Application of the Lovász-Schrijver Operator to Compact Stable Set Integer Programs

Federico Battista[*1], Fabrizio Rossi[†2], and Stefano Smriglio[‡2]

[1]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, 18015, USA
[2]DISIM, University of L'Aquila, Coppito (AQ), 67100, Italy

October 28, 2025

Abstract

The Lovász theta function $\theta(G)$ provides a very good upper bound on the stability number of a graph G. It can be computed in polynomial time by solving a semidefinite program (SDP), which also turns out to be fairly tractable in practice. Consequently, $\theta(G)$ achieves a hard-to-beat trade-off between computational effort and strength of the bound. Indeed, several attempts to improve the theta bound are documented, mainly based on playing around the application of the $N_+(\cdot)$ lifting operator of Lovász and Schrijver to the classical formulation of the maximum stable set problem (SSP). Experience shows that solving such SDPs often struggles against practical intractability and requires highly specialized methods. We investigate the application of such an operator to two different linear formulations of the SSP based on clique and nodal inequalities, respectively. These two formulations are described by fewer inequalities than the natural formulation based on edge inequalities, yet they guarantee that the resulting SDP bound is at least as strong as $\theta(G)$. Our computational experience, including larger graphs than those previously documented, shows that upper bounds stronger than $\theta(G)$ can be accessed by a reasonable additional effort using the clique-based formulation on sparse graphs and the nodal-based one on dense graphs.

Keywords semidefinite programming · lift-and-project operator · stable set problem;
MSC 90C22 · 90C27 · 05C69;

1 Introduction

Given a simple undirected graph $G = (V, E)$, with vertex set V and edge set E, a *stable* (or *independent*) *set* in G is a set of pairwise non-adjacent vertices. A fundamental problem in combinatorial optimization is the *Stable Set Problem* (SSP), which consists of computing a stable

---

[*]feb223@lehigh.edu

[†]fabrizio.rossi@univaq.it

[‡]stefano.smriglio@univaq.it

set of maximum cardinality [33, 50]. This value, denoted by $\alpha(G)$, is the *stability number* of G. For the sake of presentation, we do not explicitly address the weighted version of this problem, where a weight vector $w \in \mathbb{Q}_+^{|V|}$ is given, although all the results presented remain valid in that setting. The SSP is NP-hard in the strong sense and even hard to approximate [35], it is equivalent to the *Maximum Clique* and *Set Packing* problems [53] and has a wide range of applications in different practical contexts, see, e.g., those illustrated in [14, 16, 57] and those surveyed in [10, 68]. The SSP is naturally formulated as a binary program by the so-called *edge inequalities*

$$
\begin{aligned}
\alpha(G) = \max \quad & \sum_{i \in V} x_i \\
\text{s.t.} \quad & x_i + x_j \leq 1, \quad \{i, j\} \in E, \\
& x_i \in \{0, 1\}, \quad i \in V.
\end{aligned}
\tag{1}
$$

As is customary in the literature, we denote by

$$
\text{FRAC}(G) = \{x \in [0, 1]^{|V|} : (1) \text{ hold}\}
$$

the *fractional stable set polytope*, while

$$
\text{STAB}(G) = \text{conv}\{x \in \{0, 1\}^{|V|} : (1) \text{ hold}\}
$$

denotes the *stable set polytope*, that is, the convex hull of the incidence vectors of all stable sets in G. The intense study of $\text{STAB}(G)$ has led, since the early seventies [2], to the identification of several classes of valid inequalities, often associated with specific "facet-producing" sub-graphs [11, 15, 33]. In practice, however, linear relaxations based on these inequalities often yielded rather weak upper bounds on the stability number of unstructured graphs. As a consequence, basic branch-and-cut algorithms are not particularly effective [49] and much more sophisticated cut generation procedures, see [19, 20, 30, 31, 55, 56], are required to reduce the performance gap with fast combinatorial exact algorithms, such as those devised in [52, 58, 62].

In his seminal paper, Lovász [42] introduced the celebrated *theta number* of a graph, denoted by $\theta(G)$. It represents an upper boundon $\alpha(G)$, which can be computed in polynomial time (up to arbitrary precision) by solving a Semidefinite Program (SDP). Since its introduction, $\theta(G)$, along with the associated non-polyhedral convex *theta body* $\text{TH}(G)$, turned out to be powerful tools. For instance, they allowed to prove that the SSP can be solved in polynomial time when G is a perfect graph. For the description of these results we refer the reader to the book of Grötschel et al. [33]. Indeed, the practical relevance of $\theta(G)$ is no less. On one side, it often turns out to be a significantly stronger upper bound than those from linear relaxations, as documented in [28, 39]. On the other side, the structure of the associated SDP relaxation often mitigates numerical difficulties that typicallyaffects SDP algorithms. As a consequence, $\theta(G)$ realizes a very good trade-off between the quality of the upper bound and computational effort, which makes it attractive toincorporate it within branch-and-bound algorithms, as studied by, e.g., Wilson [67] and more recently by Gaar et al. [25].

In a successive landmark paper Lovász and Schrijver [43] introduced the $N_+(\cdot)$ *lift-and-project* operator, which can be applied to the LP relaxation of any 0-1 LP and returns a stronger SDP relaxation. Its application to $\text{FRAC}(G)$ has also been intensively investigated and has resulted in a tighter relaxation than the Lovász theta relaxation. From a theoretical perspective, its strength has been documented by showing implications of (additional) facet defining inequalities of the stable set polytope (Giandomenico and Letchford [27], Lovász and Schrijver [43]). Moreover, Bianchi et al.

[7, 8] showed that it provides an effective tool to identify other classes of graphs for which the SSP can be solved in polynomial time. Recently, graph classes yielding stable set polytopes with high rank with respect to $N_+(\cdot)$ have been studied in [1].

From the computational point of view, things are somewhat more involved. Experiments conducted with $N_+(\mathrm{FRAC}(G))$ by Dash [21], Burer and Vandenbussche [12] and with related variants by Dukanovic and Rendl [23], Gruber and Rendl [34], show that these relaxations may yield stronger upper bounds than $\theta(G)$ but at the price of considerably increasing the computational effort. In fact, solving these large-scale SDPs requires specialized methods, and large instances can hardly be managed. Other methods, based on strengthening SDP relaxations by linear inequalities not related to the $N_+(\cdot)$ operator, have also been presented by Gaar and Rendl [24] and Locatelli [41]. These achieve strong bounds but still require significant computational effort.

Overall, we currently face a dichotomic picture, where even minimal progress from the Lovász θ bound may be paid with a very high computational burden. This paper aims to present new SDP relaxations that can bridge such a gap. These are derived by applying the $N_+(\cdot)$ operator to alternative LP relaxations. The first relaxation is obtained by replacing edge inequalities with (clearly stronger) clique inequalities. This is unconventional since the linear description may have exponentially many inequalities, making the approach somehow unattractive from theoretical and practical perspectives. However, we show that a natural selection of clique inequalities allows for more compact, tractable, yet stronger, relaxations. An opposite rationale derives the other SDP relaxations: accepting to start from more compact but weaker LP relaxations and relying on the ability of the $N_+(\cdot)$ operator to recover their potential weakness. Based on this rationale, we investigate a hierarchy of SDP relaxations where the primary level consists of applying the operator to a surrogate relaxation of $\mathrm{FRAC}(G)$, introduced in Della Croce and Tadei [22], with $O(|V|)$ (so-called *nodal*) inequalities. Coefficient strengthening procedures then derive two progressively stronger relaxations, the first of which takes polynomial time while the second requires the solution of NP-hard (sub-)problems to be built.

In this work, we first prove that all the proposed SDP relaxations are at least as strong as the Lovász theta relaxation and draw a complete theoretical picture by comparing them to each other and to $N_+(\mathrm{FRAC}(G))$. Then, we discuss implementation issues in handling them and illustrate extensive experiments where the practical strength of relaxations is evaluated and contrasted to the theoretical expectation. We show that significant progress with respect to $\theta(G)$ is achieved in several cases. Relaxations obtained from clique inequalities are more effective for sparse graphs, while those from nodal inequalities are the best as graph density grows. Furthermore, the increase of computational workload required to improve the θ bound is significantly smoother than that of previous approaches. Indeed, this increased efficiency allowed us experimentation based on significantly larger graphs than previously documented. This experience also sheds new light on the practical potential of the $N_+(\cdot)$ operator, which emerges more clearly than in the studies documented so far. All the material and code presented is available at `https://github.com/febattista/SDP_lift_and_project` and organized to facilitate the reproduction of relaxations and experiments.

2 Preliminaries

This paper will adopt the following standard notation. The set of integers $\{1, ..., k\}$ is denoted by $[k]$. The *complement* of a simple graph $G = (V, E)$ is the graph $\bar{G} = (V, \bar{E})$, where $\bar{E} = \{\{i, j\} \subset V : \{i, j\} \in (V \times V) \setminus E, i \neq j\}$. Given a vertex $i \in V$, we let $\Gamma_G(i)$ denote the set of *neighbors*

of i in G, i.e., $\Gamma_G(i) = \{j \in V \setminus \{i\} : \{i,j\} \in E\}$. When no confusion arises, we let $\Gamma_G(i) = \Gamma(i)$. Finally, given a set $S \subseteq V$, we let $G[S] = (S, E(S))$ denote the subgraph of G induced by S, and $r(S)$ denote its stability number $\alpha(G[S])$ (the so-called *rank* of S). We also let $\mathcal{S}_n$ be the set of real symmetric square matrices of order n and $\mathcal{S}_n^+ \subset \mathcal{S}_n$ denotes the cone of those which are positive semidefinite (PSD). PSDness of a matrix $Y \in \mathcal{S}_n^+$ is also denoted by $Y \succeq 0$. Given any square matrix $X \in \mathbb{R}^{n \times n}$ we denote by X_{ij} the element at the i-th row and the j-th column, by $\mathrm{diag}(X) \in \mathbb{R}^n$ its main diagonal and by $\mathrm{rank}(X)$ its rank. We denote by $\langle X, Y \rangle = \mathrm{tr}(XY)$ the standard inner product in $\mathcal{S}_n$, where $\mathrm{tr}(A)$ is the trace of the square matrix A. At last, we denote by J the all-ones square matrix and e the all-ones vector; the dimensions should be clear from the context where not explicit. Given $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$, the linear inequality $a^\top x \leq b$ is *valid* for $\mathrm{STAB}(G)$ if $\left\{ x \in \mathbb{R}^n \mid a^\top x \leq b \right\} \supseteq \mathrm{STAB}(G)$.

2.1 The lift-and-project operator of Lovász and Schrijver

We now review the general statement of the $N_+(\cdot)$ operator to make the presentation self-consistent, referring the reader to [18, 21, 32, 43] for an exhaustive treatment.

Let us consider a 0-1 linear program of the form

$$\max \left\{ c^\top x \mid Ax \leq b, \ x_i \in \{0,1\} \text{ for } i \in [n] \right\}, \tag{IP}$$

where $c \in \mathbb{R}^n$, the inequality system $Ax \leq b$ has $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ and contains the inequalities $0 \leq x_i \leq 1$, for $i \in [n]$. Each linear inequality of the system $Ax \leq b$ is denoted by $(a^k)^\top x \leq b^k$, for $k \in [m]$. The polytope $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ is the linear relaxation of (IP) and $P_I = \mathrm{conv}\{x \in \{0,1\}^n \mid Ax \leq b\}$ denotes the convex hull of the feasible solutions to (IP).

The application of the *lift-and-project* operator to P involves the three following steps:

Step 1 (Lifting)

Generate the quadratic inequalities

$$x_i \cdot ((a^k)^\top x - b^k) \leq 0 \quad i \in [n], k \in [m], \tag{2}$$

$$(1 - x_i) \cdot ((a^k)^\top x - b^k) \leq 0 \quad i \in [n], k \in [m]. \tag{3}$$

which are valid for P as $x_i \geq 0$ and $1 - x_i \geq 0$ for all $x \in P$. These inequalities can be linearized through the introduction of the matrix variable $X = xx^\top$ and (2) and (3) can be rewritten as

$$\sum_{j \in I} a_j^k X_{ij} - b^k x_i \leq 0 \quad i \in [n], k \in [m], \tag{4}$$

$$\sum_{j \in I} a_j^k x_j - b^k - \sum_{j \in I} a_j^k X_{ij} + b^k x_i \leq 0 \quad i \in [n], k \in [m], \tag{5}$$

$$X = xx^\top.$$

Then, the quadratic constraint $X = xx^\top$ is relaxed with $X - xx^\top \succeq 0$. By applying the Schur's complement definition $X - xx^\top \succeq 0$ is equivalent to

$$\begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0. \tag{6}$$

4

Step 2 (Strengthening)

Add to constraints (4)–(6)

$$X_{ii} = x_i \qquad i \in [n], \tag{7}$$

as any binary vector $x \in P_I$ satisfies $x_i = x_i^2$, whereas such equality is not guaranteed for points in P. This yields to the definition of the convex set

$$M_+(P) = \left\{ \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0 \ \Big| \ (4), (5), (7) \text{ hold} \right\},$$

where $M_+(\cdot)$ is referred here as *lift operator*.

Step 3 (Projection)

The projection of $M_+(P)$ onto the original x-space is

$$N_+(P) = \left\{ x \in \mathbb{R}^n \ \Big| \ \exists \, Y \in M_+(P) \text{ with } \begin{pmatrix} 1 \\ x \end{pmatrix} = \text{diag}(Y) \right\}.$$

Lovász and Schrijver [43] showed that $N_+(P)$ is contained in P and is a valid relaxation for P_I, i.e. $P_I \subseteq N_+(P) \subseteq P$ holds. For the ease of presentation, in the reminder of this paper we will refer to both $M_+(P)$ and $N_+(P)$ as a relaxation of P_I, despite the former "lives" in the lifted variables space. Furthermore, note that all the linear inequalities $Ax \leq b$ are trivially satisfied by points in $M_+(P)$.

