Les Cahiers du GERAD ISSN: 0711–2440

An Interior-Point Algorithm with Selective Addition of Inequalities for Solving Doubly Non-Negative Relaxations of Maximum-Stable-Set and Maximum-Clique Problems

A. Engau, M.F. Anjos,
I. Bomze

G–2011–08

February 2011
Revised: November 2011

An Interior-Point Algorithm with Selective Addition of Inequalities for Solving Doubly Non-Negative Relaxations of Maximum-Stable-Set and Maximum-Clique Problems

Alexander Engau

Department of Mathematical & Statistical Sciences
University of Colorado Denver
Denver, Colorado, U.S.A.
aengau@alumni.clemson.edu

Miguel F. Anjos

GERAD & Département de mathématiques et génie industriel
École Polytechnique de Montréal
Montréal (Québec) Canada, H3C 3A7
anjos@stanfordalumni.org

Immanuel Bomze

Applied Mathematics and Statistics
University of Vienna
1010 Vienna, Austria
immanuel.bomze@univie.ac.at

February 2011
Revised: November 2011

Les Cahiers du GERAD
G–2011–08

Abstract

The maximum-stable-set and maximum-clique problems are operations research problems that arise in numerous areas such as social networking, electrical engineering, environmental forest planning, bioinformatics clustering and prediction, and computational chemistry. The doubly nonnegative relaxation is known to provide high-quality bounds on the size of the global optimal solution of both problems. However, it is an expensive conic optimization problem to solve in general. We propose an integrated interior-point cutting-plane method to efficiently handle the large number of nonnegativity constraints in the relaxation. The approach is based on a recent interior-point algorithm that selectively adds inequalities in a dynamic fashion. We present computational results showing the significant benefits of the integrated algorithm in comparison to a standard interior-point cutting-plane method.

Key Words: stable set, maximum clique, theta number, semidefinite programming, interior-point algorithms, cutting-plane methods, combinatorial optimization.

Acknowledgments: Research of the first author was partially supported by the DFG Emmy Noether project "Combinatorial Optimization in Physics (COPhy)" and MITACS, a Network of Centres of Excellence for the Mathematical Sciences in Canada. The second author's research was partially supported by the Natural Sciences and Engineering Research Council of Canada, and by a Humboldt Research Fellowship.

1 Introduction

The maximum-stable-set problem and the closely related maximum-clique-problem are operations research problems on undirected graphs. Given such a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a set $V \subseteq \mathcal{V}$ is said to be a stable set or a clique in $\mathcal{G}$ if for all pairs of distinct nodes $i \in V$ and $j \in V$ with $i \neq j$, $\{i, j\} \notin \mathcal{E}$ or $\{i, j\} \in \mathcal{E}$, respectively. The stability number $\alpha(\mathcal{G})$ and the clique number $\omega(\mathcal{G})$ are the maximum cardinality of any stable set and clique in $\mathcal{G}$, respectively. Since a set $V \subseteq \mathcal{V}$ is stable in $\mathcal{G}$ if and only if V is a clique in its complement $\bar{\mathcal{G}}$, the stability number of $\mathcal{G}$ is identical to the clique number of $\bar{\mathcal{G}}$. Hence we mostly focus in this paper on the stability number $\alpha(\mathcal{G})$. Practical applications of stable sets and maximum cliques arise in areas as varied as social networking [22], electrical engineering [27], environmental forest planning [4], bioinformatics clustering and prediction [49], computational chemistry [44], and coding theory [47].

