RAIRO-Theor. Inf. Appl. 45 (2011) 181–196
DOI: 10.1051/ita/2011006

Available online at:
www.rairo-ita.org

AN IMPROVED DERANDOMIZED APPROXIMATION ALGORITHM FOR THE MAX-CONTROLLED SET PROBLEM *, **

Carlos Martinhon[1] and Fábio Protti[1]

Abstract. A vertex i of a graph $G = (V, E)$ is said to be *controlled* by $M \subseteq V$ if the majority of the elements of the neighborhood of i (including itself) belong to M. The set M is a *monopoly* in G if every vertex $i \in V$ is controlled by M. Given a set $M \subseteq V$ and two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ where $E_1 \subseteq E_2$, the MONOPOLY VERIFICATION PROBLEM (MVP) consists of deciding whether there exists a sandwich graph $G = (V, E)$ (*i.e.*, a graph where $E_1 \subseteq E \subseteq E_2$) such that M is a monopoly in $G = (V, E)$. If the answer to the MVP is No, we then consider the MAX-CONTROLLED SET PROBLEM (MCSP), whose objective is to find a sandwich graph $G = (V, E)$ such that the number of vertices of G controlled by M is maximized. The MVP can be solved in polynomial time; the MCSP, however, is NP-hard. In this work, we present a deterministic polynomial time approximation algorithm for the MCSP with ratio $\frac{1}{2} + \frac{1+\sqrt{n}}{2n-2}$, where $n = |V| > 4$. (The case $n \leq 4$ is solved exactly by considering the parameterized version of the MCSP.) The algorithm is obtained through the use of randomized rounding and derandomization techniques based on the method of conditional expectations. Additionally, we show how to improve this ratio if good estimates of expectation are obtained in advance.

Mathematics Subject Classification. 68W20, 68W25.

Keywords and phrases. Derandomization, Monte Carlo method, Randomized rounding, sandwich problems.

* *A preliminary version of this paper appeared in the Proc. of the Third International Workshop on Experimental and Efficient Algorithms, WEA2004, Angra dos Reis-RJ, Brazil; Lect. Notes Comput Sci.* **3059** *(2004) 341–355.*

** *This work was partially supported by CNPq and FAPERJ (E-26/110.552/2010), Brazilian Research Agencies.*

[1] Fluminense Federal University, Institute of Computing, Rua Passo da Pátria 156, Bloco E, 24210-230, Niterói, RJ, Brazil; mart@dcc.ic.uff.br; fabio@ic.uff.br

Article published by EDP Sciences

1. Preliminaries

Given two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ such that $E_1 \subseteq E_2$, we say that $G = (V, E)$,where $E_1 \subseteq E \subseteq E_2$, is a *sandwich graph* for some property Π if $G = (V, E)$ satisfies Π. A *sandwich problem* consists of deciding whether there exists some sandwich graph satisfying Π. Many different properties may be considered in this context. In general, the property Π is non-hereditary by (not induced) subgraphs (otherwise G_1 would trivially be a solution, if any) and non-ancestral by supergraphs (otherwise G_2 would trivially be a solution, if any.) As discussed by Golumbic *et al.* [9], sandwich problems generalize recognition problems arising in various situations (when $G_1 = G_2$, the sandwich problem becomes simply a recognition problem).

One of the most known sandwich problems is the CHORDAL SANDWICH PROBLEM, where we require G to be a chordal graph (a graph where every cycle of length at least four possesses a chord – an edge linking two non-consecutive vertices in the cycle). The CHORDAL SANDWICH PROBLEM is closely related to the MINIMUM FILL-IN PROBLEM [22]: given a graph G, find the minimum number of edges to be added to G so that the resulting graph is chordal. The MINIMUM FILL-IN PROBLEM has applications to areas such as solution of sparse systems of linear equations [18]. Another important sandwich problem is the INTERVAL SANDWICH PROBLEM, where we require the sandwich graph G to be an interval graph (a graph whose vertices are in a one-to-one correspondence with intervals on the real line in such a way that there exists an edge between two vertices if and only if the corresponding intervals intersect). Kaplan and Shamir [10] describe applications to DNA physical mapping *via* the INTERVAL SANDWICH PROBLEM. In this work we consider a special kind of sandwich problem, the MAX-CONTROLLED SET PROBLEM (MCSP) [14], which is described in the sequel.

Given an undirected graph $G = (V, E)$ and a set of vertices $M \subseteq V$, a vertex $i \in V$ is said to be *controlled* by M if $|N_G[i] \cap M| \geq |N_G[i]|/2$, where $N_G[i] = \{i\} \cup \{j \in V | (i, j) \in E\}$. The set M defines a *monopoly* in G if every vertex $i \in V$ is controlled by M. Following the notation of [14], if $cont(G, M)$ denotes the set of vertices controlled by M in G, M will be a monopoly in G if and only if $cont(G, M) = V$.

In order to defined formally the MCSP, we first define the MONOPOLY VERIFICATION PROBLEM (MVP): given a set $M \subseteq V$ and two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, where $E_1 \subseteq E_2$, the question is to decide whether there exists a set E such that $E_1 \subseteq E \subseteq E_2$ and M is a monopoly in $G = (V, E)$. If the answer of the MVP applied to M, G_1, and G_2 is No, we then consider the MCSP, whose goal is to find a set E such that $E_1 \subseteq E \subseteq E_2$ and the number of vertices controlled by M in $G = (V, E)$ is maximized.

The MVP can be solved in polynomial time by formulating it as a network flow problem [14]. If the answer to the MVP is No, then a natural alternative is to solve the MCSP. Unfortunately, the MCSP is NP-hard, even for those instances where G_1 is an empty graph; and also for those instances where G_2 is a complete graph.

In [14] a reduction from INDEPENDENT SET to the MCSP is given. In the same work, an approximation algorithm for the MCSP with ratio $\frac{1}{2}$ is presented.

