Improved worst-case complexity for the MIN 3-SET COVERING problem

Federico Della Croce, Bruno Escoffier, Vangelis Paschos

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
CNRS UMR 7024

CAHIER DU LAMSADE

232

Janvier 2006

Improved worst-case complexity for the MIN 3-SET COVERING problem

Federico Della Croce, Bruno Escoffier, Vangelis Th. Paschos

Improved worst-case complexity for the MIN 3-SET COVERING problem*

Federico Della Croce[1] Bruno Escoffier[2] Vangelis Th. Paschos[2]

[1] D.A.I., Politecnico di Torino, Italy, federico.dellacroce@polito.it
[2] LAMSADE, CNRS UMR 7024 and Université Paris-Dauphine, France
{escoffier,paschos}@lamsade.dauphine.fr

January 13, 2006

Abstract

We consider MIN SET COVERING when the subsets are constrained to have maximum cardinality three. We propose an exact algorithm whose worst case complexity is bounded above by $O^*(1.3957^n)$. This is an improvement, based on a refined analysis, of a former result ($O^*(1.4492^n)$) by F. Della Croce and V. Th. Paschos, *Computing optimal solutions for the* MIN 3-SET COVERING *problem*, Proc. ISAAC'05, LNCS 3827, pp. 685–692.

Keywords: Worst-case complexity, Exact algorithm, MIN SET COVERING

In MIN SET COVERING, we are given a universe U of elements and a collection $\mathcal{S}$ of (non-empty) subsets of U. The aim is to determine a minimum cardinality sub-collection $\mathcal{S}' \subseteq \mathcal{S}$ which covers U, i.e., $\cup_{S \in \mathcal{S}'} S = U$ (we assume that $\mathcal{S}$ covers U). The frequency f_i of $u_i \in U$ is the number of subsets $S_j \in \mathcal{S}$ in which u_i is contained. The cardinality d_j of $S_j \in \mathcal{S}$ is the number of elements $u_i \in U$ that S_j contains. We say that S_j *hits* S_k if both S_j and S_k contain an element u_i and that S_j *double-hits* S_k if both S_j and S_k contain at least two elements u_i, u_l. Finally, we denote by n the size (cardinality) of $\mathcal{S}$ and by m the size of U. In what follows, we restrict ourselves to MIN SET COVERING-instances such that:

1. no element $u_i \in U$ has frequency $f_i = 1$;
2. no set $S_i \in \mathcal{S}$ is a subset of another set $S_j \in \mathcal{S}$.
3. no pair of elements u_i, u_j exists such that every subset $S_i \in \mathcal{S}$ containing u_i contains also u_j.

Indeed, if item 1 is not verified, then the set containing u_i belongs to any feasible cover of U. On the other hand, if item 2 is not verified, then S_i can be replaced by S_j in any solution containing S_i and the resulting cover will not be worse than the one containing S_i. Finally, if item 3 is not verified, then element u_j can be ignored as any sub-collection $\mathcal{S}'$ covering u_i will necessarily cover also u_j. So, for any instance of MIN SET COVERING, a preprocessing of data, obviously performed in polynomial time, leads to instances where all items 1, 2 and 3 are verified.

Let $T(\cdot)$ be a super-polynomial and $p(\cdot)$ be a polynomial, both on integers. In what follows, using notations in [9], for an integer n, we express running-time bounds of the form $p(n).T(n)$ as $O^*(T(n))$, the asterisk meaning that we ignore polynomial factors. We denote by $T(n)$ the

*Part of this research has been performed while the first author was in visit at the LAMSADE on a research position funded by the CNRS

worst case time required to exactly solve the MIN SET COVERING problem with n subsets. We recall (see, for instance, [5]) that, if it is possible to bound above $T(n)$ by a recurrence expression of the type $T(n) \leqslant \sum T(n-r_i)+O(p(n))$, we have $T(n) = O^*(\alpha(r_1, r_2, \ldots)^n)$ where $\alpha(r_1, r_2, \ldots)$ is the largest zero of the function $f(x) = 1 - \sum x^{-r_i}$.