Another closely related concept is that of graph coloring. Given an integer $k \geq 1$, a graph $\mathcal{G}$ is said to be *k-colorable* if there is an assignment of k colors to the nodes $\mathcal{V}$ of $\mathcal{G}$ such that adjacent nodes receive different colors, or equivalently, if $\mathcal{V}$ can be partitioned into k stable sets. The coloring number or chromatic number $\chi(\mathcal{G})$ is the smallest integer k so that $\mathcal{G}$ is k-colorable. The computation of stability, clique, and chromatic numbers is NP-hard [32]. Because in any k-coloring the nodes in a clique must receive different colors, it follows that $\alpha(\mathcal{G}) = \omega(\bar{\mathcal{G}}) \leq \chi(\bar{\mathcal{G}})$.. Several instances are known in which this inequality holds with equality, for example, if $\mathcal{G}$ (or equivalently $\bar{\mathcal{G}}$) is perfect, i.e. $\omega(\bar{\mathcal{G}}') = \chi(\bar{\mathcal{G}}')$ also for every induced subgraph $\bar{\mathcal{G}}'$ of $\bar{\mathcal{G}}$. In these cases, the stability, clique, and chromatic numbers can be computed in polynomial time using the Lovász theta function $\vartheta(\mathcal{G})$ (see Section 2).

The generalization of interior-point methods (IPMs) for linear programming (LP) to convex conic optimization [40], and especially to semidefinite programming (SDP) [1], has had a major impact on the development of tractable approximations for many NP-hard combinatorial problems. Indeed, the fact that $\vartheta(\mathcal{G})$ can be computed efficiently as the optimal value of an SDP is a truly beautiful and powerful consequence of the polynomial-time convergence of IPMs. Thus, IPMs provide the means to both efficiently solve the maximum-stable-set, maximum-clique, and minimum-graph-coloring problems for perfect graphs, and to prove their membership in P for which no purely combinatorial method is known [33].

Research on both theoretical and computational aspects of stability, clique, and chromatic numbers remains very active. In particular, several hierarchies of LP and SDP relaxations have been proposed based on an exact formulation of stability and clique numbers as solutions to certain copositive programs [5, 10, 12, 42]. Copositive programming is linear optimization over the cone of copositive matrices; see the recent surveys [6, 8, 13] and the bibliography [7] for details. However, although both the copositive cone and its completely positive dual cone are convex and have self-concordant barriers, these functions cannot be evaluated in polynomial time so that copositive programming remains NP-hard and IPMs cannot be used efficiently. The existing relaxation hierarchies therefore approximate the copositive cone. Unfortunately even their lower-order approximations are typically too large to be useful for computational purposes.

One of the simplest approximations of the copositive cone is the intersection of the semidefinite cone and the nonnegative cone. This is known as the doubly nonnegative (DNN) relaxation presented as problem (4) below. The DNN relaxation provides the bound $\vartheta'(\mathcal{G})$ which improves on the Lovász theta bound. However, formulating the DNN relaxation requires a quadratic number of nonnegativity constraints which makes the resulting SDP expensive to solve.

This paper is concerned with optimizing over the DNN relaxation using IPMs. The use of IPMs is attractive not only because of their proven polynomial-time convergence but also because they have consistently shown impressive robustness and efficiency in practice for general conic optimization problems. Specifically, we propose to view the nonnegativity constraints as cutting planes and to apply an interior-point cutting-plane scheme with selective addition of inequalities that dynamically adds the necessary constraints within the interior-point method.. From a theoretical perspective, the algorithm we propose is supported by the convergence results presented in the recent technical report [20]. From a computational perspective, the results in this paper extend those in the paper [18] and show the significant benefits from this approach for a different and generally difficult class of SDP relaxations.

