HAL
open science

Greedy algorithms for on-line set-covering

Giorgio Ausiello, Aristotelis Giannakos, Vangelis Th. Paschos

To cite this version:

Giorgio Ausiello, Aristotelis Giannakos, Vangelis Th. Paschos. Greedy algorithms for on-line set-covering. 2006. hal-00120329

HAL Id: hal-00120329

https://hal.science/hal-00120329v1

Preprint submitted on 14 Dec 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Greedy algorithms for on-line set-covering

Giorgio Ausiello*, Aristotelis Giannakos†, Vangelis Th. Paschos†

Abstract

We study the following on-line model for set-covering: elements of a ground set of size n arrive one-by-one and with any such element c_i, arrives also the name of some set S_{i_0} containing c_i and covering the most of the uncovered ground set-elements (obviously, these elements have not been yet revealed). For this model we analyze a simple greedy algorithm consisting of taking S_{i_0} into the cover, only if c_i is not already covered. We prove that the competitive ratio of this algorithm, that uses only $O(\log m)$ space, where m is the size of the set-system, is $\sqrt{n}$ and that it is asymptotically optimal for the model dealt, since no on-line algorithm can do better than $\sqrt{n/2}$. We next show that this model can also be used for solving minimum dominating set with competitive ratio bounded above by the square root of the size of the input graph. We also study a slightly simplified, but more memory consuming, on-line model: any time a ground set-element arrives, the whole sequence of the names of the sets containing it is revealed. For this model, we propose an algorithm achieving competitive ratio bounded above by the maximum frequency of the instance, i.e., by the maximum number of sets containing a ground set-element. 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. The objective is to maximize the savings on the initial budget. We show that when this budget is at least equal to $\sqrt{n}$ times the size of the optimal (off-line) solution of the instance under consideration, then the natural greedy off-line algorithm is asymptotically optimal.

*Dipartimento di Informatica e Sistemistica, Università degli Studi di Roma "La Sapienza", Via salaria 113, 00198, Roma, Italy. `ausiello@dis.uniroma1.it`

†LAMSADE, Université Paris-Dauphine and CNRS UMR 7024, Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France. `{giannako,paschos}@lamsade.dauphine.fr`

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 $c_i \in C$, we set $F_i = \{S_j \in \mathcal{S} : c_i \in S_j\}$ 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 ([7]) and $O(\log n)$-approximable for both weighted and unweighted cases (see [2], for the former and [6, 8, 10], for the latter; see also [9] for a comprehensive survey on the subject). This approximation ratio is the best achievable, unless **NP** $\subseteq$ **DTIME**$(\mathbf{n}^{\mathbf{O}(\log\log \mathbf{n})})$, i.e., unless problems in **NP** could be proved solvable by slightly super-polynomial algorithms ([3]).

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 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 c_i, arrives also the name of some set S_{i_0} containing c_i and covering the most of the ground set-elements that have not been yet covered. Clearly, any uncovered element is yet unrevealed. For this model we analyze a simple greedy algorithm consisting of taking S_{i_0} into the cover, only if c_i is not already covered. We prove that the competitive ratio of this algorithm is $\sqrt{n}$ and that it is asymptotically optimal for the model dealt, since no on-line algorithm can do better than $\sqrt{n/2}$. This model generalizes the one proposed in [1] and, furthermore, it uses a very simple, fast and intuitive algorithm that could be seen as the on-line counterpart of the natural greedy (off-line) set-covering algorithm.

2 Preliminaries

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 and studied in Section 3, is inspired, yet quite different, from the one in [1]. Given C, $\mathcal{S}$ (not known in advance as [1] assumes) and 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 $\{\sigma_1, \ldots, \sigma_i\} \subseteq \cup_{S \in \mathcal{S}'_i} S$. For any σ_i, $i = 1, \ldots$, we denote by S_i^j, $j = 1, \ldots, f_i$, the sets of $\mathcal{S}$ containing σ_i, by $\bar{S}_i^j$ the subset of the elements of S_i^j still remaining uncovered and by δ_i^j the cardinality of $\bar{S}_i^j$. By f_i, we denote the frequency of σ_i, i.e., the number of sets in $\mathcal{S}$ containing σ_i. When σ_i switches on, the only information revealed is the name of some set $S_i^{j_0} \in \operatorname{argmax}\{\delta_i^j, j = 1, \ldots, f_i\}$. So, no a priori knowledge of the topology of the instance $(\mathcal{S}, C)$ is assumed by the model. In particular, we do not have to know which are the yet uncovered elements of $S_i^{j_0}$ but only the fact that their number is maximum with respect to any other S_i^j.

