Journal of Global Optimization **21:** 111–137, 2001.

Finding independent sets in a graph using continuous multivariable polynomial formulations

JAMES ABELLO[1], SERGIY BUTENKO[2], PANOS M. PARDALOS[2,⋆] and MAURICIO G.C. RESENDE[1]

[1]*Shannon Laboratory, AT&T Labs Research, Florham Park, NJ 07932 USA. (e-mails: abello@research.att.com and mgcr@research.att.com*

[2]*Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611 USA. (e-mails: butenko@ufl.edu and pardalos@ufl.edu)*

Abstract. Two continuous formulations of the maximum independent set problem on a graph $G = (V, E)$ are considered. Both cases involve the maximization of an n-variable polynomial over the n-dimensional hypercube, where n is the number of nodes in G. Two (polynomial) objective functions $F(x)$ and $H(x)$ are considered. Given any solution to x_0 in the hypercube, we propose two polynomial-time algorithms based on these formulations, for finding maximal independent sets with cardinality greater than or equal to $F(x_0)$ and $H(x_0)$, respectively. A relation between the two approaches is studied and a more general statement for dominating sets is proved. Results of preliminary computational experiments for some of the DIMACS clique benchmark graphs are presented.

Key words: Maximum independent set, continuous approach, global optimization

1. Introduction

Let $G = (V, E)$ be a simple undirected graph with vertex set $V = \{1, \ldots, n\}$ and set of edges E. The *complement graph* of G is the graph $\bar{G} = (V, \bar{E})$, where $\bar{E}$ is the complement of E. For a subset $W \subset V$ let $G(W)$ denote the subgraph induced by W on G. $N(i)$ will denote the set of neighbors of vertex i and $d_i = |N(i)|$ is the degree of vertex i. We denote by $\Delta \equiv \Delta(G)$ the maximum degree of G.

A subset $I \subset V$ is called an *an independent set (stable set, vertex packing)* if the edge set of the subgraph induced by I is empty. An independent set is *maximal* if it is not a subset of any larger independent set, and *maximum* if there are no longer independent sets in the graph. The *independence number* $\alpha(G)$ (also called the *stability number*) is the cardinality of a maximum independent set in G.

Along with the maximum independent set problem, we consider some other problems for graphs, namely the *maximum clique*, the *minimum vertex cover*, and the *maximum matching* problems. A *clique* C is a subset of V such that the subgraph $G(C)$ induced by C on G is complete. The maximum clique problem is to

⋆ The author was partially supported by NSF under Grants DBI 9808210 and EIA 9872509

find a clique of maximum cardinality. The *clique number* $\omega(G)$ is the cardinality of a maximum clique in G. A *vertex cover* V' is a subset of V, such that every edge $(i, j) \in E$ has at least one endpoint in V' is a subset of V, such that every edge $(i, j) \in E$ has at least one endpoint in V'. The minimum vertex cover problem is to find a vertex cover of minimum cardinality. Two edges in a graph are *incident* if they have an endpoint in common. A set of edges is independent if no two of them are incident. A set M of independent edges in a graph $G = (V, E)$ is called a *matching*. The maximum matching problem is to find a matching of maximum cardinality.

It is easy to see that I is a maximum independent set of G if and only if I is a maximum clique of $\bar{G}$ and if and only if $V \setminus I$ is a minimum vertex cover of G. The last fact yields Gallai's identity (Gallai, 1959)

$$\alpha(G) + |S| = |V(G)|, \tag{1}$$

where S is a minimum vertex cover of the graph G.

The maximum independent set, the maximum clique, and the minimum vertex cover problems are NP-hard (Garey and Johnson, 1979), so it is unlikely that a polynomial-time algorithm for computing the independence number of a graph can be devised. Alternatively, the maximum matching problem can be solved in polynomial time even for the weighted case (see, for instance, Papadimitriou and Steiglitz, 1988).

König's theorem (see p. 30 in Diestel, (1997) and Rizzi (2000) for a shorrt proof) states that the maximum cardinality of a matching in a bipartite graph G is equal to the minimum cardinality of a vertex cover.

Practical applications of these optimization problems are abundant. They appear in information retrieval, signal transmission analysis, classification theory, economics, scheduling, experimental design, and computer vision. See Abello et al. (2000), Avondo-Bodeno (1962), Bomze et al. (1999), Balas and Yu (1986), Berge (1962), Deo (1974), Pardalos and Xue (1992) and Trotter (1973) for details.

The remainder of this paper is organized as follows. In Section 2 we review some integer programming and continuous formulations of the maximum independent set problem. Two new polynomial formulations are proposed in Section 3. In Section 4, we show how the Motzkin–Straus theorem can be obtained from one of these formulations. Two algorithms that use the polynomial formulations to find maximal independent sets are presented in Section 5. In Section 6, these two algorithms are shown to be equivalent in some sense. Examples are presented in Section 7. In Section 8, one of the polynomial formulations proposed in this paper is extended for dominating sets. Preliminary computational results, illustrating the approach, are described in Section 9. Finally, concluding remarks are made in Section 10.

2. Problem formulations

The maximum independent set problem has many equivalent formulations as an integer programming problem and as a continuous nonconvex optimization problem (Pardalos and Xue, 1992; Bomze et al., 1999). In this section we will give a brief review of some existing approaches.

2.1. INTEGER PROGRAMMING FORMULATIONS

Given a vector $w \in \mathbb{R}^n$ of positive weights w_i (associated with each vertex $i, i = 1, \ldots, n$ the *maximum weight independent set* problem asks for independent sets of maximum weight. Obviously, it is a generalization of the maximum independent set problem. One of the simplest formulations of the maximum weight independent set problem is the following *edge formulation*:

$$\max f(x) = \sum_{i=1}^{n} w_i x_i, \tag{2}$$

subject to

$$x_i + x_j \leqslant 1, \forall (i, j) \in E, \tag{2a}$$

$$x_i \in \{0, 1\}, i = 1, \ldots, n. \tag{2b}$$

An alternative formulation of this problem is the following *clique formulation* (Grötschel et al., 1993);

$$\max f(x) = \sum_{i=1}^{n} w_i x_i, \tag{3}$$

subject to

$$\sum_{i \in S} x_i \leqslant 1, \forall S \in \mathcal{C} \equiv \{\text{maximal cliques of} G\}, \tag{3a}$$

$$x_i \in \{0, 1\}, i = 1, \ldots, n. \tag{3b}$$

The advantage of formulation (3) over (2) is a smaller gap between the optimal values of (3) and its linear relaxation. However, since there is an exponential number of constraints in (3a), finding an efficient solution of (3) is difficult.

We mention now one more integer programming formulation. Let A_G be the adjacency matrix of a graph G, and let J denote the $n \times n$ identity matrix. The maximum independent set problem is equivalent to the global quadratic zero-one problem

$$\max f(x) = x^T A x, \tag{4}$$

subject to

$$x_i \in \{0, 1\}, i = 1, \ldots, n, \tag{4a}$$

where $A = J - A_G$. If x^* is a solution to (4), then the set I defined by $I = \{i \in V : x_i^* = 1\}$ is a maximum independent set of G with $|I| = f(x^*)$. See Pardalos and Rodgers (1992) for details.

2.2. CONTINUOUS FORMULATIONS

Shor (1990) considered an interesting formulation of the maximum weight independent set problem by noticing that formulation (2) is equivalent to the quadratically constrained global optimization problem

$$\max f(x) = \sum_{i=1}^{n} w_i x_i, \tag{5}$$

subject to

$$x_i x_j = 0, \quad \forall (i, j) \in E, \tag{5a}$$

$$x_i^2 - x_i = 0, i = 1, 2, \ldots, n. \tag{5b}$$

Applying dual quadratic estimates, Shor reported good computational results and presented a new way to compute the Lovász number of a graph (Lovász, 1979).

Motzkin and Straus (1965) established a remarkable connection between the maximum clique problem and a certain standard quadratic progbramming problem. The original proof of the Motzkin–Straus theorem was by induction. Below we present a new proof. Let A_G be the adjacency matrix of G and let e be the n-dimensional vector with all components equal to 1.