We are aware of only relatively few papers that either explore similar approaches to the one described in this paper or offer extensive computational results specifically for computing $\vartheta'(\mathcal{G})$. To our knowledge, the most closely related study is that by Gruber and Rendl [26] who compute $\vartheta(\mathcal{G})$ with a primal-dual interior-point cutting-plane scheme similar to the strategy by Helmberg and Rendl [29]. More recently, Dukanovic and Rendl [11, 12] also test some of the newer copositive formulations with their primary focus on the resulting improvement of the bounds for stability, clique, and chromatic numbers. Although this second paper also reports on some results for $\vartheta'(\mathcal{G})$, we did not find running times or any other measures of computational efficiency discussed in either of these papers. Using a significantly different approach, two sets of highly competitive results for computing $\vartheta'(\mathcal{G})$ are given by two recent articles that develop methods of augmented-Lagrangian type based on Newton conjugate gradients or an alternating direction approach [52, 53]. Another recent development is the application of the augmented primal-dual method proposed in [30] to computing $\vartheta'(\mathcal{G})$ [9]. While the problems sizes and running times in [9, 52, 53] are unlikely to be met with any current IPM implementation, the results in this paper demonstrate the possibility of a fundamentally different and innovative way to handle large number of inequalities in the IPM framework that is also applicable to other broad classes of problems.

This paper is structured as follows. In Section 2, we cover the technical details of the problem formulations and relaxations described above.. Algorithms and cutting-plane methods, including our proposed method and its new modifications for the maximum-stable-set problem, are discussed in Section 3. Section 4 presents our computational findings from computing $\vartheta'(\mathcal{G})$ using both standard techniques IPM and our own method on different sets of instances, including instances arising from coding theory [47]. Section 5 summarizes our results and concludes the paper.

2 Problem Formulations and Relaxations

We denote by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ an undirected connected loopless graph with n nodes or vertices $\mathcal{V} = \{1, 2, \ldots, n\}$ and m edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. We let $\bar{\mathcal{G}} = (\mathcal{V}, \bar{\mathcal{E}})$ denote the complement graph of $\mathcal{G}$ for which $\{i, j\} \in \bar{\mathcal{E}}$ if and only if $\{i, j\} \notin \mathcal{E}$ and $i \neq j$. The symmetric $n \times n$ adjacency matrix of a graph $\mathcal{G}$ is denoted by $A(\mathcal{G})$. We write $e = (1, 1, \ldots, 1)^T \in \mathbb{R}^n$ for the vector of all ones, $E = ee^T \in \mathbb{R}^{n\times n}$ for the matrix of all ones, and $I \in \mathbb{R}^{n\times n}$ for the n-dimensional identity matrix. The set $\mathcal{S}^n := \{S \in \mathbb{R}^{n\times n} : S = S^T\}$ defines the cone of real symmetric $n \times n$ matrices, and we use

$$\begin{aligned} \mathcal{P}^n &:= \{S \in \mathcal{S}^n : u^T S u \geq 0 \text{ for all } u \in \mathbb{R}^n\} \\ \mathcal{N}^n &:= \{S \in \mathcal{S}^n : S_{ij} \geq 0 \text{ for all } (i, j)\} \end{aligned}$$

to denote the n-dimensional positive semidefinite and nonnegative cones respectively. For a positive semidefinite (nonnegative) matrix $X \in \mathcal{P}^n$ ($X \in \mathcal{N}^n$) we also write $X \succeq 0$ ($X \geq 0$), and for two matrices X and Y we define their inner product $X \bullet Y = \sum_{i=1}^n \sum_{j=1}^n X_{ij} Y_{ij} = \mathrm{Tr}(XY)$ where $\mathrm{Tr}(Z)$ is the trace of matrix Z.

The stability number $\alpha(\mathcal{G})$ is the optimal value of the linear binary program

$$\begin{aligned} \alpha(\mathcal{G}) = \quad \max \quad & \sum_{i=1}^n x_i && \text{(1a)} \\ \text{s.t.} \quad & x_i + x_j \leq 1 \text{ for all } \{i, j\} \in \mathcal{E} && \text{(1b)} \\ & x_i \in \{0, 1\} \text{ for all } i \in \mathcal{V} && \text{(1c)} \end{aligned}$$

where x is the incidence vector of the subset $V \subseteq \mathcal{V}$. The convex hull of all feasible incidence vectors of stable sets defines the stable set polytope

$$\mathrm{STAB}(\mathcal{G}) := \mathrm{conv}\{x \in \{0, 1\}^n : x_i + x_j \leq 1 \text{ for all } \{i, j\} \in \mathcal{E}\}.$$

