.*

Proof. Denote by $\sigma_1, \ldots, \sigma_k$ the critical elements of Σ, i.e., the elements having entailed the introduction of $S_1, \ldots, S_k$ in $\mathcal{S}'$. Denote also by $\mathcal{S}^*$ an optimum off-line solution. Obviously, for any of the critical elements, a distinct set is needed to cover it, in any feasible cover for C; hence:

$$|\mathcal{S}^*| \geqslant k \tag{1}$$

On the other hand, since, for $i = 1, \ldots, k$, $f_i \leqslant f$:

$$\left|\mathcal{S}'\right| \leqslant kf \tag{2}$$

Combining (1) and (2), the competitive ratio is immediately derived.

In order to show tightness, consider an instance with ground set $C = \{1, \ldots, n\}$ and the family of all 2^{n-1} sets formed by 1 union any other set in $2^{C\setminus\{1\}}$. With an arrival sequence starting with 1, the competitive ratio of `TAKE-ALL` would be $2^{n-1} = f$. ∎

Note that `TAKE-ALL` is similar to the approximation algorithm for minimum set-covering presented in [4] and, furthermore, it guarantees the same approximation ratio.

Note also that, from Proposition 2, `TAKE-ALL` gives a much worse competitiveness than n.

4.2 Algorithm `TAKE-AT-RANDOM`

Recall that `TAKE-AT-RANDOM` chooses at random one set in F_i per revealed uncovered element σ_i and puts it in the solution. It can be immediately seen that, with the same arguments as the ones at the end of Section 3, `TAKE-AT-RANDOM` achieves competitive ratio $O(n)$. In the following proposition, we show that even its expected competitive ratio, if it chooses one of the sets covering σ_i with uniform probability, cannot be much better than $O(n)$.

Theorem 1. *For any* $\epsilon > 0$*, there exists an instance of the on-line set-covering with* n *ground elements such that the expected competitive ratio of* `TAKE-AT-RANDOM` *is* $\Omega(n^{1-(1/\epsilon)})$.

Proof. For any $\epsilon > 0$, fix an integer $k > 1/\epsilon$ and let $N > 2^k$. Consider the instance with ground set $C = \{1, \ldots, n = N^k\}$. Family $\mathcal{S}$ contains the following sets:

- a partition of *class* sets $S(i) = \{j \in C : (j-1) \div N = (i-1)\}$; clearly, $|S(i)| = N$ and there exist N^{k-1} class sets;
- for any $j \in S(i)$ for some i, there exist 2^{N-1} *internal* sets, each one containing j plus the elements of one of all possible subsets of $S(i)$ (including the empty set);
- the ground set C itself.

Consider now an arbitrary arrival sequence, and compute the expected value of the cover, which will be equal to the expected competitive ratio of `TAKE-AT-RANDOM` (equality holds, since the optimum for this instance is C).

Every element belongs to one class set and to 2^{N-1} internal sets. We note by $\mathbf{E}(q)$ the expected value of `TAKE-AT-RANDOM` on the instance of q elements defined as before. Then:

$$\begin{aligned}
\mathbf{E}\left(N^k\right) &= \frac{1}{2^{N-1}+1}\left(1 + \sum_{l=0}^{N-1}\binom{N-1}{l}\left(1 + \mathbf{E}\left(N^k - l - 1\right)\right)\right) \\
&\geq \frac{1}{2^{N-1}+1}\left(1 + 2^{N-1} + 2^{N-1}\mathbf{E}\left(N^k - N\right)\right) \;\approx\; 1 + \mathbf{E}\left(N^k - N\right)
\end{aligned}$$

The recursive relation yields then directly $\mathbf{E}(N^k) \geq N^{k-1}$, i.e., $\mathbf{E}(n) = \Omega(n^{1-(1/\epsilon)})$. ∎

Example 1. We now give an example of construction of Theorem 1. Consider $N = 3$ and $k = 3$ (these values of N and k are not conformal with their definition but, in a first time, we use them for simplicity). Then $C = \{1, 2, \ldots, 27\}$ and we have:

- class sets: $\{1, 2, 3\}$, $\{4, 5, 6\}$, $\{7, 8, 9\}$, $\{10, 11, 12\}$, $\{13, 14, 15\}$, $\{16, 17, 18\}$, $\{19, 20, 21\}$, $\{22, 23, 24\}$ and $\{25, 26, 27\}$;
- for any class set $\{a, b, c\}$, there exist the internal sets: $\{a\}$, $\{b\}$, $\{c\}$, $\{a, b\}$, $\{a, c\}$ and $\{b, c\}$;
- finally $C = \{1, 2, \ldots, 27\} \in \mathcal{S}$.

Let us assume that $\sigma_1 = 17$. With it will be revealed the following $2^2 + 1 = 5$ sets: $\{17\}$, $\{17, 18\}$, $\{17, 19\}$, $\{17, 18, 19\}$ and $\{1, 2, \ldots, 27\}$. The average cover for the whole instance will be of size 9, independently on the arrival sequence.