Remark 1. *Let $(a^k)^\top x \leq b^k$, for $k \in [m]$, be an inequality in the linear system defining P, then from the linear inequalities defining $M_+(P)$ one can sum (4) and (5), for any $i \in [n]$, to recover the original constraint.*

Moreover, Bodur et al. [9] noted that the definition of $M_+(P)$ and $N_+(P)$ does not depend on the representation of P but only on the set of points in P. Hence, the following property holds

Property 1. *Given two polytopes P, P' with $P' \subseteq P$ then $M_+(P') \subseteq M_+(P)$.*

Additionally, given any polytope P containing $\text{STAB}(G)$, Lovász and Schrijver [43] provided necessary conditions for an inequality $a^\top x \leq b$ to be valid for $N_+(P)$.

Lemma 1. *Let $a^\top x \leq b$ be a valid inequality for $\text{STAB}(G)$. W.l.o.g. assume $a \in \mathbb{R}_+^n$ and $b \in \mathbb{R}_+$. If $a^\top x \leq b$ is valid for $P \cap \{x : x_i = 1\}$ for all $i \in V$, then $a^\top x \leq b$ is valid for $N_+(P)$.*

3 Continuous relaxations of the SSP

In this section we recall known results about linear and semidefinite relaxation of the SSP. These are widely researched and we will only concentrate on elements relevant to our study.

3.1 Linear relaxations of the SSP

Several classes of valid inequalities for $\mathrm{STAB}(G)$ have been identified since early seventies, as is illustrated in [11, 33]. Among them, *clique* inequalities $\sum_{i \in C} x_i \leq 1$, where $C \subseteq V$ is a maximal clique (set of pairwise adjacent vertices), induce facets of $\mathrm{STAB}(G)$ [53]. As customary in the literature, we define the polytope

$$\mathrm{QSTAB}(G) = \{x \in [0,1]^{|V|} : \sum_{i \in C} x_i \leq 1, \ C \in \mathcal{K}\}$$

where $\mathcal{K}$ denotes the collection of all maximal cliques of G. As clique inequalities clearly imply edge inequalities, $\mathrm{STAB}(G) \subseteq \mathrm{QSTAB}(G) \subseteq \mathrm{FRAC}(G)$ and inclusion is generally strict [33]. Clique inequalities can be exponentially many, and the associated separation problem is NP-hard in the strong sense. However, effective separation heuristics make this class of inequalities computationally easy to handle, as shown in Marzi et al. [45]. We say that a clique $C \in \mathcal{K}$ *covers* an edge $\{i, j\} \in E$ if $i, j \in C$. An *edge clique cover* of G is a collection $\mathcal{C}$ of (not necessarily maximal) cliques of G such that each edge in E is covered by at least one clique in $\mathcal{C}$. Let $\mathcal{Q}$ be the set of all edge clique covers of G. Then, the 0-1 linear program

$$\begin{aligned} \max &\sum_{i \in V} x_i \\ \mathrm{s.t.} &\sum_{i \in C} x_i \leq 1 \quad C \in \mathcal{C} \\ &x_i \in \{0,1\} \quad i \in V. \end{aligned} \tag{8}$$

is a valid formulation for the SSP, for all $\mathcal{C} \in \mathcal{Q}$. Extending the standard notation, we denote by $\mathrm{QSTAB}(G, \mathcal{C})$ the polytope associated with its continuous relaxation. Unless stated otherwise, the results presented in the remainder of the paper are valid for any $\mathcal{C} \in \mathcal{Q}$. It is not hard to see that $\mathrm{QSTAB}(G) \subseteq \mathrm{QSTAB}(G, \mathcal{C}) \subseteq \mathrm{FRAC}(G)$. If $\mathcal{C} \in \mathcal{Q}$ is a *minimal* edge clique cover, then $|\mathcal{C}| \leq |E|$.

In practice, formulations based on greedily computed edge clique covers $\mathcal{C}$, which are made up of maximal cliques, contain considerably fewer inequalities than those describing FRAC(G), and also provide a good approximation of $\mathrm{QSTAB}(G)$, as shown in several studies (see, e.g., Letchford et al. [40] and references therein).

Other *rank inequalities* of the form $\sum_{i \in S} x_i \leq \alpha(G[S])$ have been studied for vertex subsets $S \subseteq V$ inducing special graphs (see e.g., Borndörfer [11] and Giandomenico et al. [28] for a survey). A well-known example arises when $S = H$, $|H| \geq 5$, induces a chordless cycle of odd cardinality, yielding the *odd-hole* inequality $\sum_{i \in H} x_i \leq \lfloor |H|/2 \rfloor$. Among other subgraphs, we recall *webs* and *antiwebs* introduced by Trotter Jr [63]. Let p and q be integers satisfying $p > 2q + 1$ and $q > 1$, and use arithmetic modulo p. A $(p, q)-$web is a graph with vertex set $\{1, ..., p\}$ and with edges from i to $\{i + q, \ldots, i - q\}$, for every $1 \leq i \leq p$. A $(p, q)-$antiweb is the complement of a $(p, q)-$web. The web inequalities take the form $\sum_{i \in W} x_i \leq q$ for every vertex set W inducing a (p, q)-web, and the antiweb inequalities take the form $\sum_{i \in AW} x_i \leq \lfloor p/q \rfloor$ for every vertex set AW inducing a (p, q)-antiweb.

A different modeling rationale has been pursued by Della Croce and Tadei [22] who considered the *surrogate* relaxation of formulation based on (1) obtained by summing up, for every $i \in V$, the edge inequalities over all $j \in \Gamma(i)$. This yields the so called *nodal inequality* $\sum_{j \in \Gamma(i)} x_j + |\Gamma(i)|x_i \leq |\Gamma(i)|$. Murray and Church [48] observed that this relaxation can be strengthened by replacing $|\Gamma(i)|$ with any value $r_i \geq \alpha(G[\Gamma(i)])$, which returns a class of 0-1 linear programs of the form

$$\max \sum_{i \in V} x_i$$
$$\text{s.t.} \sum_{j \in \Gamma(i)} x_j + r_i\, x_i \leq r_i \quad i \in V \tag{9}$$
$$x_i \in \{0, 1\} \quad i \in V.$$

The strongest formulation, corresponding to $r_i = \alpha(G[\Gamma(i)])$, has been extensively investigated in [40]. Of course, computing coefficients $r_i = \alpha(G[\Gamma(i)])$ is NP-hard, as it amounts to solve the SSP on the subgraphs $G[\Gamma(i)]$. Nevertheless, Letchford et al. [40] showed that these formulations can be handled efficiently for many graphs of interest. Here, we are interested in a further choice of coefficients r_i, namely, $r_i = \theta(G[\Gamma(i)])$ where $\theta(G[\Gamma(i)])$ is the Lovász [42] θ number of the graph $G[\Gamma(i)]$. This represents an upper bound to $\alpha(G[\Gamma(i)])$ which can be computed in polynomial time up to an arbitrary precision, as reviewed in Section 3.2.1. Then, we introduce the following notation

$$\mathrm{NOD}(G, r) = \left\{ x \in [0, 1]^{|V|} : \sum_{j \in \Gamma(i)} x_j + r_i\, x_i \leq r_i, \; i \in V \right\}$$

to denote the polytope defined from (9) (and non-negativity), where $r \in \{\Gamma, \theta, \alpha\}$ specifies the coefficient used. Clearly, we have

$$\mathrm{NOD}(G, \alpha) \subseteq \mathrm{NOD}(G, \theta) \subseteq \mathrm{NOD}(G, \Gamma).$$

It is worth noting that nodal inequalities in general do not imply edge inequalities, even in the case $r = \alpha$. On the other hand, if $r = \alpha$, they imply the *wheel inequalities* (see Borndörfer [11] for their definition) which are not implied by clique inequalities.

3.2 SDP relaxations of the SSP

Three well-known SDP relaxations are related to our study.

3.2.1 The Lovász theta relaxation

The Lovász theta relaxation [42] has been introduced in 1979, yet we refer to the presentation of Lovász and Schrijver [43]. Introduce the quadratic variable X_{ij}, representing the product $x_i x_j$ for all $\{i, j\} \in V \times V$ and let $X = xx^\top$ be the associated symmetric matrix of order $n = |V|$. An upper boundon $\alpha(G)$ is given by

$$\theta(G) = \max \sum_{i \in V} x_i$$
$$\text{s.t.} \quad X_{ii} = x_i, \quad i \in V, \tag{SDP-θ}$$
$$X_{ij} = 0, \quad \{i, j\} \in E,$$
$$\begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0.$$

The projection TH(G) of the feasible region of (SDP-θ) onto the subspace of the x_i variables is known as the *theta body*, a convex but not a polyhedral set in general. Remarkably, Grötschel et al. [33] proved that STAB(G) $\subseteq$ TH(G) $\subseteq$ QSTAB(G), where equality holds if and only if G is perfect. This implies that there exists a polynomial-time separation algorithm for a class of inequalities which includes all clique inequalities. A thorough comparison between the original formulation in [42] and (SDP-θ) is provided by Galli and Letchford [26]. $\theta(G)$ can be computed in polynomial time up to an arbitrary precision and often provides a strong bound to the stability number $\alpha(G)$, typically better than those obtained from linear relaxations see, e.g., [28, 30, 38, 69]. Another interesting feature of (SDP-θ) concerns its computational behavior. In fact, (SDP-θ) is handled better than similarly sized unstructured SDPs by general SDP solvers. Moreover, several algorithms have been designed to solve it efficiently as in, e.g., [44, 54, 29].

Therefore, $\theta(G)$ achieves a good compromise between strength of the upper bound and computational burden. From a theoretical viewpoint, Busygin and Pasechnik [13] showed that, unless P=NP, no polynomially computable upper bound to $\alpha(G)$ which is provably smaller than $\theta(G)$ can be found. However, several stronger relaxations have been obtained by adding linear inequalities to (SDP-θ), as reviewed in the following subsections.

3.2.2 The Schrijver relaxation

Schrijver [59] observed that the PSD condition in (SDP-θ) does not imply the non-negativity on X. Hence, by adding the inequalities

$$X_{ij} \geq 0, \quad \{i,j\} \in \bar{E} \tag{SDP-θ^+a}$$

to (SDP-θ), one yields a model that we denote as (SDP-θ^+) and the corresponding upper bound as $\theta^+(G)$. Accordingly, TH$^+$(G) denotes the convex set obtained by its projection onto the subspace of x_i variables.

3.2.3 The Lovász-Schrijver relaxation

The Lovász-Schrijver relaxation for the SSP is obtained by applying the operator $N_+(\cdot)$ to FRAC(G). $N_+(\text{FRAC}(G))$ corresponds to (SDP-θ) plus the following linear inequalities:

$$X_{ij} \geq 0, \qquad \{i,j\} \in \bar{E}, \tag{11a}$$

$$X_{ik} + X_{jk} \leq x_k, \qquad \{i,j\} \in E, k \neq i,j, \tag{11b}$$

$$x_i + x_j + x_k \leq 1 + X_{ik} + X_{jk}, \quad \{i,j\} \in E, k \neq i,j. \tag{11c}$$

Lovász and Schrijver [43] assessed the theoretical strength of $N_+(\text{FRAC}(G))$ by showing that it satisfies several well-known classes of valid inequalities for STAB(G). Namely, they proved that $N_+(\text{FRAC}(G))$ satisfies all clique, odd cycle, odd antihole and odd wheel inequalities. Giandomenico and Letchford [27] showed that, in addition, it satisfies all web inequalities. Optimizing $\sum_{i \in V} x_i$ over $N_+(\text{FRAC}(G))$ returns an upper boundon $\alpha(G)$ that we denote as $\lambda(G)$. From a more general perspective, it is interesting to observe the key role of the positive semidefiniteness constraint. In fact, Burer and Vandenbussche [12] compared $\lambda(G)$ with the optimal value of the corresponding LP relaxation obtained by the Sherali-Adams procedure, observing a substantial improvement due to it.

Handling such a strong relaxation in practice is not straightforward. The first related computational results, obtained by a lift-and-project cutting plane method, appear in [3]. Results

onoptimizing directly over $M_+(\mathrm{FRAC}(G))$ are illustrated in Dash [21] and, more recently, in Burer and Vandenbussche [12] using aspecialized augmented Lagrangian method. The resulting upper boundson $\alpha(G)$ may be considerably better than $\theta^+(G)$ but at the expense of a large increase in running times. In practice, this relaxation turns out to be hardly tractable for graphs of medium/large size.

3.2.4 Further SDP relaxations

Variants of the previous relaxations are presented in Dukanovic and Rendl [23] and in Gruber and Rendl [34]. In the former, extra triangle inequalities are added to $N_+(\mathrm{FRAC}(G))$ to obtain a very strong relaxation. In the latter, relaxations are specialized with respect to graph density to improve efficiency. We refer to Galli and Letchford [26] for an exhaustive review of these methods. Two SDP-related methodologies are also presented in the literature. In Gaar and Rendl [24], strong SDP relaxations of several combinatorial optimization problems are investigated by exploiting the *exact subgraph inequalities*. Despite the power of this technique, the experience reveals that strengthening the natural SDP relaxation of the SSP is typically harder than other problems, e.g., *max-cut*, and confirms that improving the θ bound towards $\alpha(G)$, even by a small amount, is actually challenging. Another effective method (presented in the context of the clique number) based on adding non-valid inequalities has been developed in Locatelli [41]. This turns out to achieve quite strong bounds for both structured and random graphs with up to 150 vertices.

Summarizing, the evidence from state-of-the-art methodologies is that achieving even a small improvement of $\theta(G)$ requires a considerable additional computational cost. Namely, adding extra linear inequalities in the SDP models yields much harder SDPs and does require specialized methods. In the next section we introduce new relaxations with the goal of bridging such a gap.

4 New SDP relaxations

We now apply the lift operator $M_+(\cdot)$ to $\mathrm{QSTAB}(G,\mathcal{C})$ and $\mathrm{NOD}(G,r)$ and compare the resulting SDP relaxations. Let us first observe that all of them contain variable bound constraints, the lifting step of which consists in multiplying $0 \leq x_i \leq 1$ by x_j and $(1-x_j)$, for $i,j \in V, i < j$:

$$x_i x_j \rightarrow \quad X_{ij} \geq 0, \tag{12a}$$
$$x_i(1-x_j) \rightarrow \quad x_i \geq X_{ij}, \tag{12b}$$
$$(1-x_i)x_j \rightarrow \quad x_j \geq X_{ij}, \tag{12c}$$
$$(1-x_i)(1-x_j) \rightarrow \quad x_i + x_j - X_{ij} \leq 1. \tag{12d}$$

The linear inequalities (12a)–(12d) are known as McCormick [46] inequalities and describe the convex hull of the set $\{(x_i, x_j, y_{ij}) \in \{0,1\}^3 : y_{ij} = x_i x_j\}$. It is easy to see that they are not implied by the PSD condition (6) and should be explicitly added to the relaxations illustrated below unless implied by other constraints.

