

# Greedy algorithms for on-line set-covering and related problems

Giorgio Ausiello<sup>1</sup>

Aristotelis Giannakos<sup>2</sup>

Vangelis Th. Paschos<sup>2</sup>

<sup>1</sup>Dipartimento di Informatica e Sistemistica  
Università degli Studi di Roma “La Sapienza”  
Viaalaria 113, 00198, Roma, Italy  
Email: ausiello@dis.uniroma1.it

<sup>2</sup>LAMSADE  
Université Paris-Dauphine and CNRS UMR 7024  
Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France  
Emails: {giannako,paschos}@lamsade.dauphine.fr

## 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 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 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.

*Keywords:* Set-covering, On-line algorithm, Competitive ratio, Dominating set, Budget saving

## 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  $f = \max\{f_i : i = 1, \dots, n\}$ .

The set covering problem has been extensively studied over the past decades. It has been shown to be **NP**-hard in Karp (1972) and  $O(\log n)$ -approximable for both weighted and unweighted cases (see Chvátal (1979), for the former, and Johnson (1974), Lovász (1975) and Slavík (1996), for the latter; see also Paschos (1997) for a comprehensive survey on the subject). As it is shown by Feige (1998), this approximation ratio is the best achievable, unless **NP**  $\subseteq$  **DTIME**( $n^{O(\log \log n)}$ ), i.e., unless problems

in **NP** could be proved solvable by slightly super-polynomial algorithms.

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  $m(x, y)/\text{opt}(x)$ , 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  $\text{opt}(x)$  the value of an optimal off-line solution. This measure for on-line computation has been introduced by Sleator *et al.* (1985).

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 Alon *et al.* (2003) 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.

In Alon *et al.* (2003), 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 2, is inspired, yet quite different, from the one of Alon *et al.* (2003). Given  $C$ ,  $\mathcal{S}$  (not known in advance as Alon *et al.* (2003) assumes) and an arrival sequence  $\Sigma = (\sigma_1, \dots, \sigma_n)$  of the elements of  $C$  (i.e., elements of  $C$  are switched on following the order  $\sigma_1, \dots, \sigma_n$ ), the objective is to find, for any  $i \in \{1, \dots, n\}$ , a family  $\mathcal{S}'_i \subseteq \mathcal{S}$  such that  $\{\sigma_1, \dots, \sigma_i\} \subseteq \cup_{S \in \mathcal{S}'_i} S$ . For any  $\sigma_i$ ,  $i = 1, \dots$ , we denote by  $S_i^j$ ,  $j = 1, \dots, f_i$ , the sets of  $\mathcal{S}$  containing  $\sigma_i$ , by  $\bar{S}_i^j$  the subset of the elements of  $S_i^j$  still remaining uncovered and by  $\delta_i^j$  the cardinality of  $\bar{S}_i^j$ . By  $f_i$ , we denote the frequency of  $\sigma_i$ , i.e., the number of sets in  $\mathcal{S}$  containing  $\sigma_i$ . When  $\sigma_i$  switches on, the only information revealed is the name of some set  $S_i^{j_0} \in \operatorname{argmax}\{\delta_i^j, j = 1, \dots, f_i\}$ . 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  $\sigma_i$  is not already covered, then set  $S_i^{j_0} \in \operatorname{argmax}\{\delta_i^j, j = 1, \dots, f_i\}$  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  $\sqrt{n/2}$ . We then show that the set-covering model dealt 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 3, 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 **LGRREEDY** 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 Alon *et al.* (2003). 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 resources.

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 4. 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  $B(x) = \sqrt{n} \operatorname{opt}(x)$  (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  $B(x) - \operatorname{opt}(x)$ . We show in Section 4 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 Gambosi *et al.* (1997). 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 Gambosi *et al.* (1997) that if solution  $\mathcal{S}'$  has been produced by application of the natural greedy algorithm achieving approximation ratio  $O(\log n)$  (see Chvátal (1979)), then after  $O(\log n)$  such insertions initial solution  $\mathcal{S}'$  still guarantees the same approximation ratio.