THEOREM 1 (Motzkin–Straus). *The global optimal value of the following quadratic program*

$$\max f(x) = \frac{1}{2} x^T A_G x, \tag{6}$$

subject to

$$e^T x = 1, \tag{6a}$$

$$x \geqslant 0. \tag{6b}$$

is given by

$$\frac{1}{2}\left(1 - \frac{1}{\omega(G)}\right),$$

where $\omega(G)$ is the clique number of G.

Proof. We will use the following well known inequality for the proof:

$$\sum_{i=1}^{n} a_i^2 \geqslant \frac{\left(\sum_{i=1}^{n} a_i\right)^2}{n}, \tag{7}$$

where $a_1, a_2, \ldots, a_n$ are positive numbers. Equality takes place if and only if $a_1 = a_2 = \cdots = a_n$.

Now consider the program (6). Let J denote the $n \times n$ identity matrix and let O be the $n \times n$ matrix of all ones. Then

$$A_G = O - J - A_{\bar{G}}$$

and

$$y^T A_G y = y^T O y - y^T J y - y^T A_{\bar{G}} y = 1 - (y^T J y + y^T A_{\bar{G}} y),$$

where $A_{\bar{G}}$ is the adjacency matrix of the complement graph $\bar{G}$.

Let $R(x) = x^T J x + x^T A_{\bar{G}} x$. Program (8) is equivalent to (6).

$$\begin{aligned} &\min R(x) = x^T J x + x^T A_{\bar{G}} x, \\ &\text{s.t. } e^T x = 1, \\ &x \geqslant 0. \end{aligned} \tag{8}$$

To check that there always exists an optimal solution x^* of (8) such that $x^{*T} A_{\bar{G}} x^* + 0$, consider any optimal solution $\hat{x}$ of (8). Assume that $\hat{x}^T A_{\bar{G}} \hat{x} > 0$. Then there exists a pair $(i, j) \in \bar{E}$ such that $\hat{x}_i \hat{x}_j > 0$. Consider the following representation for $R(x)$:

$$R(x) = R_{ij}(x) + \bar{R}_{ij}(x),$$

where

$$R_{ij}(x) = x_i^2 + x_j^2 + 2x_i x_j + 2x_i \sum_{(i,k)\in\bar{E}, k\neq j} x_k + 2x_j \sum_{(j,k)\in\bar{E}, k\neq i} x_k;$$

$$\bar{R}_{ij}(x) = R(x) - R_{ij}(x).$$

Without loss of generality, assume that

$$\sum_{(i,k)\in\bar{E}, k\neq i} \hat{x}_k \leqslant \sum_{(j,k)\in\bar{E}, k\neq j} \hat{x}_k.$$

Then, if we set

$$\tilde{x}_k = \begin{cases} \hat{x}_i + \hat{x}_j, & \text{if } k = i, \\ 0, & \text{if } k = j, \\ \hat{x}_k, & \text{otherwise}, \end{cases}$$

we have:

$$\begin{aligned} R(\tilde{x}) &= R_{ij}(\tilde{x}) + \bar{R}_{ij}(\tilde{x}) \\ &= (\hat{x}_i + \hat{x}_j)^2 + 2(\hat{x}_i + \hat{x}_j) \cdot \sum_{(i,k)\in\bar{E}, k\neq i} \hat{x}_k \\ &\leqslant \hat{x}_i^2 + \hat{x}_j^2 + 2\hat{x}_i\hat{x}_j + 2\hat{x}_i \cdot \sum_{(i,k)\in\bar{E}, k\neq j} \hat{x}_k + 2\hat{x}_j \cdot \sum_{(j,k)\in\bar{E}, k\neq i} \hat{x}_k = R(\hat{x}). \end{aligned}$$

If we denote by $Z(x) = \{(i, j) \in \bar{E} : x_i x_j > 0\}$, then $\tilde{x}$ is an optimal solution of (8) with $|Z(\tilde{x})| < |Z(\hat{x})|$. Repeating this procedure a finite number of times we will finally obtain an optimal solution for x^* for which $|Z(x^*)| = 0$ and thus $x^{*T} A_{\bar{G}} x^* = 0$.

Note that $x^{*T} A_{\bar{G}} x^* = 0$ if and only if $\forall (i, j) \in \bar{E} : x_i^* x_j^* = 0$. This means that if we consider the set $C = \{i : x_i^* > 0\}$ then C is a clique.

Without loss of generality, assume that $x_i^* > 0$ for $i = 1, 2, \ldots, m$. and $x_i^* = 0$ for $m + 1 \leqslant i \leqslant n$. Consider the objective function of (8),

$$R(x^*) = x^{*T} J x^* = \sum_{i=1}^{m} x_i^{*2}.$$

By inequality (7) and the feasibility of x^* for (8),

$$\sum_{i=1}^{n} x_i^{*2} \geqslant \frac{\left(\sum_{i=1}^{n} x_i^*\right)}{m} = \frac{1}{m}.$$

Since C is a clique of cardinality m, it follows $m \leqslant \omega(G)$ and

$$R(x^*) \geqslant \frac{1}{m} \geqslant \frac{1}{\omega(G)}.$$

On the other hand, if we consider

$$x_k^* = \begin{cases} \dfrac{1}{\omega(G)}, & \text{if } k \in C^*, \\ 0, & \text{otherwise}, \end{cases}$$

where C^* is a maximum clique of G, then x^* is feasible and $R(x^*) = 1/\omega(G)$. Thus x^* is an optimal solution of (8). Returning back to the original quadratic program, the result of the theorem follows. □

This result is extended in Gibbons et al. (1997), by providing a characterization of maximal cliques in terms of local solutions. Moreover, optimality conditions of the Motzkin–Straus program have been studied and properties of a newly introduced

parameterization of the corresponding QP have been investigated. A further generalization of the same theorem to hypergraphs can be found in (Sós and Straus, 1982).

3. Polynomial formulations

In this section we consider some of the continuous formulations proposed in (Harant, 2000; Harant et al., 1999). We prove deterministically that the independence number of a graph G can be characterized as an optimization problem based on these formulations. Probabilistic proofs of these results were given in (Harant, 2000; Harant et al., 1999). We consider two polynomial formulations, a degree $(\Delta + 1)$ formulation and a quadratic formulation.

3.1. DEGREE $(\Delta + 1)$ POLYNOMIAL FORMULATION

Consider the degree $(\Delta + 1)$ polynomial of n variables

$$F(x) = \sum_{i=1}^{n}(1 - x_i) \prod_{(i,j)\in E} x_j, x \in [0, 1]^n.$$

The following theorem characterizes the independence number of the graph G as the maximization of $F(x)$ over the n-dimensional hypercube.

THEOREM 2 *Let $G = (V, E)$ be a simple graph on n nodes $V = \{1, \ldots, n\}$ and set of edges E, and let $\alpha(G)$ denote the independence number of G. Then*

$$\begin{aligned}\alpha(G) &= \max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} F(x) \\ &= \max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n}(1 - x_i) \prod_{(i,j)\in E} x_j, \qquad \textbf{(P1)}\end{aligned}$$

where each variable x_i corresponds to node $i \in V$.

Proof. Denote the optimal objective function value by $f(G)$, i.e.

$$\begin{aligned}f(G) &= \max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} F(x) \\ &= \max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n}(1 - x_i) \prod_{(i,j)\in E} x_j. \qquad (9)\end{aligned}$$

We want to show that $\alpha(G) = f(G)$.

First we show that (9) always has an optimal 0-1 solution. This is so because $F(x)$ is a continuous function and $[0, 1]^n = \{(x_1, x_2, \ldots, x_n) : 0 \leqslant x_i \leqslant 1, i = 1, \ldots, n\}$ is a compact set. Hence, there always exists $x^* \in [0, 1]^n$ such that $F(x^*) = \max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} F(x)$.