4.1 Relaxation $M_+(\mathrm{QSTAB}(G,\mathcal{C}))$

In order to investigate $M_+(\mathrm{QSTAB}(G,\mathcal{C}))$ we first describe the application of the operator $M_+(\cdot)$ to $\mathrm{QSTAB}(G)$. The lifting step consists in multiplying each clique inequality $\sum_{i\in C} x_i \leq 1$, $C \in \mathcal{K}$, by x_j and $(1-x_j)$, for all $j \in V$. Then, adding the condition $X_{ii} = x_i$, one obtains

$$x_j \cdot \left(\sum_{i \in C} x_i \le 1 \right) \rightarrow \begin{cases} \displaystyle\sum_{i \in C \setminus \{j\}} X_{ij} \le 0, & \text{if } j \in C, \quad (13\text{a}) \\[2ex] \displaystyle\sum_{i \in C} X_{ij} - x_j \le 0, & \text{if } j \notin C, \quad (13\text{b}) \end{cases}$$

$$(1 - x_j) \cdot \left(\sum_{i \in C} x_i \le 1 \right) \rightarrow \begin{cases} \displaystyle\sum_{i \in C} x_i - \sum_{i \in C \setminus \{j\}} X_{ij} \le 1, & \text{if } j \in C, \quad (13\text{c}) \\[2ex] \displaystyle\sum_{i \in C} (x_i - X_{ij}) + x_j \le 1, & \text{if } j \notin C. \quad (13\text{d}) \end{cases}$$

Inequalities (13b), called *clique-variable* inequalities, have been introduced in [28]. One can observe that

1. Constraints (13a) imply $X_{ij} = 0$ for all $\{i,j\} \in E$ and constraints (13c) reduce to the corresponding clique inequality $\sum_{i \in C} x_i \le 1, C \in \mathcal{K}$, which is already implied from (13b) and (13d) by Remark 1;

2. $X_{ij} = 0$ for all $\{i,j\} \in E$, $X_{ij} \ge 0$ for all $\{i,j\} \in \bar{E}$ and constraints (13b) imply inequalities (12b) and (12c);

3. Constraints (13a), (13b) and (13d) imply (12d).

Then, the description of $M_+(\text{QSTAB}(G))$ reduces to

$$M_+(\text{QSTAB}(G)) = \left\{ \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0 : \right.$$
$$\sum_{i \in C} X_{ij} - x_j \le 0, \quad C \in \mathcal{K}, j \in V \setminus C,$$
$$\sum_{i \in C} (x_i - X_{ij}) + x_j \le 1, \quad C \in \mathcal{K}, j \in V \setminus C,$$
$$x_i = X_{ii}, \quad i \in V,$$
$$X_{ij} = 0, \quad \{i,j\} \in E,$$
$$X_{ij} \ge 0, \quad \{i,j\} \in \bar{E}$$
$$\left. x_i \ge 0, \quad i \in V \right\}.$$

Moreover, one has

Theorem 1. $M_+(\text{QSTAB}(G)) \subseteq M_+(\text{FRAC}(G))$.

Proof Since $\text{QSTAB}(G) \subseteq \text{FRAC}(G)$, the statement follows from Property 1. $\qquad\square$

We denote the resulting upper bound on $\alpha(G)$ by

$$\mu(G) = \max \left\{ \sum_{i \in V} x_i : \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \in M_+(\text{QSTAB}(G)) \right\}$$

The following example shows that the containmentof Theorem 1 can be strict.

Example 1. *Let G be the $(10, 3)$-antiweb graph of Figure [2a]. We have $\mu(G) = 3$. Differently, optimizing the cardinality objective function $\sum_{i \in V} x_i$ over $N_+(\text{FRAC}(G))$ one obtains the optimal solution $x_F = (0.3106, \ldots, 0.3106)$ with value $z_F^* = 3.106$. Therefore, $x_F \in N_+(\text{FRAC}(G))$ lies outside $N_+(\text{QSTAB}(G))$.*

The example indeed suggests a general result. In fact, we can show that, unlike $N_+(\text{FRAC}(G))$, the relaxation $N_+(\text{QSTAB}(G))$ implies the class of antiweb inequalities.

Theorem 2. $N_+(\text{QSTAB}(G))$ *implies antiweb inequalities.*

Proof Let G be the (p, q)-antiweb graph with $V = \{1, \ldots, p\}$. In what follows, we use arithmetic modulo p. Note that, by definition, any consecutive q nodes in G form a clique. By denoting $\alpha = \lfloor p/q \rfloor$, the (p, q)-antiweb inequality is

$$\sum_{i \in V} x_i \leq \alpha.$$

Given any $i \in V$, we denoteby $G \setminus i$ the subgraph induced by the vertex set

$$V(G \setminus i) = V \setminus (\{i\} \cup \Gamma(i)) = \{i + q, \ i + q + 1, \ldots, \ i - q - 1, \ i - q\}.$$

The goal is to prove that inequality

$$\sum_{j \in V(G \setminus i)} x_j \leq \alpha - 1, \tag{14}$$

is valid for $\text{QSTAB}(G \setminus i)$, for all $i \in V$, then the theorem's statement will follow from Lemma [1]. Let i be any node in V and consider the following $\alpha - 1$ cliques in $G \setminus i$

$$\begin{aligned}
C^1 &= \{i + q, \ i + q + 1, \ldots, \ i + 2q - 1\}, \\
C^2 &= \{i + 2q, \ i + 2q + 1, \ldots, \ i + 3q - 1\}, \\
&\vdots \\
C^k &= \{i + kq, \ i + kq + 1, \ldots, i + (k + 1)q - 1\}, \\
&\vdots \\
C^{\alpha - 1} &= \{i + (\alpha - 1)q, \ i + (\alpha - 1)q + 1, \ldots, \ i - q\}.
\end{aligned}$$

Note that the set $\{C^1, \ldots, C^{\alpha - 1}\}$ forms a partition of $V(G \setminus i)$, i.e.,

$$\bigcup_{k = 1, \ldots, \alpha - 1} C^k = V(G \setminus i)$$

and $C^\ell \cap C^k = \emptyset$, for all $\ell, k = 1, \ldots, \alpha - 1$, with $\ell \neq k$. Hence, the sum of clique inequalities $\sum_{j \in C^k} x_j \leq 1$, for $k = 1, \ldots, \alpha - 1$yields the inequality

$$\sum_{i + q \leq j \leq i - q} x_j = \sum_{j \in V(G \setminus i)} x_j \leq \alpha - 1,$$

which is valid for $\text{QSTAB}(G \setminus i)$. By symmetry, this argument is independent of the choice of i. $\square$

Theorem 1 completes the theoretical picture for $N_+(\text{QSTAB}(G))$ along the line drawn by Lovász and Schrijver [43]. In fact, all of the traditional combinatorial inequalities, namely, clique, odd cycle, odd antihole, odd wheel, web and antiweb inequalities turn out to be satisfied by $N_+(\text{QSTAB}(G))$. It is interesting to mention that Giandomenico et al. [28] proved the same result for a non-compact linear relaxation obtained by the application of the $N(K,K)$ lifting operator by Lovász and Schrijver to $\text{QSTAB}(G)$. The comparison between $N(K,K)$ and $N_+(\cdot)$ was indeed suggested by Lovász and Schrijver [43] as an interesting open issue.

Another insight of the progress of $N_+(\text{QSTAB}(G))$ towards $N_+(\text{FRAC}(G))$ stems from the analysis of graph $\hat{G}$ of Figure 1 which has been shown in [6] that $N_+(\text{FRAC}(\hat{G}))$ gives rise to a non-polyhedral relaxation. Here we have $\lambda(\hat{G}) = 2.146$. Now, optimizing over $N_+(\text{QSTAB}(\hat{G}))$ one gets the stability number $\mu(\hat{G}) = \alpha(\hat{G}) = 2$. This may suggest that $N_+(\text{QSTAB}(\hat{G}))$ is actually polyhedral.
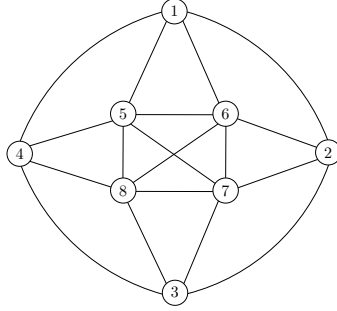
Figure 1: The $\hat{G}$ graph.

From a computational perspective, a major issue of $N_+(\text{QSTAB}(G))$ is that the linear description of $\text{QSTAB}(G)$, in general, contains exponentially many inequalities. To get rid of this issue, we consider $\text{QSTAB}(G,\mathcal{C})$, for any edge clique cover $\mathcal{C} \in \mathcal{Q}$. The application of the lifting operator to this relaxation yields the convex set

$$M_+(\text{QSTAB}(G,\mathcal{C})) = \left\{ \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0 \, : \, x_i = X_{ii}, i \in V \text{ and } (13a) - (13d) \text{ hold} \right\}.$$

Since $\text{QSTAB}(G,\mathcal{C}) \subseteq \text{FRAC}(G)$, we have $M_+(\text{QSTAB}(G,\mathcal{C})) \subseteq M_+(\text{FRAC}(G))$. Also note that $M_+(\text{QSTAB}(G,\mathcal{C}))$ still implies McCormick inequalities (12a)–(12d). Remarkably, it can be built in polynomial time as a minimal edge clique cover $\mathcal{C} \in \mathcal{Q}$ can be determined by a greedy algorithm (e.g., the one proposed in Letchford et al. [40]), and the number of inequalities describing $\text{QSTAB}(G,\mathcal{C})$ is at most $|E|$ (for increasing graph density, it is also considerably smaller than $|E|$). Optimizing $\sum_{i \in V} x_i$ over $M_+(\text{QSTAB}(G,\mathcal{C}))$ returns an upper bound that we denote as $\mu(G,\mathcal{C})$. We will experiment with such a relaxation in Section 6.

4.2 Relaxation $N_+(\text{NOD}(G,r))$

Let us now consider a generic nodal inequality, with $r_i \geq \alpha(G[\Gamma(i)])$, for all $i \in V$. The lifting step along with $x_i = X_{ii}$ returns:

$$x_j \cdot \left(\sum_{k \in \Gamma(i)} x_k + r_i x_i \leq r_i \right) \rightarrow$$

12

$$
\rightarrow \begin{cases} \sum_{k\in\Gamma(i)} X_{ik} \le 0, & \text{if } j = i \quad (15\text{a}) \\[2mm] (1-r_i)x_j + \sum_{k\in\Gamma(i)\setminus j} X_{jk} + r_i X_{ij} \le 0, & \text{if } j \in \Gamma(i) \quad (15\text{b}) \\[2mm] \sum_{k\in\Gamma(i)} X_{jk} + r_i X_{ij} - r_i x_j \le 0, & \text{if } j \notin \Gamma(i) \quad (15\text{c}) \end{cases}
$$

$$
(1 - x_j) \cdot \left(\sum_{k\in\Gamma(i)} x_k + r_i x_i \le r_i \right) \rightarrow
$$

$$
\rightarrow \begin{cases} \sum_{k\in\Gamma(i)} (x_k - X_{ik}) + r_i x_i \le r_i, & \text{if } j = i \quad (16\text{a}) \\[2mm] \sum_{k\in\Gamma(i)\setminus j} (x_k - X_{jk}) + r_i x_i + r_i x_j - r_i X_{ij} \le r_i, \text{if } j \in \Gamma(i) \quad (16\text{b}) \\[2mm] \sum_{k\in\Gamma(i)} (x_k - X_{jk}) + r_i x_i + r_i x_j - r_i X_{ij} \le r_i, & \text{if } j \notin \Gamma(i) \quad (16\text{c}) \end{cases}
$$

One can observe that:

1. Constraints (15a) imply $X_{ij} = 0$ for all $\{i,j\} \in E$; this is a relevant fact if one recalls that edge inequalities are not implied by nodal inequalities (9);

2. $X_{ij} = 0$, for all $\{i,j\} \in E$, $X_{ij} \ge 0$ for all $\{i,j\} \in \bar{E}$ and constraints (15c) imply inequalities (12b) and (12c);

3. $X_{ij} = 0$, for all $\{i,j\} \in E$, inequalities (12b) and (12c) and constraints (16b) imply edge inequalities (and (12d) for $\{i,j\} \in E$);

4. By $X_{ij} = 0$, for all $\{i,j\} \in E$, (16a) boils down to the original nodal inequality (9) and can be dropped, since Remark 1 can be applied to both pairs (15b), (16b) and (15c), (16c);

5. $X_{ij} \ge 0$, for all $\{i,j\} \in \bar{E}$, inequalities (12b) and (12c) and constraints (16c) imply (12d) for $\{i,j\} \in \bar{E}$.

Thanks to these properties $M_+(\text{NOD}(G,r))$ boils down to

$$M_+(\mathrm{NOD}(G,r)) = \left\{ \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0 : \right.$$

$$(1-r_i)x_j + \sum_{k\in\Gamma(i)\setminus j} X_{jk} \le 0, \quad i\in V, j\in\Gamma(i),$$

$$\sum_{k\in\Gamma(i)} X_{jk} + r_i X_{ij} - r_i x_j \le 0, \quad i\in V, j\notin\Gamma(i),$$

$$\sum_{k\in\Gamma(i)\setminus j} (x_k - X_{jk}) + r_i x_i + r_i x_j \le r_i, \quad i\in V, j\in\Gamma(i),$$

$$\sum_{k\in\Gamma(i)} (x_k - X_{jk}) + r_i x_i + r_i x_j - r_i X_{ij} \le r_i, \quad i\in V, j\notin\Gamma(i),$$

$$x_i = X_{ii}, \quad i\in V,$$
$$X_{ij} = 0, \quad \{i,j\}\in E,$$
$$X_{ij} \ge 0, \quad \{i,j\}\in\bar{E},$$
$$\left. x_i \ge 0, \quad i\in V \right\}.$$

Then, $M_+(\mathrm{NOD}(G,r))$ can be obtained by adding linear inequalities (15b), (15c), (16b), (16c) and $X_{ij}\ge 0$, for $\{i,j\}\in\bar{E}$, to (SDP-θ^+) (it is straightforward to observe that inequalities (15b) are implied by Mc-Cormick inequalities for $r=\Gamma$). Therefore, its projection $N_+(\mathrm{NOD}(G,r))$ onto the x-space is at least as strong as the Schrijver relaxation.