## 2 A greedy on-line algorithm

As already mentioned, the model studied in this section assumes an arrival sequence  $\Sigma = (\sigma_1, \dots, \sigma_n)$  of the elements of  $C$ , and the objective is to find, for any  $i \in \{1, \dots, n\}$ , a family  $\mathcal{S}'_i \subseteq \mathcal{S}$  such that  $\{\sigma_1, \dots, \sigma_i\} \subseteq \cup_{S \in \mathcal{S}'_i} S$ . Once an element  $\sigma_i$ ,  $i = 1, \dots$ , switches on, the encoding for  $S_i^{j_0} \in$

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

**Theorem 1.** *Consider an instance  $(S, C)$  of minimum set covering with  $|C| = n$ . Consider also the on-line model introduced above, and denote by  $S^* = \{S_1^*, \dots, S_{k^*}^*\}$  an optimal off-line solution on  $(S, C)$ . 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  $\Sigma = (\sigma_1, \dots, \sigma_n)$  and denote by  $c_1, \dots, c_k$ , its *critical elements*, i.e., the elements having entailed introduction of a set in  $S'$ . In other words, critical elements of  $\Sigma$  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  $S'$  consists of  $k$  sets, namely,  $S_1, \dots, S_k$ , where  $S_1$  has been introduced in  $S'$  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  $S'$  for critical element  $c_i = \sigma_j$ ,  $\delta(S_i) = \max\{\delta_j^1, \dots, \delta_j^{f_j}\}$ ). We have:

$$\delta(S_1) = |S_1| \quad (1)$$

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

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

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

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

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

$$r \leq k^* \quad (4)$$

For any  $S_i^*$ , let  $c_1^i, \dots, c_{h(S_i^*)}^i$  be the elements of its critical content ordered according to their position in the arrival sequence  $\Sigma$ ; in other words, following our assumptions,  $\hat{S}_i^* = \{c_1^i, \dots, c_{h(S_i^*)}^i\}$  (recall that  $\hat{S}_i^* \subseteq S_i^*$ ).

Suppose, without loss of generality, that, for  $\ell = 1, \dots, h(S_i^*)$ , the set  $S_{j_\ell} \in S$  has been introduced in  $S'$  when the critical element  $c_\ell^i$  has been activated. At

the moment of the arrival of  $c_\ell^i$ , the set  $S_i^*$  is also a candidate set for  $S'$ . The fact that  $S_{j_1}$  has been chosen instead of  $S_i^*$  means that  $\delta(S_{j_1}) \geq \delta(S_i^*)$ ; hence, since as noticed just above,  $\hat{S}_i^* \subseteq S_i^*$ , the following holds immediately:  $\delta(S_{j_1}) \geq \delta(S_i^*) \geq |\hat{S}_i^*| = h(S_i^*)$ . When  $c_\ell^i$  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  $c_\ell^i$  (covered by  $S_{j_1}$ ). So, following the arguments developed just above for  $S_{j_1}$ ,  $\delta(S_{j_2}) \geq h(S_i^*) - 1$ , and so on (quantities  $\delta(\cdot)$  are defined either by (1), or by (2)). So, dealing with  $c_\ell^i$ , the following holds:

$$h(S_i^*) - \ell + 1 \leq \delta(S_{j_\ell}) \quad (5)$$

For example, consider the illustration of Figure 1. Let  $S^*$  be a set of the fixed optimal cover  $S^*$  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  $c^4$ ).