There exist to our knowledge few results on worst-case complexity of exact algorithms for MIN SET COVERING or for cardinality-constrained versions of it. Let us note that an exhaustive algorithm computes any solution for MIN SET COVERING in $O(2^n)$. For MIN SET COVERING the most recent non-trivial result is the one of [6] (that has improved the result of [8]) deriving a bound (requiring exponential space) of $O^*(1.2301^{(m+n)})$. We consider here, the most notorious cardinality-constrained version of MIN SET COVERING, the MIN 3-SET COVERING, namely, MIN SET COVERING where $d_j \leqslant 3$ for all $S_j \in \mathcal{S}$ (notice that the bound of [6], for the case where $f_i = 2$, $u_i \in U$, and $d_j = 3$, for any $S_j \in S$ corresponds to $O^*(1.2301^{(5n/2)}) \approx O^*(1.6782^n)$). It is well known that MIN 3-SET COVERING is **NP**-hard, while MIN 2-SET COVERING (where any set has cardinality at most 2) is polynomially solvable by matching techniques ([2, 7]).

Our purpose is to devise an exact (optimal) algorithm with provably improved worst-case complexity for MIN 3-SET COVERING. We propose a search tree-based algorithm with running time $O^*(1.3957^n)$. This result, largely inspired by the one of [4], further improves it by reducing the complexity of the tree-based algorithm from $O^*(1.4492^n)$ down to $O^*(1.3957^n)$. This outcome is due to a different complexity analysis of the algorithm by the introduction of a kind of weights on the fixed sets. This technique seems to be quite close to the one very recently introduced in [6].

The following straightforward lemma holds, inducing some useful domination conditions for the solutions of MIN SET COVERING.

Lemma 1. *There exists at least one optimal solution of* MIN SET COVERING *where:*

1. *for any subset S_j with $d_j = 2$ containing elements u_i, u_p, if S_j is included in $\mathcal{S}'$, then all subsets S_k hitting S_j are excluded from $\mathcal{S}'$;*

2. *for any subset S_j with $d_j = 3$ containing elements u_i, u_p, u_q, where S_j double-hits another subset S_k with $d_k = 3$ on u_i and u_p, if S_j is included in $\mathcal{S}'$ then S_k must be excluded from $\mathcal{S}'$ and viceversa;*

3. *for any subset S_j with $d_j = 3$ containing elements u_i, u_p, u_q, if S_j is included in $\mathcal{S}'$, then either all subsets S_k hitting S_j on element u_i are excluded from $\mathcal{S}'$, or all subsets S_k hitting S_j on elements u_p and u_q are excluded from $\mathcal{S}'$.*

Proof. We only prove item 1, items 2 and 3 being proved by the same kind of analysis. Assume, without loss of generality, that S_j hits S_k on u_i and S_l on u_p. Suppose by contradiction that the optimal solution S' includes S_j and S_k. Then, it cannot include no more S_l, or else, it would not be optimal as a better cover would be obtained by excluding S_j from S'. On the other hand, suppose that S' includes S_j, S_k but does not include S_l. Then, an equivalent optimal solution can be derived by swapping S_j with S_l. ∎

In what follows, we consider the following counting. When we fix the status of a set of size 3, then our benefit is 1. When we do not fix a set of size 3 but cover one element of this set (hence this set will have size 2 is the remaining instance), we consider that our benefit is $\alpha \leqslant 1$. Obviously, when a set of size 2 is fixed, we can only consider that (in the worst case) our benefit is $1 - \alpha$. Hence, in some cases, the benefit is increasing with α while, in other cases, it is decreasing. An optimal value for α, following our analysis, is $\alpha = 0.297$.

The rest of the paper is devoted to the proof of the following result.