The algorithm that we study for this model, called `LGREEDY` in the sequel, informally works as follows: once an element $\sigma_i \in C$ switches on, if σ_i is not already covered, then set $S_i^{j_0} \in \operatorname{argmax}\{\delta_i^j, j = 1, \ldots, f_i\}$ is added in the cover under construction. Clearly, by the way `LGREEDY` works, the content of $\bar{S}_i^{j_0}$ is still unrevealed. This algorithm follows the same principle as the natural greedy algorithm for (off-line) minimum set covering, called `FGREEDY` in the sequel, modulo the fact that this principle applies not to the whole instance $(\mathcal{S}, C)$ that is to be finally revealed, but to the part of $(\mathcal{S}, C)$ induced by the elements of C that, at a given moment, are switched off (even if the topology of this part is not known). We prove that the competitive ratio of this algorithm is $\sqrt{n}$ and also that there exist arbitrarily large instances for which this ratio is at least $\sqrt{n/2}$.

In Section 4, we provide a lower bound, equal to $\sqrt{n/2}$ for the competitiveness of a whole class of on-line algorithms running on our model. These algorithms are the ones that construct a cover by taking, at any activation step, at least one set containing some not yet covered recently activated ground element. Based upon this result, one can conclude

that LGRREDY is asymptotically optimal for this class and for the model adopted.

There exist several reasons motivating, to our opinion, the study of the model dealt in this paper. The first one is that the algorithm used is as it has already been mentioned, a kind of on-line alternative of the famous greedy algorithm for set-covering. Hence, analysis of its competitiveness is interesting by itself. The second reason is that a basic and very interesting feature of the model dealt is its very small memory requirement, since the only information needed is the binary encoding of the name of $S_i^{j_0}$. 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 algorithm dealt uses $O(\log m)$ space.

In Section 5.1, we show that the set-covering model just presented can be used to solve also minimum dominating set, within competitive ratio $\sqrt{n}$ where n is the order of the input graph. Minimum dominating set is defined as follows: given a graph $G(V, E)$, we wish to determine the least subset $V' \subseteq V$ that dominates the rest of the vertices, i.e., a subset $V' \subseteq V$ such that for all $u \in V \setminus V'$ there exists $v \in V'$ for which $(u, v) \in E$.

In Section 5.2, a slightly simplified, but somewhat more memory consuming on-line model for minimum set covering is adopted. Any time an element σ_i of the arrival sequence Σ switches on, the whole sequence $(S_i^1, \ldots, S_i^{f_i})$, where S_i^j, $j = 1, \ldots, f_i$ are as above, is revealed. For this model, we propose an algorithm achieving competitive ratio $f = \max_{\sigma_i \in \Sigma}\{f_i\}$. 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 6. We assume that two algorithms collaborate to solve it: the LGREEDY and the FGREEDY. 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 $B(x) = \sqrt{n}\,\mathrm{opt}(x)$ (this is in order that at least

LGREEDY 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 6 that there exists a natural algorithm-cost model such that FGREEDY is asymptotically optimal for maximum budget saving.

Before closing this section, let us quote another very interesting approach that could be considered to be at midway between semi-on-line approaches and solutions-stability ones, developed in [4]. 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 a set covering instance $(\mathcal{S}, C)$ and a solution $\mathcal{S}'$ for it. 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? It is shown in [4] 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.

3 A greedy on-line algorithm

As already mentioned, the model studied in this section assumes 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, the encoding for $S_i^{j_0} \in \mathrm{argmax}\{\delta_i^j, j = 1, \ldots, f_i\}$ is also revealed.

For this model, we propose the following algorithm, LGREEDY, where, although it is not necessary, we suppose for reasons of simplicity of algorithm's specification that n is known to it:

- set $\mathcal{S}'_0 = \emptyset$;
- for $i = 1$ to n do (σ_i switches on): if σ_i is not already covered by $\mathcal{S}'_{i-1}$,
  - then set $\mathcal{S}'_i = \mathcal{S}'_{i-1} \cup \{\mathrm{argmax}\{\delta_i^j : j = 1, \ldots, f_i\}\}$;
  - else set $\mathcal{S}'_i = \mathcal{S}'_{i-1}$;
- output $\mathcal{S}' = \mathcal{S}'_n$.