Theorem 3. $N_+(\mathrm{NOD}(G,r)) \subseteq \mathrm{TH}^+(G)$.

This result also implies

Corollary 1. $N_+(\mathrm{NOD}(G,r))$ *implies all clique inequalities.*

Corollary 2. $\nu(G,r) = \max\left\{ \sum_{i\in V} x_i : \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \in M_+(\mathrm{NOD}(G,r)) \right\} \le \theta^+(G)$.

Theorem 3 holds in particular for $r_i = |\Gamma(i)|$ corresponding to the case where the initial formulation is the surrogate relaxation of FRAC(G). Even in this case, applying the $N_+(\cdot)$ operator can recover the initial formulation's weakness and yield an upper bound that is at least as good as Schrijver's bound. Furthermore, a hierarchy of relaxations derives from strengthening the lifting coefficient r_i, that is, looking at $r_i = \theta(G[\Gamma(i)])$ and $r_i = \alpha(G[\Gamma(i)])$. Thanks to Property 1 we have

Theorem 4. $M_+(\mathrm{NOD}(G,\alpha)) \subseteq M_+(\mathrm{NOD}(G,\theta)) \subseteq M_+(\mathrm{NOD}(G,\Gamma))$.

This hierarchy poses a complexity issue. The strongest relaxation comes from applying the operator to an LP relaxation, the construction of which, in general, cannot be carried out in polynomial time. An interesting polynomial special case arises when the subgraphs induced by neighbor sets are perfect and, according to the result in [33], in this case, we have $\theta(G[\Gamma(i)]) = \alpha(G[\Gamma(i)])$, for all $i\in V$. Differently, $M_+(\mathrm{NOD}(G,\theta))$ can always be constructed (and solved) in polynomial time. Section 6 documents the practical trade-off between strength and computational tractability for all relaxations in this hierarchy.

4.3 Comparison among relaxations

A natural question concerns with relationships between $N_+(\mathrm{FRAC}(G))$, $N_+(\mathrm{QSTAB}(G))$ and $N_+(\mathrm{NOD}(G,r))$. The following example shows that neither $N_+(\mathrm{NOD}(G,r))$ contains $N_+(\mathrm{FRAC}(G))$ nor the reverse.

Example 2. *Consider the odd-hole with seven vertices of Figure (2b). According to [43], $N_+(\mathrm{FRAC}(G))$ satisfies the facet-defining inequality $\sum_{i\in V} x_i \leq 3$ which implies $\lambda(G) = 3$. Differently, the optimal value from $N_+(\mathrm{NOD}(G,r))$ is $\nu(G,\theta) = \nu(G,\alpha) = 3.317$ corresponding to solution $(0.47395,\ldots,0.47395)$, which therefore falls outside $N_+(\mathrm{FRAC}(G))$. However, one can find a feasible solution to $N_+(\mathrm{FRAC}(G))$ which is outside $N_+(\mathrm{NOD}(G,r))$ as well. Consider the $(10,3)$-antiweb graph G of Figure (2a). Here, $\nu(G,\theta) = \nu(G,\alpha) = \alpha(G) = 3$. On the contrary, optimizing over $N_+(\mathrm{FRAC}(G))$ we obtain the solution*

$$x_F = (0.31055,\ldots,0,31055)$$

with value 3.105 which is then outside $N_+(\mathrm{NOD}(G,r))$, for $r \in \{\theta,\alpha\}$.

(a) $(10,3)$–antiweb   (b) 7-hole
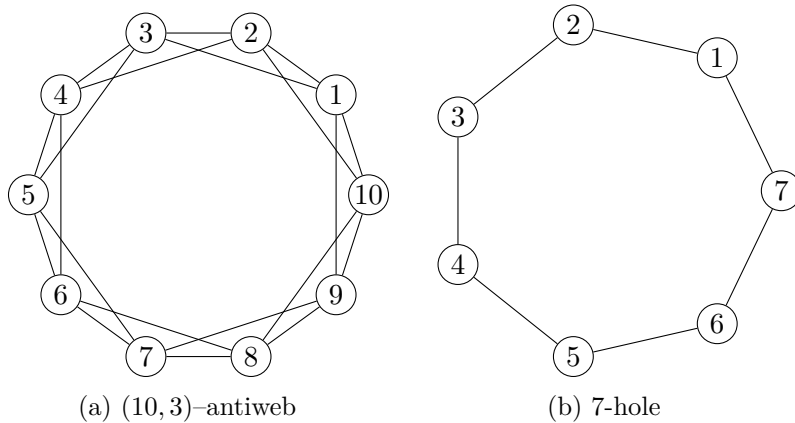
Figure 2: Two example graphs.

This example also shows that the odd-hole inequalities are not implied by $N_+(\mathrm{NOD}(G,r))$ even with $r = \alpha$. This appears as a weakness indicator of $N_+(\mathrm{NOD}(G,r))$ if one recalls that odd-hole inequalities are contained in the first Chvàtal closure of the formulation based on edge inequalities (1) [53]. The overall theoretical picture is summarized in Figure 3. The results related to the implication of odd-hole and antiweb inequalities seem to suggest that $N_+(\mathrm{QSTAB}(G))$ (as well as $N_+(\mathrm{FRAC}(G))$) should be stronger than $N_+(\mathrm{NOD}(G,r))$. Interestingly, the computational analysis will reveal significant supremacy of the latter as graphs get dense, for $r \in \{\theta,\alpha\}$.

5 Implementation

The software implemented for this work, along with the datasets used in the experiments (see Section 6), is available at `https://github.com/febattista/SDP_lift_and_project`. The repository comprises the following Python (`.py`) and MATLAB (`.m`) files:

Optimal value	Relaxation
$\lambda(G)$	$N_+(\text{FRAC}(G))$
$\mu(G)$	$N_+(\text{QSTAB}(G))$
$\mu(G,\mathcal{C})$	$N_+(\text{QSTAB}(G,\mathcal{C}))$
$\nu(G,r)$	$N_+(\text{NOD}(G,r))$, for $r \in \{\Gamma, \theta, \alpha\}$

Table 1: Notation for SDPs optimal values.
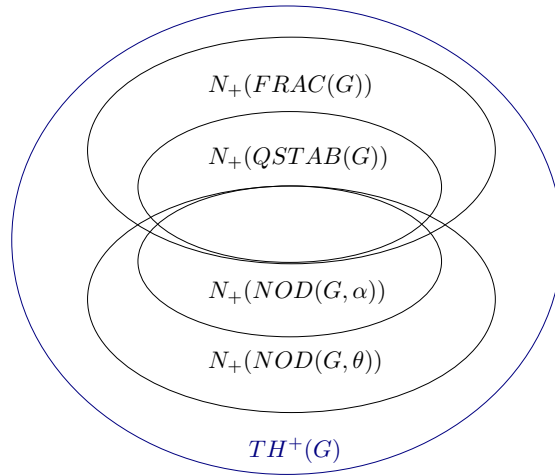
Figure 3: Containment relationships among relaxations.

- `LinearFormulations.py`: this module creates all linear formulations presented in Section 3.1 and exports them in standard `.lp` format. It relies on the max-clique solver `cliquer` [52] and `ADMMsolver.py` (see below) to compute coefficients for nodal inequalities (9).

- `SDPLifting.py`: starting from a linear formulation in standard `.lp` format, this module implements the $M_+(\cdot)$ operator described in Section 2.1, exporting the resulting SDP in the `.mat` MATLAB format.

- `ADMMsolver.py`: this module implements the SDP solver proposed in [4, 5] and is exclusively employed as a subroutine during the LP formulation.

- `kelley_cutting_plane.m`: this module implements the cutting plane method described in Algorithm 2.

The following Python scripts are provided to simplify the use of the software:

- `install.py` downloads and builds all required dependencies (i.e. `SDPNAL+` [61] and `cliquer` [52]).

- `parameters.py` sets up the computational experiments, allowing for (i) selection of the datasets to be used and (ii) tuning of all the parameters of Algorithm 2 and the SDP solvers involved.

- `analyze_results.py` collects the data from the experiments and replicates all the tables presented in this work.

Selection of the SDP solver

Under mild assumptions, semidefinite programs can be solved in polynomial time up to any arbitrary fixed precision [51]. Intuitively, the computational burden mainly depends on the order n of the matrix variable and the number m of linear constraints. However, the density of the latter should also be taken into account as most SDP solvers rely on sparse matrices implementations. *Interior-point methods* (IPMs) [47, 60, 64] provide a good-precision solution in reasonable time for "small" and "medium" SDPs (both in term of n and m). However, they require to store and factorize large Hessian matrices at each iteration, which may become prohibitive for large SDPs. In fact, we preliminarily tested the IPM implemented in the commercial state-of-the-art solver `MOSEK` [47] and observed that problems with $m \approx 10^5$ and $n \approx 200$ are intractable. Although this experiment already provided interesting insights on the strength of the relaxations, testing larger graphs is crucial for a reliable experimental picture. Therefore, we resorted to Alternating Direction Methods of Multipliers (ADMMs), which, as a variant of Augmented Lagrangian methods, is a popular first-order alternative to IPMs and can scale on much larger SDPs at the price of possibly slowing down the convergence to high-precision solutions. For a detailed overview, we refer the reader to [4, 5, 17, 54, 61, 65, 66]. Furthermore, ADMMs allow the efficient handling of bound constraints (e.g., nonnegativity) on the matrix variable via iterative projections on the cone of positive semidefinite matrices. In our case, it enables the computation of $\theta^+(G)$ with no substantial additional effort with respect to $\theta(G)$. We selected `SDPNAL+` proposed in [61], which represents a state-of-the-art ADMM general-purpose SDP solver implemented in MATLAB equipped with efficient subroutines in `C` included via `.mex` files, and has proven to be numerically stable in our experiments.

Model building

The $M_+(\cdot)$ operator has been applied to four different compact LP relaxations of the SSP: QSTAB$(G, \mathcal{C})$ and NOD(G, r) for $r \in \{\Gamma, \theta, \alpha\}$. The construction of these LP models entails different complexity levels. QSTAB$(G, \mathcal{C})$ requires the computation of anedge clique cover $\mathcal{C} \in \mathcal{Q}$. This task has been accomplished by the greedy algorithm presented in Letchford et al. [40] and is reported in Algorithm 1. Starting from the empty collection $\mathcal{C} = \emptyset$, the heuristic iteratively adds a maximal clique of G to $\mathcal{C}$, until every edge is covered. At each iteration, the selected maximal clique C_k covers at least one new edge not yet covered by cliques in $\mathcal{C}$ (line 8). Consequently, even if it can be shown that the resulting edge clique cover $\mathcal{C}$ may not be minimal, its size is at most $|E|$ and, in practice, comprises significantly fewer cliques [40].

Algorithm 1 Greedy edge clique cover

1: **Input**: Graph $G = (V, E)$
2: **Output**: An edge clique cover $\mathcal{C}$ of G
3: Initialize $E' \leftarrow E$, $G' \leftarrow (V, E')$ $\mathcal{C} \leftarrow \emptyset$, $k \leftarrow 0$
4: **while** $E' \neq \emptyset$ **do**
5: $\quad k \leftarrow k + 1$
6: $\quad u \leftarrow \arg\max_{i \in V} \{|\Gamma_{G'}(i)|\}$
7: $\quad v \leftarrow \arg\max_{i \in \Gamma_{G'}(u)} \{|\Gamma_{G'}(i)|\}$ $\qquad\qquad\qquad\qquad \triangleright E' \neq \emptyset \Rightarrow \Gamma_{G'}(u)$ is non-empty
8: $\quad$ Find a maximal clique C_k of G such that $u, v \in C_k$
9: $\quad \mathcal{C} \leftarrow \mathcal{C} \cup \{C_k\}$
10: $\quad E' \leftarrow E' \setminus E(C_k)$
11: **end while**

Building NOD(G, Γ) is straightforward, as $r_i = |\Gamma(i)|$. On the contrary, NOD(G, θ) requires the evaluation of $r_i = \theta(G[\Gamma(i)])$, for all $i \in V$. This corresponds to solve $|V|$ SDPs, namely, those corresponding to (SDP-θ) on the subgraph induced by $\Gamma(i)$. In practice, the SDPs are completely decomposed and can be solved in parallel. Moreover, r_i can be tightened to $\lfloor \theta(G[\Gamma(i)]) \rfloor$ and one can stop the computation as long as the SDP solver returns a valid upper (dual) bound on $\theta(G[\Gamma(i)])$. Several approaches to compute a valid dual bound throughout solver's iterations and a posteriori have been proposed for ADMM algorithms (see, e.g., [17, 36, 66]). Their specific application in SDP relaxations of the SSP have been implemented in [4, 5]. Then, for each iteration of the ADMM implemented in `ADMMsolver.py`, a valid dual bound is computed and this, in turn, allows stopping the computation of θ as soon as a "moderate" precision is achieved. In Table 2, we report the average/max/min CPU times to calculate coefficients $\theta(G[\Gamma(i)])$on a subset of the instances used in our computational experiments, whose detailed introduction is deferred to Section 6.1. Finally, building NOD(G, α) is the most challenging case as it requires calculating $\alpha(G[\Gamma(i)])$, for all $i \in V$. Despite the theoretical hardness of determining $\alpha(G[\Gamma(i)])$, Letchford et al. [40] showed that in practice, the computational burden to build NOD(G, α) is, in fact, accessible for large classes of graphs. We refer the interested reader to their discussion; here we mention that even in this case the evaluation of coefficients $r_i = \alpha(G[\Gamma(i)])$ can be parallelized and, as in the previous case, we report in Table 2 the average/max/min CPU times to evaluate $\alpha(G[\Gamma(i)])$ via the max-clique solver `cliquer` [52].