Theorem 1. MIN 3-SET COVERING *can be optimally solved within time $O^*(1.396^n)$.*

The algorithm either reduces the MIN 3-SET COVERING instance according to assumptions 1, 2 and 3 on the form of the instance (by detecting a subset S_j to be immediately included in (excluded from) $\mathcal{S}'$ or an element u_i to be ignored (correspondingly reducing the size of several subsets)), or applies a branching on subset S_j, where the following exhaustive relevant branching cases may occur.

1. $d_j = 2$: then no double-hitting occurs to S_j or else, due to the preprocessing step of the algorithm, S_j can be excluded from $\mathcal{S}'$ without branching. The following subcases occur.

   (a) S_j contains elements u_i, u_k with $f_i = f_k = 2$ where S_j hits S_l on u_i and S_m on u_k. Due to Lemma 1, if S_j is included in $\mathcal{S}'$, then both S_l and S_m must be excluded from $\mathcal{S}'$; alternatively, S_j is excluded from $\mathcal{S}'$ and, correspondingly, both S_l and S_m must be included in $\mathcal{S}'$ to cover elements u_i, u_k. For the analysis, consider the two following cases.

      i. $d_l = 3$, or $d_m = 3$, say $d_l = 3$. Then, in both cases (including or excluding S_j) we fix $3 - 2\alpha$ (1 for S_l, (at least) $1 - \alpha$ for S_j and S_m).

      ii. $d_l = d_m = 2$, S_l contains u_i and u_l and S_m contains u_k and u_m, (with $u_l \neq u_m$, otherwise no need to branch). By including S_j we fix $3(1 - \alpha)$. Otherwise, u_l is contained in S_p and u_m in S_q. If $S_p \neq S_q$, then we fix at least $3(1-\alpha)+2\alpha = 3-\alpha$. Indeed, we fix $1-\alpha$ for any of the sets S_j, S_l, S_m; by covering u_m, we fix α (resp., $1 - \alpha \geqslant \alpha$) if $d_p = 3$ (resp., if $d_p = 2$, since we can exclude S_p), and the same holds for covering u_k. Note that this is still valid if $S_p = S_q$, since in this case we can exclude this set, which gives at least $1 - \alpha \geqslant 2\alpha$.

   In case 1(a)i, we have $T(n) \leqslant 2T(n - 3 + 2\alpha) + O(p(n))$. This results in a time-complexity of $O^*(1.334^n)$. In case 1(a)ii, we have $T(n) \leqslant T(n - 3 + 3\alpha) + T(n - 3 + \alpha) + O(p(n))$. This results in a time-complexity of $O^*(1.336^n)$.

   (b) S_j contains elements u_i, u_k with $f_i = 2$ and $f_k \geqslant 3$, where S_j hits S_l on u_i and S_m, S_p on u_k. Due to Lemma 1, if S_j is included in $\mathcal{S}'$, then S_l, S_m, S_p must be excluded from $\mathcal{S}'$; alternatively, S_j is excluded from $\mathcal{S}'$ and, correspondingly, S_l must be included in $\mathcal{S}'$ to cover element u_i. For the analysis, consider the two following cases.

      i. $d_l = 2$, i.e., S_l contains u_i, u_l; then, $f_l \geqslant 3$ (or else we are in case 1a). Then, by including S_j, we fix $4(1 - \alpha)$ ($(1 - \alpha)$ for any of the sets S_j, S_l, S_m, S_p); by excluding S_j, we fix $2(1 - \alpha) + 2\alpha = 2$ ($(1 - \alpha)$ for any of the sets S_j, S_l, and (at least) α for each set containing u_l).

      ii. If $d_l \geqslant 3$, i.e., S_l contains at least u_i, u_l, u_m, then by including S_j, we fix $3(1-\alpha)+1$ (since now fixing S_l gives benefit 1); by excluding S_j, we fix $(1 - \alpha) + 1 + 2\alpha = 2 + \alpha$ (α from covering u_l, and α from covering u_m, with the same reasoning as in case 1(a)ii).

   The worst case is 1(b)i where we get $T(n) \leqslant T(n - 2) + T(n - 4 + 4\alpha) + O(p(n))$, resulting in a time-complexity of $O^*(1.338^n)$.