Indeed, for $k = 3$, $N \geqslant 8$. Taking $N = 8$, $n = 512$. In this case, the class sets would be the partition of $\{1, 2, \ldots, 512\}$ into 64 subsequent 8-tuples. For any class set there would be 254 internal sets. With any element of the arrival sequence they would arrive 1 class set, plus 63 internal sets plus set $\{1, 2, \ldots, 512\}$, i.e., $8^{3-1} + 1 = 65$ sets. The average cover size would be in this case 64. ∎

5 The nasty flaw of greediness

In this section, we consider a slightly enriched model for on-line set-covering by assuming that together with any element σ_i of the arrival sequence, arrive not only the names of the sets containing them to the final instance but also some information about the cover potential of some of these sets.

In what follows, we show that, even for this revised model, no on-line algorithm can achieve competitive ratio better than $\sqrt{n/2}$, even if it is allowed to choose at any step more than one set to be introduced in the solution.

Theorem 2. *Consider an on-line model for set-covering where together with any element σ_i of the arrival sequence, arrive not only the names of the sets containing them to the final instance but also some information about the cover potential of some of these sets. Then, on-line algorithm for this model such that, at any step, it takes in the cover at least one set containing some not yet covered arriving element can achieve competitive ratio less than $\sqrt{n/2}$, even if one assumes that with any σ_i, $\tilde{S}_i$ is also revealed.*

Proof. Consider the following set-covering instance built, for any integer N, upon a ground set $S = \{x_{ij} : 1 \leqslant j \leqslant i \leqslant N\}$; obviously, $|C| = n = N(N+1)/2$. A *path-set of order i* is defined as a set containing $N - i + 1$ elements $\{x_{ij_i}, \ldots, x_{Nj_N}\}$. The set-system $\mathcal{S}$ of the instance contains all possible path-sets of each order i, $1 \leqslant i \leqslant N$. Clearly, there exist $N!/0!$ path-sets of order 1, $N!/1!$ path-sets of order 2, and so on and, finally, $N!/(N-1)!$ path-sets of order N, i.e., in all $N!(1 + \ldots + 1/(N-1)!) \approx eN!$ path-sets. Finally, the set-system $\mathcal{S}$ is completed with an additional set Y containing all elements of C but those of some path-set of order 1, that will be specified later (hence, $|Y| = n - N$).

As long as there exist uncovered elements, the adversary may choose to have an uncovered element x_{ij} of the lowest possible i arriving, which will be contained only in all path-sets of order less than or equal to i. Notice that as long as algorithm `A` has $r < N$ sets inserted in the cover, there will be at least one element $x_{(r+1)j}$ for some j, $1 \leqslant j \leqslant k+1$, not yet covered. Suppose that after the arrival of σ_t, the size of the cover computed by `A` gets equal to, or greater than, N. Clearly, $1 \leqslant t \leqslant N$. At time $t+1$, a new element arrives, contained in some path-sets and in Y,

which can be now specified as consisting of all elements in C except of the elements of some path-set S^* of order 1 containing $\sigma_1, \ldots, \sigma_t$; the rest of the arrival sequence is indifferent.

Clearly the optimum cover in this case would have been path-set S^* together with set Y; hence, $k_{\mathtt{A}}/k^* \geqslant N/2$, with N tending to $\sqrt{2n}$ as n increases.

It is easy to see that the above construction can be directly generalized so that the same result holds also in the case that the on-line algorithm is allowed to take more than one sets at a time in the cover: if $\sigma_1 = x_{11}$, then as long as the size of the online cover is less than N, there exists always some $i_{\ell-1} < i_\ell \leqslant N$ and some j_{i_ℓ} for which $x_{i_\ell j_{i_\ell}}$ is yet uncovered. Hence, if σ_ℓ is this element, then the algorithm will have to put some sets in the cover. Finally, the algorithm will have put N sets in the cover, while the optimum will always be of size 2. ∎

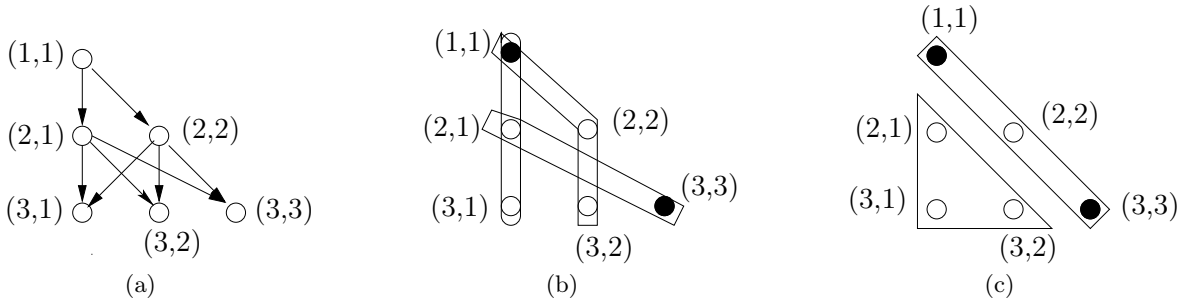

Figure 1: The counter-example of Theorem 2 for $N = 5$.