Now, fix any $i \in V$. We can rewrite $F(x)$ in the form

$$F(x) = (1 - x_i)A_i(x) + x_i B_i(x) + C_i(x), \tag{10}$$

where

$$A_i(x) = \prod_{(i,j)\in E} x_j, \tag{11}$$

$$B_i(x) = \sum_{(i,k)\in E} (1 - x_k) \prod_{(k,j)\in E, j\neq i} x_j, \tag{12}$$

$$C_i(x) = \sum_{(i,k)\notin E} (1 - x_k) \prod_{(k,j)\in E} x_j. \tag{13}$$

Expressions (11)–(13) can be interpreted in terms of neighborhoods. $A_i(x)$ and $B_i(x)$ characterize the first- and the second-order neighborhoods of vertex i, respectively, and $C_i(x)$ is complementary to $B_i(x)$ with respect to i in the sense that it describes neighborhoods of all vertices, other than i, which are not characterized by $B_i(x)$.

Notice that x_i is absent in (11)–(13), and therefore $F(x)$ is linear with respect to each variable. It is also clear from the above representation that if x^* is any optimal solution of (9), then $x_i^* = 0$ if $A_i(x^*) > B_i(x^*)$, and $x_i^* = 1$, if $A_i(x^*) < B_i(x^*)$. Finally, if $A_i(x^*) = B_i(x^*)$, we can set $x_i^* = 1$. This shows that (9) always has an optimal 0–1 solution.

To show that $f(G) \geqslant \alpha(G)$, assume that $\alpha(G) = m$ and let I be a maximum independent set. Set

$$x_i^* = \begin{cases} 0, & \text{if } i \in I; \\ 1, & \text{otherwise.} \end{cases} \tag{14}$$

Then, $f(G) = \max_{0\leqslant x_i\leqslant 1, i=1,\dots,n} F(x) \geqslant F(x^*) = m = \alpha(G)$.

To complete the proof, we need to show that $f(G) \leqslant \alpha(G)$. Since the considered problem always has an optimal 0–1 solution, it follows that $f(G)$ must be integer. Assume $f(G) = m$ and take any optimal 0–1 solution of (9). Without loss of generality we can assume that this solution is $x_1^* = x_2^* = \cdots = x_k^* = 0$; $x_{k+1}^* = x_{k+2}^* = \cdots = x_n^* = 1$, for some k. Then we have:

$$(1 - x_1^*) \prod_{(1,j)\in E} x_j^* + (1 - x_2^*) \prod_{(2,j)\in E} x_j^* + \cdots + (1 - x_k^*) \prod_{(k,j)\in E} x_j^* = m \tag{15}$$

Each term in (15) is either 0 or 1. Therefore $k \geqslant m$ and there exists a subset $I \subset \{1, \dots, k\}$ such that $|I| = m$ and

$$\forall i \in I : \prod_{(i,j)\in E} x_j^* = 1.$$

Therefore, if $(i, j) \in E$, then $x_j^* = 1$. Note that since $x_1^* = x_2^* = \cdots = x_k^* = 0$, it follows that $\forall\{i, j\} \subset I$ we have $(i, j) \notin E$ and so I is an independent set by definition. Thus, $\alpha(G) \geqslant |I| = m = f(G)$, which completes the proof. □

COROLLARY 1 *The clique number $\omega(G)$ in a simple undirected graph $G = (V, E)$ satisfies*

$$\omega(G) = \max_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n} (1 - x_i) \prod_{(i,j) \notin E, i \neq j} x_j.$$

Proof. The result follows from Theorem 2 and Gallai's identity (1). □

COROLLARY 2 *The size $|M|$ of a maximum matching M in a bipartite graph $G = (V, E)$ satisfies*

$$|M| = n - \max_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n} (1 - x_i) \prod_{(i,j) \in E} x_j.$$

Proof. The statement follows from Corollary 2 and König's theorem. □

3.2. QUADRATIC POLYNOMIAL FORMULATION

Consider now the quadratic polynomial

$$H(x) = \sum_{i=1}^{n} x_i - \sum_{i,j) \in E} x_i x_j,$$

defined for $x \in [0, 1]^n$.

The following theorem characterizes the independence number of the graph G as the maximization of $H(x)$ over the n-dimensional hypercube.

THEOREM 3 *Let $G = (V, E)$ be a simple graph on n nodes $V = \{1, \ldots, n\}$ and set of edges E, and let $\alpha(G)$ denote the independence number of G. Then*

$$\begin{aligned} \alpha(G) &= \max_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} H(x) \\ &= \max_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} \left(\sum_{i=1}^{n} x_i - \sum_{(i,j) \in E} x_i x_j \right), \end{aligned} \tag{P2}$$

where each variable x_i corresponds to node $i \in V$.

Proof. Denote the optimal objective funcrtion value by $h(G)$, i.e.

$$h(G) = \max_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} \left(\sum_{i=1}^{n} x_i - \sum_{(i,j)\in E} x_i x_j \right) \tag{16}$$

and let I be a maximum independent set. To prove that $h(G) \geqslant \alpha(G)$, let

$$x_i^* = \begin{cases} 1, & \text{if } i \in I; \\ 0, & \text{otherwise.} \end{cases} \tag{17}$$

Since I is an independent set and $x_i^* = 0$ for $i \notin I$, then $\sum_{(i,j)\in E} x_i^* x_j^* = 0$. Furthermore, $\sum_{i=1}^{n} x_i^* = |I| = \alpha(G)$. This yields $h(G) \geqslant H(x^*) = \alpha(G)$.

To complete the proof, we need to show that $h(G) \leqslant \alpha(G)$. Assume $h(G) = m$. Since $H(x)$ is linear with respect to each variable, problem (16) always has an optimal 0–1 solution. Take any optimal 0–1 solution x of (16). Suppose, that there exists $(i_0, j_0) \in E$ such that $x_{i_0} = x_{j_0} = 1$. Changing x_{i_0} to 0 decreases $\sum_{i=1}^{n} x_i$ by 1 and decreases $\sum_{(i,j)\in E} x_i x_j$ by at least 1. Thus, the objective function will not decrease. Doing this for all such pairs (i_0, j_0) will finally lead to an optimal solution x^* such that $\forall (i, j) \in E : x_i^* x_j^* = 0$, and an independent set $I = \{i : x_i^* = 1\}$ of cardinality $h(G)$. This yields $h(G) \leqslant \alpha(G)$ and the theorem is proved. □

4. Motzkin–Straus revisited

The next statement is a reformulation the Motzkin–Straus theorem for the maximum independent set problem. We show how it can be obtained from formulation (P2).

THEOREM 4 *The global optimal value of the following quadratic program,*

$$\max f(x) = \frac{1}{2} x^T A_{\bar{G}} x, \tag{18}$$

subject to

$$e^T x = 1, \tag{18a}$$

$$x \geqslant 0 \tag{18b}$$

is given by

$$\frac{1}{2}\left(1 - \frac{1}{\alpha(G)}\right),$$

where $\alpha(G)$ is the independence number of G.

Proof. Consider formulation (P2):

$$\alpha(G) = \max_{0 \leqslant x_i \leqslant 1, i=1,\dots,n} \left(e^T x - \frac{1}{2} x^T A_G x \right).$$

Note that changing the feasible region from $[0, 1]^n$ in the last quadratic program to the following

$$\{x \geqslant 0 : e^T x = \alpha(G)\}$$

does not change the optimal objective function value. Changing the variables to $y = \frac{1}{\alpha(G)} x$, we obtain:

$$\alpha(G) = \max \left(\alpha(G) e^T y - \frac{1}{2} \alpha(G)^2 y^T A_G y \right) \tag{19}$$

subject to

$$e^T y = 1,$$
$$y \geqslant 0.$$

If I is a maximum independent set of G, then

$$y_i^* = \begin{cases} \dfrac{1}{\alpha(G)}, & \text{if } i \in I \\ 0, & \text{otherwise,} \end{cases}$$

is an optimal solution for the last program.