   (c) S_j contains elements u_i, u_k with $f_i = 3$ and $f_k \geqslant 3$ where S_j hits S_l, S_m on u_i and (at least) S_p, S_q on u_k. Note that we can suppose that S_j hits at least one set of size 3. Due to Lemma 1, if S_j is included in $\mathcal{S}'$, then S_l, S_m, S_p, S_q must be excluded from $\mathcal{S}'$; alternatively, S_j is excluded from $\mathcal{S}'$. For the analysis, consider the three following cases.

      i. If $d_l = d_m = d_p = d_q = 3$, then we fix either $5 - \alpha$, or $1 - \alpha$.

ii. If $d_l = 2$ or $d_m = 2$, say $d_l = 2$, then we fix either $5-4\alpha$, or $1-\alpha$. But in the case where we exclude S_j from $\mathcal{S}'$, then S_l has size 2 and contains u_i, whose frequency is now 2. Hence, we are either in case 1a or in case 1b. In the worst case, the branching gives (with case 1(b)i) $5-4\alpha$, $5(1-\alpha)$ and $3-\alpha$.

iii. Finally, if $d_l = d_m = 3$, then we can suppose that $f_k \geqslant 4$ (otherwise we are either in case 1(c)i or in case 1(c)ii). In this case, by including S_j we fix $2+4(1-\alpha)$ and by excluding S_j we fix $1-\alpha$.

In case 1(c)i, we get $T(n) \leqslant T(n-1+\alpha) + T(n-5+\alpha) + O(p(n))$, i.e., a time-complexity of $O^*(1.3953^n)$. In case 1(c)ii, we get $T(n) \leqslant T(n-3+\alpha) + T(n-5+5\alpha) + T(n-5+4\alpha) + O(p(n))$. This results in a time-complexity of $O^*(1.3942^n)$. In case 1(c)iii, we get $T(n) \leqslant T(n-6+4\alpha) + T(n-1+\alpha) + O(p(n))$, i.e., a time-complexity of $O^*(1.389^n)$.

(d) S_j contains elements u_i, u_k with $f_i \geqslant 4$ and $f_k \geqslant 4$ where S_j hits S_l, S_m, S_p on u_i and S_q, S_r, S_s on u_k. Note that we can suppose that S_j hits at least one set of size 3. Due to Lemma 1, if S_j is included in $\mathcal{S}'$, then $S_l, S_m, S_p, S_q, S_r, S_s$ must be excluded from $\mathcal{S}'$; alternatively, S_j is excluded from $\mathcal{S}'$. Then, we fix either $7-6\alpha$ or $1-\alpha$ getting $T(n) \leqslant T(n-1+\alpha) + T(n-7+6\alpha) + O(p(n))$, resulting so in a time-complexity of $O^*(1.366^n)$.

2. $d_j = 3$ (that is, there does not exist $S_k \in S$ such that $d_k = 2$) and there is at least one element u_i with $f_i = 2$. Then, S_j contains u_i, u_j, u_k, and S_k contains u_i, u_l, u_m (notice that no double crossing can occur between S_j and S_k due to the preprocessing step of the algorithm). Then, either we include S_j, and we fix $1+3\alpha$ new sets, or we exclude S_j, and we have to include S_k fixing so $2+2\alpha$ new sets. In this case, we get $T(n) \leqslant T(n-1-3\alpha) + T(n-2-2\alpha) + O(p(n))$. This results in a time-complexity of $O^*(1.366^n)$.

3. $d_j = 3$, all elements have a frequency at least 3, with S_j double-hitting one or more subsets. The following exhaustive subcases may occur.

(a) S_j double-hits at least three subsets S_k, S_l, S_m. Due to Lemma 1, if S_j is included in $\mathcal{S}'$ then S_k, S_l, S_m must be excluded from $\mathcal{S}'$; alternatively, S_j is excluded from $\mathcal{S}'$. This can be seen as a binary branching where either one subset (S_j) is fixed, or four subsets (S_j, S_k, S_l, S_m) are fixed and hence, $T(n) \leqslant T(n-1) + T(n-4) + O(p(n))$. This results in a time-complexity of $O^*(1.3803^n)$.