Theorem 1. *Consider an instance $(\mathcal{S}, C)$ of minimum set covering with $|C| = n$. Consider also the on-line model introduced above, and denote by $\mathcal{S}^* = \{S_1^*, \ldots, S_{k^*}^*\}$ an optimal off-line solution on $(\mathcal{S}, C)$. Then, the competitive ratio of LGREEDY 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:

$$\delta\left(S_1\right) = |S_1| \tag{1}$$

and, for $2 \leqslant i \leqslant k$,

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

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,

$$\sum_{i=1}^{r} h\left(S_i^*\right) = k \tag{3}$$

$$r \leqslant k^* \tag{4}$$

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 (1), or by (2)). So, dealing with c_i^ℓ, the following holds:

$$h\left(S_i^*\right) - \ell + 1 \leqslant \delta\left(S_{j_\ell}\right) \tag{5}$$

For example, consider the illustration of Figure 1. 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 `LGREEDY` to cover c^2. The shadowed parts of S^*, $\hat{S}$ and S correspond to elements already covered by `LGREEDY` 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).

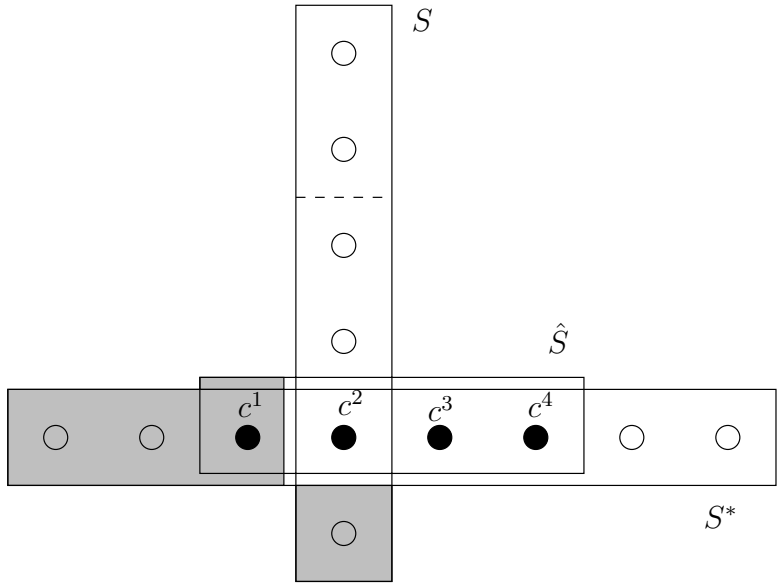

Figure 1: An example for (5).

Summing up inequalities (5), 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:

$$\frac{h\left(S_i^*\right)\left(h\left(S_i^*\right)+1\right)}{2} \leqslant \sum_{\ell=1}^{h\left(S_i^*\right)} \delta\left(S_{j_\ell}\right) = n_i \Longrightarrow h\left(S_i^*\right) \leqslant \sqrt{2n_i} \tag{6}$$

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

$$\sum_{i=1}^{r} \sqrt{\alpha_i} \leqslant \sqrt{r} \tag{7}$$

Using (3), (4), (6) and (7), we get:

$$k = \sum_{i=1}^{r} h\left(S_i^*\right) \leqslant \sqrt{2n} \sum_{i=1}^{r} \sqrt{\alpha_i} \leqslant \sqrt{r}\sqrt{2n} \leqslant \sqrt{k^*}\sqrt{2n} \tag{8}$$

Dividing the first and the last members of (8) by k^*, we get:

$$\frac{k}{k^*} \leqslant \sqrt{\frac{2n}{k^*}} \tag{9}$$

On the other hand, remark that, if $k^* = 1$, i.e., if there exists $S^* \in \mathcal{S}$ such that $\mathcal{S}^* = \{S^*\}$, then LGREEDY 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 (9),

$$\frac{k}{k^*} \leqslant \sqrt{n} \tag{10}$$

Combination of (9) and (10) 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 $(1, \ldots, N(N+1)/2)$. LGREEDY 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.