Example 2. In order to illustrate the construction in Theorem 2, consider the instance of Figure 1, with $N = 5$ (the elements of C are depicted as cycles labelled by (i, j) for $1 \leqslant j \leqslant i \leqslant 3$). The S_i sets can be thought of as paths terminating to a sink on the directed graph of Figure 1(a). Assume that $(1, 1)$ arrives, and algorithm A chooses sets $\{(1, 1), (2, 1), (3, 1)\}, \{(1, 1), (2, 2), (3, 2)\}$ for covering it; the uncovered element $(3, 3)$ arrives next, so A has to cover it by, say, the set $\{(2, 1), (3, 3)\}$ (Figure 1(b)). The optimal cover might consist of set $\{(1, 1), (2, 2), (3, 3)\}$ together with a big set consisting of the rest of the elements, that could not have been revealed to A upon arrival of $(1, 1)$, or of $(3, 3)$ (Figure 1(c)). ∎

Let us now see what is the situation if we assume that together with F_i is also revealed the name of some set $\hat{S}_i \in \text{argmax}\{|S_i^j|, j = 1, \ldots, f_i\}$ and we implement TAKE-LARGEST that consists of taking $\hat{S}_i$ in the solution.

Observe first that the discussion about the competitiveness of deterministic algorithms that take a specific set containing a new (uncovered) element, holds also for TAKE-LARGEST. Hence its competitive ratio is bounded above by n. We now show that this ratio is asymptotically tight.

Consider the following set-covering instance: a ground set $C = \{1, \ldots, 2N\}$, a family of sets $\mathcal{S} = \{S_0, \ldots, S_N\}$ with $S_i = \{i, \ldots, N + i\}$. Assume an arrival sequence starting with $N, N + 1, \ldots, 2N$. Then, TAKE-LARGEST, might take into the cover sets $S_1, \ldots, S_N$, while the optimum cover would be consisting of only S_0, S_N, thus yielding a competitive ratio of $N/2$.

6 The power of look-ahead

The discussion in Section 5 shows a large gap between the lower bound provided for any algorithm for this model in Theorem 2 and the (tight) competitiveness of TAKE-LARGEST, that is one of the most natural rules one could think about. So, an immediate question can be addressed: "what an on-line algorithm has to know in order to achieve a competitive ratio $O(\sqrt{n})$? is it possible to devise such an algorithm?" The following result tackles these questions.

Theorem 3. *Consider an instance $(\mathcal{S}, C)$ of minimum set-covering with $|C| = n$. Assume an arrival sequence $\Sigma = (\sigma_1, \ldots, \sigma_n)$ and suppose that once an element σ_i, $i = 1, \ldots,$ switches on, the encoding for $\tilde{S}_i \in \text{argmax}\{\delta_i^j, j = 1, \ldots, f_i\}$ is also revealed together with . Consider implementation of* `TAKE-LARGEST-ON-FUTURE-ITEMS` *that, if σ_i is not already covered in one of the previous steps, it takes $\tilde{S}_i$ in the solution under construction. Denote by $\mathcal{S}^* = \{S_1^*, \ldots, S_{k^*}^*\}$ an optimal off-line solution on $(\mathcal{S}, C)$. Then, the competitive ratio of* `TAKE-LARGEST-ON-FUTURE-ITEMS` *is bounded above by $\min\{\sqrt{2n/k^*}, \sqrt{n}\}$. Furthermore, there exist large enough instances for which this ratio is at least $\sqrt{n/2}$.*

Proof. Fix an arrival sequence $\Sigma = (\sigma_1, \ldots, \sigma_n)$ and denote by $c_1, \ldots, c_k$, its *critical elements*, i.e., the elements having entailed introduction of a set in $\mathcal{S}'$. In other words, critical elements of Σ are all elements c_i such that c_i was not yet covered by the cover under construction upon its arrival. Assume also that the final cover $\mathcal{S}'$ consists of k sets, namely, $S_1, \ldots, S_k$, where S_1 has been introduced in $\mathcal{S}'$ due to c_1, S_2 due to c_2, and so on.

Let $\delta(S_i)$ be the increase of the number of covered elements just after having taken S_i in the greedy cover (recall that if S_i has been added in $\mathcal{S}'$ for critical element $c_i = \sigma_j$, $\delta(S_i) = \max\{\delta_j^1, \ldots, \delta_j^{f_j}\}$). We have:

$$\delta(S_1) = |S_1| \tag{3}$$

$$\delta(S_i) = \left|\bigcup_{\ell=1}^{i} S_\ell\right| - \left|\bigcup_{\ell=1}^{i-1} S_\ell\right|, \quad 2 \leqslant i \leqslant k \tag{4}$$

Fix now an optimal off-line solution $\mathcal{S}^*$ of cardinality k^*. Any of the critical elements $c_1, \ldots, c_k$ can be associated to the set of smallest index in $\mathcal{S}^*$ containing it. For any $S_i^* \in \mathcal{S}$, we denote by $\hat{S}_i^*$, the set of the critical elements associated with S_i^* (obviously, $\hat{S}_i^* \subseteq S_i^*$). The *critical content* $h(S_i^*)$ of any $S_i^* \in \mathcal{S}^*$ is defined as the number of critical elements associated to it as described before, i.e., $h(S_i^*) = |\hat{S}_i^*|$.