Consider now

$$A_G = O - J - A_{\bar{G}}.$$

We have

$$y^T A_G y = y^T O y - y^T J y - y^T A_{\bar{G}} y = 1 - y^T J y - y^T A_{\bar{G}} y,$$

where $A_{\bar{G}}$ is the adjacency matrix of the complement graph $\bar{G}$. If $\mathcal{F} = \{y \geqslant 0 : e^T y = 1\}$, then (20) can be rewritten as

$$\alpha(G) = \max_{y \in \mathcal{F}} \left(\alpha(G) + \frac{1}{2} \alpha(G)^2 (-1 + y^T J y + y^T A_{\bar{G}} y) \right)$$

which yields

$$1 = \max_{y \in \mathcal{F}} (y^T J y + y^T A_{\bar{G}} y).$$

Since the maximum is reached in y^*, we have

$$1 - \frac{1}{\alpha(G)} = y^{*T} A_{\bar{G}} y^* \leqslant \max_{y \in \mathcal{F}} y^T A_{\bar{G}} y.$$

Now assume that for some $\hat{y}$,

$$\hat{y}^T A_{\bar{G}} \hat{y} = \max_{y \in \mathcal{F}} y^T A_{\bar{G}} y > 1 - \frac{1}{\alpha(G)}.$$

Then there exists $\tilde{y}$ with $|\{(i, j) \in E : \tilde{y}_i \tilde{y}_j > 0\}| = 0$ such that $\tilde{y}^T A_{\bar{G}} \tilde{y} \geqslant \hat{y}^T A_{\bar{G}} \hat{y}$. For $\tilde{y}$ we have

$$1 - \tilde{y}^T J \tilde{y} \geqslant \hat{y}^T A_{\bar{G}} \hat{y} > 1 - \frac{1}{\alpha(G)},$$

which yields

$$\tilde{y}^T J \tilde{y} < \frac{1}{\alpha(G)},$$

Then, from (7), we obtain

$$\frac{1}{|\{i : \tilde{y}_i > 0\}|} \leqslant \tilde{y}^T J \tilde{y} < \frac{1}{\alpha(G)}.$$

Note that $I = \{i : \tilde{y}_i > 0\}$ is an independent set and the last inequality implies $|I| > \alpha(G)$, which contradicts the definition of $\alpha(G)$. Thus, $\max_{y \in \mathcal{F}} y^T A_{\bar{G}} y = 1 - \frac{1}{\alpha(G)}$. □

5. Algorithms for finding maximal independent sets

In this section, we discuss two algorithms for finding maximal independent sets using the formulations discussed in Section 3.

5.1. AN ALGORITHM BASED ON FORMULATION (P1)

We discuss the algorithm proposed in Harant et al. (1999). As pointed out before, the function $F(x)$ is linear with respect to each variable, so $A_i(x)$ and $B_i(x)$ can be computed for any $i \in \{1, 2, \ldots, n\}$. To produce a maximal independent set using $F(x)$, first let $x^0 \in [0, 1]^n$ be any starting point. The procedure described below produces a sequence of n points $x^1, x^2, \ldots, x^n$ such that x^n corresponds to a maximal independent set. Let $\mathcal{V} = \{1, 2, \ldots, n\}$ and consider some $i \in \mathcal{V}$. From (10)–(13) it follows that if we set

$$x_i^1 = \begin{cases} 0, & \text{if } A_i(x^0) > B_i(x^0); \\ 1, & \text{otherwise.} \end{cases}$$

and $x_j^1 = x_j^0$, if $j \neq i$, we obtain for the point $x^1 = (x_1^1, x_2^1, \ldots, x_n^1)$ that $F(x^1) \geqslant F(x^0)$.

If we update $\mathcal{V} = \mathcal{V}\backslash\{i\}$, we can construct the next point x^2 from x^1 in the same manner. Running this procedure n times, we obtain a point x^n which satisfies the inequality.

$$F(x^n) \geqslant F(x^0),$$

The following theorem states that x^n has an independent set associated with it.

THEOREM 5 *If $I = \{i \in \{1, 2, \ldots, n\} : x_i^n = 0\}$, then I is an independent set.*

Proof. Consider any $(i, j) \in E$. We need to show, that $\{i, j\}$ is not a subset of I. Without loss of generality, assume that we check x_i on the kth iteration of the above procedue. If $x_i^k = 1$, then $i \notin I$. Alternatively, if $x_i^k = 0$, i.e. $i \in I$, we need to show that $j \notin I$. Let $l > k$ be an iteration on which we check x_j. Then

$$A_j(x^{l-1}) = \prod_{(i,j)\in E} x_i = 0,$$

and therefore $A_j(x^{l-1}) \leqslant B_j(x^{l-1})$ and $x_j^l = 1$, which implies that $j \notin I$. □

From the discussion above, we have the following algorithm to find a maximal independent set.

Algorithm 1:
INPUT: $x^0 \in [0, 1]^n$
OUTPUT: Maximal independent set I

0. $v := x^0$;
1. **for** $i = 1, \ldots, n$ **do if** $A_i(v) > B_i(v)$ **then** $v_i := 0$**, else** $v_i := 1$;
2. **for** $i = 1, \ldots, n$ **do if** $A_i(v) = 1$ **then** $v_i = 0$;
3. $I = \{i \in \{1, 2, \ldots, n\} : v_i = 0\}$;

END

THEOREM 6 *Algorithm 1 is correct.*

Proof. We have already discussed steps 1 and 3 of the algorithm which guarantee that an independent set is produced. We need to show that step 2 guarantees that the independent set is maximal. Indeed, assume that after running step 1 we have, for some index i, $v_i = 1$ and

$$A_i(v) = \prod_{(i,j)\in E} v_j = 1.$$

This means that neither i nor any node from the neighborhood of i is included in the independent set that we obtained after step 1. Thus, we can increase the cardinality of this set including i in it by setting $v_i = 0$. □

The time complexity of the proposed algorithm is $O(\Delta^2 n)$, since $A_i(v)$ and $B_i(v)$ can be calculated in $O(\Delta^2)$ time.

5.2. AN ALGORITHM BASED ON FORMULATION (P2)

We now focus our attention on an algorithm, similar to Algorithm 1, based on formulation (P2)

Algorithm 2:
INPUT: $x^0 \in [0,1]^n$
OUTPUT: Maximal independent set I
0. $v := x^0$;
1. **for** $i = 1, \ldots, n$ **do if** $\sum_{(i,j)\in E} v_j < 1$ **then** $v_i := 1$**, else** $v_i := 0$;
2. **for** $i = 1, \ldots, n$ **do if** $\sum_{(i,j)\in E} v_j = 0$ **then** $v_i = 1$;
3. $I = \{i \in \{1, 2, \ldots, n\} : v_i = 1\}$;
END

THEOREM 7 *Algorithm 2 is correct.*

Proof. Algorithm 2 is similar to Algorithm 1. In step 1 it finds an independent set $I_1 = \{i \in \{1, 2, \ldots, n\} : v_i = 1\}$. Set I_1 is independent because after step 1 we have that $\forall (i, j) \in E$ such that $i < j$, if $v_i = 1$ then $\sum_{(j,k)\in E} v_k \geqslant 1$ and $v_j = 0$. If I_1 is not a maximal independent set, then there exists i such that $v_i + \sum_{(i,j)\in E} v_j = 0$. We can increase the cardinality of I_1 by one, including i in it, by setting $v_i = 1$ in step 2. The resulting set is independent, because $\sum_{(i,j)\in E} v_j = 0$, which requires that $\forall j$ such that $(i, j) \in E : v_j = 0$. Thus no neighbors of i are included in the set. □

The time complexity of this algorithm is $O(\Delta n)$.

6. An interesting observation

In this section, we study a relation between Algorithm 1 and Algorithm 2.

If we define

$$F'(x) = \max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n} x_i \prod_{(i,j\in E} (1 - x_j),$$

then formulation (P1) can be rewritten as

$$\begin{aligned} \alpha(G) &= \max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} F'(x) \\ &= \max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n} x_i \prod_{(i,j)\in E} (1 - x_j). \end{aligned} \qquad \textbf{(P1')}$$

The following theorem offers an alternate characterization of the independence number of a graph G.