	Graph	brock800_1	brock800_2	brock800_3	brock800_4	C500-9	p_hat300-1	p_hat300-2	p_hat300-3	p_hat500-1	p_hat500-2	p_hat500-3	p_hat700-1	p_hat700-2	p_hat700-3	sanr400_0.5	sanr400_0.7
θ_i	Min	2.94	2.83	3.07	2.81	0.76	1.96	1.39	0.44	4.63	2.61	0.58	8.89	5.30	0.77	1.27	0.69
	Max	4.98	4.97	4.94	4.91	10.92	7.43	20.35	9.33	10.73	78.22	13.28	25.81	123.65	24.58	2.64	2.04
	Average	3.93	3.90	3.93	3.93	2.40	3.53	6.14	1.86	7.66	15.95	3.84	15.06	31.25	6.98	1.78	1.09
α_i	Min	0.89	1.08	0.68	0.63	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	0.01	< 0.01
	Max	28.87	34.41	26.17	25.91	< 0.01	< 0.01	0.02	0.02	0.02	1.83	2.74	0.07	62.41	108.48	0.04	0.13
	Average	6.00	6.17	5.93	6.04	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	0.10	0.10	0.02	1.64	2.78	0.02	0.03

Table 2: CPU time statistics for computing coefficients θ and α for (9) on DIMACS graphs.

Lifting operation

The function `m_plus_lifting()` in `SDPLifting.py` carries out Steps 1 and 2 of the procedure outlined in Section 2.1. Initially, it parses the polytope P, provided as input through a standard `.lp` format text file. Then, it constructs the symmetric matrices corresponding to constraints (4), (5) and (7). Finally, the model is exported in `.mat` file format, ready to be loaded into MATLAB and solved using `SDPNAL+`.

SDP optimization

In general, we provide as input to `SDPNAL+` SDPs of the form

$$\text{opt(R)} = \max_{Y \in \text{R}} \langle C, Y \rangle,$$

where

$$\text{R} = \left\{ Y \in \mathcal{S}_{n+1}^+ \; : \; Y \geq 0, \langle A_i, Y \rangle \leq b_i, \; i \in [m] \right\},$$

and $m \in \mathbb{N}$, $C, A_1, \ldots, A_m \in \mathcal{S}_{n+1}$, $b \in \mathbb{R}^m$. In a preliminary experiment we observed that `SDPNAL+` converges quickly on (SDP-θ^+) while it requires very large (and practically unacceptable) convergence times on $M_+(\text{QSTAB}(G, \mathcal{C}))$ and $M_+(\text{NOD}(G, r))$ relaxations. This is due to the number of linear constraints in the former, and their density in the latter. However, we proved that $M_+(\text{QSTAB}(G, \mathcal{C}))$ and $M_+(\text{NOD}(G, r))$ are contained in (SDP-θ^+), thus one can resort to a classical cutting plane method to optimize over them. The cutting plane algorithm starts optimizing over (SDP-θ^+). At the k-th iteration, given the optimal solution $Y^k \in \mathcal{S}_{n+1}^+$, one can compute

$$V_i(Y^k) = \langle A_i, Y^k \rangle - b_i, \; i \in [m].$$

If $V_i(Y^k) > \varepsilon$, for a "small" $\varepsilon > 0$, then the i-th constraint is violated by the current solution. Let $c \leq m$ be a positive integer parameter, and let $I_c(Y^k) = \{i_1, i_2, \ldots, i_c\}$ denote the set of indexes corresponding to the c most violated constraints. Then, the constraints $\left\{ Y \in \mathcal{S}_{n+1} \; : \; \langle A_i, Y \rangle \leq b_i, \; i \in I_c(Y^k) \right\}$ are added to the current SDP relaxation and a new optimal solution Y^{k+1} is computed. The algorithm terminates either when no violated constraints are found or when a tailing off condition is met. The detailed algorithm is reported in Algorithm 2.

Algorithm 2 Cutting-plane scheme

1: **Input**: Any SDP R $\subseteq$ (SDP-θ^+), $c \in \mathbb{N}$, $\varepsilon > 0, \delta \in \mathbb{R}_+$
2: Initialize $\Pi^0 \leftarrow \mathcal{S}_{n+1}, \Delta^0 = +\infty, \; k = 0$
3: Compute $p^0 = \text{opt}(\text{SDP-}\theta^+)$ and let Y^0 be its optimal solution
4: **while** Y^k violates constraints in R **or** $\Delta^k > \delta$ **do**
5:      Compute the vector $V(Y^k)$ of violations and $I_c(Y^k)$
6:      $\Pi^{k+1} \leftarrow \Pi^k \cap \left\{ Y \in \mathcal{S}_{n+1} \; : \; \langle A_i, Y \rangle \leq b_i, \; i \in I_c(Y^k) \right\}$
7:      Compute $p^{k+1} = \text{opt}(\text{SDP-}\theta^+ \cap \; \Pi^{k+1})$ and let Y^{k+1} be its optimal solution
8:      $\Delta^{k+1} \leftarrow |p^{k+1} - p^k|$
9:      $k \leftarrow k + 1$
10: **end while**

---

6 Computational Experiments

We now illustrate the results obtained by solving relaxations $M_+(\mathrm{QSTAB}(G,\mathcal{C}))$ and $M_+(\mathrm{NOD}(G,r))$ for $r \in \{\Gamma, \theta, \alpha\}$ with Algorithm 2. Experiments are designed to address three main questions:

(i) Can relaxations based on operator $M_+(\cdot)$ provide upper bounds on $\alpha(G)$ that are significantly stronger than $\theta(G)$?

(ii) What is the additional computational cost for achieving such an improvement?

(iii) How much does the choice of r affect the quality of the bound from $M_+(\mathrm{NOD}(G,r))$?

As a byproduct, we document the quality of the bound provided by $M_+(\mathrm{NOD}(G,\theta))$, that is, the strongest bound which can be computed in polynomial time in our hierarchy. Finally, we investigate the extent to which the bound $\mu(G,\mathcal{C})$ is sensitive to the choice of $\mathcal{C}$ and, in particular, the degree to which it can be improved with a more refined selection of the edge clique cover.

Experiments are run on a computer with Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz, 256GB RAM, under OS Ubuntu 16.04.7 LTS (GNU/Linux 4.15.0-128-generic x86_64). After preliminary tuning experiments, the parameters of Algorithm 2 have been set to $\varepsilon = 10^{-3}, \delta = 10^{-1}$ and $c = 1000$ with 7200 seconds time limit (excluding formulation building time). The precision of the SDP solver `SDPNAL+` is set to 10^{-6}. The precision of `ADMMsolver.py` for evaluating coefficients θ_i is set to 10^{-4}. Denoting by opt(R) the optimal (with the given precision) value of relaxation R computed by Algorithm 2 at termination, tables below report, for each relaxation, four statistics: the percentage gap to $\alpha(G)$, that is, $\frac{\mathrm{opt}(\mathrm{R})-\alpha(G)}{\alpha(G)}\%$; the number of cutting plane iterations and the corresponding CPU time; the number of added cuts. The latter is detailed for each class of inequalities and a class is not reported if none of its members has ever been detected as violated.

6.1 Instances

The numerical experiments are based on the following two collections of graphs, available at `https://github.com/febattista/SDP_lift_and_project` in the standard edge-list format.

Random graphs

A Erdös–Rényi graph $G(n,p)$ has n vertices and each edge appears with probability $p \in [0,1]$. We have considered the collection of graphs from [40], generated by

$$n \in \{200, 250, 300\}, \quad p \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\},$$

where, for each combination of n and p, five instances with different random seeds have been created for a total of 135 graphs.

DIMACS graphs

Graphs from the Second DIMACS Implementation Challenge [37] form the standard benchmark for max-clique algorithms. We complemented the graphs to convert the max-clique instances into SSP instances (`DSJC*` graphs are not complemented, as these belong to the "coloring" benchmark set). We excluded the `c-fat`, `johnson`, and `san` graphs because the integrality gap is completely closed by linear relaxations, so they do not provide information to our analysis. We also dropped `keller5`

and the largest `p-hat` instances (with 1000 and 1500 vertices) as the SDP relaxations turned out to be too large to be handled. The final collection includes 38 graphs. Statistics about the time required to compute lifting coefficients of linear formulations $N_+(\mathrm{NOD}(G,\theta))$ and $N_+(\mathrm{NOD}(G,\alpha))$ are reported in Table 2. Recall that their computation can be carried out in parallel, so the impact on the overall time is often negligible.

6.2 Experiment 1: random graphs

Table 3 collects gap statistics for Erdös–Rényi graphs, where the first column reports the integrality gap of the LP relaxation $\mathrm{NOD}(G,\alpha)$.Tables 4 and 5 report cutting plane iterations and CPU times, and the number of added cuts for each class of inequalities, respectively.

Let us first notice that $\lambda(G) \simeq \theta^+(G)$ in all cases but $n = 200$, $p \in \{0.1, 0.9\}$, evidencing a rather disappointing behavior of $M_+(\mathrm{FRAC}(G))$. On the contrary, replacing edge inequalities with a compact collection of clique inequalities in the initial formulation pays off, as $\mu(G,\mathcal{C})$ is always smaller than $\lambda(G)$. The average portion of additional gap closed by $\mu(G,\mathcal{C})$ w.r.t. $\lambda(G)$ is 6.7%. This is achieved by a few cutting plane iterations, adding about one to three thousand inequalities of class (13b), resulting in an average 80% CPU time increase. Interestingly, no inequalities in the class (13d) have been detected as violated.

This experiment clearly classifies relaxations strength as a function of edge probability p and graph size n. When $p \leq 0.4$, $M_+(\mathrm{QSTAB}(G,\mathcal{C}))$ turns out to be the strongest relaxation and the sparser the graph, the larger the gap closed by $\mu(G,C)$ w.r.t. other relaxations, while formulations based on nodal inequalities fail to improve on $\theta^+(G)$. However, the advantage of $M_+(\mathrm{QSTAB}(G,\mathcal{C}))$ over other relaxations tends to thin out as n increases. The outcome is completely reversed when $p \geq 0.5$, where relaxations based on nodal inequalities are more competitive and, mainly, $M_+(\mathrm{NOD}(G,\alpha))$ is the clear winner. $\nu(G,\alpha)$ closes 45.3% of the gap left by $\lambda(G)$ and 40.0% of the one left by $\mu(G,\mathcal{C})$, on average. To our knowledge, such an improvement on $\theta^+(G)$ has rarely been documented in the literature on SDP approaches, even for smaller graphs. Another nice evidence is that this advantage of $M_+(\mathrm{NOD}(G,\alpha))$ is independent of the graph size. Observe that, also in this case, one class of inequalities is predominant, namely, (15b). Notice also that some violated inequalities in (16b) and (16c) have only been detected for the densest graphs $p = 0.9$. Very good upper bounds are also achieved by $M_+(\mathrm{NOD}(G,\theta))$ when $p \geq 0.7$. In fact, $\nu(G,\theta)$ remarkably improves on $\theta^+(G)$ and, unlike $\nu(G,\alpha)$, can be computed in polynomial time. Also, the computational effort required to achieve strong bounds is reasonable. Differently, the weakest relaxation $M_+(\mathrm{NOD}(G,\Gamma))$ always returns $\nu(G,\Gamma) = \theta^+(G)$, that is, never closes additional gap in our tests. Finally, notice that for highly dense graphs, the gap of the LP relaxation $\mathrm{NOD}(G,\alpha)$ may be smaller than that of the weakest SDP relaxations. However, it is largely improved by the strongest SDP relaxation, which confirms the operator's power and the PSD constraint's key role.

6.3 Experiment 2: DIMACS instances

The DIMACS test set, a standard for max-clique/stable set studies, includes, for the most part, structured graphs from various applications. The breadth of the graph collection supports the relevance of our analysis. To our knowledge, DIMACS graphs with more than 300-400 vertices have never been documented in SDP-based studies [23, 24, 25, 41]. The graphs we have considered are significantly larger than those in case studies applying the $N_+(\cdot)$ operator in [12, 21]. In our experience, experimenting with larger graphs is crucial to draw reliable conclusions about the

Graph					% gap			
n	p	$\mathrm{NOD}(G,\alpha)$	$\theta^+(G)$	$\lambda(G)$	$\mu(G,\mathcal{C})$	$\nu(G,\Gamma)$	$\nu(G,\theta)$	$\nu(G,\alpha)$
	0.1	46.957	19.330	18.723	**16.033**	19.330	19.330	19.330
	0.2	102.539	27.924	27.912	**26.060**	27.924	27.924	27.924
	0.3	93.334	30.774	30.773	**29.393**	30.774	30.774	30.774
	0.4	73.543	30.550	30.550	**29.322**	30.550	30.550	30.537
200	0.5	56.527	30.597	30.597	**29.355**	30.597	30.593	29.540
	0.6	39.889	25.708	25.708	24.318	25.708	25.143	**20.138**
	0.7	30.543	25.579	25.579	23.679	25.579	19.897	**12.494**
	0.8	9.720	13.474	13.474	10.819	13.474	2.871	**1.836**
	0.9	9.600	10.123	10.120	5.346	10.123	**5.057**	**5.057**
	0.1	64.521	27.371	27.112	**24.908**	27.371	27.371	27.371
	0.2	118.348	35.462	35.459	**34.284**	35.462	35.462	35.462
	0.3	103.026	39.136	39.136	**38.370**	39.136	39.136	39.136
	0.4	78.520	38.335	38.335	**37.737**	38.335	38.335	38.312
250	0.5	54.333	33.563	33.563	32.943	33.563	33.563	**31.421**
	0.6	46.178	39.416	39.416	38.669	39.416	39.163	**28.532**
	0.7	39.114	38.531	38.531	37.459	38.531	36.305	**23.679**
	0.8	19.867	20.012	20.012	18.473	20.012	9.419	**4.433**
	0.9	5.950	11.717	11.716	9.524	11.717	**3.219**	**3.219**
	0.1	75.661	31.124	31.047	**29.389**	31.124	31.124	31.124
	0.2	137.919	46.238	46.237	**45.551**	46.238	46.238	46.238
	0.3	113.088	48.426	48.426	**48.033**	48.426	48.426	48.426
	0.4	89.040	50.228	50.228	**49.918**	50.228	50.228	50.166
300	0.5	62.433	46.061	46.061	45.791	46.061	46.061	**41.439**
	0.6	42.224	39.956	39.956	39.595	39.956	39.848	**26.444**
	0.7	23.650	31.619	31.619	31.059	31.619	28.272	**11.347**
	0.8	23.333	29.809	29.809	28.718	29.809	19.807	**11.667**
	0.9	7.090	23.160	23.160	20.318	23.160	5.512	**5.486**

Table 3: $N_+(\cdot)$ bounds on random instances. The % gap of the LP relaxation $\mathrm{NOD}(G,\alpha)$ is reported as a reference. The best gaps are shown in boldface.