Let $S_1^*, \ldots, S_r^*$ be the sets in $\mathcal{S}^*$ of positive critical contents $h(S_1^*), \ldots, h(S_r^*)$, respectively. Clearly,

$$\sum_{i=1}^{r} h(S_i^*) = k \tag{5}$$

$$r \leqslant k^* \tag{6}$$

For any S_i^*, let $c_i^1, \ldots, c_i^{h(S_i^*)}$ be the elements of its critical content ordered according to their position in the arrival sequence Σ; in other words, following our assumptions, $\hat{S}_i^* = \{c_i^1, \ldots, c_i^{h(S_i^*)}\}$ (recall that $\hat{S}_i^* \subseteq S_i^*$).

Suppose, without loss of generality, that, for $\ell = 1, \ldots, h(S_i^*)$, the set $S_{j_\ell} \in \mathcal{S}$ has been introduced in $\mathcal{S}'$ when the critical element c_i^ℓ has been activated. At the moment of the arrival of c_i^1, the set S_i^* is also a candidate set for $\mathcal{S}'$. The fact that S_{j_1} has been chosen instead of S_i^* means that $\delta(S_{j_1}) \geqslant \delta(S_i^*)$; hence, since as noticed just above, $\hat{S}_i^* \subseteq S_i^*$, the following holds immediately: $\delta(S_{j_1}) \geqslant \delta(S_i^*) \geqslant |\hat{S}_i^*| = h(S_i^*)$.

When c_i^2 gets activated, the set S_i^* has lost some of its elements that have been covered by some sets already chosen by the algorithm. In any case, it has lost c_i^1 (covered by S_{j_1}). So, following the arguments developed just above for S_{j_1}, $\delta(S_{j_2}) \geqslant h(S_i^*) - 1$, and so on (quantities $\delta(\cdot)$ are defined either by (3), or by (4)). So, dealing with c_i^ℓ, the following holds:

$$h(S_i^*) - \ell + 1 \leqslant \delta(S_{j_\ell}) \tag{7}$$

For example, consider the illustration of Figure 2. Let S^* be a set of the fixed optimal cover $\mathcal{S}^*$ and denote by $\hat{S}$ the set of its critical elements, c^1, c^2 and c^3 (ranged in the order they have been activated). Let S be the set chosen by `TAKE-LARGEST-ON-FUTURE-ITEMS` to cover c^2. The shadowed parts of S^*, $\hat{S}$ and S correspond to elements already covered by `TAKE-LARGEST-ON-FUTURE-ITEMS` at the moment of arrival of c^2. At this moment, S must contain at least as many uncovered elements as S^* does and a fortiori at least one uncovered element for any yet uncovered critical element of S^* (two uncovered elements for S appear below the dashed line for c^3 and c^4).

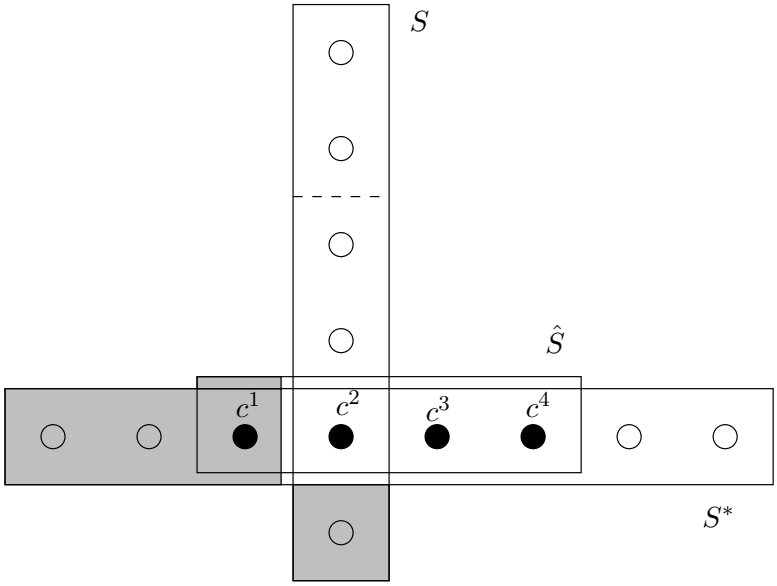

Figure 2: An example for (7).