The notion of monopoly has applications to local majority voting in distributed environments and agreement in agent systems [2,7,13,16,19,20]. For instance, suppose that the agents must agree on one industrial standard between two proposed candidate standards. Suppose also that the candidate standard supported by the majority of the agents is to be selected. When every agent knows the opinion of his neighbors, a natural heuristic to obtain a reasonable agreement is: every agent i takes the majority opinions in $N[i]$. This is known as the deterministic local majority polling system. In such a system, securing the support by the members of a monopoly M implies securing unanimous agreement. In this context, the motivation for the MCSP is to find an efficient way of controlling the maximum number of objects by modifying the system's topology.

In this work, we present a linear integer programming formulation and a randomized rounding procedure for the MCSP. As far as we know, our procedure achieves the best polynomial time approximation ratio for the MCSP. If y^* denotes the optimum value of the linear relaxation and $y^* > A(k)$ (for some fixed $k \in (1, 2]$ and some function $A(k) > 4$), the approximation ratio $\frac{1}{k} + \frac{1+\sqrt{y^*}}{k(y^*-1)}$ improves the $\frac{1}{2}$-approximation algorithm presented in [14]. As described later, the case $y^* \leq A(k)$ may be solved exactly by considering a polynomial time algorithm for the parameterized version of the MCSP. This procedure is based on the ideas presented in [14] for the MVP.

This paper is organized as follows. In Section 2 some basic notation and results from [14] are presented. These are fundamental for the development of our algorithm. In Section 3, we introduce the PARAMETERIZED MCSP. For a given parameter $A \geq 0$, we solve exactly the PARAMETERIZED MCSP in time $O(n^A)$. Section 4 gives a detailed description of our MCSP formulation and outlines our randomized rounding procedure. In randomized rounding techniques, we first solve the linear relaxation and "round" the resulting solution to produce feasible solutions. In Section 5 we present an approximation analysis *via* the probabilistic method (introduced by Erdös and Spencer [6]). In this case, the main objective is to construct probabilistic existence proofs of some particular combinatorial structure for actually exhibiting this structure. This is performed through the use of derandomization techniques. In Section 6 we describe a derandomized procedure *via* the method of conditional expectations, achieving an improved deterministic approximation algorithm for the MCSP with performance ratio $\frac{1}{2} + \frac{1+\sqrt{n}}{2n-2}$. Finally, in Section 7, we present some conclusions and suggestions for future work.

2. THE $\frac{1}{2}$-APPROXIMATION ALGORITHM FOR THE MCSP

Consider a maximization optimization problem P and an arbitrary input instance I of P. Denote by $\hat{z}(I)$ the optimal objective function value for I, and by $z_H(I)$ the value of the objective function delivered by an algorithm H. Without loss of generality, it is assumed that each feasible solution for I has a non-negative objective function value. Recall that H is a φ-approximation algorithm for some $\varphi \in (0,1]$, if and only if, a feasible solution with value $z_H(I) \geq \varphi \hat{z}(I)$ is delivered for all instances I of P.

From now on, we suppose that the answer for the MVP when applied to M, G_1 and G_2 is No. Let us briefly describe the deterministic $\frac{1}{2}$-approximation algorithm for the MCSP presented in [14]. For $A, B \subseteq V$, define the edge set $D(A,B) = \{(i,j) \in E_2 \backslash E_1 \mid i \in A, j \in B\}$. Let $U = V \backslash M$. Two reduction rules are used: a new edge set E_1^* is obtained by the union of E_1 and $D(M,M)$, and a new edge set E_2^* is obtained by removing $D(U,U)$ from E_2. Since we are maximizing the total number of vertices controlled by M, these reduction rules do not modify the optimal solution. In other words, the edge set E in the sandwich graph G satisfies $E_1 \cup D(M,M) \subseteq E \subseteq E_1 \cup D(M,M) \cup D(U,M)$.

For simplicity, assume from now on that $E_1 = E_1^*$ (Reduction Rule 1) and $E_2 = E_2^*$ (Reduction Rule 2). In the MYK algorithm, $W1, W2 \subseteq V$ denote the sets of vertices controlled by M in $G = (V,E)$ for $E = E_1$ and $E = E_2$, respectively.

```
Algorithm 1: MYK ALGORITHM [14]
1. compute $|W_1|$ by removing from $G_2$ the edge set $D(U,M)$;
2. compute $|W_2|$ by adding to $G_1$ the edge set $D(U,M)$;
3. $z_H \leftarrow \max\{|W_1|, |W_2|\}$.
```

Formally, they proved the following result:

Theorem 2.1. *The value z_H returned by Algorithm 1 satisfies $z_H(I) \geq \frac{1}{2}\hat{z}(I)$, for all instances I of the* MCSP.

3. Parameterizing the MCSP

In this section we introduce the PARAMETERIZED MCSP. Let A be a fixed nonnegative integer. The objective is to find, in polynomial time, a solution for the MCSP with value at least A. In other words, we require the parameter A to be a lower bound for the maximum number of vertices that can be controlled by M in a sandwich graph $G = (V,E)$ with $E_1 \subseteq E \subseteq E_2$.

Let us describe an $O(n^A)$ algorithm for the PARAMETERIZED MCSP. We first consider a partition of V into six special subsets (some of them are implicitly described in [14]):

- M_C and U_C, consisting of the vertices in M and U, respectively, which are controlled by M in any sandwich graph (vertices which are "always controlled");
- M_N and U_N, consisting of the vertices in M and U, respectively, which are not controlled by M in any sandwich graph (vertices which are "never controlled");
- M_R and U_R, defined as $M_R = M \backslash (M_C \cup M_N)$ and $U_R = U \backslash (U_C \cup U_N)$.

Define the binary variables $x_{ij} \in \{0,1\}$ for $i, j \in V$ and assume that $x_{ij} = x_{ji}$, $\forall i, j \in V$. Define binary constants $a_{ij} \in \{0,1\}$ such that $a_{ij} = 1$ if and only if $i = j$ or $(i,j) \in E_2$. Consider now the following auxiliary equations:

$$B_i = \sum_{j \in M} a_{ij} x_{ij} \quad - \sum_{j \in U} a_{ij} x_{ij}, \quad \text{for} \quad i = 1, \ldots, |V|. \tag{1}$$

In these equations, assume that: $x_{ii} = 1$ for every $i \in V$, $x_{ij} = 1$ for every $(i,j) \in E_1$, and $x_{ij} = 0$ for every $(i,j) \notin E_2$. This means that the remaining binary variables are associated to edges in $E_2 \backslash E_1$ (set of optional edges). It is clear that $B_i \geq 0$ for some 0-1 assignment (to variables associated to optional edges) if and only if vertex i can be controlled by M in some sandwich graph. Observe that the subsets M_C, U_C, M_N, U_N can be thus characterized by the following properties:

- $i \in M_C \cup U_C$ if and only if $B_i \geq 0$ for every 0-1 assignment;
- $i \in M_N \cup U_N$ if and only if $B_i < 0$ for every 0-1 assignment.