THEOREM 8 *The independence number of $G = (V, E)$ can be characterized as*

$$\alpha(G) = \max_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} H'(x) = \sum_{i=1}^{n} x_i \left(1 - \sum_{(i,j)\in E} x_j\right).$$

Proof. We first show that $\max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} H'(x) \leqslant \alpha(G)$. For any $x \in [0, 1]^n$, we have

$$\begin{aligned} H'(x) &= \sum_{i=1}^{n} \left(x_i - \sum_{(i,j)\in E} x_i x_j\right) \\ &= \sum_{i=1}^{n} x_i - 2 \sum_{(i,j)\in E} x_i x_j \\ &\leqslant H(x) \\ &\leqslant \alpha(G). \end{aligned}$$

Thus, $\max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} H'(x) \leqslant \alpha(G)$.

To show that $\max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} H'(x) \geqslant \alpha(G)$, let I be an independent set, and consider

$$x_i^* = \begin{cases} 1, & \text{if } i \in I; \\ 0, & \text{otherwise.} \end{cases}$$

Then $H'(x^*) = H(x^*) = \alpha(G)$, which yields $\max_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} H'(x) \geqslant \alpha(G)$. This completes the proof. □

Consider a logical expression, which can be obtained from $H'(x)$ by changing arithmetic operations to logical ones as follows. Summation is changed to $\bigvee$ (logical OR); product is changed to $\bigwedge$ (logical AND); and $1 - x_i$ is changed to $\overline{x_i}$ (logical negation). Then, we have

$$\bigvee_{i=1}^{n} \left(x_i \bigwedge \overline{\left[\bigvee_{(i,j)\in E} x_j\right]}\right) = \bigvee_{i=1}^{n} \left(x_i \bigwedge \left[\bigwedge_{(i,j)\in E} \overline{x_j}\right]\right). \tag{21}$$

Changing logical operations in (21) back to arithmetic operations, we obtain the expression for $F'(x)$.

Now consider Algorithms 1 and 2. Since (**P1**′) is obtained from (**P1**) by changing all variables x_i to $1 - x_i$, we can derive an algorithm for finding maximal independent sets based on (**P1**′) from Algorithm 1 by changing $v = (v_1, \ldots, v_n)$ to $v' = (v'_n) = (1 - v_1, \ldots, 1 - v_n)$ as follows.

Algorithm 1′:
INPUT: $x^0 \in [0,1]^n$
OUTPUT: Maximal independent set I

0. $v := x^0$;
1. **for** $i = 1, \ldots, n$ **do if** $A_i'(v) > B_i'(v)$ **then** $v_i := 1$**, else** $v_i := 0$;
2. **for** $i = 1, \ldots, n$ **do if** $A_i'(v) = 1$ **then** $v_i := 1$;
3. $I = \{i \in \{1, 2, \ldots, n\} : v_i = 1\}$;

END

In Algorithm 1′

$$A_i'(v) = A_i(v') = \prod_{(i,j)\in E} (1 - v_j);$$

$$B_i'(v) = B_i(v') = \sum_{(i,k)\in E} v_k \prod_{(k,j)\in E, i\neq j} (1 - v_j).$$

We have $A_i'(v) > B_i'(v)$ if and only if

$$1 - [B_i'(v) + (1 - A_i'(v))] > 0. \tag{22}$$

Then, logical expression L_{11} corresponding to the left-hand side of (22) is

$$L_{11} = \overline{\left[B_i'(v) \bigvee \overline{A_i'(v)}\right]} = \overline{B_i'(v)} \bigwedge A_i'(v).$$

Since $A_i'(v) = \bigwedge_{(i,j)\in E} \overline{v_j}$ and

$$\overline{B_i'(v)} = \bigwedge_{(i,k)\in E} \overline{\left[v_k \bigwedge_{(k,j)\in E, i\neq j} \overline{v_j}\right]} = \bigwedge_{(i,k)\in E} \left[\overline{v_k} \bigvee_{(k,j)\in E, i\neq j} v_j\right],$$

we have

$$L_{11} = \left[\bigwedge_{(i,k)\in E} \left[\overline{v_k} \bigvee_{(k,j)\in E, i\neq j} v_j\right]\right] \bigwedge \left[\bigwedge_{(i,j)\in E} \overline{v_j}\right] = \bigwedge_{(i,j)\in E} \overline{v_j}.$$

The logical expression L_{21} corresponding to the left-hand side of the inequality $1 - \sum_{(i,j)\in E} v_j$ from step 1 of Algorithm 2 is:

$$L_{21} = \overline{\left[\bigvee_{(i,j)\in E} v_j\right]} = \bigwedge_{(i,j)\in E} \overline{v_j} = L_{11},$$

Similarly, the logical expression L_{12} corresponding to the left-hand side of the equality $1 - A_i'(v) = 0$ taken from step 2 of Algorithm 1′ can be written as

$$L_{12} = \overline{[A_i'(v)]} = \overline{\left[\bigwedge_{(i,j)\in E} \overline{v_j}\right]} = \bigvee_{(i,j)\in E} v_j = L_{22},$$

where L_{22} is the logical expression corresponding to the left-hand side of the equality from step 2 of Algorithm 2.

This shows that Algorithms 1 and 2 are syntactically related by substituting each variable v_j by $1 - v_j$, for $j = 1, \ldots, n$ in Algorithm 1.

7. Examples

The algorithms presented build a maximal independent set from any given point $x^0 \in [0, 1]^n$ in polynomial time. The output, however, depends on the choice of x^0. An interesting question that arises is how to choose such input point x^0, so that a maximum independent set can be found? The problem of finding such a point cannot be solved in polynomial time, unless P = NP. A related question is to improve the lower bound on the independence number. The best known bound, by Caro and Tuza, Wei (1991, 1981), is expressed by

$$\alpha(G) \geqslant \sum_{i \in V} \frac{1}{d_i + 1}.$$

Though we are unaware of a way to improve this bound using formulations (**P1**) or (**P2**), we can show on simple examples that in some cases even starting with a 'bad' starting point $x^0 (F(x^0) \leqslant 1$ or $H(x^0) \leqslant 1)$, we obtain a maximum independent set as the output.

7.1. EXAMPLE 1

Consider the graph in Figure 1. For this example

$$\begin{aligned}
x &= (x_1, x_2, x_3, x_4) \in [0, 1]^4;\\
F(x) &= (1 - x_1)x_2x_3x_4 + (1 - x_2)x_1 + (1 - x_3)x_1 + (1 - x_4)x_1;\\
A_1(x) &= x_2x_3x_4;\\
B_1(x) &= (1 - x_2) + (1 - x_3) + (1 - x_4);\\
A_2(x) &= A_3(x) = A_4(x) = x_1;\\
B_2(x) &= (1 - x_1)x_3x_4;\\
B_3(x) &= (1 - x_1)x_2x_4;\\
B_4(x) &= (1 - x_1)x_2x_3;\\
H(x) &= x_1 + x_2 + x_3 + x_4 - x_1x_2 - x_1x_3 - x_1x_4.
\end{aligned}$$

Consider Algorithm 1 with $x^0 = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})$. Since $A_1(x^0) = \frac{1}{8}$, $B_1(x^0) = \frac{3}{2}$, and since $\frac{1}{8} < \frac{3}{2}$, the next point is $x^1 = (1, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})$.

Next, $A_2(x^1) = 1$, $B_2(x^1) = 0$, and $x^2 = (1, 0, \frac{1}{2}, \frac{1}{2})$. After two more iterations we get $x^4 = (1, 0, 0, 0)$ with $I = \{2, 3, 4\}$, which is the maximum independent

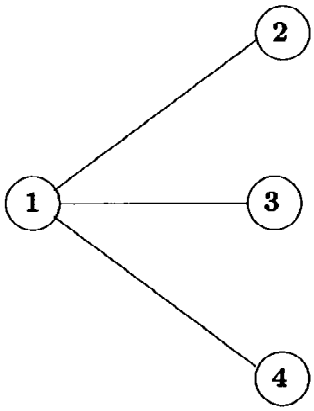

Figure 1. Example 1

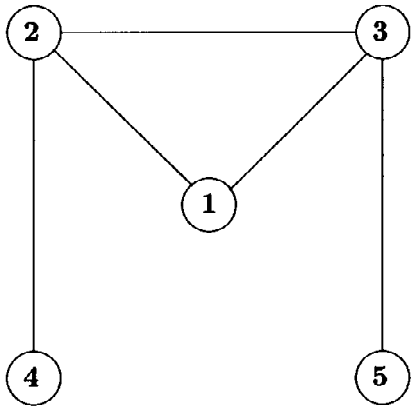

Figure 2. Example 2

set of the given graph. We have $|I| = 3$, $F(x^0) = \frac{13}{16}$, and the objective function increase is $|I| - F(x^0) = \frac{35}{16}$.