Graph		$\theta^+(G)$	$\lambda(G)$		$\mu(G,\mathcal{C})$		$\nu(G,\Gamma)$		$\nu(G,\theta)$		$\nu(G,\alpha)$	
n	p	Time	# iter	Time	# iter	Time	# iter	Time	# iter	Time	# iter	Time
	0.1	7.24	2.0	26.64	3.2	48.24	0.2	8.74	0.2	8.64	0.2	8.76
	0.2	4.29	1.0	9.10	2.0	13.96	0.0	4.34	0.0	4.34	0.0	4.34
	0.3	2.64	0.8	4.43	2.0	8.53	0.0	2.71	0.0	2.71	0.0	2.71
	0.4	1.74	0.2	2.65	2.0	6.51	0.0	1.82	0.2	2.13	1.0	3.77
200	0.5	1.64	0.0	2.46	2.0	6.37	0.0	1.74	1.0	3.60	1.8	12.12
	0.6	1.82	0.0	2.85	2.0	7.00	0.0	1.93	1.0	6.56	2.0	22.02
	0.7	2.18	0.0	3.26	2.0	10.90	0.0	2.30	2.0	29.37	2.4	58.03
	0.8	3.05	0.0	4.42	2.0	14.14	0.0	3.18	1.4	49.62	1.4	40.79
	0.9	5.43	0.6	10.33	2.0	37.73	0.0	5.57	2.6	236.08	2.6	236.57
	0.1	9.62	1.6	28.92	3.0	56.76	0.0	9.68	0.0	9.68	0.0	9.68
	0.2	5.15	1.0	11.53	2.0	18.05	0.0	5.25	0.0	5.25	0.0	5.25
	0.3	4.05	0.4	5.82	2.0	11.34	0.0	4.19	0.0	4.18	0.0	4.18
	0.4	2.40	0.2	4.17	1.0	5.12	0.0	2.57	0.0	2.57	1.0	4.95
250	0.5	2.10	0.0	4.08	1.0	4.78	0.0	2.30	0.6	3.51	2.0	23.23
	0.6	2.05	0.0	4.21	1.0	4.79	0.0	2.26	1.0	6.46	3.0	53.63
	0.7	2.07	0.0	4.66	1.0	5.87	0.0	2.30	2.0	29.51	3.0	65.98
	0.8	2.52	0.0	5.04	1.0	8.39	0.0	2.80	2.8	112.29	2.6	162.95
	0.9	5.77	0.2	9.94	1.2	20.54	0.0	6.05	3.0	430.18	3.0	429.29
	0.1	10.92	1.0	25.16	3.0	68.62	0.0	11.04	0.0	11.03	0.0	11.03
	0.2	8.02	0.6	13.87	2.0	25.53	0.0	8.18	0.0	8.19	0.0	8.18
	0.3	5.26	0.0	6.88	1.0	9.61	0.0	5.47	0.0	5.47	0.0	5.48
	0.4	3.31	0.0	5.72	1.0	6.59	0.0	3.60	0.0	3.58	1.0	7.26
300	0.5	2.67	0.0	5.62	1.0	5.64	0.0	3.01	0.0	3.00	2.0	31.62
	0.6	2.54	0.0	6.24	1.0	5.89	0.0	2.90	1.0	7.51	3.0	66.31
	0.7	2.37	0.0	6.86	1.0	5.80	0.0	2.84	2.0	40.36	4.0	135.51
	0.8	3.78	0.0	7.42	1.0	10.76	0.0	4.31	2.2	97.62	3.6	197.18
	0.9	6.61	0.0	11.98	1.8	30.75	0.0	7.11	4.8	1250.1	5.0	1566.21

Table 4: CPU time and cutting plane's iterations on random graphs.

Graph		$\lambda(G)$	$\mu(G,\mathcal{C})$	$\nu(G,\theta)$				$\nu(G,\alpha)$			
n	p	(11b)	(13b)	(15b)	(15c)	(16b)	(16c)	(15b)	(15c)	(16b)	(16c)
	0.1	1018	3126	0	0	0	0	0	0	0	0
	0.2	30	2000	0	0	0	0	0	0	0	0
	0.3	2	1517	0	0	0	0	0	0	0	0
	0.4	0	1326	0	1	0	0	8	2	0	0
200	0.5	0	1232	7	1	0	0	569	65	0	0
	0.6	0	1213	322	28	0	0	1459	128	0	0
	0.7	0	1247	1508	52	0	0	2137	157	0	0
	0.8	0	1273	1332	27	0	0	1395	5	0	0
	0.9	1	1280	1800	0	567	60	1800	0	567	60
	0.1	615	3000	0	0	0	0	0	0	0	0
	0.2	7	1821	0	0	0	0	0	0	0	0
	0.3	0	1281	0	0	0	0	0	0	0	0
	0.4	0	972	0	0	0	0	21	3	0	0
250	0.5	0	907	1	1	0	0	1095	145	0	0
	0.6	0	862	201	11	0	0	2799	201	0	0
	0.7	0	908	1122	32	0	0	2924	76	0	0
	0.8	0	994	2759	41	0	0	2378	122	0	0
	0.9	0	755	2400	0	232	24	2400	0	232	24
	0.1	256	3000	0	0	0	0	0	0	0	0
	0.2	1	1462	0	0	0	0	0	0	0	0
	0.3	0	946	0	0	0	0	0	0	0	0
	0.4	0	657	0	0	0	0	69	8	0	0
300	0.5	0	534	0	0	0	0	1822	178	0	0
	0.6	0	547	128	4	0	0	2828	172	0	0
	0.7	0	662	1772	60	0	0	3939	61	0	0
	0.8	0	848	2138	62	0	0	3574	26	0	0
	0.9	0	1179	4000	0	460	63	4200	0	568	73

Number of violated inequalities added in Algorithm 2

Table 5: Number of added cutting planes on random instances.

strength of relaxations for the stable set problem, as it typically tends to degrade with graph size. Looking at Table 6, as already noted for random graphs, $\lambda(G)$ does not prove to be better than $\theta^+(G)$: some improvement is observed in two cases (`DSJC125-1` and `MANN_a27`), it is negligible in nine cases and null in the remaining ones. Concerning the relaxations we have proposed, $M_+(\text{NOD}(G, \alpha))$ shows the most promising results. $\nu(G, \alpha)$ is the best bound in 11 cases while $\mu(G, \mathcal{C})$ is the winner in as many as 23 cases. Nevertheless, $\nu(G, \alpha)$ achieves the smallest gap on average, thanks to its remarkable strength on selected instances. According to the evidence gathered from Erdös–Rényi graphs, $M_+(\text{QSTAB}(G, \mathcal{C}))$ is more suited for sparse instances, and $M_+(\text{NOD}(G, \alpha))$ turns out to be particularly tight as the density increases. $\mu(G, \mathcal{C})$ is the best bound for the `C` and `MANN` collections and the sparsest among `p_hat` graphs. On the other hand, $\nu(G, \alpha)$ is very strong for `DSJC500-5`, `p_hat300-1`, `p_hat500-1`, `p_hat700-1`, `sanr400_0.5`, where it closes more than 50% of the gap left by $\theta^+(G)$ and $\lambda(G)$ (about 74% for `p_hat-1` graphs). The computational price for this is reported in Table 7: computing $\mu(G, \alpha)$ is about 27 times slower than computing $\theta^+(G)$ except `p_hat700-1` where it is 185 times slower. But this experiment also shows that the tightness of $M_+(\text{NOD}(G, \alpha))$ is less sensitive to the graph size with respect to the other relaxations. This results in an interesting outcome for the hard `brock800` collection that, with a density of 35%, one expects to be favorable to $M_+(\text{FRAC}(G))$ and $M_+(\text{QSTAB}(G, \mathcal{C}))$. $\nu(G, \alpha)$ is the only bound stronger than $\theta^+(G)$ on these graphs, and the improvement is significant. Table 7 shows that computing $\nu(G, \alpha)$ is about four times slower than computing $\theta^+(G)$ (including the time to build the LP relaxation), an acceptable price for such a valuable result. Table 8 also reveals the key contribution to closing the gap given by inequalities (15b).

Also in this experiment, the polynomial bound $\nu(G, \theta)$ improves on $\theta^+(G)$ in six cases by a meaningful amount (namely, `DSJC125-5`, `MANN_a27`, `p_hat300-1`, `p_hat500-1`, `p_hat700-1`). The comparison between $\nu(G, \theta)$ and $\nu(G, \alpha)$ reveals that strengthening the lifting coefficient pays off, as in eleven cases, $\nu(G, \alpha)$ is significantly smaller than $\nu(G, \theta)$, in other eight cases the improvement is negligible while no improvement is observed in the remaining sixteen cases. On the other hand, $\nu(G, \Gamma)$ has never enhanced $\theta^+(G)$ even if a few violated cuts have occasionally been detected.

Looking at Tables 7 and 8, one can observe that relevant improvements on $\theta^+(G)$ are achieved by adding several inequalities in a few iterations of the cutting plane algorithm. Similarly to what is observed for random graphs, inequalities (13d) are only added in a few special graphs, namely the extremely sparse `MANN` graphs. Differently, classes of inequalities (15b) and (15c) are relevant to describe $M_+(\text{NOD}(G, \theta))$, $M_+(\text{NOD}(G, \alpha))$, while no violated members of classes (16b) and (16c) have ever been found. Interestingly, our numerical results indicate that inequalities generated during the lifting step by multiplying by $x_i \geq 0$ are more frequently violated than those obtained by multiplying by $1 - x_i \geq 0$. This trend is consistent across all reported relaxations, with only a few exceptions.

Algorithm 2 has shown a nice numerical behavior even for the largest graphs in the DIMACS collection, and re-optimization is cost-effective (Table 7): its extra time is only paid when the cuts yield a significant improvement of the upper bound. In contrast, in the other cases, CPU times do not deviate significantly from those of $\theta^+(G)$.

Overall, the computational analysis shows that the new SDP relaxations $M_+(\text{QSTAB}(G, \mathcal{C}))$ and, mainly, $M_+(\text{NOD}(G, \alpha))$, may be a viable option to compute upper boundson $\alpha(G)$ which are remarkably stronger than $\theta(G)$ at a reasonable computational price. Interestingly, some experimental evidence complements the theoretical insights illustrated in Section 4.3, where $M_+(\text{QSTAB}(G, \mathcal{C}))$ looks to be likely to achievetighter bounds than $M_+(\text{NOD}(G, \alpha))$.

Graph		θ⁺(G)		λ(G)		μ(G,C)		ν(G,Γ)		ν(G,θ)		ν(G,α)	
Name	d(%)	UB	% gap	UB	% gap	UB	% gap	UB	% gap	UB	% gap	UB	% gap
brock200_1	25	27.20	29.508	27.20	29.505	27.12	29.119	27.20	29.508	27.20	29.508	27.20	29.508
brock200_2	50	14.13	17.758	14.13	17.758	14.12	17.674	14.13	17.758	14.13	17.757	14.02	16.795
brock200_3	39	18.67	24.479	18.67	24.479	18.65	24.356	18.67	24.479	18.67	24.478	18.67	24.467
brock200_4	34	21.12	24.242	21.12	24.242	21.10	24.101	21.12	24.242	21.12	24.242	21.12	24.242
brock400_1	25	39.33	45.670	39.33	45.670	39.33	45.662	39.33	45.670	39.33	45.670	39.33	45.670
brock400_2	25	39.20	35.160	39.20	35.160	39.20	35.156	39.20	35.160	39.20	35.160	39.20	35.160
brock400_3	25	39.16	26.324	39.16	26.324	39.16	26.310	39.16	26.324	39.16	26.324	39.16	26.324
brock400_4	25	39.23	18.883	39.23	18.883	39.23	18.877	39.23	18.883	39.23	18.883	39.23	18.883
brock800_1	35	41.87	82.032	41.87	82.032	41.87	82.032	41.87	82.032	41.87	82.032	41.55	80.646
brock800_2	35	42.10	75.435	42.10	75.435	42.10	75.435	42.10	75.435	42.10	75.435	41.73	73.896
brock800_3	35	41.88	67.530	41.88	67.530	41.88	67.530	41.88	67.530	41.88	67.530	41.51	66.043
brock800_4	35	42.00	61.541	42.00	61.541	42.00	61.541	42.00	61.541	42.00	61.541	41.61	60.039
C125-9	10	37.55	10.431	36.75	8.074	36.23	6.566	37.55	10.431	37.54	10.424	37.55	10.429
C250-9	10	55.82	26.856	55.71	26.618	55.20	25.446	55.82	26.856	55.82	26.856	55.82	26.856
C500-9	10	83.58	46.630	83.58	46.630	83.51	46.515	83.58	46.630	83.58	46.630	83.58	46.630
DSJC125.1	9	38.04	11.896	37.28	9.650	36.79	8.207	38.04	11.895	38.04	11.881	38.04	11.881
DSJC125.5	50	11.40	14.021	11.40	14.017	11.36	13.598	11.40	14.021	11.38	13.810	11.35	13.531
DSJC125.9	90	4.00	0.000	4.00	0.000	4.00	0.000	4.00	0.000	4.00	0.000	4.00	0.000
DSJC500-5	50	22.57	73.621	22.57	73.621	22.57	73.621	22.57	73.621	22.57	73.621	20.54	58.014
MANN_a9	7	17.48	9.219	17.09	6.811	17.00	6.250	17.48	9.219	17.47	9.201	17.47	9.201
MANN_a27	1	132.76	5.367	131.11	4.057	131.01	3.980	132.76	5.367	131.99	4.751	131.99	4.751
johnson32-2-4	12	16.00	0.000	16.00	0.000	16.00	0.000	16.00	0.000	16.00	0.000	16.00	0.000
keller4	35	13.47	22.417	13.46	22.388	13.39	21.709	13.47	22.417	13.45	22.236	13.45	22.236
p_hat300-1	76	10.02	25.253	10.02	25.253	10.02	25.194	10.02	25.253	9.58	19.691	8.58	7.288
p_hat300-2	51	26.71	6.855	26.58	6.317	26.49	5.950	26.71	6.855	26.70	6.813	26.69	6.768
p_hat300-3	26	40.70	13.057	40.62	12.829	40.44	12.320	40.70	13.057	40.70	13.054	40.70	13.053
p_hat500-1	75	13.01	44.533	13.01	44.533	13.01	44.526	13.01	44.533	12.65	40.583	10.68	18.721
p_hat500-2	50	38.56	48.306	38.44	47.863	38.38	47.598	38.56	48.306	38.56	48.293	38.54	48.225
p_hat500-3	25	57.81	15.622	57.76	15.524	57.67	15.342	57.81	15.622	57.81	15.622	57.81	15.621
p_hat700-1	75	15.05	36.774	15.05	36.774	15.05	36.773	15.05	36.774	14.86	35.047	11.30	2.709
p_hat700-2	50	48.44	10.091	48.30	9.776	48.24	9.646	48.44	10.091	48.44	10.087	48.41	10.023
p_hat700-3	25	71.76	15.734	71.71	15.660	71.64	15.545	71.76	15.734	71.75	15.734	71.75	15.733
sanr200_0.7	30	23.63	31.296	23.63	31.291	23.59	31.048	23.63	31.296	23.63	31.296	23.63	31.296
sanr200_0.9	10	48.90	16.440	48.63	15.786	48.03	14.356	48.90	16.440	48.90	16.440	48.90	16.440
sanr400_0.5	50	20.18	55.217	20.18	55.217	20.18	55.216	20.18	55.217	20.18	55.217	19.10	46.951
sanr400_0.7	30	33.97	61.746	33.97	61.746	33.97	61.741	33.97	61.746	33.97	61.746	33.97	61.746
Means		-	30.832	-	30.528	-	30.248	-	30.832	-	30.489	-	27.772