In fact, it is easy to construct these four sets, since it is sufficient to look at "worst-case" assignments. For instance, if $i \in M$ then $i \in M_C$ if and only if $B_i \geq 0$ for a 0-1 assignment which sets $x_{ij} = 1$ for every $(i,j) \in D(\{i\}, U)$.

Use now the following new reduction rules:

- set $x_{ij} = 1$ for every $(i,j) \in D(M_C \cup M_N, U_R)$ (Reduction Rule 3);
- set $x_{ij} = 0$ for every $(i,j) \in D(M_R, U_C \cup U_N)$ (Reduction Rule 4);
- set arbitrary values to variables x_{ij} for $(i,j) \in D(M_C \cup M_N, U_C \cup U_N)$ (Reduction Rule 5).

It is clear that if $A \leq |M_C| + |U_C|$ then the algorithm for the PARAMETERIZED MCSP answers Yes. Hence, from now on, assume that $A > |M_C| + |U_C|$.

Clearly, $M_R = \emptyset$ if and only if $U_R = \emptyset$. Thus, if $M_R = \emptyset$ or $U_R = \emptyset$, the algorithm must answer No. Add then the assumption $M_R, U_R \neq \emptyset$.

We fix an arbitrary subset $S \subseteq M_R \cup U_R$ of cardinality $|S| = A - |M_C| - |U_C|$, and check whether it is possible to control S. Similarly to the reduction rules described above, set $x_{ij} = 1$ for every edge $(i,j) \in D(M_R \backslash S, U_R \cap S)$, and $x_{ij} = 0$ for every edge $(i,j) \in D(M_R \cap S, U_R \backslash S)$. Finally, set $x_{ij} = 0$ for every edge $(i,j) \in D(M_R \cap S, U_R \cap S)$, and calculate the corresponding B_i's according to equations (1).

Following the ideas in [14], we construct a network $\mathcal{N}$ whose vertex set consists of S together with two additional vertices s, t. Create an edge (s,i) with capacity B_i for every $i \in S \cap M_R$, an edge (j,t) with capacity $B'_j = \max\{-B_j, 0\}$ for every $j \in S \cap U_R$, and an edge (i,j) with capacity 1 for every $(i,j) \in D(S \cap M_R, S \cap U_R)$. Notice that $\mathcal{N}$ can be constructed in constant time, since its edge set contains at most $O(A^2)$ elements. Now, if the maximum flow in $\mathcal{N}$ is equal to $\sum_{j \in S \cap U_R} B'_j$ then S can be controlled by selecting the edges of the form $(i,j) \in D(S \cap M_R, S \cap U_R)$ with unitary flow value. This maximum flow problem can be solved in constant time, depending on A.

By repeating this procedure for every $S \subseteq M_R \cup U_R$ such that $|S| = A - |M_C| - |U_C|$, we obtain an algorithm with complexity $O(n^A)$.

4. An improved randomized rounding procedure for the mcsp

The definition of performance ratio in randomized approximation algorithms is the same as in the deterministic ones. In this case, however, $z_H(I)$ is replaced by $E(z_H(I))$, where the expectation is taken over the random choices made by the algorithm. Then, an algorithm H for a maximization problem P is a randomized φ-approximation algorithm for some $\varphi \in (0,1]$ if and only if $E(z_H(I)) \geq \varphi\hat{z}(I)$ is delivered for all instances I of P.

In randomized rounding techniques (introduced by Raghavan and Thompson [17]), one usually solves a relaxation of a combinatorial optimization problem (by using linear or semidefinite programming), and uses randomization to return from the relaxation to the original optimization problem. The main idea is to use fractional solutions to define tuned probabilities in the randomized rounding procedure.

In order to introduce a new integer programming formulation for the mcsp, we define the binary variables z_i for $i \in V$, which determine whether vertex i is controlled or not by M. Binary variables x_{ij} are used to decide whether optional edges belonging to $E_2 \backslash E_1$ will be included or not in the sandwich graph. The objective function (2) computes the maximum number of controlled vertices. As defined before, binary constants $a_{ij} \in \{0,1\}$ are associated to edges $(i,j) \in E_2$ with $a_{ij} = 1$ if and only if $i = j$ or $(i,j) \in E_2$. (Assume that $a_{ij} = a_{ji}, \forall i,j \in V$.) Inequalities (3) guarantee that every time a vertex i is controlled by M, the left hand side will be greater than or equal to 1. On the other hand, if the left hand side is less than 1, vertex i will not be controlled by M and z_i will be set to 0. The divisions by n are used to maintain the difference between the two summations always greater than -1. Equalities (4) define the set of fixed edges. The linear programming relaxation is obtained by replacing integrality constraints (5) and (6) by $x_{ij} \in [0,1]$ and $z_i \in [0,1]$, respectively:

$$\hat{z} = \max \sum_{i \in V} z_i \tag{2}$$

subject to :

$$\sum_{j \in M} (a_{ij}/n)x_{ij} \ - \sum_{j \in V} (a_{ij}/2n)x_{ij} + 1 \geq z_i, \forall i \in V \tag{3}$$

$$x_{ij} = 1, \forall (i,j) \in E_1 \tag{4}$$

$$x_{ij} \in \{0,1\}, \forall (i,j) \in E_2 \backslash E_1 \tag{5}$$

$$z_i \in \{0,1\}, \forall i \in V. \tag{6}$$

It is assumed from now on that (x^*, z^*) and y^* will denote, respectively, an optimal solution of the relaxed integer programming formulation and its associated objective function value. The value of the original integer problem will be denoted by $\hat{z}$.