The basic linear relaxation of $\mathrm{STAB}(\mathcal{G})$ replaces the binary constraint by box or nonnegativity constraints

$$\mathrm{FRAC}(\mathcal{G}) := \{x \in \mathbb{R}^n : x \geq 0,\ x_i + x_j \leq 1 \text{ for all } \{i, j\} \in \mathcal{E}\}.$$

FRAC($\mathcal{G}$) can be tightened either by adding the clique inequalities

$$\text{QSTAB}(\mathcal{G}) := \{x \in \mathbb{R}^n : x \geq 0,\ \sum_{i \in Q} x_i \leq 1 \text{ for all cliques } Q \subseteq \mathcal{V}\}$$

or by adding other types of valid inequalities such as the odd-cycle inequalities

$$\text{CSTAB}(\mathcal{G}) := \{x \in \text{FRAC}(\mathcal{G}) : \sum_{i \in V(C)} x_i \leq \tfrac{1}{2}(|V| - 1) \text{ for all odd cycles } C \subseteq \mathcal{E}\}.$$

While linear optimization over QSTAB remains NP-hard even for perfect graphs [23], the relaxations FRAC($\mathcal{G}$) and CSTAB($\mathcal{G}$) for arbitrary graphs can be solved in polynomial-time using IPMs [24] but provide much weaker bounds than the SDP relaxations yielding $\vartheta(\mathcal{G})$ and $\vartheta'(\mathcal{G})$ below. More details on LP and polyhedral relaxations can be found in the references [25, 33].

The maximum stable set problem can also be formulated as the quadratic program

$$\begin{aligned} \alpha(G) = \quad & \max \quad x^T x && \text{(2a)} \\ & \text{s.t.} \quad x_i x_j = 0 \text{ for all } \{i,j\} \in \mathcal{E} && \text{(2b)} \\ & \qquad\ \ x_i^2 = x_i \text{ for all } i \in \mathcal{V}.. && \text{(2c)} \end{aligned}$$

Lovász [34] showed that if a vector $x \in \mathbb{R}^n$ is feasible for (2), then the diagonal of the matrix $Y = \binom{x}{1}\binom{x}{1}^T = \begin{pmatrix} xx^T & x \\ x^T & 1 \end{pmatrix}$ is contained in

$$\text{TH}(\mathcal{G}) := \left\{ x \in \mathbb{R}^n : \exists\ Y = \begin{pmatrix} X & x \\ x^T & 1 \end{pmatrix} \succeq 0,\ \text{diag}(X) = x,\ X_{ij} = 0 \text{ for } \{i,j\} \in \mathcal{E} \right\}.$$

TH($\mathcal{G}$) is known as the theta body and satisfies STAB($\mathcal{G}$) $\subseteq$ TH($\mathcal{G}$) $\subseteq$ QSTAB($\mathcal{G}$) with equality if and only if $\mathcal{G}$ is perfect. Optimizing over TH($\mathcal{G}$) yields the Lovász theta number

$$\vartheta(\mathcal{G}) := \max \left\{ e^T x : x \in \text{TH}(\mathcal{G}) \right\}.$$

The theta number satisfies the sandwich inequality $\alpha(\mathcal{G}) \leq \vartheta(\mathcal{G}) \leq \chi(\bar{\mathcal{G}})$, thus yielding the equalities $\alpha(\mathcal{G}) = \vartheta(\mathcal{G})$ and $\omega(\mathcal{G}) = \chi(\mathcal{G}) = \vartheta(\bar{\mathcal{G}})$ if $\mathcal{G}$ is perfect.