For example, consider Figure 2. For Σ starting with 1, 6, 10, 13, 15, LGREEDY 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 the two 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 (9), 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 1.

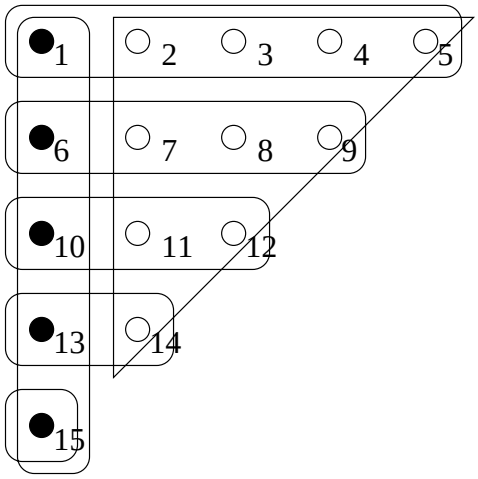

Figure 2: The ratio $\sqrt{n/2}$ for LGREEDY is asymptotically attained.

Corollary 1. *The competitive ratio of LGREEDY is bounded above by* $\sqrt{2\Delta}$.

The set-covering model dealt here is very economic and thus suitable to solve very large instances. Indeed, its memory requirements are extremely reduced since the only information LGREEDY needs at any step i is the encoding of the name of a set $S_i^{j_0} \in \operatorname{argmax}\{\delta_i^j, j = 1, \ldots, f_i\}$. This is not the case for the intensive computations implied by the model of [1].

The counter-example instance given in the proof of Theorem 1 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. 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. LGREEDY 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 $\sqrt{\rho n/2}$.

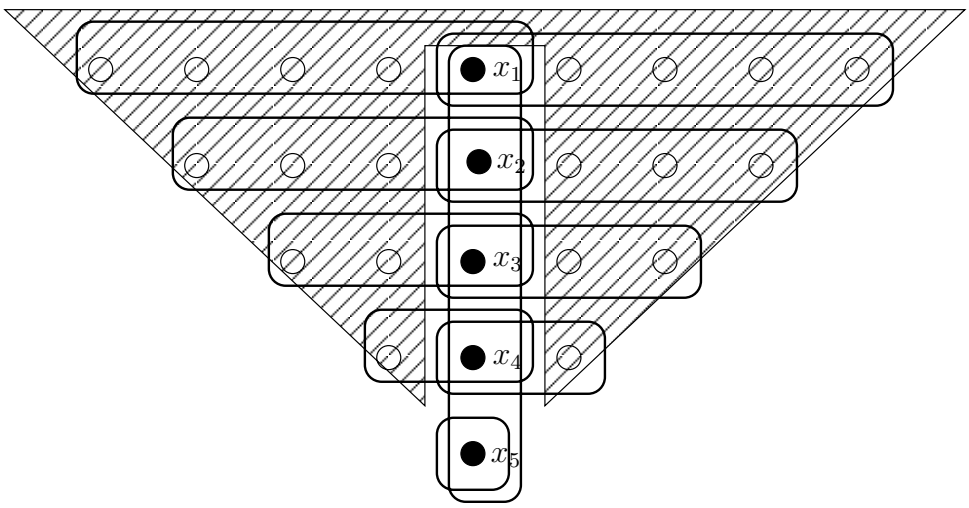

Figure 3: A counter-example for the case where the algorithm is allowed to take a constant number of sets containing a recently arrived element.

For example, set $\rho = 2$ and $N = 5$ and consider the instance of Figure 3. 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 3).

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 `LGREEDY` 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.

4 A lower bound for a whole class of on-line algorithms

We now prove that, in the general model, no on-line algorithm can achieve, for the model dealt, 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.

Proposition 1. *Let* A *be an on-line algorithm for set-covering such that, at any step, it takes in the cover at least one set containing some not yet covered arriving element. Let* $k_{\mathtt{A}}$ *be the size of the cover computed by* A *and* k^* *be the size of the optimal cover. Then,* $k_{\mathtt{A}}/k^* \geqslant \sqrt{n/2}$.

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.

For example consider the instance of Figure 4, 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 4(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 4(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 4(c)).