The value of linear programming relaxation may be reduced if the reduction rules for the MCSP are used. As will be observed in Section 5, the performance ratio of our randomized algorithm is based on the value of the linear relaxation and an improvement of this ratio is attained if good upper bounds are obtained. Thus, assume without loss of generality that MCSP instances satisfy Reduction Rules 1 to 5.

Algorithm 2, based on randomized rounding techniques, is a Monte Carlo procedure and delivers, in polynomial time and with high probability, a value within a prescribed approximation ratio. In Step 3 of the algorithm we define a function $A(k)$ for a given parameter $k \in (1, 2]$ conveniently chosen. The construction of $A(k)$ will be detailed in the next section. For the time being, an "oracle" is used.

```
Algorithm 2: RANDOMIZED ALGORITHM FOR THE MCSP
1. compute $z_1$ using Algorithm 1;
2. solve the linear programming relaxation and return $x^*$ and $y^*$;
3. compute $A(k)$ for some $k \in (1, 2]$ (conveniently chosen);
4. if $y^* \leq A(k)$

     then compute $z_H$ by executing the algorithm for the PARAMETERIZED
          MCSP in Section 3 for parameters $A = \lfloor y^* \rfloor, \lfloor y^* \rfloor - 1, \ldots, 1, 0$, until
          obtaining a Yes answer
     else for each $(i, j) \in E_2 \backslash E_1$ do
               $Pr(\bar{x}_{ij} = 1) = x^*_{ij}$   (constructing the integer feasible solution)
               $Pr(\bar{x}_{ij} = 0) = 1 - x^*_{ij}$;
          compute $z_2$ by using the integer feasible function $\bar{x}$;
          $z_H \leftarrow \max\{z_1, z_2\}$;

5. return $z_H$.
```

We can use, for example, an interior point method in Step 2 (introduced by Karmarkar [11]) to compute the fractional solution x^*, yielding in this way a polynomial time execution for Algorithm 2. Observe that Algorithm 2 always produce a feasible solution, and additional executions of Step 4 (for $y^* > A(k)$) arbitrarily reduce the failure probability, provided that a prescribed approximation ratio is given. Moreover, it is obviously a $\frac{1}{2}$-approximation algorithm since Algorithm 1 was used in Step 1. As will be pointed out in the next section, this will directly help us to build an improved approximation algorithm with ratio $\frac{1}{k} + \frac{1+\sqrt{y^*}}{k(y^*-1)}$ for some $k \in (1, 2]$, conveniently chosen. It is straightforward to observe that, even for $k = 2$, this ratio is strictly greater than $\frac{1}{2}$, thus improving the previous result of [14]. In addition, recall that all those instances with $y^* \leq A(k)$ (for some parameter $A = \lfloor y^* \rfloor$) are polynomially solved by the algorithm for the PARAMETERIZED MCSP given in Section 3.

5. Approximation analysis

Before to proceed to the approximation analysis, consider the following auxiliary definitions and lemmas. We first present the notion of negative association.

Definition 5.1 (negative association)**.** Let $X = (X_1, X_2, \dots, X_n)$ be a vector of random variables. The random variables X are negatively associated if for every two disjoint index sets $I, J \subseteq \{1, 2, \dots, n\}$, $E(f(X_i, i \in I)g(X_j, j \in J)) \leq E(f(X_i, i \in I))E(g(X_j, j \in J))$ for all functions $f : \Re^{|I|} \to \Re$ and $g : \Re^{|J|} \to \Re$ that are either both non-decreasing or both non-increasing.

For a more detailed study concerning negative dependence see Dubhashi and Ranjan [5].

The next lemma ensures that the lower Chernoff-Hoeffding bound (lower CH bound) may be applied to not necessarily independent random variables. See Motwani and Raghavan [15] and Dubhashi and Ranjan [5] for the proof. An analogous result may be established for the upper CH bound.

Lemma 5.1 (lower Chernoff-Hoeffding bound and negative association)**.** *Let $X_1, X_2, \dots, X_n$ be negatively associated Poisson trials such that, for $1 \leq i \leq n$, $Pr(X_i = 1) = p_i$, where $0 < p_i < 1$. Then, for $X = \sum_{i=1}^{n} X_i$, $\mu = E(X) = \sum_{i=1}^{n} p_i$, and any $0 < \delta \leq 1$, we have that $Pr(X < (1-\delta)\mu) < \exp(-\mu\delta^2/2)$.*

Finally, consider the following auxiliary lemma:

Lemma 5.2. *Let X, Y be arbitrary random variables. Then $E(\min(X,Y)) \leq \min(E(X), E(Y))$.*

Now, in order to describe the approximation analysis of Algorithm 2, we define random variables $Z_i \in \{0,1\}$ for every $i \in V$. These variables denote the set of vertices controlled by M. We also define random variables $X_{ij} \in \{0,1\}$ for every $i, j \in V$. Assume $X_{ii} = 1$ for every $i \in V$, and $X_{ij} = 1$ for every $(i,j) \in E_1$. Observe that variables X_{ij} for $(i,j) \in E_2 \backslash E_1$ are associated to the set of optional edges. Additionally, let Z_H be the sum of not necessarily independent random variables $Z_i \in \{0,1\}$ for $i \in V$. Thus, we have the following preliminary result:

Lemma 5.3. *The random variables Z_i for all $i \in V$ are negatively associated.*

Proof. Consider two arbitrary disjoint index sets $I, J \subseteq \{1, 2, \dots, n\}$. Then we want to show that:

$$E\left(\sum_{i \in I} Z_i \sum_{j \in J} Z_j\right) - E\left(\sum_{i \in I} Z_i\right) E\left(\sum_{j \in J} Z_j\right) = \sum_{i \in I, j \in J} (E(Z_i Z_j) - E(Z_i)E(Z_j)) \leq 0.$$