For Algorithm 2, starting with $x^0 = (1, 1, 1, 1)$, for which $H(x^0) = 1$, we obtain the maximum independent set after step 1. Note, that the Caro–Wei bound for this graph is $\frac{7}{4}$.

7.2. EXAMPLE 2

For the graph in Figure 2, we have $x = (x_1, x_2, x_3, x_4, x_5) \in [0, 1]^5$ and

$$F(x) = (1 - x_1)x_2x_3 + (1 - x_2)x_1x_3x_4 + (1 - x_3)x_1x_2x_5 + (1 - x_4)x_2 + (1 - x_5)x_3.$$

Applying Algorithm 1 for this graph with initial point $x^0 = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})$, we obtain, at the end of step 1, the solution $x^5 = (1, 1, 1, 0, 0)$, which corresponds to

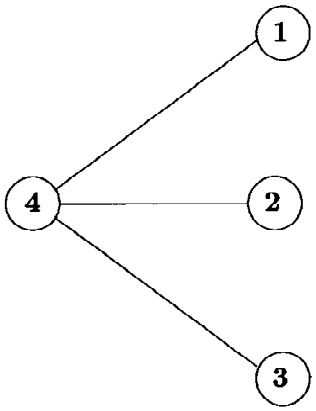

Figure 3. Example 3

the independent set $I = \{4, 5\}$. At the end of step 2, the solution is $(0, 1, 1, 0, 0)$, which corresponds to the maximum independent set $I = \{1, 4, 5\}$. For this case we have $|I| = 3$, $F(x^0) = \frac{7}{8}$, and the objective function improvement is $\frac{17}{8}$. With the initial point $x^0 = (0, 0, 0, 0, 0)$, $H(x^0) = 0$, and Algorithm 2 finds the maximum independent set after step 1. For this example, the Caro–Wei bound equals $\frac{11}{6}$.

7.3. EXAMPLE 3

This example shows, that the output of Algorithm 1 and Algorithm 2 depends not only on initial point x^0, but also on the order in which we examine variables in steps 1 and 2. For example, if we consider the graph from Figure 1 with a different order of nodes (as in Figure 3), and run Algorithm 1 and Algorithm 2 for this graph with initial point $x^0 = (1, 1, 1, 1)$, we obtain $I = \{4\}$ as output for both algorithms. Note that, for the graph from Figure 1, both outputs would be the maximum independent set of the graph.

As Example 3 shows, we may be able to improve both algorithms by including two procedures (one for each step) which, given a set of remaining nodes, choose a node to be examined next. Consider Algorithm 2. Let `index1()` and `index2()` be procedures for determining the order of examining nodes on step 1 and step 2 of the algorithm, respectively. Then we have the following algorithm

Algorithm 3:
INPUT: $x^0 \in [0, 1]^n$
OUTPUT: Maximal independent set I

0. $c := x^0$; $V_1 := V$; $V_2 := V$;
1. **while** $V_1 \neq \emptyset$ **do**
   a) = `index1`(V_1);
   b) **if** $\sum_{(k,j)\in E} x_j < 1$ **then** $v_k := 1$**, else** $v_k := 0$;

c) $V_1 : V_1 \backslash \{v_k\}$;

2. **while** $V_2 \neq \emptyset$ **do**

a) $k = \texttt{index2}(V_2)$;

b) **if** $\sum_{(k,j)\in E} v_j = 0$ **then** $v_k := 1$;

c) $V_2 := V_2 \backslash \{v_k\}$;

3. $I = \{i \in \{1, 2, \ldots, n\} : v_i = 1\}$;

END

In general, procedures `index1()` and `index2()` can be different. We propose the same procedure `index()` for `index1()` and `index2()`:

$$\texttt{index}(V_0 = \operatorname*{argmax}_{k\in V_0} \left\{ \sum_{(k,j)\in E} v_j \right\}$$

breaking ties in favor of the node with the smallest neighborhood in $V \backslash V_0$ and at random if any nodes remain tied.

8. A generalization for dominating sets

For a graph $G = (V, E)$ with $V = \{1, \ldots, n\}$, let $l = (k_1, \ldots, k_n)$ be a vector of integers such that $1 \leqslant k_i \leqslant d_i$ for $i \in V$, where d_i is the degree of vertex $i \in V$. An l-dominating set (Harant et al., 1999) is a set $D_l \subset V$ such that every vertex $i \in V \backslash D_l$ has at least k_i neighbors in D_l. The l-domination number $\gamma_l(G)$ of G is the cardinality of a smallest l-dominating set of G.

For $k_1 = \cdots = k_n = 1$, l-domination corresponds to the usual definition of domination. The domination number $\gamma(G)$ of G is the cardinality of a smallest dominating set of G. If $k_i = d_i$ for $i = 1, \ldots, n$, then $I = V \backslash D_l$ is an independent set and $\gamma_d(G) = n - \alpha(G)$ with $d = (d_1, \ldots, d_n)$.

The following theorem characterizes the domination number. A probabilistic proof of this theorem is found in Harant et al. (1999).

THEOREM 9 *The domination number can be expressed by*

$$\gamma_l(G) = \min_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} f_l(x) = \min_{0\leqslant x_i\leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} (1 - x_i) \times \left(\sum_{p=0}^{k_i-1} \sum_{\{i_1,\ldots i_p\}\subset N(i)} \prod_{m\in\{i_1,\ldots,i_p\}} x_m \prod_{m\in N(i)\backslash\{i_1,\ldots,i_p\}} (1 - x_m) \right). \quad (23)$$

Proof. Denote the objective function by $g(G)$, i.e.

$$g(G) = \min_{0\leqslant x_i\leqslant 1, i=1,\dots,n} \sum_{i=1}^{n} x_i + \sum_{i=1}^{n}(1-x_i) \times \left(\sum_{p=0}^{k_i-1} \sum_{\{i_1,\dots i_p\}\subset N(i)} \prod_{m\in\{i_1,\dots,i_p\}} x_m \prod_{m\in N(i)\setminus\{i_1,\dots,i_p\}} (1-x_m) \right). \tag{24}$$

We want to show that $\gamma_l(G) = g(G)$.

We first show that (24) always has an optimal 0–1 solution. Since $f_l(x)$ is a continuous function and $[0,1]^n = \{(x_1, x_2, \dots, x_n) : 0 \leqslant x_i \leqslant 1, i = 1, \dots, n\}$ is a compact set, there always exists $x^* \in [0,1]^n$ such that $f_l(x^*) = \min_{0\leqslant x_i\leqslant 1, i=1,\dots,n} f_l(x)$. The statement follows from linearity of $f_l(x)$ with respect to each variable.

Now we show that $g(G) \leqslant \gamma_l(G)$. Assume $\gamma_l(G) = m$. Set

$$x_i^l = \begin{cases} 1, & \text{if } i \in D_l; \\ 0, & \text{otherwise.} \end{cases}$$

Then, $g(G) = \min_{0\leqslant x_i\leqslant 1, i=1,\dots,n} f_l(x) \leqslant f_l(x^l) = m = \gamma_l(G)$.