The theta number can be computed in polynomial time using an IPM to solve the SDP problem

$$\begin{aligned} \vartheta(\mathcal{G}) = \quad & \max \quad E \bullet X && \text{(3a)} \\ & \text{s.t.} \quad I \bullet X = 1 && \text{(3b)} \\ & \qquad\ \ X_{ij} = 0 \text{ for all } \{i,j\} \in \mathcal{E} && \text{(3c)} \\ & \qquad\ \ X \succeq 0. && \text{(3d)} \end{aligned}$$

Problem (3) yields $\alpha(\mathcal{G})$ if the additional constraint $\text{rank}(X) = 1$ is enforced. However, this constraint is non-convex and cannot be handled efficiently.

It is nonetheless possible to improve the Lovász theta bound in an efficient manner. McEliece, Rodemich, and Rumsey [36] and Schrijver [45] considered the DNN relaxation

$$\begin{aligned} \vartheta'(\mathcal{G}) := \quad & \max \quad E \bullet X && \text{(4a)} \\ & \text{s.t.} \quad I \bullet X = 1 && \text{(4b)} \\ & \qquad\ \ X_{ij} = 0 \text{ for } \{i,j\} \in \mathcal{E} && \text{(4c)} \\ & \qquad\ \ X \succeq 0 && \text{(4d)} \\ & \qquad\ \ X \geq 0 && \text{(4e)} \end{aligned}$$

Alternatively, Szegedy [48] defined

$$\vartheta^+(\mathcal{G}) := \quad \max \quad E \bullet X \tag{5a}$$
$$\text{s.t.} \quad I \bullet X = 1 \tag{5b}$$
$$X_{ij} \leq 0 \text{ for } \{i,j\} \in \mathcal{E} \tag{5c}$$
$$X \succeq 0 \tag{5d}$$

which weakens the edge constraints by means of inequalities (5c) and thereby improves the bound on the chromatic number, yielding the amended chain of relaxation bounds

$$\alpha(\mathcal{G}) \leq \vartheta'(\mathcal{G}) \leq \vartheta(\mathcal{G}) \leq \vartheta^+(\mathcal{G}) \leq \chi(\bar{\mathcal{G}}).$$

In addition, several other classes of cutting-plane inequalities can also be derived from the various techniques of lift-and-project for general integer programs [3, 35, 46], some of which are nicely described for maximum-stable-set relaxations in the paper by Gruber and Rendl [26].

In this paper we focus on the computation of $\vartheta'(\mathcal{G})$, but we note that our proposed algorithm could also be used for computing $\vartheta^+(\mathcal{G})$ if desired.

3 Interior-Point Cutting-Plane Methods and New Algorithm

The basic idea of any cutting-plane method (CPM) is to replace a difficult problem containing a large number of inequalities by a series of easier relaxations that are successively augmented with violated inequalities until finding an optimal solution at which all the inequalities are satisfied, whether they have been added or not.

The standard CPM that we consider for the computation of $\vartheta'(\mathcal{G})$ is outlined as Algorithm 1. We begin by solving the smaller SDP problem (3) for an optimal solution X^*, and subsequently add only constraints $X_{ij} \geq 0$ for which $X^*_{ij} < 0$ and solve again. We define the set of added inequality constraints as

$$\mathcal{E}^+ := \{(i,j) : X_{ij} \geq 0 \text{ is present in the current relaxation}\}.$$

By repeating this process, we eventually must obtain an optimal solution X^* such that $X^* \geq 0$ and $\vartheta'(\mathcal{G}) = E \bullet X^*$.

The above scheme can be modified in several ways. One common adjustment exploits the fact that a constraint $\{i,j\} \in \mathcal{E}^+$ frequently turns out to be unnecessary after some number of updates of $\mathcal{E}^+$ because $X^*_{ij} > 0$ in subsequent optimal solutions. When this happens, we can remove that inequality from $\mathcal{E}^+$. Sophisticated tests for the redundancy of constraints exist; in the context of IPMs, the tests are often based on vanishing dual variables [14]. A second aspect with noticeable impact on the computational performance is the number of new inequalities that are added per iteration. We will study this issue in our computational experiments in Section 4. Finally, several techniques exist to utilize a priori knowledge from the previously solved relaxations for the choice of a better initial point for solving the updated problem. The state-of-the-art in such warm start strategies is documented in [15, 38].