In particular, it is easy to observe (from the definition of the MCSP) that Z_i and Z_j (for $i \neq j$) are independent random variables if they simultaneously belong to M (or U). However, Z_i and Z_j are negatively associated if they are not in the same set. Generally, for arbitrary index sets I and J, we can establish that $Pr(Z_j = 1 \mid Z_i = 1) \leq Pr(Z_j = 1)$ or, equivalently, $Pr(Z_i = 1 \mid Z_j = 1) \leq Pr(Z_i = 1)$. Thus, for every pair $i \in I$ and $j \in J$, we have that $E(Z_iZ_j) = Pr(Z_iZ_j = 1) = Pr(Z_i = 1)Pr(Z_j = 1 \mid Z_i = 1) \leq Pr(Z_i = 1)Pr(Z_j = 1) = E(Z_i)E(Z_j)$, which proves the lemma. □

Now, consider our relaxed integer programming formulation. For $i, j \in V$, assume $X_{ij} = 1$ if $i = j$ or $(i, j) \in E_1$. Assume also we assign, as described in Algorithm 2, arbitrarily values X_{ij} for every $(i, j) \in E_2 \backslash E_1$. If Z_H is the sum of random variables denoting the value of the randomized solution, it follows from constraints (3)–(6) that:

$$Z_H = \sum_{i \in V} Z_i \leq \sum_{i \in V} \min \left\{ 1; \sum_{j \in M} (a_{ij}/n) X_{ij} - \sum_{j \in V} (a_{ij}/2n) X_{ij} + 1 \right\}. \qquad (7)$$

From Lemma 5.2 and the linearity of expectation one obtains:

$$E(Z_H) = \sum_{i \in V} E(Z_i) \leq \min \left\{ 1; \sum_{j \in M} (a_{ij}/n) x^*_{ij} - \sum_{j \in V} (a_{ij}/2n) x^*_{ij} + 1 \right\},$$

where $E(X_{ij}) = x^*_{ij}$. Therefore:

$$E(Z_H) \leq \sum_{i \in V} \min\{1; z^*_i\} \Rightarrow E(Z_H) \leq \sum_{i \in V} z^*_i = y^* \text{ for } z^*_i \in [0, 1]. \qquad (8)$$

Recall that Step 1 Algorithm 2 guarantees a performance ratio equal to $\frac{1}{2}$. Therefore, each iteration of Algorithm 2 returns a solution with $Z_H \geq \hat{z}/2$ (where $\hat{z}$ denotes the value of the optimal integer solution). Now, as the optimal solution itself may be generated at random, one may concludes, without loss of generality, that $E(Z_H)$ is strictly greater than $\hat{z}/2$ (otherwise, the solution generated by Algorithm 1 would be optimal). Thus, we assume from (8) that $\hat{z}/2 < E(Z_H) \leq y^*$, where $E(Z_H) = \mu = y^*/\beta$ for some $\beta \in [1, 2)$.

Now, for some $\alpha > 1$, to be considered later, define a bad event $B \equiv (Z_H < y^*/\alpha)$. Equivalently to the definition of a randomized approximation algorithm (described in the preceding section), Z_H defines an $\frac{1}{\alpha}$-approximation solution for the MCSP if $B^c \neq \emptyset$ holds (complementary event).

How small a value for α can we achieve while guaranteeing good events $B^c \neq \emptyset$? Since we expect to obtain an approximation algorithm with a superior performance ratio (greater than $\frac{1}{2}$), it suffices to consider $\alpha \in (\beta, k)$ for some $k \in (\beta, 2]$. The parameter k will be fixed later. This give us an improved $\frac{1}{\alpha}$-approximation Z_H with nonzero probability. As discussed later, this solution will be made deterministic through derandomization techniques, namely, the method of conditional expectations.

Therefore, a bad event B occurs if $Z_H < y^*/\alpha$. Then:

$$Pr(B) = Pr\left(Z_H < \frac{y^*}{\alpha}\right) = Pr\left(Z_H < \frac{\beta\mu}{\alpha}\right) = Pr\left(Z_H < (1-\delta)\mu\right),$$

where $\delta = 1 - \frac{\beta}{\alpha} > 0$.

In order to apply the lower CH bound, in addition to the negative association (Lem. 5.3), all random variables must assume values in the interval (0,1). In our case, however, as observed in Section 3, $Pr(Z_i = 1) = 1$ for every $i \in M_C \cup U_C$ (set of vertices which are always controlled by M) and $Pr(Z_i = 1) = 0$ for every $i \in M_N \cup U_N$ (set of vertices which are never controlled by M). Despite of that, CH bounds may be applied, since the linear programming relaxation is being solved by some interior point method (see Wright [21]).

Therefore, from the lower CH bound and assuming $\mu > \hat{z}/2$, it follows that:

$$Pr(B) < \frac{1}{\exp\left((1-\beta/\alpha)^2\ \frac{\mu}{2}\right)} < \frac{1}{\exp\left((1-\beta/\alpha)^2\ \frac{\hat{z}}{4}\right)}. \tag{9}$$

This implies:

$$Pr(B) < \frac{1}{\exp((1-\beta/\alpha)^2\ \frac{\hat{z}}{4} - \frac{1}{4})}. \tag{10}$$

We expect that $Pr(B) < 1$ (probability of bad event). Thus, if we impose this last condition, it follows from (10) that:

$$Pr(B) = \frac{1}{\exp((1-\beta/\alpha)^2\ \frac{\hat{z}}{4} - \frac{1}{4})} < 1 \text{ for some } \alpha \in (\beta, k). \tag{11}$$

Additional executions of Step 4 in Algorithm 2 for $y > A(k)$ arbitrarily reduce the failure probability (Monte Carlo method). Therefore, without loss of generality, if $Pr(B) = C < 1$ is the probability of a bad event, and $\xi > 0$ is a given error, $\lceil \mid \log\xi / \log C \mid \rceil$ iterations are sufficient to ensure a $\frac{1}{\alpha}$-approximation algorithm with probability $1 - \xi > 0$.