Summing up inequalities (7), for $\ell = 1, \ldots, h(S_i^*)$, and setting $\sum_{\ell=1}^{h(S_i^*)} \delta(S_{j_\ell}) = n_i$, we finally get for S_i:

$$\frac{h\left(S_i^*\right)\left(h\left(S_i^*\right)+1\right)}{2} \leqslant \sum_{\ell=1}^{h\left(S_i^*\right)} \delta\left(S_{j_\ell}\right) = n_i \Longrightarrow h\left(S_i^*\right) \leqslant \sqrt{2n_i} \tag{8}$$

Set, for $1 \leqslant i \leqslant r$, $n_i = \alpha_i n$, for some $\alpha_i \in [0,1]$. Then, $\sum_{i=1}^{r} \alpha_i = 1$ and

$$\sum_{i=1}^{r} \sqrt{\alpha_i} \leqslant \sqrt{r} \tag{9}$$

Using (5), (6), (8) and (9), we get:

$$k = \sum_{i=1}^{r} h\left(S_i^*\right) \leqslant \sqrt{2n} \sum_{i=1}^{r} \sqrt{\alpha_i} \leqslant \sqrt{r}\sqrt{2n} \leqslant \sqrt{k^*}\sqrt{2n} \tag{10}$$

Dividing the first and the last members of (10) by k^*, we get:

$$\frac{k}{k^*} \leqslant \sqrt{\frac{2n}{k^*}} \tag{11}$$

On the other hand, remark that, if $k^* = 1$, i.e., if there exists $S^* \in \mathcal{S}$ such that $\mathcal{S}^* = \{S^*\}$, then `TAKE-LARGEST-ON-FUTURE-ITEMS` would have chosen it from the beginning of its running

in order to cover σ_1; next, no additional set would have entered the $\mathcal{S}'$. Consequently, we can assume that $k^* \geqslant 2$ and, using (11),

$$\frac{k}{k^*} \leqslant \sqrt{n} \tag{12}$$

Combination of (11) and (12) concludes the competitive ratio claimed.

Fix an integer N and consider the following instance $(\mathcal{S}, C)$ of minimum set-covering:

$$\begin{aligned}
C &= \left\{1, \ldots, \frac{N(N+1)}{2}\right\} \\
S_1 &= \{1, \ldots, N\} \\
S_2 &= \{N+1, \ldots, 2N-1\} \\
&\vdots \\
S_N &= \left\{\frac{N(N+1)}{2}\right\} \\
S_{N+1} &= \left\{(i-1)N - \frac{i(i-3)}{2} : i = 1, \ldots, N\right\} \\
S_{N+2} &= C \setminus S_{N+1}
\end{aligned}$$

Consider the arrival sequence $(1, \ldots, N(N+1)/2)$. `TAKE-LARGEST-ON-FUTURE-ITEMS` might compute the cover $\mathcal{S}' = \{S_i, 1 \leqslant i \leqslant N\}$, while the optimal one is $\mathcal{S}^* = \{S_{N+1}, S_{N+2}\}$. Hence, the competitive ratio in this case would be $N/2$, with $N = (-1 + \sqrt{1+8n})/2$ which is asymptotically equal to $\sqrt{n/2}$ as claimed.

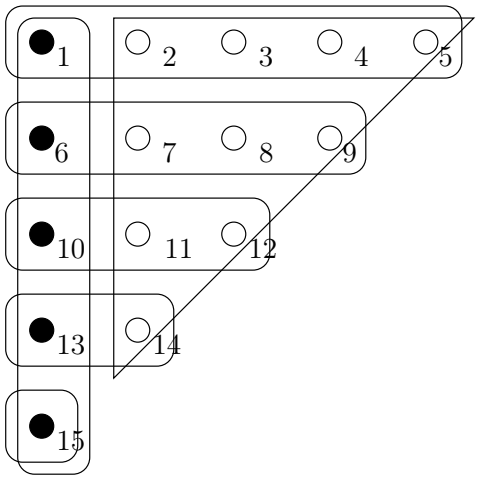

Figure 3: The ratio $\sqrt{n/2}$ for `TAKE-LARGEST-ON-FUTURE-ITEMS` is asymptotically attained.

For example, consider Figure 3. If Σ starts with 1, 6, 10, 13, 15, `TAKE-LARGEST-ON-FUTURE-ITEMS` may have chosen sets $\{1,2,3,4,5\}, \{6,7,8,9\}, \{10,11,12\}, \{13,14\}, \{15\}$, respectively, while the optimal cover would consist of sets $\{1,6,10,13,15\}$ and $\{2,3,4,5,7,8,9,11,12,14\}$. The proof of the theorem is now complete. ∎

Revisit (11), set $\Delta = \max_{S_i \in \mathcal{S}}\{|S_i|\}$ and take into account the obvious inequality: $k^* \geqslant n/\Delta$. Then, the following result is immediately derived from Theorem 3.

Corollary 1. *The competitive ratio of `TAKE-LARGEST-ON-FUTURE-ITEMS` is bounded above by* $\sqrt{2\Delta}$.

It can be easily seen from the proof of Theorem 3 that it also works even if one assumes that the arrival sequence does not contain all the elements of C but only a part of them. In this case, the competitive ratio achieved is $\sqrt{2n/k^*} \leqslant \sqrt{2n}$, since the hypothesis on the size of k^* (discussed at the end of the proof of Theorem 3) is no more valid. This is important for TAKE-LARGEST-ON-FUTURE-ITEMS because it can be seen as an algorithm working also for the on-line model in [1] with provably upper competitive bound.