(b) S_j double-hits two subsets S_k, S_l. Note that the double-hit elements must be contained by another set. Note also that (at least) one element, say u_i, is in S_j, S_k and S_l. Consider the two following cases.

i. If $f_i \geqslant 4$, then either we include S_j and then, by Lemma 1, we can exclude S_k and S_l, or we exclude S_j. Then, either we fix $3+3\alpha$ (3 for S_j, S_k, S_l, and 3α since u_i, u_j and u_k belong to at least one other set) or 1.

ii. If $f_i = 3$, then we must include at least one set among S_j, S_k, S_l, but we can suppose that we do not include two such sets. In other words, we have a branching on the three following choices:
- taking S_j (and not S_k, S_l),
- taking S_k (and not S_j, S_l),
- taking S_l (and not S_j, S_k).

In any case, we fix $3+2\alpha$ (2α since each element has a frequency at least 3)

In the first case, $T(n) \leqslant T(n-1) + T(n-3-3\alpha) + O(p(n))$. This results in a time-complexity of $O^*(1.388^n)$. In the second case, $T(n) \leqslant 3T(n-3-2\alpha)+O(p(n))$, and this results in a time-complexity of $O^*(1.358^n)$.

(c) S_j contains elements u_i, u_k, u_l and double-hits one subset S_k on elements u_i, u_k. The following exhaustive subcases must be considered.

i. $f_i = 3$, $f_k \geqslant 3$, $f_l \geqslant 3$, with u_i contained by S_j, S_k, S_m, u_k contained at least by S_j, S_k, S_p and u_l contained at least by S_j, S_q, S_r. A composite branching can be devised.

- Suppose that S_j is included in $\mathcal{S}'$ and then S_k is excluded from $\mathcal{S}'$. In this case, we fix $2+4\alpha$ (α from reduction of the sizes of S_m, S_p, S_q, S_r).
- Suppose that S_j is excluded from $\mathcal{S}'$ and S_k is included in $\mathcal{S}'$. In this case, we fix $2+4\alpha$ (since no other double hit occurs on S_k).
- Suppose finally that S_j and S_k are excluded from $\mathcal{S}'$. In this case, we have to include S_m in $\mathcal{S}'$. Since $d_m = 3$, all elements have frequency at least 3, and at most one double crossing occurs on S_m; we can see that S_m hits at least three new sets. Hence, we fix $3+3\alpha$.

ii. $f_i \geqslant 4$, $f_k \geqslant 4$, $f_l \geqslant 3$, with u_i contained at least by S_j, S_k, S_m, S_p, u_k contained at least by S_j, S_k, S_q, S_r and u_l contained at least by S_j, S_u, S_v. Either we include S_j in $\mathcal{S}'$, and then we can exclude S_k from $\mathcal{S}'$ and fix $2+6\alpha$, or we exclude S_j and fix 1.

In case 3(c)i, we get $T(n) \leqslant 2T(n-2-4\alpha) + T(n-3-3\alpha) + O(p(n))$. This results in a time-complexity of $O^*(1.381^n)$. In case 3(c)ii, we get $T(n) \leqslant T(n-1) + T(n-2-6\alpha) + O(p(n))$. This results in a time-complexity of $O^*(1.3957^n)$.

4. $d_j = 3$ and no double-hitting occurs to S_j (nor to any other subset) that contains elements u_i, u_k, u_l. The following subcases occur.

(a) $f_i = 3$, $f_k \geqslant 3$, $f_l \geqslant 3$ with u_i contained by S_j, S_k, S_l, u_k contained by S_j, S_m, S_p and u_l contained at least by S_j, S_q, S_r. A composite branching can be devised:

- if S_j is included in $\mathcal{S}'$, then we fix $1+6\alpha$ new sets;
- if S_j is excluded from $\mathcal{S}'$ and S_k is included in $\mathcal{S}'$, then there exist at least five other subsets hitting S_k and hence we fix $2+5\alpha$;
- finally, if S_j, S_k are excluded from $\mathcal{S}'$, then we have to include S_l in $\mathcal{S}'$ (in order to cover u_i); there exist at least four other subsets hitting S_l and hence we fix $3+4\alpha$.