The CPM outlined so far is still generic in the sense that we have not specified how to solve each individual relaxation in Step 2 of Algorithm 1. Whereas the standard cutting-plane method for LP is traditionally based on a dual simplex method to exploit its favorable warmstarting capabilities, CPMs for SDP are typically based on bundle or interior-point methods [15, 38]. On the one hand, bundle methods often offer better ways to exploit known structure and sparsity. This has led to some extremely good results for the max-cut relaxation whose polyhedral structure is well understood [21, 28, 43, 41]. On the other hand, IPMs are often preferred due to their overall robustness and efficiency in practice [2, 26, 29, 37, 39]. Moreover, IPMs can compete with alternative methods if many constraints are added at once, or if it is necessary to stabilize a cutting-plane or column-generation process [38]. IPMs also suffer far less from effects of degeneracy [37]. For the latter reasons, we use in this paper a primal-dual path-following IPM.

3.1 Basic Scheme of Primal-Dual IPMs

To establish some notation for the subsequent discussion of our new algorithm, let us briefly recall the major steps of primal-dual IPMs for solving the SDP relaxations in Step 2 of Algorithm 1. Starting from the initial

relaxation (3), in each iteration we have an SDP of the form

$$\begin{aligned} \max \quad & E \bullet X && \text{(6a)}\\ \text{s.t.} \quad & I \bullet X = 1 && \text{(6b)}\\ & X_{ij} = 0 \text{ for all } \{i,j\} \in \mathcal{E} && \text{(6c)}\\ & X_{ij} \geq 0 \text{ for all } \{i,j\} \in \mathcal{E}^+ && \text{(6d)}\\ & X \succeq 0. && \text{(6e)} \end{aligned}$$

The dual of problem (6) is given by

$$\min \left\{ t : S = tI + \sum\nolimits_{i\in\mathcal{E}} s_{ij}E_{ij} + \sum\nolimits_{i\in\mathcal{E}^+} s^+_{ij}E_{ij} - E \succeq 0,\ s^+_{ij} \geq 0 \right\} \tag{7}$$

where $S \in \mathbb{R}^{n\times n}$ is a symmetric, positive-definite matrix, t and s_{ij} for $\{i,j\} \in \mathcal{E}$ are free scalar variables, s^+_{ij} for $\{i,j\} \in \mathcal{E}^+$ are nonnegative scalar variables, and E_{ij} is the matrix with entries (i,j) and (j,i) equal to $\frac{1}{2}$ and 0 elsewhere.

Using a suitable symmetrization strategy [50] followed by Newton's method to solve these optimality conditions, we can compute an associated search direction $(\Delta X, \Delta S, \Delta t, \Delta s, \Delta s^+)$ at any point $(\hat{X}, \hat{S}, \hat{t}, \hat{s}, \hat{s}^+)$ from the resulting linear system

$$\begin{aligned} & I \bullet \Delta X = 1 - I \bullet \hat{X} && \text{(8a)}\\ & \Delta X_{ij} = -\hat{X}_{ij} \text{ for } \{i,j\} \in \mathcal{E} && \text{(8b)}\\ & \Delta S - (\Delta t)I - \sum\nolimits_{i\in\mathcal{E}} (\Delta s_{ij})E_{ij} + \sum\nolimits_{i\in\mathcal{E}^+} (\Delta s^+_{ij})E_{ij} \\ & = \hat{t}I + \sum\nolimits_{i\in\mathcal{E}} \hat{s}_{ij}E_{ij} + \sum\nolimits_{i\in\mathcal{E}^+} \hat{s}^+_{ij}E_{ij} - E - \hat{S} && \text{(8c)}\\ & X\Delta S + (\Delta X)S = \mu I - \hat{X}\hat{S} && \text{(8d)}\\ & \hat{X}_{ij}\Delta s^+_{ij} + (\Delta X_{ij})\hat{s}^+_{ij} = \mu - \hat{X}_{ij}\hat{s}^+_{ij} \text{ for all } \{i,j\} \in \mathcal{E}^+. && \text{(8e)} \end{aligned}$$