Corollary 2. *The competitive ratio of TAKE-LARGEST-ON-FUTURE-ITEMS when assumed that only a subset of C will finally be revealed is bounded above by $\sqrt{2n}$.*

The counter-example instance given in the proof of Theorem 3 can be slightly modified to fit the case where, at each step, whenever a yet uncovered element arrives, the algorithm is allowed to take in the cover a constant number of sets containing it and such that the number of elements yet switched off that belong to these sets is maximized.

Let $\bar{F}_i^{[\rho]} = \text{argmax}\{|\bar{S}_i^{j_1} \cup \bar{S}_i^{j_2} \cup \ldots \cup \bar{S}_i^{j_\rho}| : S_i^{j_1}, S_i^{j_2}, \ldots, S_i^{j_\rho} \in F_i\}$, assume that together with F_i also arrives some encoding for $\bar{F}_i^{[\rho]}$ and consider the modification of TAKE-LARGEST-ON-FUTURE-ITEMS where, instead of $\tilde{S}_i$, the members of $\bar{F}_i^{[\rho]}$ enter the solution under construction. Then, the following holds.

Proposition 3. *The competitive ratio of modified TAKE-LARGEST-ON-FUTURE-ITEMS is bounded below by $\sqrt{\rho n}/2$.*

Proof. For some $\rho > 1$ and for some integer N, consider the following instance:

$$\begin{aligned}
\mathcal{S} &= \left\{X, Y, S_i^j : 1 \leqslant i \leqslant N, 1 \leqslant j \leqslant \rho\right\} \\
C &= \bigcup_{i=1}^{N} \bigcup_{j=1}^{\rho} S_i^j \quad \left(|C| = \rho\frac{N(N-1)}{2} + N = n\right) \\
X &= \{x_1, \ldots, x_N\} \\
\left|S_i^j\right| &= N - i + 1 \text{ for } i = 1, \ldots, N \\
S_i^j \bigcap S_l^k &= \emptyset, \text{ if } i \neq l \\
S_i^j \bigcap S_i^k &= \{x_i\}, \text{ if } j \neq k \\
Y &= C \setminus X
\end{aligned}$$

Consider the arrival sequence where $x_1, \ldots, x_N$ are firstly revealed. TAKE-LARGEST-ON-FUTURE-ITEMS might take in the cover all the S_i^j's, while the optimal cover is $\{X, Y\}$. In this case, the competitive ratio is $\rho N/2$, with:

$$N = \frac{\rho - 2}{2\rho} + \sqrt{\left(\frac{\rho - 2}{2\rho}\right)^2 + 2\frac{n}{\rho}}$$

i.e., the value of the ratio is asymptotically $\sqrt{\rho n/2}$.

For example, set $\rho = 2$ and $N = 5$ and consider the instance of Figure 4. For Σ starting with x_1, x_2, x_3, x_4, x_5, the algorithm may insert to the cover the sets depicted as "rows", while the optimal cover would consist of the "column"-set $\{x_1, x_2, x_3, x_4, x_5\}$ together with the "big" set containing the rest of the elements (drawn striped in Figure 4). ∎

Finally, let us note that one cannot do better if a rule concerning the already covered elements is added; the arrival sequence can be set in such a way that always arrive elements not yet covered, until the greedy online cover reaches all elements.

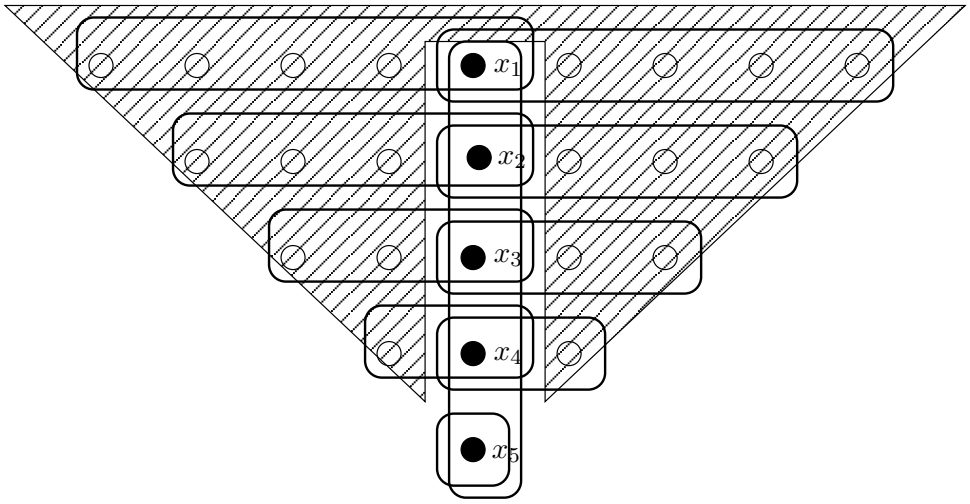

Figure 4: A counter-example for the case where the algorithm is allowed to take a constant number of sets containing a recently arrived element.