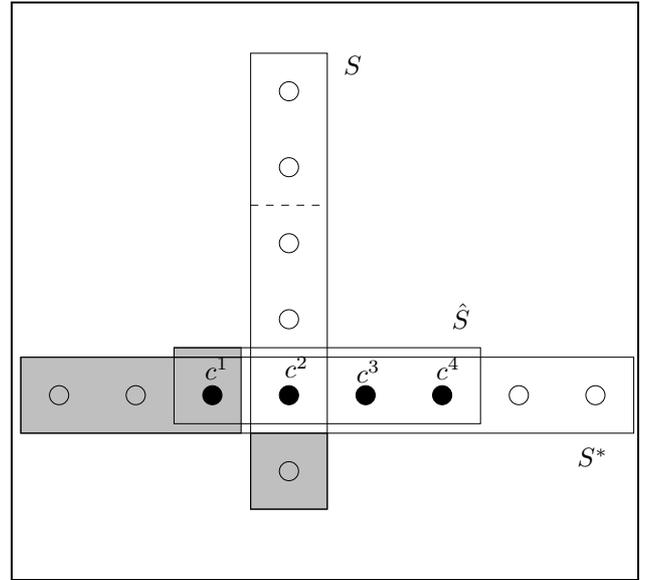


Figure 1: An example for (5)

Summing up inequalities (5), for  $\ell = 1, \dots, h(S_i^*)$ , and setting  $\sum_{\ell=1}^{h(S_i^*)} \delta(S_{j_\ell}) = n_i$ , we finally get for  $S_i^*$ :

$$\begin{aligned} \frac{h(S_i^*) (h(S_i^*) + 1)}{2} &\leq \sum_{\ell=1}^{h(S_i^*)} \delta(S_{j_\ell}) = n_i \\ \implies h(S_i^*) &\leq \sqrt{2n_i} \end{aligned} \quad (6)$$

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

$$\sum_{i=1}^r \sqrt{\alpha_i} \leq \sqrt{r} \quad (7)$$

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

$$k = \sum_{i=1}^r h(S_i^*) \leq \sqrt{2n} \sum_{i=1}^r \sqrt{\alpha_i} \leq \sqrt{r} \sqrt{2n} \leq \sqrt{k^*} \sqrt{2n} \quad (8)$$

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

$$\frac{k}{k^*} \leq \sqrt{\frac{2n}{k^*}} \quad (9)$$

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  $\sigma_1$ ; next, no additional set would have entered the  $\mathcal{S}'$ . Consequently, we can assume that  $k^* \geq 2$  and, using (9),

$$\frac{k}{k^*} \leq \sqrt{n} \quad (10)$$

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:

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

Consider the arrival sequence  $(1, \dots, N(N+1)/2)$ . LGREEDY might compute the cover  $\mathcal{S}' = \{S_i, 1 \leq i \leq N\}$ , 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  $\Sigma$  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:

$$\{1, 6, 10, 13, 15\} \quad (11)$$

$$\{2, 3, 4, 5, 7, 8, 9, 11, 12, 14\} \quad (12)$$

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:  $k^* \geq n/\Delta$ . Then, the following result is immediately derived from Theorem 1.

**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  $S_i^{j_0} \in \operatorname{argmax}\{\delta_i^j, j = 1, \dots, f_i\}$ . This is not the case for