Table 6: Lift-and-project bounds on DIMACS instances. The column d(%) reports the density of each graph.

Graph Name	$d(\%)$	$\theta^+(G)$ Time	$\lambda(G)$ # iter	Time	$\mu(G,\mathcal{C})$ # iter	Time	$\nu(G,\Gamma)$ # iter	Time	$\nu(G,\theta)$ # iter	Time	$\nu(G,\alpha)$ # iter	Time
brock200_1	25	3.72	1	7.22	1	6.86	0	3.79	0	3.78	0	3.79
brock200_2	50	1.63	0	2.51	1	4.02	0	1.73	1	3.79	2	14.22
brock200_3	39	2.07	1	4.27	1	4.44	0	2.15	1	3.92	1	4.47
brock200_4	34	2.66	0	3.12	1	4.79	0	2.75	0	2.75	1	4.35
brock400_1	25	11.59	0	15.10	1	20.59	0	12.00	0	12.07	0	11.99
brock400_2	25	12.35	0	15.47	1	21.18	0	12.75	0	12.77	0	12.76
brock400_3	25	12.17	0	14.87	1	21.40	0	12.59	0	12.58	0	12.61
brock400_4	25	12.05	0	14.84	1	20.76	0	12.45	0	12.44	0	12.44
brock800_1	35	20.60	0	51.54	0	27.57	0	24.54	0	24.60	2	77.95
brock800_2	35	20.26	0	53.58	0	27.05	0	24.25	0	24.22	2	86.15
brock800_3	35	20.55	0	52.85	0	28.19	0	25.02	0	24.63	2	79.55
brock800_4	35	19.86	0	50.82	0	27.46	0	23.98	0	24.00	2	76.29
C125-9	10	3.46	2	15.68	4	32.44	1	7.10	1	7.44	1	7.55
C250-9	10	10.45	2	32.52	2	42.68	0	10.52	0	10.51	0	10.52
C500-9	10	25.12	1	57.41	1	53.55	0	25.52	0	25.50	0	25.50
DSJC125.1	9	4.38	2	19.28	4	34.02	1	8.38	1	10.27	1	9.02
DSJC125.5	50	1.58	1	3.17	1	3.72	0	1.61	1	4.35	1	4.64
DSJC125.9	90	1.65	0	1.98	0	1.72	0	1.70	0	1.70	0	1.70
DSJC500-5	50	6.35	0	20.49	0	8.85	0	7.71	0	7.83	4	126.12
MANN_a9	7	0.79	1	2.15	2	4.38	0	0.80	1	2.20	1	1.82
MANN_a27	1	9.93	1	73.41	3	1992.72	0	10.00	1	57.22	1	60.32
johnson32-2-4	12	4.16	0	6.92	0	4.28	0	4.58	0	4.58	0	4.60
keller4	35	3.97	1	7.70	1	13.20	0	4.03	1	15.51	1	15.12
p_hat300-1	76	9.07	0	12.62	1	22.93	0	9.49	2	134.81	3	261.52
p_hat300-2	51	80.54	2	220.85	2	221.98	0	80.85	1	300.55	1	279.48
p_hat300-3	26	17.66	1	32.39	2	44.90	0	17.84	1	39.16	1	39.27
p_hat500-1	75	19.08	0	40.72	1	42.03	0	21.13	2	164.90	5	858.88
p_hat500-2	50	201.27	2	661.29	2	652.87	0	202.66	1	580.22	1	893.64
p_hat500-3	25	35.75	1	88.95	2	116.21	0	36.50	1	79.32	1	90.28
p_hat700-1	75	35.53	0	82.10	1	79.12	0	41.34	2	260.41	7	6580.82
p_hat700-2	50	425.90	2	1270.35	2	1409.98	0	429.64	1	920.86	1	1251.71
p_hat700-3	25	81.42	1	204.92	2	278.70	0	83.93	1	189.04	1	204.45
sanr200_0.7	30	3.05	1	5.39	1	5.54	0	3.11	0	3.11	0	3.11
sanr200_0.9	10	7.77	2	29.39	3	43.35	0	7.81	1	15.36	1	14.86
sanr400_0.5	50	4.40	0	11.74	1	8.96	0	5.09	0	5.11	3	77.60
sanr400_0.7	30	7.69	0	11.58	1	12.80	0	8.16	0	8.17	0	8.26

Table 7: CPU time and cutting plane's iterations on DIMACS graphs.

	Number of violated inequalities added in Algorithm 2							
Graph Name	$\lambda(G)$ (11b)	$\mu(G,\mathcal{C})$ (13b)	(13d)	$\nu(G,\Gamma)$ (15c)	$\nu(G,\theta)$ (15b)	(15c)	$\nu(G,\alpha)$ (15b)	(15c)
brock200_1	10	471	0	0	0	0	0	0
brock200_2	0	81	0	0	5	4	583	76
brock200_3	1	152	0	0	1	1	2	5
brock200_4	0	139	0	0	0	0	1	0
brock400_1	0	34	0	0	0	0	0	0
brock400_2	0	38	0	0	0	0	0	0
brock400_3	0	49	0	0	0	0	0	0
brock400_4	0	33	0	0	0	0	0	0
brock800_1	0	0	0	0	0	0	1845	155
brock800_2	0	0	0	0	0	0	1780	220
brock800_3	0	0	0	0	0	0	1825	175
brock800_4	0	0	0	0	0	0	1840	160
C125-9	1413	2248	3	1	0	35	0	14
C250-9	565	1988	0	0	0	0	0	0
C500-9	1	633	0	0	0	0	0	0
DSJC125.1	1435	2249	11	8	0	36	0	36
DSJC125.5	8	210	0	0	62	17	117	24
DSJC125.9	0	0	0	0	0	0	0	0
DSJC500-5	0	0	0	0	0	0	3772	228
MANN_a9	720	900	36	0	0	504	0	504
MANN_a27	1000	2000	1000	0	0	1000	0	1000
johnson32-2-4	0	0	0	0	0	0	0	0
keller4	48	296	0	0	96	48	96	48
p_hat300-1	0	48	0	0	1911	89	2923	77
p_hat300-2	988	1379	0	0	121	346	329	398
p_hat300-3	761	1878	0	0	0	60	0	75
p_hat500-1	0	13	0	0	1982	18	4959	41
p_hat500-2	1042	1363	0	0	30	288	576	424
p_hat500-3	548	1625	0	0	0	28	0	72
p_hat700-1	0	2	0	0	1960	40	6962	38
p_hat700-2	1107	1549	0	0	16	286	718	282
p_hat700-3	520	1414	0	0	0	47	1	187
sanr200_0.7	3	286	0	0	0	0	0	0
sanr200_0.9	1060	2216	0	0	0	1	0	1
sanr400_0.5	0	3	0	0	0	0	2806	194
sanr400_0.7	0	29	0	0	0	0	0	0

Table 8: Number of added cutting planes on DIMACS instances.

6.4 Sensitivity to the choice of $\mathcal{C}$

To enhance the results of our computational analysis, we investigate whether refining the edge clique cover $\mathcal{C}$ obtained from Algorithm 1 can lead to a significant improvement in the bound $\mu(G, \mathcal{C})$, while also considering the associated computational overhead.

To this end, we consider the edge clique cover $\mathcal{C}_{\text{cut}} \in \mathcal{Q}$ proposed by Letchford et al. [40]. The collection $\mathcal{C}_{\text{cut}}$ is obtained by adding to $\mathcal{C}$ all cliques identified through a cutting plane algorithm that employs the separation heuristic of Marzi et al. [45]. Since $\mathcal{C} \subseteq \mathcal{C}_{\text{cut}}$, QSTAB$(G, \mathcal{C}_{\text{cut}})$ provides a tighter upper bound on $\alpha(G)$ than that of QSTAB$(G, \mathcal{C})$. Marzi et al. [45] and Letchford et al. [40] also show that this upper bound is very close to the bound provided by QSTAB(G). However, this improvement may require a large number of clique inequalities.

Tables 9 and 10 report the results of this sensitivity analysis on a subset of the previously tested DIMACS instances. We selected graphs of varying densities for which we expect that $\mathcal{C}_{\text{cut}}$ has the potential to overturn the conclusion drawn from the previous experiment. As anticipated, the percentage gaps in Table 9 show an improvement of $\mu(G, \mathcal{C}_{\text{cut}})$ over $\mu(G, \mathcal{C})$. This gain, however, generally comes at a substantial additional cost in terms of cutting-plane iterations and CPU time (up to a factor of 2), as reported in Table 10. Interestingly, $\mathcal{C}_{\text{cut}}$ has a negligible effect on the bounds for dense graphs, where $\nu(G, \alpha)$ remains superior. Overall, this experiment suggests that the greedy edge clique cover provides a good trade-off between bound quality and computational time.

A notable exception is `keller4`, where $\mu(G, \mathcal{C}_{\text{cut}})$ attains a tight upper bound of 12.64 on $\alpha(G) = 11$, computed in a reasonable time. To the best of the authors' knowledge, this represents the strongest bound reported in the literature; see, e.g., Locatelli [41], Giandomenico et al. [30] and references therein.

7 Conclusions

We have presented a way to improve the classical Lovász θ bound for the stability number of a graph by applying the Lovász-Schrijver $N_+(\cdot)$ lift-and-project operator to tailored LP relaxations. Unlike previous approaches, some of the latter may not be constructed in polynomial time. In the case of clique inequalities, we have to handle exponentially many inequalities (with an associated NP-hard separation problem), while building strong nodal inequalities requires solving the SSP on specific subgraphs. We show that these expedients, if properly handled, are indeed helpful in letting the resulting SDP relaxations improve the θ bound without excessive extra effort. This study reveals that the Lovász and Schrijver $N_+(\cdot)$ operator can be of practical interest besides being a powerful theoretical tools. Therefore, its application to other combinatorial optimization problems that admit strong SDP relaxations, such as the graph coloring problem, is a natural research line. From a theoretical standpoint, a few questions remain open. First, from Theorem 3 we have $N_+(\text{NOD}(G, \Gamma)) \subseteq \text{TH}^+(G)$, whereas the numerical experiments suggest $N_+(\text{NOD}(G, \Gamma)) = \text{TH}^+(G)$. However, it still remains an open question. Secondly, the experiments showed a strong predominance of violated constraints lifted by multiplying by $x_i \geq 0$ on those multiplied by $1 - x_i \geq 0$. This deserves a further theoretical investigation. Finally, an interesting research direction consists of finding classes of graphs where $N_+(\text{NOD}(G, \theta)) = \text{STAB}(G)$, as for them the SSP would be polynomially solvable.

Graph	$\theta^+(G)$		$\mu(G, \mathcal{C}_{\text{cut}})$		$\mu(G, \mathcal{C})$		$\nu(G, \alpha)$	
Name	UB	% gap	UB	% gap	UB	% gap	UB	% gap
brock200_1	27.20	29.508	**26.71**	**27.183**	27.12	29.119	27.20	29.508
brock200_2	14.13	17.758	**13.98**	**16.510**	14.12	17.674	14.02	16.795
brock200_3	18.67	24.479	**18.30**	**22.013**	18.65	24.356	18.67	24.467
brock200_4	21.12	24.242	**20.78**	**22.246**	21.10	24.101	21.12	24.242
C125-9	37.55	10.431	**35.87**	**5.487**	36.23	6.566	37.55	10.430
DSJC125.1	38.04	11.896	**36.42**	**7.125**	36.79	8.207	38.04	11.881
DSJC125.5	11.40	14.021	**11.04**	**10.359**	11.36	13.598	11.35	13.531
keller4	13.47	22.417	**12.64**	**14.873**	13.39	21.709	13.45	22.236
p_hat300-1	10.02	25.253	9.81	22.622	10.02	25.194	**8.58**	**7.288**
p_hat300-2	26.71	6.855	**26.00**	**4.015**	26.49	5.950	26.69	6.768
p_hat300-3	40.70	13.056	**39.74**	**10.384**	40.44	12.321	40.70	13.053
p_hat500-1	13.01	44.533	12.95	43.847	13.01	44.526	**10.68**	**18.721**
p_hat500-2	38.56	48.306	**37.94**	**45.934**	38.38	47.597	38.54	48.225
p_hat700-1	15.05	36.774	15.02	36.589	15.05	36.773	**11.30**	**2.709**
p_hat700-2	48.44	10.091	**47.89**	**8.831**	48.24	9.647	48.41	10.023
sanr200_0.7	23.63	31.296	**23.22**	**28.976**	23.59	31.048	23.63	31.296
sanr200_0.9	48.90	16.439	**47.49**	**13.061**	48.03	14.356	48.90	16.440
sanr400_0.5	20.18	55.217	20.17	55.142	20.18	55.216	**19.10**	**46.951**

Table 9: Semidefinite bounds with improved edge clique covers.