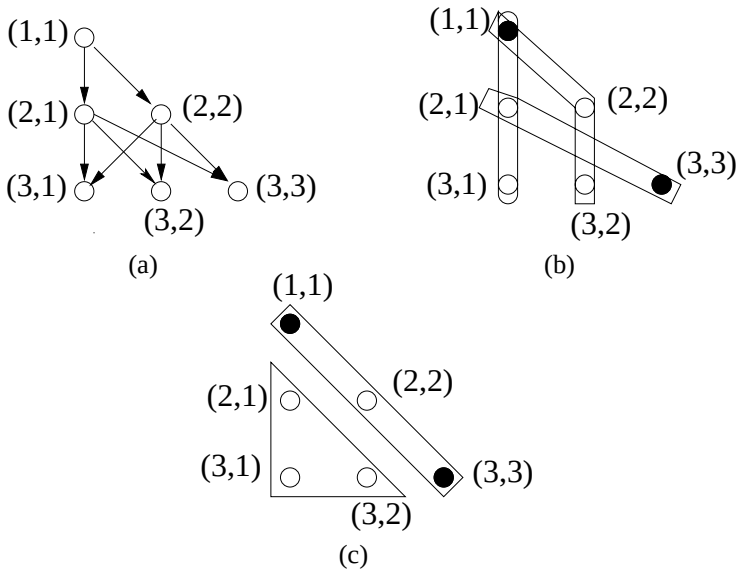

Figure 4: The counter-example of Proposition 1 for $N = 5$.

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. ∎

5 Other on-line models

Let us first note that the greedy algorithm 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.

5.1 On-line minimum dominating set

Let us consider a different but related problem, the on-line version of the minimum dominating set problem. 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 does so,

the name of its neighbor with the most neighbors still switched off is announced. Denote by v_{i_0} such neighbor of σ_i. If σ_i is not yet dominated by the partial dominating set V' already constructed, then v_{i_0} enters V'.

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 of Section 3. Consequently, the following result follows immediately.

Proposition 2. *The on-line set-covering algorithm of Theorem 1 is $\sqrt{n}$-competitive for minimum dominating set in graphs of order n.*

5.2 A simplified set-covering on-line model

In this section, we assume the following set-covering model. Any time an element σ_i of the arrival sequence Σ switches on, only the sequence $(S_i^1, \ldots, S_i^{f_i})$, where S_i^j, $j = 1, \ldots, f_i$ are as in Section 3, is revealed.

For this model, we consider the following algorithm, denoted by `LGREEDY2`:

- set $\mathcal{S}' = \emptyset$;
- while a new element σ_i switches on, if σ_i is not already covered, then set $\mathcal{S}' = \mathcal{S}' \cup \{S_i^1, \ldots, S_i^{f_i}\}$;
- output $\mathcal{S}' = \{S_1, \ldots, S_k\}$.

Proposition 3. *The competitive ratio of `LGREEDY2` is bounded above by f.*

Proof. Denote by $c_1, \ldots, c_k$ 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

$$|\mathcal{S}^*| \geqslant k \tag{11}$$

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

$$|\mathcal{S}'| \leqslant kf \tag{12}$$

Combining (11) and (12), the competitive ratio is immediately derived. ∎

Note that algorithm LGREEDY2 is similar to the approximation algorithm for minimum set covering presented in [5] and, furthermore, it guarantees the same approximation ratio.

6 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; this is algorithm LGREEDY presented in Section 3.

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 1, 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 2. *Under the model adopted, FGREEDY 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 FGREEDY and LGREEDY, respectively. By what has been assumed just above, denoting by $c_{\mathtt{F}}$ the cost of using FGREEDY, 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 1:

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

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

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 FGREEDY

is also computed with respect to the algorithm of [1]), then a similar analysis as in the proof of Theorem 2 leads to the same result, i.e., that FGREEDY 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.

7 Conclusions

We have introduced an on-line model associated with a natural greedy on-line algorithm achieving non-trivial competitive ratio $\sqrt{n}$. Moreover, we have shown that this simple algorithm is strongly competitive since no on-line algorithm for this model, even if it introduces in the cover more than one sets at a time, can guarantee better than $\sqrt{n/2}$. One of the features of our model is that the algorithm can run with an extremely small amount of memory and disk requirements (using, for example, only $O(\log m)$ space) and hence it is suitable to solve very large instances.

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.

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

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

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

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

[7] 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.

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

[9] 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.

[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.