Thus, $T(n) \leqslant T(n-1-6\alpha) + T(n-2-5\alpha) + T(n-3-4\alpha) + O(p(n))$, resulting in a time-complexity of $O^*(1.378^n)$.

(b) $f_i \geqslant 4$, $f_k \geqslant 4$, $f_l \geqslant 4$, u_i is contained by S_j, S_k, S_l, S_m, u_k is contained by S_j, S_p, S_q, S_r and u_l is contained at least by S_j, S_t, S_u, S_v. A composite branching on S_j can be devised:

- if S_j is excluded from $\mathcal{S}'$, then we fix 1;
- if S_j is included in $\mathcal{S}'$, then S_k, S_l, S_m are excluded from $\mathcal{S}'$; in this case we fix $4+6\alpha$;
- finally, if S_j is included in $\mathcal{S}'$, then $S_p, S_q, S_r, S_t, S_u, S_w$ are excluded from $\mathcal{S}'$; in this case we fix $7+3\alpha$.

Hence, $T(n) \leqslant T(n-1) + T(n-4-6\alpha) + T(n-7-3\alpha) + O(p(n))$. This results in a time-complexity of $O^*(1.355^n)$.

Putting things together, the global worst case complexity is $O^*(1.3957^n)$ and the proof of the theorem is complete.

As a last word, let us note that a straightforward (improvable) analysis along the lines of Theorem 1, leads to an $O^*(1.1679^n)$ time bound for minimum vertex covering in graphs of maximum size 3. Such a bound is the best-known dealing with search tree-based algorithms and is only dominated by the bounds in [1, 3], ($O^*(1.1252^n)$ and $O^*(1.152^n)$, respectively) that are not based upon such algorithms. Note also, dealing with minimum dominating set in graphs of maximum size 3, analysis along the same lines reaches $O^*(1.344^n)$, which is always the best-known search-tree complexity.

References

[1] R. Beigel. Finding maximum independent sets in sparse and general graphs. In *Proc. Symposium on Discrete Algorithms, SODA'99*, pages 856–857, 1999.

[2] C. Berge. *Graphs and hypergraphs.* North Holland, Amsterdam, 1973.

[3] J. Chen, L Liu, and W. Jia. Improvement on vertex cover for low-degree graphs. *Networks*, 35:253–259, 2000.

[4] F. Della Croce and V. Th. Paschos. Computing optimal solutions for the MIN 3-SET COVERING problem. In X. Deng and D. Du, editors, *Proc. International Symposium on Algorithms and Computation, ISAAC'05*, volume 3827 of *Lecture Notes in Computer Science*, pages 685–692. Springer-Verlag, 2005.

[5] D. Eppstein. Improved algorithms for 3-coloring, 3-edge-coloring, and constraint satisfaction. In *Proc. Symposium on Discrete Algorithms, SODA'01*, pages 329–337, 2001.

[6] F. V. Fomin, F. Grandoni, and D. Kratsch. Measure and conquer: domination – a case study. Reports in Informatics 294, Department of Informatics, University of Bergen, 2005. To appear in the Proceedings of ICALP'05.

[7] M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness.* W. H. Freeman, San Francisco, 1979.

[8] F. Grandoni. A note on the complexity of minimum dominating set. *J. Discr. Algorithms*, 2005. To appear.

[9] G. J. Wœginger. Exact algorithms for NP-hard problems: a survey. In M. Juenger, G. Reinelt, and G. Rinaldi, editors, *Combinatorial Optimization - Eureka! You shrink!*, volume 2570 of *Lecture Notes in Computer Science*, pages 185–207. Springer-Verlag, 2003.