Conflict of interest

The authors declare that they have no conflict of interest.

Graph	$\mu(G, \mathcal{C}_{\text{cut}})$		$\mu(G, \mathcal{C})$		$\nu(G, \alpha)$	
Name	# iter	Time	# iter	Time	# iter	Time
brock200_1	2.0	10.60	1.0	6.31	0.0	3.34
brock200_2	2.0	7.04	1.0	3.87	2.0	13.88
brock200_3	2.0	7.03	1.0	4.18	1.0	4.26
brock200_4	2.0	7.29	1.0	4.56	1.0	4.54
C125-9	4.0	32.91	4.0	33.90	1.0	9.95
DSJC125.1	4.0	32.81	4.0	33.64	1.0	9.22
DSJC125.5	2.0	5.53	1.0	2.97	1.0	3.64
keller4	4.0	29.15	1.0	11.57	1.0	15.01
p_hat300-1	2.0	36.83	1.0	21.78	3.0	262.60
p_hat300-2	3.0	311.08	2.0	168.35	1.0	232.25
p_hat300-3	4.0	75.37	2.0	43.34	1.0	36.93
p_hat500-1	1.0	37.54	1.0	40.53	5.0	854.09
p_hat500-2	3.0	841.79	2.0	588.09	1.0	927.59
p_hat700-1	1.0	76.46	1.0	78.54	7.0	6307.61
p_hat700-2	3.0	1405.20	2.0	1332.51	1.0	1208.12
sanr200_0.7	2.0	9.51	1.0	5.50	0.0	3.19
sanr200_0.9	3.0	43.39	3.0	43.01	1.0	20.16
sanr400_0.5	1.0	8.78	1.0	8.70	3.0	76.85

Table 10: CPU time and cutting-plane iterations with improved edge clique covers.

References

[1] Y. H. Au and L. Tunçel. Stable set polytopes with high lift-and-project ranks for the Lovász–Schrijver SDP operator. *Mathematical Programming ser. A*, 2024. URL `https://doi.org/10.1007/s10107-024-02093-0`.

[2] E. Balas and M. W. Padberg. Set partitioning: A survey. *SIAM review*, 18(4):710–760, 1976.

[3] E. Balas, S. Ceria, G. Cornuejols, and G. Pataki. Polyhedral methods for the maximum clique problem. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 1994.

[4] F. Battista. *On semidefinite lift-and-project of combinatorial optimization problems*. PhD thesis, Università di Roma Sapienza, 2023.

[5] F. Battista and M. De Santis. Dealing with inequality constraints in large-scale semidefinite relaxations for graph coloring and maximum clique problems. *4OR. A Quarterly Journal of Operations Research*, pages 1–31, 2024. doi: 10.1007/s10288-024-00569-5.

[6] S. M. Bianchi, M. Escalante, G. L. Nasini, and L. Tunçel. Some advances on Lovász–Schrijver semidefinite programming relaxations of the fractional stable set polytope. *Discrete Applied Mathematics*, 164:460–469, Feb. 2014. ISSN 0166-218X. doi: 10.1016/j.dam.2013.03.028. URL `https://www.sciencedirect.com/science/article/pii/S0166218X13001765`.

[7] S. M. Bianchi, M. Escalante, G. L. Nasini, and L. Tunçel. Lovász–Schrijver SDP-operator, near-perfect graphs and near-bipartite graphs. *Mathematical Programming*, 162:201–223, 2017.

[8] S. M. Bianchi, M. S. Escalante, G. L. Nasini, and A. K. Wagler. Lovász–Schrijver PSD-operator and the stable set polytope of claw-free graphs. *Discrete Applied Mathematics*, 332:70–86, 2023.

[9] M. Bodur, S. Dash, and O. Günlük. A new lift-and-project operator. *European Journal of Operational Research*, 257(2):420–428, 2017.

[10] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo. The maximum clique problem. *Handbook of Combinatorial Optimization: Supplement Volume A*, pages 1–74, 1999.

[11] R. Borndörfer. *Aspects of set packing, partitioning, and covering*. PhD thesis, Technischen Universität Berlin, 1998.

[12] S. Burer and D. Vandenbussche. Solving lift-and-project relaxations of binary integer programs. *SIAM Journal on Optimization*, 16(3):726–750, 2006.

[13] S. Busygin and D. V. Pasechnik. On NP-hardness of the clique partition–independence number gap recognition and related problems. *Discrete Mathematics*, 306(4):460–463, 2006.

[14] A. Cabello, S. Severini, and A. Winter. Graph-theoretic approach to quantum correlations. *Physical Review Letters*, 112(4):040401, 2014.

[15] L. Cánovas, M. Landete, and A. Marın. New facets for the set packing polytope. *Operations Research Letters*, 27(4):153–161, 2000.

[16] A. Caprara and J. J. S. González. Separating lifted odd-hole inequalities to solve the index selection problem. *Discrete Applied Mathematics*, 92(2-3):111–134, 1999.

[17] M. Cerulli, M. De Santis, E. Gaar, and A. Wiegele. Improving ADMMs for solving doubly nonnegative programs through dual factorization. *4OR. A Quarterly Journal of Operations Research*, 19:415–448, 2021.

[18] M. Conforti, G. Cornuéjols, G. Zambelli, M. Conforti, G. Cornuéjols, and G. Zambelli. *Integer programming models.* Springer, Heidelberg, 2014.

[19] S. Coniglio and S. Gualandi. Optimizing over the closure of rank inequalities with a small right-hand side for the maximum stable set problem via bilevel programming. *INFORMS Journal on Computing*, 34(2):1006–1023, 2022.

[20] R. C. Corrêa, D. Delle Donne, I. Koch, and J. Marenco. General cut-generating procedures for the stable set polytope. *Discrete Applied Mathematics*, 245:28–41, 2018.

[21] S. Dash. *On the matrix cuts of Lovász and Schrijver and their use in Integer Programming.* Rice University, Houston, TX, 2001.

[22] F. Della Croce and R. Tadei. A multi-KP modeling for the maximum-clique problem. *European Journal of Operational Research*, 73(3):555–561, 1994.

[23] I. Dukanovic and F. Rendl. Semidefinite programming relaxations for graph coloring and maximal clique problems. *Mathematical Programming*, 109(2-3):345–365, 2007.

[24] E. Gaar and F. Rendl. A computational study of exact subgraph based SDP bounds for Max-Cut, stable set and coloring. *Mathematical Programming*, 183(1-2):283–308, 2020.

[25] E. Gaar, M. Siebenhofer, and A. Wiegele. An SDP-based approach for computing the stability number of a graph. *Mathematical Methods of Operations Research*, 95(1):141–161, 2022.

[26] L. Galli and A. N. Letchford. On the lovász theta function and some variants. *Discrete Optimization*, 25:159–174, 2017.

[27] M. Giandomenico and A. N. Letchford. Exploring the relationship between max-cut and stable set relaxations. *Mathematical Programming*, 106:159–175, 2006.

[28] M. Giandomenico, A. N. Letchford, F. Rossi, and S. Smriglio. An application of the Lovász–Schrijver $M(K,K)$ operator to the stable set problem. *Mathematical Programming*, 120:381–401, 2009.

[29] M. Giandomenico, A. N. Letchford, F. Rossi, and S. Smriglio. Approximating the Lovász θ function with the subgradient method. *Electronic Notes in Discrete Mathematics*, 41:157–164, 2013.

[30] M. Giandomenico, F. Rossi, and S. Smriglio. Strong lift-and-project cutting planes for the stable set problem. *Mathematical Programming*, 141:165–192, 2013.

[31] M. Giandomenico, A. N. Letchford, F. Rossi, and S. Smriglio. Ellipsoidal relaxations of the stable set problem: theory and algorithms. *SIAM Journal on Optimization*, 25(3):1944–1963, 2015.

[32] M. X. Goemans and L. Tuncel. When does the positive semidefiniteness constraint help in lifting procedures? *Mathematics of Operations Research*, 4(26):796–815, 2001.

[33] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, Heidelberg, 2012.

[34] G. Gruber and F. Rendl. Computational experience with stable set relaxations. *SIAM Journal on Optimization*, 13(4):1014–1028, 2003.

[35] J. Hastad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 627–636. IEEE, 1996.

[36] C. Jansson, D. Chaykin, and C. Keil. Rigorous error bounds for the optimal value in semidefinite programming. *SIAM Journal on Numerical Analysis*, 46(1):180–200, 2008.

[37] D. S. Johnson and M. A. Trick. *Cliques, coloring, and satisfiability: second DIMACS implementation challenge, October 11-13, 1993*, volume 26. American Mathematical Soc., Providence, RI, 1996.

[38] F. Juhász. The asymptotic behaviour of Lovász' θ function for random graphs. *Combinatorica*, 2(2):153–155, 1982.

[39] A. N. Letchford, F. Marzi, F. Rossi, and S. Smriglio. Strengthening Chvátal-Gomory cuts for the stable set problem. In *International Symposium on Combinatorial Optimization*, pages 201–212. Springer, 2016.

[40] A. N. Letchford, F. Rossi, and S. Smriglio. The stable set problem: Clique and nodal inequalities revisited. *Computers & Operations Research*, 123:105024, 2020.

[41] M. Locatelli. Improving upper bounds for the clique number by non-valid inequalities. *Mathematical Programming*, 150:511–525, 2015.

[42] L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, 25(1):1–7, 1979.

[43] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0–1 optimization. *SIAM Journal on Optimization*, 1(2):166–190, 1991.

[44] J. Malick, J. Povh, F. Rendl, and A. Wiegele. Regularization methods for semidefinite programming. *SIAM Journal on Optimization*, 20(1):336–356, 2009.

[45] F. Marzi, F. Rossi, and S. Smriglio. Computational study of separation algorithms for clique inequalities. *Soft Computing*, 23(9):3013–3027, 2019.

[46] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Mathematical Programming*, 10(1):147–175, 1976.

[47] MOSEK. The mosek optimization toolbox for matlab manual. version 9.0, 2019. URL http://docs.mosek.com/9.0/toolbox/index.html.

[48] A. T. Murray and R. L. Church. Facets for node packing. *European Journal of Operational Research*, 101(3):598–608, 1997.

[49] G. L. Nemhauser and G. Sigismondi. A strong cutting plane/branch-and-bound algorithm for node packing. *Journal of the Operational Research Society*, 43(5):443–457, 1992.

[50] G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, Hoboken, NJ, 1999.

[51] Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming.* SIAM, Philadelphia, PA, 1994.

[52] P. R. Östergård. A fast algorithm for the maximum clique problem. *Discrete Applied Mathematics*, 120(1-3):197–207, 2002.

[53] M. W. Padberg. On the facial structure of set packing polyhedra. *Mathematical programming*, 5(1):199–215, 1973.

[54] J. Povh, F. Rendl, and A. Wiegele. A boundary point method to solve semidefinite programs. *Computing*, 78:277–286, 2006.

[55] S. Rebennack, M. Oswald, D. O. Theis, H. Seitz, G. Reinelt, and P. M. Pardalos. A branch and cut solver for the maximum stable set problem. *Journal of Combinatorial Optimization*, 21:434–457, 2011.

[56] F. Rossi and S. Smriglio. A branch-and-cut algorithm for the maximum cardinality stable set problem. *Operations Research Letters*, 28(2):63–74, 2001.

[57] F. Rossi and S. Smriglio. A set packing model for the ground holding problem in congested networks. *European Journal of Operational Research*, 131(2):400–416, 2001.

[58] P. San Segundo, D. Rodríguez-Losada, and A. Jiménez. An exact bit-parallel algorithm for the maximum clique problem. *Computers & Operations Research*, 38(2):571–581, 2011.

[59] A. Schrijver. A comparison of the Delsarte and Lovász bounds. *IEEE Transactions on Information Theory*, 25(4):425–429, 1979.

[60] J. F. Sturm. Using SeDuMi 1.02, A MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653, 1999. doi: 10.1080/10556789908805766. URL https://doi.org/10.1080/10556789908805766.

[61] D. Sun, K.-C. Toh, Y. Yuan, and X.-Y. Zhao. SDPNAL+: A Matlab software for semidefinite programming with bound constraints. *Optimization Methods and Software*, 35(1):87–115, 2020.

[62] E. Tomita and T. Kameda. An efficient branch-and-bond algorithm for finding a maximum clique with computational experiments. *Journal of Global Optimization*, 37:95–111, 2007.

[63] L. E. Trotter Jr. A class of facet producing graphs for vertex packing polyhedra. *Discrete Mathematics*, 12(4):373–388, 1975.

[64] R. H. Tütüncü, K.-C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical Programming*, 95:189–217, 2003.

[65] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented Lagrangian methods for semidefinite programming. *Mathematical Programming Computation*, 2(3):203–230, 2010.

[66] A. Wiegele and S. Zhao. Sdp-based bounds for graph partition via extended admm. *Computational Optimization and Applications*, 82(1):251–291, 2022.

[67] A. T. Wilson. *Applying the boundary point method to an SDP relaxation of the maximum independent set problem for a branch and bound algorithm.* PhD thesis, New Mexico Institute of Mining and Technology, 2009.

[68] Q. Wu and J.-K. Hao. A review on algorithms for maximum clique problems. *European Journal of Operational Research*, 242(3):693–709, 2015.

[69] E. A. Yildirim and X. Fan-Orzechowski. On extracting maximum stable sets in perfect graphs using Lovász's theta function. *Computational Optimization and Applications*, 33(2-3):229–247, 2006.