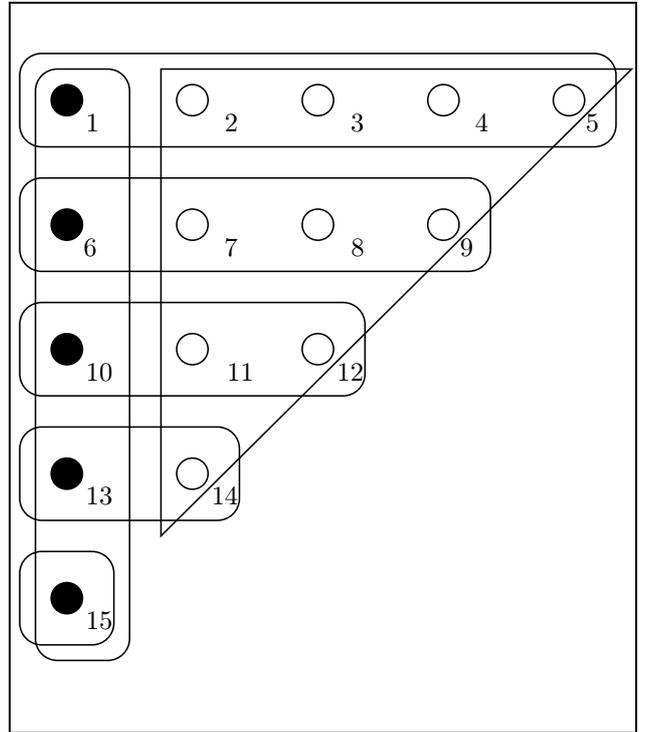


Figure 2: The ratio  $\sqrt{n/2}$  for LGREEDY is asymptotically attained

the intensive computations implied by the model of Alon *et al.* (2003).

Let us note that the model dealt above can be used to solve a natural on-line version of the minimum dominating set problem. Given a graph  $G(V, E)$  with  $|V| = n$ , assume that its vertices switch on one-by-one. Any time a vertex  $\sigma_i$  does so, the name of its neighbor with the most neighbors still switched off is announced. Denote by  $v_{i_0}$  such neighbor of  $\sigma_i$ . If  $\sigma_i$  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: the vertex-set  $V$  of the input-graph  $G$  becomes both the family of subsets and the ground set of the set covering instance (hence, both items have size  $n$ ) and for any vertex  $v_i \in V$ , the corresponding set contains  $v_i$  itself together with its neighbors in  $G$ . It is easy to see that any set cover of size  $k$  in the so-constructed set covering instance 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 the transformation just sketched, the set-covering model dealt before. Consequently, the following result follows immediately.

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

Note also that 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:

$$\mathcal{S} = \left\{ X, Y, S_i^j : 1 \leq i \leq N, 1 \leq j \leq \rho \right\}$$



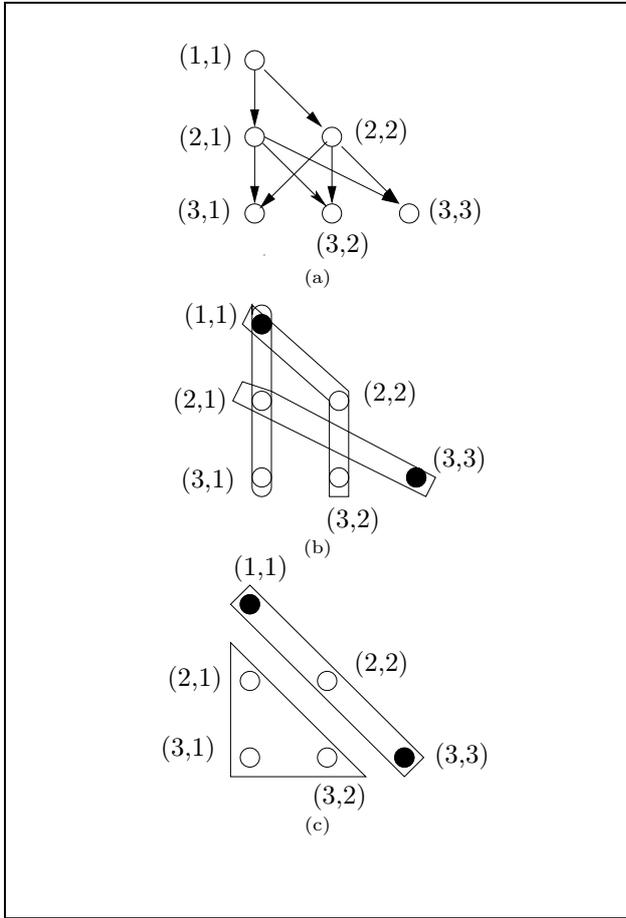


Figure 4: The counter-example of Proposition 2 for  $N = 5$

#### 4 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 2.

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 The-

orem 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  $\Sigma$  for  $C$  such that, for any  $\sigma_i \in \Sigma$ , an oracle can always choose to cover  $\sigma_i$  with the same set with which  $\sigma_i$  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  $c_A(\mathcal{S}, C)$  the cost of algorithm A when solving minimum set-covering on  $(\mathcal{S}, C)$ , the approximation ratio of maximum set saving is equal to:

$$\frac{B(\mathcal{S}, C) - c_A(\mathcal{S}, C)}{B(\mathcal{S}, C) - k^*} \quad (13)$$