Then, we need to determine if there is some value $\alpha \in (\beta, k)$ (where $\beta \geq 1$ and $k \leq 2$) for which inequality (11) makes sense. Equivalently, we expect to obtain $(\hat{z}-1)\alpha^2 - (2\hat{z}\beta)\alpha + \beta^2\hat{z} > 0$ for some $\alpha \in (\beta, k)$. By solving the quadratic equation, we obtain the roots:

$$\alpha' = \frac{\beta(\hat{z} - \sqrt{\hat{z}})}{\hat{z} - 1} \text{ and } \alpha'' = \frac{\beta(\hat{z} + \sqrt{\hat{z}})}{\hat{z} - 1}.$$

Since $\alpha > \beta$, it is easy to observe that inequality $(\hat{z} \pm \sqrt{\hat{z}})/(\hat{z} - 1) > 1$ holds only for α'' with $\hat{z} > 1$. In addition, we expect that $\alpha'' < k$ for some $k \in (\beta, 2]$. Thus, since $\beta = y^*/\mu \geq 1$, it follows that:

$$\alpha'' = \frac{y^*(\hat{z} + \sqrt{\hat{z}})}{\mu(\hat{z} - 1)} < k \Rightarrow \frac{y^*(\hat{z} + \sqrt{\hat{z}})}{k(\hat{z} - 1)} < \mu \leq y^*. \tag{12}$$

Therefore:

$$\frac{y^*(\hat{z}+\sqrt{\hat{z}})}{k(\hat{z}-1)} < y^* \Rightarrow \hat{z}+\sqrt{\hat{z}} < k(\hat{z}-1). \tag{13}$$

Now, inequality (13) holds only for $\hat{z} > A(k)$ with:

$$A(k) = \frac{2k(k-1)+1+\sqrt{4k(k-1)+1}}{2(k-1)^2}.$$

Notice that constraint $\hat{z} > 1$ above is immediately verified since we have $A(k) \geq 4$ for every $k \in (1, 2]$. Finally, from expression (12), since $E(Z_H) = \mu$, $\hat{z} \leq y^*$ and $y^* > A(k)$, it follows that:

$$E(Z_H) > \left(\frac{\hat{z}+\sqrt{\hat{z}}}{\hat{z}-1}\right)\left(\frac{y^*}{k}\right) > \left(\frac{y^*+\sqrt{y^*}}{k(y^*-1)}\right)\hat{z} = \left(\frac{1}{k}+\frac{1+\sqrt{y^*}}{k(y^*-1)}\right)\hat{z}. \tag{14}$$

Moreover, observe from the above expression that:

$$\frac{1+\sqrt{y^*}}{k(y^*-1)} > 0 \quad \text{and} \quad \lim_{y^*\to\infty}\frac{1+\sqrt{y^*}}{k(y^*-1)} = 0 \qquad \text{for every } y^* > A(k).$$

Thus, inequality (14) gives us a randomized $\frac{1}{k}+\frac{1+\sqrt{y^*}}{k(y^*-1)}$-approximation algorithm for every $y^* > A(k)$ and $k \in (\beta, 2]$. Therefore, with high probability and for a large class of instances, this ratio improves the $\frac{1}{2}$-approximation algorithm in [14]. The case $y^* \leq A(k)$ may be solved exactly in time $O(n^{A(k)})$ through the algorithm for the PARAMETERIZED MCSP in Section 3. Observe for instance that $A(k_1) > A(k_2)$ for every $k_1, k_2 \in (\beta, 2]$ with $k_1 < k_2$. In other words, despite the increase in the computational time of the algorithm for the PARAMETERIZED MCSP, small values of k (for $k > \beta$) guarantee improved approximation ratios for every $y^* > A(k)$. Formally, we proved the following result:

Theorem 5.1. *Consider y^* and μ as above. Then, for a given parameter $k \in (\beta, 2]$ with $\beta = y^*/\mu$, Algorithm 2 defines a randomized $\frac{1}{k}+\frac{1+\sqrt{y^*}}{k(y^*-1)}$-approximation algorithm for the* MCSP.

Unfortunately, we do not know explicitly the value of $\beta = y^*/\mu$ since the expectation μ is unknown and hard to compute. Moreover, we cannot guarantee a parameter k strictly less than 2. This problem is minimized if some good estimations of $E(Z_H) = \mu$, and thus of β, are obtained. By running independent experiments with respect to Z_H, the recent work of Dagum *et al.* [3] ensures, for a given ξ and ϵ, an estimator μ' of μ within a factor $1+\epsilon$ and probability at least $1-\xi$. Therefore, if this approximation is performed in advance, and if we assume $k = \min\{2, \frac{y^*}{\mu'(1-\epsilon)}\}$, an improved randomized approximation algorithm (for every instance of the MCSP) may be achieved if $k < 2$. Notice for instance that, given an interval (β, k), the proof of Theorem 5.1 guarantees the existence of $\alpha \in (\beta, k)$, thus improving the performance ratio.

6. A derandomized algorithm

Derandomization techniques convert a randomized algorithm into a deterministic one. Here, this is performed through the probabilistic method (introduced by Erdös and Spencer [6]). The main idea is to use the existence proof of some combinatorial structure for actually exhibiting this structure.

The purpose of this section is to derandomize Algorithm 2 by using the method of conditional expectations. In this case, the goal is to convert the expected approximation ratio into a guaranteed approximation ratio while increasing the running time by a factor that is polynomial on the input size. Basically, the method of conditional expectations analyzes the behavior of a randomized approximation algorithm as a computation tree, in a such way that each path from the root to a leaf of this tree corresponds to a possible computation generated by the algorithm.

In order to describe our derandomized procedure for the MCSP, consider inequality (7). Then, it follows that:

$$Z_H = \sum_{i \in V} Z_i \le \sum_{i \in V} Y_i, \quad \text{where} \quad Y_i = \min \left\{ 1, \sum_{j \in M} (a_{ij}/n) X_{ij} - \sum_{j \in V} (a_{ij}/2n) X_{ij} + 1 \right\}.$$

Recall that $X_{ii} = 1$ for every $i \in V$, and $X_{ij} = 1, \forall (i,j) \in E_1$. In addition, suppose that all optional edges in $E_2 \backslash E_1$ are arbitrarily ordered and indexed by $k = 1, \ldots, |E_2 \backslash E_1|$. In this section, the notation $x_{ij}^{(k)} = 1$ has the following meaning: the k-th edge of $E_2 \backslash E_1$, with endpoints i and j, belongs to the sandwich graph $G = (V, E)$. Otherwise, $x_{ij}^{(k)} = 0$ means that $(i,j) \notin E$. For simplicity, we will suppress indexes i and j and simply write $x^{(k)}$. Capital letters $X^{(k)}$ mean that a value 0 or 1 was assigned to variable $x^{(k)}$, for some $k \in \{1, \ldots, |E_2 \backslash E_1|\}$. Furthermore, the notation $E(Z_H \mid x^{(k)} = 0 \text{ or } 1)$ denotes the average value produced by the randomized algorithm by computations that set $x^{(k)} = 0$ or 1.