Finally, we show that $g(G) \geqslant \gamma_l(G)$. Since (24) always has an optimal 0–1 solution, then $g(G)$ must be integer. Assume $g(G) = m$. Take an optimal 0–1 solution x^* of (23), such that the number of 1s is maximum among all 0–1 optimal solutions. Without loss of generality we can assume that this solution is $x_1^* = x_2^* = \cdots = x_r^* = 1$; $x_{r+1}^* = x_{r+2}^* = \cdots = x_n^* = 0$, for some r. Let

$$Q_i(x) = \sum_{p=0}^{k_i-1} \sum_{\{i_1,\dots i_p\}\subset N(i)} \prod_{m\in\{i_1,\dots,i_p\}} x_m \prod_{m\in N(i)\setminus\{i_1,\dots,i_p\}} (1-x_m)$$

and

$$Q(x) = \sum_{i=r+1}^{n} (1-x_i) \times \left(\sum_{p=0}^{k_i-1} \sum_{\{i_1,\dots i_p\}\subset N(i)} \prod_{m\in\{i_1,\dots,i_p\}} x_m \prod_{m\in N(i)\setminus\{i_1,\dots,i_p\}} (1-x_m) \right).$$

Let

$$D_l = \{i : x_i^* = 1\}.$$

We have

$$r + Q(x^*) = m.$$

From the last expression and nonnegativity of $Q(x)$ it follows that $|D_l| = r \leqslant m$.

We want to show that D_l is an l-dominating set. Assume that it is not. Let $S = \{i \in V\setminus D_l : |N(i) \cap D_l| < k_i\}$. Then $S \neq \emptyset$ and $\forall i \in S : Q_i(x^*) \geqslant 1$. Note,

that changing x_i^*, $i \in S$ from 0 to 1 will increase $\sum_{i=1}^n x_i^*$ by 1 and will decrease $Q(x^*)$ by at least 1, thus it will not increase the objective function.

Consider $D_l^* = D_l \cup S$ and build x' as follows:

$$x_i' = \begin{cases} 1, & \text{if } i \in D_l^*; \\ 0, & \text{otherwise.} \end{cases}$$

Then $f_l(x') \leqslant f_l(x^*)$ and $|\{i : x_i' = 1\}| > |\{i : x_i^* = 1\}|$, which contradicts the assumption that x^* is an optimal solution with the maximum number of 1's. Thus, D_l is an l-dominating set with cardinality $r \leqslant m = g(G)$ and therefore $\gamma_l(G) \leqslant g(G)$. This concludes the proof of the theorem. □

COROLLARY 3 *For the case in which $k_1 = \cdots = k_n = 1$, we have*

$$\gamma(G) = \min_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n} \left(x_i + (1 - x_i) \prod_{(i,j) \in E} (1 - x_j) \right).$$

COROLLARY 4 *For the case $k_i = d_i$, $i = 1, \ldots, n$, the result of Theorem 2 follows.*

Proof. It can be shown by induction for $|N(i)|$, that

$$\sum_{p=0}^{d_i - 1} \sum_{\{i_1, \ldots p_i\} \subset N(i)} \prod_{m \in \{i_1, \ldots, p\}} x_m \prod_{m \in N(i) \setminus \{i_1, \ldots, i_p\}} (1 - x_m) = 1 - \prod_{j \in N(i)} x_j.$$

Thus,

$$\begin{aligned} \alpha(G) &= n - \min_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n} \left(x_i + (1 - x_i) \left(1 - \prod_{(i,j) \in E} x_j \right) \right) \\ &= \max_{0 \leqslant x_i \leqslant 1, i=1,\ldots,n} \sum_{i=1}^{n} (1 - x_i) \prod_{(i,j) \in E} x_j. \end{aligned}$$ □

9. Computational experiments

This section presents preliminary computational results of the algorithms described in this paper. Please notice that the results are very inconclusive at this point. We have tested the algorithms on some of the DIMACS clique instances which can be downloaded from the URL `http://dimacs.rutgers.edu/Challenges/`. All algorithms are programmed in C and compiled and executed on an Intel Pentium III 600 Mhz PC under MS Windows NT.

Table 1. Results on benchmark instances: Algorithms 1 and 2, random x^0.

Name	Nodes	Density	$\omega(G)$	Sol. Found		Average Sol.		Time (sec.)	
				A1	A2	A1	A2	A1	A2
MANN_a9	45	0.927	16	15	16	13.05	14.45	0.01	0.01
MANN_a27	378	0.009	126	113	120	68.16	119.16	8.14	1.89
MANN_a45	1035	0.996	345	283	334	198.76	331.66	243.94	36.20
c-fat200-1	200	0.777	12	12	12	10.80	12.00	33.72	0.33
c-fat200-2	200	0.163	24	24	24	22.22	22.59	31.43	0.32
c-fat200-3	200	0.426	58	58	58	57.14	57.85	17.60	0.28
hamming6-2	64	0.905	32	32	32	30.17	21.44	0.06	0.01
hamming6-4	64	0.349	4	4	4	3.72	2.38	0.37	0.01
hamming8-2	256	0.969	128	128	121	119.54	90.95	6.45	1.07
hamming8-4	256	0.639	16	16	16	10.87	9.71	38.22	1.43
hamming10-2	1024	0.990	512	503	494	471.17	410.92	233.45	46.46
johnson8-2-4	28	0.556	4	4	4	4.00	4.00	0.01	0.01
johnson8-4-4	70	0.768	14	14	14	13.02	9.99	0.17	0.01
johnson16-2-4	120	0.765	8	8	8	8.00	8.00	1.13	0.04
johnson32-2-4	496	0.879	16	16	16	16.00	16.00	186.77	4.57
keller4	171	0.649	11	7	7	7.00	7.00	9.72	0.13
san200_0.9_1	200	0.900	70	53	61	39.08	38.86	0.14	0.14
san200_0.9_2	200	0.900	60	34	32	29.41	29.09	0.14	0.13
san200_0.9_3	200	0.900	44	31	30	27.58	26.93	0.23	0.18
san400_0.9_1	400	0.900	100	53	54	46.18	44.20	4.54	2.68

First, each algorithm was executed 100 times with random initial solutions uniformly distributed in the unit hypercube. The results of these experiments are summarizede in Tables 1 and 2. The columns 'Name,' 'Nodes,' 'Density,' and '$\omega(G)$' represent the name of the graph, the number of its nodes, its density, and its clique number, respectively. This information is available from the DIMACS web site. The column 'Sol. Found' contains the size of the largest clique found after 100 runs. The columns 'Average Sol.' and 'Time (s)' contain average solution and average CPU time (in seconds) taken over 100 runs of an algorithm, respectively. Finally, columns 'A1' and 'A2' in Table 1 represent Algorithms 1 and 2.

Table 3 contains the results of computations for all the algorithms with initial solution x^0, such that $x_i^0 = 0, i = 1, \ldots, n$. In this table, 'A3' stands for Algorithm

Table 2. Results on benchmark instances: Algorithm 3, random x^0.

Name	Nodes	Dens.	$\omega(G)$	Sol. Found	Average Sol.	Time (s)
MANN_a9	45	0.927	16	16	14.98	0.01
MANN_a27	378	0.990	126	121	119.21	4.32
MANN_a45	1035	0.996	345	334	331.57	87.78
c-fat200-1	200	0.077	12	12	11.64	0.48
c-fat200-2	200	0.163	24	24	22.47	0.47
c-fat200-5	200	0.426	58	58	57.25	0.42
hamming6-2	64	0.905	32	32	27.49	0.02
hamming6-4	64	0.349	4	4	4.00	0.02
hamming8-2	256	0.969	128	128	100.78	0.80
hamming8-4	256	0.639	16	16	12.49	1.13
hamming10-2	1024	0.990	512	512	359.53	90.11
johnson8-2-4	28	0.556	4	4	4.00	0.01
johnson8-4-4	70	0.768	14	14	11.22	0.02
johnson16-2-4	120	0.765	8	8	8.00	0.09
johnson32-2-4	496	0.879	16	16	16.00	10.20
keller4	171	0.649	11	9	7.54	0.28
san200_0.9_1	200	0.900	70	46	45.03	0.37
san200_0.9_2	200	0.900	60	37	34.94	0.38
san200_0.9_3	200	0.900	44	32	26.86	0.37
san400_0.9_1	400	0.900	100	51	50.01	2.54

3. As can be seen from the tables, the best solutions for almost all instances obtained during the experiments can be found among the results for Algorithm 3 with $x_i^0 = 0, i = 1, \ldots, n$ (see Table 3).