In the weighted version of set-covering, any set S of $\mathcal{S}$ is assigned with a non-negative weight $w(S)$, and a cover $\mathcal{S}'$ of the least possible total weight $W = \sum_{S \in \mathcal{S}'} w(S)$ has to be computed. A natural modification of `TAKE-LARGEST-ON-FUTURE-ITEMS` in order to deal with weighted set-covering is to put in the cover, whenever a still uncovered element arrives, a set S_i containing it that minimizes the quantity $w(S_i)/\delta(S_i)$. Unfortunately, this modification cannot perform satisfactorily. Consider, for example, an instance of weighted set-covering consisting of a ground set $C = \{x_1, \ldots, x_n\}$, and three sets, $S = C$ with $w(S) = n$, $X = \{x_1\}$ with $w(X) = 1$ and $Y = C \setminus \{x_1\}$ with $w(Y) = 0$. If x_1 arrives first, the algorithm could have chosen S to cover it, yielding a cover for the overall instance of total weight n, while the optimal cover would be $\{X, Y\}$ of total weight 1.

7 The maximum budget saving problem

In this section, we study a kind of dual version of the minimum set-covering, the maximum budget saving problem. Here, we are allotted an initial budget $B(\mathcal{S}, C)$ destined to cover the cost of an algorithm that solves minimum set-covering on $(\mathcal{S}, C)$. Any such algorithm has its own cost that is a function of the size of the solution produced, of the time overheads it takes in order to compute it, etc. Our objective is to maximize our savings, i.e., the difference between the initial budget and the cost of the algorithm. For simplicity, we assume that the maximum saving ever possible to be performed is $B(\mathcal{S}, C) - k^*$, where, as previously, k^* is the size of an optimum set-cover of $(\mathcal{S}, C)$.

We consider here that the set-covering instance arrives on-line. If a purely on-line algorithm is used to solve it, then its cost equals the size of the solution computed; otherwise, if the algorithm allows itself to wait in order to solve the instance (partly or totally) off-line then, its cost is the sum of the size of the solution computed plus a fine that is equal to some root, of order strictly smaller than 1, of the solution that would be computed by a purely on-line algorithm. We suppose that the budget allotted is equal to $k^*\sqrt{n}$, where $n = |C|$. This assumption on $B(\mathcal{S}, C)$ is quite natural. It corresponds to a kind of feasible cost for an algorithm; we assume that this algorithm is `TAKE-LARGEST-ON-FUTURE-ITEMS`, the best among the ones seen in the paper.

The interpretation of this model is the following. We are allotted a budget corresponding to

the cost of an algorithm always solving set-covering. In this way, we are sure that we can always construct a feasible solution for it. Furthermore, by the second part of Theorem 3, it is very risky to be allotted less than $k^*\sqrt{n}$ since there exist instances where the bound $\sqrt{n}$ is attained. On the other hand, we can have at our disposal a bunch of on-line or off-line set-covering algorithms, any one having its proper cost as described just above, from which we have to choose the one whose use will allow us to perform the maximum possible economy with respect to our initial budget. The fact that the measure of the optimum solution for maximum budget saving is $B(\mathcal{S}, C) - k^*$, has also a natural interpretation: we can assume that there exist an arrival sequence Σ for C such that, for any $\sigma_i \in \Sigma$, an oracle can always choose to cover σ_i with the same set with which σ_i is covered in an optimum off-line solution for instance $(\mathcal{S}, C)$. Under this assumption for the measure of the optimum budget saving solution, this problem is clearly **NP**-hard since it implies computation of an optimum solution for minimum set-covering. Finally, denoting by $c_{\mathtt{A}}(\mathcal{S}, C)$ the cost of algorithm `A` when solving minimum set-covering on $(\mathcal{S}, C)$, the approximation ratio of maximum set saving is equal to:

$$\frac{B(\mathcal{S}, C) - c_{\mathtt{A}}(\mathcal{S}, C)}{B(\mathcal{S}, C) - k^*} \tag{13}$$

Obviously this ratio is smaller than 1 and, furthermore, the closer the ratio to 1, the better the algorithm achieving it.

Theorem 4. *Under the model adopted,* `GREEDY` *is asymptotically optimum for maximum budget saving.*

Proof. Consider an instance $(\mathcal{S}, C)$ of minimum set-covering and denote by $k_{\mathtt{F}}$ and $k_{\mathtt{L}}$, the sizes of the solutions computed by algorithms `GREEDY` and `TAKE-LARGEST-ON-FUTURE-ITEMS`, respectively. By what has been assumed just above, denoting by $c_{\mathtt{F}}$ the cost of using `GREEDY`, there exist some $\epsilon > 0$ such that:

$$c_{\mathtt{F}}(\mathcal{S}, C)) = k_{\mathtt{F}} + k_{\mathtt{L}}^{1-\epsilon} \tag{14}$$