Thus, from de definition of conditional expectation and from its linearity property one concludes that:

$$\begin{aligned} E(Z_H) &= E(Z_H \mid x^{(1)} = 1) Pr(x^{(1)} = 1) + E(Z_H \mid x^{(1)} = 0) Pr(x^{(1)} = 0) \\ &\le \max \left\{ \sum_{i \in V} E(Z_i \mid x^{(1)} = 1) \; ; \; \sum_{i \in V} E(Z_i \mid x^{(1)} = 0) \right\} = \sum_{i \in V} E(Z_i \mid X^{(1)}) \\ &\le \max \left\{ \sum_{i \in V} E(Z_i \mid X^{(1)}, x^{(2)} = 1) \; ; \; \sum_{i \in V} E(Z_i \mid X^{(1)}, x^{(2)} = 0) \right\} \\ &= \sum_{i \in V} E(Z_i \mid X^{(1)}, X^{(2)}). \end{aligned}$$

By repeating this process for every edge in $E_2\backslash E_1$, one obtains:

$$E(Z_H) \leq \max\left\{\sum_{i\in V} E(Z_i \mid X^{(1)}, \ldots, X^{(|E_2\backslash E_1|)}); \sum_{i\in V} E(Z_i \mid X^{(1)}, \ldots, X^{(|E_2\backslash E_1|)})\right\}.$$

Therefore, within this framework, a guaranteed performance ratio is polynomially attained through an expected approximation ratio, gathering, in this way, an improved deterministic approximation solution.

Now, from the definition of conditional expectation,

$$E(Z_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0 \text{ or } 1) = Pr(Z_i = 1 \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0 \text{ or } 1),$$

for every $i \in V$ and $k = 1, \ldots, |E_2\backslash E_1|$. Unfortunately, for the MCSP, these probabilities are hard to compute. Lemma 6.1 will give us an alternate way to deal with these expectations without explicitly consider conditional probabilities.

Lemma 6.1. *Suppose that Z_i and Y_i, for some $i \in V$, are random variables as described above. Then:*

(a) $E(Z_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 1) \geq E(Z_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0) \Leftrightarrow$
$E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 1) \geq E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0)$

(b) $E(Z_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0) \geq E(Z_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 1) \Leftrightarrow$
$E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0) \geq E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 1)$.

Proof. We will prove item (a), the proof of (b) follows analogously. Consider without loss of generality that $E(Z_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 1) \geq E(Z_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0)$. Thus, since $E(Z_H) = y^*/\beta$ for some $\beta \in [1, 2)$, it follows from inequalities (7)–(8) and from the definition of conditional expectations that:

$$E(Z_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0 \text{ or } 1) = (1/\beta)\ E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0 \text{ or } 1).$$

Thus:

$$(1/\beta)E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 1) \geq (1/\beta)E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0).$$

By multiplying both sides by β, we get the desired inequality

$$E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 1) \geq E(Y_i \mid X^{(1)}, \ldots, X^{(k-1)}, x^{(k)} = 0).$$

The converse is obtained in the same way by first multiplying this last inequality by $\frac{1}{\beta}$. □

Now, from Lemma 6.1, it follows that $E(Z_H)$ is less than or equal to

$$\max\left\{\sum_{i\in V} E(Y_i \mid X^{(1)},\dots,X^{(k-1)},x^{(k)}=1);\ \sum_{i\in V} E(Y_i \mid X^{(1)},\dots,X^{(k-1)},x^{(k)}=0)\right\},$$

for $k=1,\dots,|E_2\backslash E_1|$.

We repeat the process above for every optional edge in $E_2\backslash E_1$. Therefore, the sequence $X^{(1)},\dots,X^{(|E_2\backslash E_1|)}$ is obtained deterministically in polynomial time while improving the approximation ratio.

From the preceding section, we have described a randomized algorithm whose expectation $E(Z_H)$ is greater than or equal to $\frac{1}{k}+\frac{1+\sqrt{y^*}}{k(y^*-1)}$ for some $k\in(\beta,2]$ conveniently chosen. Since we expect to obtain a deterministic procedure, it suffices to consider (in the worst case) $k=2$ and $y^*=\theta(n)$. Observe, from the preceding section, that by setting $k=2$ one obtains $A(2)=4$. This will give us (for an arbitrary instance) an improved deterministic polynomial time approximation algorithm with performance ratio equal to $\frac{1}{2}+\frac{1+\sqrt{n}}{2n-2}$.

Algorithm 3: DERANDOMIZED ALGORITHM FOR THE MCSP

```
1. compute $z_1$ using Algorithm 1;
2. solve the linear programming relaxation and return $y^*$;
3. **if** $y^* \le 4$
     **then** compute $z_H$ by executing the algorithm for the PARAMETERIZED
          MCSP in Section 3 for parameters $A=\lfloor y^*\rfloor, \lfloor y^*\rfloor-1,\dots,1,0$, until
          obtaining a Yes answer
     **else** **for** $k=1,\dots,|E_2\backslash E_1|$ **do**
                **if** $E(Y_k \mid X^{(1)},\dots,X^{(k-1)},x^{(k)}=1) \ge E(Y_k \mid X^{(1)},\dots,X^{(k-1)},x^{(k)}=0)$
                   **then** $X^{(k)} \leftarrow 1$
                   **else** $X^{(k)} \leftarrow 0$;
          compute $z_2$ by using the integer feasible function $X$;
          $z_H \leftarrow \max\{z_1, z_2\}$;
5. return $z_H$.
```