In Table 4 we compare these results with results for some other continuous based heuristics for the maximum clique problem taken from (Bomze et al., 2000). The columns 'ARH', 'PRD($\frac{1}{2}$)', 'PRD(0)' and 'CBH' contain the size of a clique found using the annealed replication heuristic (Bomze et al., 2000), the plain replicator dynamics applied for two different parameterizations (with parameters $\frac{1}{2}$ and 0) of the Motzkin–Straus formulation (Bomze et al., 1997), and the heuristic proposed in Gibbons et al. (1996), respectively. The column 'A3(0)' represents the results for Algorithm 3 with $x_i^0 = 0, i = 1, \ldots, n$.

Table 3. Results on benchmark instances: Algorithms 1–3, $x_i^0 = 0$, for $i = 1, \ldots, n$.

Name	Nodes	Dens.	$\omega(G)$	Sol. Found		Time (s)			
				A1	A2	A1	A2	A1	A2
MANN_a9	45	0.927	16	16	9	16	0.01	0.01	0.01
MANN_a27	378	0.990	126	125	27	125	8.17	2.01	3.72
MANN_a45	1035	0.996	345	340	45	342	170.31	36.42	79.15
c-fat200-1	200	0.077	12	12	12	12	34.13	0.48	1.59
c-fat200-2	200	0.163	24	24	24	24	33.15	0.79	0.92
c-fat200-3	200	0.426	58	58	58	58	21.63	0.48	12.02
hamming6-2	64	0.905	32	32	32	32	0.23	0.03	0.05
hamming6-4	64	0.349	4	2	4	4	0.42	0.03	0.06
hamming8-2	256	0.969	128	128	128	128	4.57	0.89	1.32
hamming8-4	256	0.639	16	16	16	16	39.07	1.98	1.79
hamming10-2	1024	0.990	512	512	512	512	508.30	35.97	82.19
johnson8-2-4	28	0.556	4	4	4	4	0.01	0.01	0.01
johnson8-4-4	70	0.768	14	14	14	14	0.33	0.02	0.03
johnson16-2-4	120	0.765	8	8	8	8	1.56	0.09	0.19
johnson32-2-4	496	0.879	16	16	16	16	191.24	5.97	9.81
keller4	171	0.649	11	7	7	11	7.99	0.18	0.43
san200_0.9_1	200	0.900	70	42	43	47	3.19	0.27	0.56
san200_0.9_2	200	0.900	60	29	36	40	3.21	0.27	0.54
san200_0.9_3	200	0.900	44	29	21	34	3.05	1.04	0.48
san400_0.9_1	400	0.900	100	52	35	75	88.47	3.28	6.55

These computational results are preliminary and more experiments are needed to determine if this approach is computationally competitive with other methods for the maximum clique problem.

10. Conclusion

We give deterministic proofs of two continuous formulations for the maximum independent set problem and their generalizations for dominating sets. We show that the celebrated Motzkin–Straus theorem can be obtained from these formulations and we offer three syntactically related polynomial time algorithms for finding maximal independent sets. We report on a preliminary computational investigation

Table 4. Results on benchmark instances: comparison with other continuous based approaches. $x_i^0 = 0, i = 1, \ldots, n$.

Name	Nodes	Dens.	$\omega(G)$	Sol. Found				
				ARH	PRD($\frac{1}{2}$)	PRD(0)	CBH	A3(0)
MANN_a9	45	0.927	16	16	12	12	16	16
MANN_a27	378	0.990	126	117	117	117	121	125
keller4	171	0.649	11	8	7	7	10	11
san200_0.9_1	200	0.900	70	45	45	45	46	47
san200_0.9_2	200	0.900	60	39	36	35	36	40
san200_0.9_3	200	0.900	44	31	32	33	30	34
san400_0.9_1	400	0.900	100	50	40	55	50	75

of these algorithms. A more complete computational investigation is needed to determine if this approach is competitive with existing approaches for the maximum independent set problem.

Acknowledgements

We would like to thank Immanuel Bomze and Stanislav Busygin for their valuable comments which helped us to improve the paper.

References

Abello, J., Pardalos, P.M. and Resende, M.G.C. (2000), On maximum clique problems in very large graphs, in: Abello, J. and Vitter, J.S. (eds.), External memory algorithms, Vol. 50 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society, pp. 119–130.

Avondo-Bodeno, G. (1962), *Economic applications of the theory of graphs*. Gordon and Breach Science, New York.

Balas, E. and Yu, C. (1986), Finding a maximum clique in an arbitrary graph. SIAM J. on Computing 15: 1054–1068.

Berge, C. (1962), *The theory of graphs and its applications*. Methuen.

Bomze, I.M., Budinich, M. Pardalos, P.M. and Pelillot, M. (1999), The maximum clique prob.em. in: Du, D.-Z. and Pardalos, P.M. (eds.) *Handbook of Combinatorial Oprtimization*. Kluwer Acad. Publishers, Dordrecht, pp. 1–74.

Bomze, I.M., Budinich, M., Pelillo, M. and Rossi, C. (2000), A new 'annealed' heuristic for the maximum clique problem, in: Pardalos, P.M. (ed.) *Approximation and complexity in Numerical Optimization: Continuous and Discrete Problems*. Kluwer Acad. Publishers, Dordrecht pp. 78–96.

Bomze, I.M., Pelillo, M. and Giacomini, R. (1997), Evolutionary approach to the maximum clique problem: empirical evidence on a larger scale, in: Bomze, I.M., Csendes, T., Horst,

R. and Pardalos, P.M. (eds.), *Developments of Global Optimization*. Kluwer Acad. Publishers, Dordrecht, pp. 95–108.

Caro, Y. and Tuza, Z. (1991), Improved lover bounds on k-independence. *J. Graph Theory* 15, 99–107.

Deo, N. (1974), *Graph theory with applications to engineering and computer science.* Prentice-Hall, Englewood Cliffs, NJ.

Diestel, R. (1997) *Graph Theory*. Springer Berlin.

Gallai, T. (1959), Über extreme Punkt- und Kantenmengen, in: *Ann. Univ. Sci. Budapest. Eötvös Sect. Math.*, Vol. 2. pp. 133–138.

Garey, M. and Johnson, D. (1979), *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, San Francisco, CA.

Gibbons, L.E., Hearn, D.W. and Pardalos, P.M. (1996), A continuous based heuristic for the maximum clique problem, in: Johnson, D.S. and Trick, M.A. (eds.) *Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge*, Vol 26 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society, pp. 103–124.

Grötschel, M., Lovász, L. and Schrijver, A. (1993), *Geometric Algorithms and Combinatorial Optimization.* Springer, Berlin. 2nd edition.

Harant, J. (2000), Some news about the independence number of a graph. *Discussiones Mathematicae Graph Theory* 20, 71–79.

Harant, J., Pruchnewski, A. and Voigt, M. (1999), On dominating sets and independent sets of graphs. *Combinatgorics, Probability and Computing* 8, 547–553.

Lovász, L. (1979), On the Shannon capacity of a graph. *IEEE Trans. Inform. Theory* 25, 1–7.

Motzkin, T.S. and Straus, E.G. (1965), Maxima for graphs and a new proof of a theorem of Turán. *Cand.J. Math.* 17, 533–540.

Papadimitriou, C.H. and Steiglitz, K. (1988), *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, Inc.

Pardalos, P.M. and Rodgers, G.P. (1992), A branch and bound algorithm for the maximum clique problem. *Computers Ops Res.* 19, 363–375.

Pardalos, P.M. and Xue, J. (1992), The maximum clique problem. *J. Global Optim.* 4, 301–328.

Rizzi, R. (2000), A short proof of matching theorem. *J. Graph Theory* 33, 138–139.

Shor, N.Z. (1990), Dual quadratic estimates in polynomial and Boolean programming. *Ann. Oper. Res.* 25, 163–168.

Sós, V.T. and E.G. Straus (1982), Extremal of functions on graphs with applications to graphs and hypergraphs. *J. Combin. Theory* B32, 246–257.

Trotter, L. Jr. (1973), Solution characteristics and algorithms for the vertex packing problem. Technical Report 168, Dept. of Operatiosn Research, Cornell University, Ithaca, NY.

Wei, V.K. (1981), A lower bound on the stability number of a simple graph. Technical Report TM 81-11217-9, Bell Laboratories.