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  $k_F$  and  $k_L$ , the sizes of the solutions computed by algorithms FGREEDY and LGREEDY, respectively. By what has been assumed just above, denoting by  $c_F$  the cost of using FGREEDY, there exist some  $\epsilon > 0$  such that:

$$c_F(\mathcal{S}, C) = k_F + k_L^{1-\epsilon} \quad (14)$$

Moreover, the following inequalities hold, the first one from Slavík (1996) and the second one from Theorem 1:

$$k_F \leq k^* \log n \quad (15)$$

$$k_L \leq k^* \sqrt{n} \quad (16)$$

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

$$c_F(\mathcal{S}, C) \leq k^{*1-\epsilon} n^{\frac{1-\epsilon}{2}} + k^* \log n \leq \left( n^{\frac{1-\epsilon}{2}} + \log n \right) k^* \quad (17)$$

On the other hand, as assumed above:

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

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

$$\begin{aligned} \frac{B(\mathcal{S}, C) - c_F(\mathcal{S}, C)}{B(\mathcal{S}, C) - k^*} &\geq \frac{k^* \sqrt{n} - \left( n^{\frac{1-\epsilon}{2}} + \log n \right) k^*}{k^* \sqrt{n} - k^*} \\ &= \frac{\sqrt{n} - \left( n^{\frac{1-\epsilon}{2}} + \log n \right)}{\sqrt{n} - 1} \quad (19) \end{aligned}$$

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 Alon *et al.* (2003)) and we assume that

the fine paid by algorithm **FGREEDY** is also computed with respect to the algorithm of Alon *et al.* (2003), 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 Alon *et al.* (2003), this latter algorithm is asymptotically optimum for maximum budget saving.

## 5 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 and hence it is suitable for solving 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 optimal.

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

- Alon, N., Awerbuch, B., Azar, Y., Buchvinder, N. & Naor, S. (2003), The online set cover problem, in 'Proc. STOC'03', pp. 100–105.
- Chvátal, V. (1979), 'A greedy-heuristic for the set covering problem', *Math. Oper. Res.* **4**, 233–235.
- Feige, U. (1998), 'A threshold of  $\ln n$  for approximating set cover', *J. Assoc. Comput. Mach.* **45**, 634–652.
- Gambosi, G., Protasi, M. & Talamo, M. (1997), 'Preserving approximation in the min-weighted set cover problem', *Discrete Appl. Math.* **73**, 13–22.
- Johnson, D. S. (1974), 'Approximation algorithms for combinatorial problems', *J. Comput. System Sci.* **9**, 256–278.
- Karp, R. M. (1972), Reducibility among combinatorial problems, in R. E. Miller & J. W. Thatcher, eds, 'Complexity of computer computations', Plenum Press, New York, pp. 85–103.
- Lovász, L. (1975), 'On the ratio of optimal integral and fractional covers', *Discrete Math.* **13**, 383–390.
- Paschos, V. (1997), 'A survey about how optimal solutions to some covering and packing problems can be approximated', *ACM Comput. Surveys* **29**(2), 171–209.
- Slavík, P. (1996), A tight analysis of the greedy algorithm for set cover, in 'Proc. STOC'96', pp. 435–441.
- Sleator, D. & Tarjan, R. E. (1985), 'Amortized efficiency of list update and paging rules', *Commun. ACM* **28**(2), 202–208.