Observe above that expectations $E(Y_k \mid X^{(1)},\dots,X^{(k-1)},x^{(k)}=0 \text{ or } 1)$ are easily obtained. This may be accomplished in polynomial time by solving a linear programming problem for every optional edge (settled 0 or 1). If L denotes the length of the input, the linear relaxation has complexity $O(n^3L)$ [21], and thus the total complexity of Algorithm 3 will be equal to $O(\max\{n^4,|E_2\backslash E_1|n^3L\})$. Moreover, from Theorem 5.1, it is straightforward to observe that an improvement of the approximation ratio may be attained if good upper bounds are obtained *via* the linear relaxation. This may be accomplished, for example, through the use of new reduction rules and/or through the use of additional cutting planes. Notice for instance that, even in the worst case, when $y^*=\theta(n)$, one obtains an improved approximation ratio. Formally, we can establish the following result:

Theorem 6.1. *Algorithm 3 guarantees in polynomial time an approximation ratio equal to* $\frac{1}{2}+\frac{1+\sqrt{n}}{2n-2}$ *for* $n>4$.

7. Conclusions

We presented an improved deterministic polynomial time approximation algorithm for the MAX-CONTROLLED SET PROBLEM through the use of randomized rounding and derandomization techniques. As far as we know, this is the best approximation result for the MCSP. This improves the $\frac{1}{2}$-approximation procedure presented by Makino *et al.* [14]. A new linear integer programming formulation was presented to define tuned probabilities in our randomized procedure. Through the use of the probabilistic method, we converted a probabilistic proof of existence of an approximated solution into an efficient deterministic algorithm for actually constructing this solution. Additionally, we show that if some good estimations of expectation are obtained in advance, some improved approximation ratios may be attained.

As future work, an interesting question is to decide whether the PARAMETERIZED MCSP is *fixed parameter tractable* – FPT. (A problem with parameter A is FPT if it admits an $O(f(A)n^{\gamma})$ time algorithm, for some function f and some constant γ independent of A, see [4].) Another interesting attempt of research is to use different randomized rounding strategies such as the dependent rounding as introduced in Gandhi *et al.* [8]. Their technique allows negative correlation in conjunction with various new ideas to develop improved (approximation) algorithms *via* LP-rounding.

Finally, the question of determine an improved constant ratio (equal to $1/2+\epsilon$ for some $\epsilon \in (0, 1/2]$) for the MCSP remains an open problem. Thus, to better evaluate the performance ratio attained by any approximation algorithm, an additional study concerning negative results is necessary. Therefore, as a future direction, an interesting question is to deal with non-approximation techniques (for instance, through the use of the PCP theorem [1] or the Unique Game Conjecture [12]) applied to the MCSP.

Acknowledgements. We thank Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ) for their partial financial support and the anonymous referees for their insightful comments.

References

[1] S. Arora and S. Safra, Probabilistic checking of proofs: A new characterization of NP. *J. ACM* **45** (1998) 70–122.

[2] J.-C. Bermond and D. Peleg, The power of small coalitions in graphs. *Discrete Appl. Math.* **127** (2003) 399–414.

[3] P. Dagum, R. Karp, M. Luby and S. Ross, An optimal algorithm for Monte Carlo estimation. *SIAM J. Comput.* **29** (2000) 1484–1496.

[4] R.G. Downey and M.R. Fellows, Fixed parameter tractability and completeness I: Basic results. *SIAM J. Comput.* **24** (1995) 873–921.

[5] D. Dubashi and D. Ranjan, Balls and bins: A study of negative dependence. *Random Struct. Algorithms* **13** (1998) 99–124.

[6] P. Erdös and J. Spencer, *The Probabilistic Method in Combinatorics*. Academic Press, San Diego (1974).

[7] D. Fitoussi and M. Tennenholtz, Minimal social laws. *Proc. AAAI'98* (1998) 26–31.

[8] R. Gandhi, S. Khuler, S. Parthasarathy and A. Srinivasan, Dependent rounding and its applications to approximation algorithms. *J. ACM* **53** (2006) 324–360.

[9] M.C. Golumbic, H. Kaplan and R. Shamir, Graph sandwich problems. *J. Algorithms* **19** (1994) 449–473.

[10] H. Kaplan and R. Shamir, Bounded degree interval sandwich problems. *Algorithmica* **24** (1999) 96–104.

[11] N. Karmarkar, A new polynomial time algorithm for linear programming. *Combinatorica* **4** (1984) 375–395.

[12] S. Khot, On the power of unique 2-prover 1-round games, in *STOC '02: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, NY, USA, ACM Press (2002) 767–775.

[13] N. Linial, D. Peleg, Y. Rabinovich and N. Saks, Sphere packing and local majorities in graphs. *Proc. 2nd Israel Symposium on Theoretical Computer Science*, IEEE Computer Society Press, Rockville, MD (1993) 141–149.

[14] K. Makino, M. Yamashita and T. Kameda, Max-and min-neighborhood monopolies. *Algorithmica* **34** (2002) 240–260.

[15] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge University Press, London, 1995.

[16] D. Peleg, *Local majority voting, small coalitions and controlling monopolies in graphs: A review*. Technical Report CS96-12, Weizmann Institute, Rehovot (1996).

[17] P. Raghavan and C.D. Thompson, Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica* **7** (1987) 365–374.

[18] J.D. Rose, A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations, in *Graph Theory and Computing*, edited by R.C. Reed, Academic Press, New York (1972) 183–217.

[19] Y. Shoham and M. Tennenholtz, Emergent conventions in multi-agent systems: Initial experimental results and observations. *Proc. International Conference on Principles of Knowledge Representation and Reasoning* (1992) 225–231.

[20] Y. Shoham and M. Tennenholtz, On the systhesis of useful social laws for artificial agent societies. *Proc. AAAI'92* (1992) 276–281.

[21] S.J. Wright, *Primal-Dual Interior-Point Methods*. SIAM (1997).

[22] M. Yannakakis, Computing the minimum fill-in is NP-complete. *SIAM J. Algebr. Discrete Methods* **2** (1981) 77–79.

Communicated by C. Choffrut.
Received June 16, 2009. Accepted October 4, 2010.