Moreover, the following inequalities hold, the first one from [10] and the second one from Theorem 3:

$$k_{\mathtt{F}} \leqslant k^* \log n \tag{15}$$

$$k_{\mathtt{L}} \leqslant k^* \sqrt{n} \tag{16}$$

Using (14), (15) and (16), we get the following inequality for $c_{\mathtt{F}}(\mathcal{S}, C))$:

$$c_{\mathtt{F}}(\mathcal{S}, C)) \leqslant k^{*1-\epsilon} n^{\frac{1-\epsilon}{2}} + k^* \log n \leqslant \left(n^{\frac{1-\epsilon}{2}} + \log n\right) k^* \tag{17}$$

On the other hand, as assumed above:

$$B(\mathcal{S}, C) = k^* \sqrt{n} \tag{18}$$

Using (13), (17) and (18), we obtain:

$$\frac{B(\mathcal{S}, C) - c_{\mathtt{F}}(\mathcal{S}, C)}{B(\mathcal{S}, C) - k^*} \geqslant \frac{k^* \sqrt{n} - \left(n^{\frac{1-\epsilon}{2}} + \log n\right) k^*}{k^* \sqrt{n} - k^*} = \frac{\sqrt{n} - \left(n^{\frac{1-\epsilon}{2}} + \log n\right)}{\sqrt{n} - 1} \tag{19}$$

It is easy to see that, for n large enough, the last term of (19) tends to 1, and the statement claimed by the theorem is true. ∎

Remark also that if we are allotted with a budget equal to $k^* \log n \log m$ (i.e., the cost of the on-line algorithm of [1]) and we assume that the fine paid by algorithm `GREEDY` is also computed with respect to the algorithm of [1]), then a similar analysis as in the proof of Theorem 4 leads to the same result, i.e., that `GREEDY` remains asymptotically optimum.

Also, if the budget allotted is $k^*\sqrt{n}$ and one calls the on-line algorithm of [1], this latter algorithm is asymptotically optimum for maximum budget saving.

8 Discussion

We have introduced several simple on-line models for set-covering and analyzed greedy rules for them. Many of these rules are strongly competitive since no on-line algorithm for the models that they treat can achieve better ratios than they do. One of the features of the models studied here is that they are very economic and thus suitable to solve very large instances. Indeed, their memory requirements are extremely reduced since the only information needed are the names of m sets. Note that this is not the case for the intensive computations implied by the very interesting model of [1].

Next, we have introduced and studied the maximum budget saving problem. Here, we have relaxed irrevocability in the solution construction by allowing the algorithm to delay its decisions modulo some fine to be paid. For such a model we have shown that the natural greedy off-line algorithm is asymptotically optimum.

A subject for further research is the extension of our models to deal with minimum-weight set-covering. For this version work is in progress.

Finally, let us note that the on-line models described in the paper can be extended to apply to a different but related problem, the minimum dominating set. Consistently with the model that we have adopted for the set-covering problem, our model for this latter problem is as follows. Given a graph $G(V, E)$ with $|V| = n$, assume that its vertices switch on one-by-one. Any time a vertex σ_i switches on, the names of its neighbors are announced.

Consider the following classical reduction from minimum dominating set to set-covering:

- $\mathcal{S} = C = V$;
- the set $S_i \in \mathcal{S}$, corresponding to the vertex $v_i \in V$, contains elements $c_{i_1}, c_{i_2}, \ldots$, of C corresponding to the neighbors $v_{i_1}, v_{i_2}, \ldots$, of v_i in G.

The set-covering instance $(\mathcal{S}, C)$ so constructed, has $|\mathcal{S}| = |C| = n$. Furthermore, it is easy to see that any set cover of size k in $(\mathcal{S}, C)$ corresponds to a dominating set of the same size in G and vice-versa. Remark also that the dominating set model just assumed on G is exactly, with respect to $(\mathcal{S}, C)$, the set-covering model dealt in the paper.

References

[1] N. Alon, B. Awerbuch, Y. Azar, N. Buchvinder, and S. Naor. The online set cover problem. In *Proc. STOC'03*, pages 100–105, 2003.

[2] V. Chvátal. A greedy-heuristic for the set covering problem. *Math. Oper. Res.*, 4:233–235, 1979.

[3] G. Gambosi, M. Protasi, and M. Talamo. Preserving approximation in the min-weighted set cover problem. *Discrete Appl. Math.*, 73:13–22, 1997.

[4] D. S. Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM J. Comput.*, 11(3):555–556, 1982.

[5] D. S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.*, 9:256–278, 1974.

[6] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of computer computations*, pages 85–103. Plenum Press, New York, 1972.

[7] L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete Math.*, 13:383–390, 1975.

[8] V. Th. Paschos. A survey about how optimal solutions to some covering and packing problems can be approximated. *ACM Comput. Surveys*, 29(2):171–209, 1997.

[9] R. Raz and S. Safra. A sub-constant error probability low-degree test and a sub-constant error probability PCP characterization of NP. In *Proc. STOC'97*, pages 475–484, 1997.

[10] P. Slavík. A tight analysis of the greedy algorithm for set cover. In *Proc. STOC'96*, pages 435–441, 1996.

[11] D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985.

[12] O. A. Telelis and V. Zissimopoulos. Dynamic maintenance of approximate set covers. Manuscript, 2004.