Choosing an appropriate step size β, we then update the current iterate

$$(\hat{X}, \hat{S}, \hat{t}, \hat{s}, \hat{s}^+) \leftarrow (\hat{X}, \hat{S}, \hat{t}, \hat{s}, \hat{s}^+) + \beta(\Delta X, \Delta S, \Delta t, \Delta s, \Delta s^+) \tag{9}$$

and for some parameter $\gamma \in [0, 1)$ reduce the barrier parameter according to the formula

$$\mu \leftarrow \gamma \left(\hat{X} \bullet \hat{S} + \sum\nolimits_{\{i,j\}\in\mathcal{E}^+} \hat{X}_{ij}\hat{s}^+_{ij} \right) / \left(n + \left|\mathcal{E}^+\right| \right). \tag{10}$$

3.2 New Cutting-Plane Interior-Point Algorithm

Several improvements have been proposed and implemented for the special case of interior-point CPMs that solve the relaxations in Step 2 of Algorithm 1 using an IPM as outlined in Algorithm 2. We already mentioned that the primal-dual nature of IPMs can be exploited effectively for several tests to detect vanishing variables. Moreover, the second-order nature of IPMs guarantees a reasonable level of robustness which enables the addition of relatively large numbers of equality or inequality constraints at intermediate iterates. Arguably the biggest challenge for IPMs is to warm start subsequent relaxations from previous solutions that are often not sufficiently interior; however, several researchers have recently addressed this question and proposed several strategies for significant new progress into this direction [15].

The robustness of IPMs, together with the observation that high-accuracy solutions are unnecessary during earlier stages of CPMs, motivate another modification to the above scheme. Rather than adding and removing constraints only after solving subsequent relaxations to approximate optimality, these new methods separate and add new violated inequalities when the current point is still relatively far from the optimal solution but sufficiently interior to possibly produce much stronger cuts [26, 29, 39]. In the context of LP relaxations, the early addition of violated cuts was suggested by Mitchell [37] and Mitchell and Borchers [39] who terminate

at suboptimal solutions and warm start successive relaxations by manually increasing small values of the current iterate or choosing the most recent previous primal-dual iterate that is strictly feasible, respectively.. The extension to SDP was explored by Helmberg and Rendl [29] who further distinguish between small and large adds at intermediate and final iterates of different rounds. In their method, a series of small adds allow the early addition of a certain number of violated inequalities which are compensated by pushing the current iterate back into the interior so that all inequalities are again satisfied, before each current relaxation is eventually solved for an optimal solution for a larger add of new constraints and a restart of the next relaxation round from a newly chosen initial point.

While IPMs are quite capable of absorbing such changes to the problem and accommodate the integration of a new sets of constraints, special care needs to be taken when initializing the new dual variables or modifying the previous iterate to maintain the algorithm's good performance. Unlike these previous methods, a new technique was recently proposed for handling large numbers of inequalities in cutting-plane schemes while avoiding a restart or the modification of the current iterate [18]. Most notably, the corresponding algorithm does not distinguish successive relaxations but solves the original problem in a single round, selects cut sets dynamically, and only introduces or redefines sets of new auxiliary variables to temporarily relax the interiority constraints of the original problem. In particular, the observed stability of the underlying primal-dual IPM enables the algorithm to continue without any changes to the original primal-dual iterate, which is updated only by the corresponding Newton steps that guarantees the iterate's convergence to an optimal solution. The new algorithm is outlined in Algorithm 3 and basically works like the regular path-following IPM in Algorithm 2 with an integrated cutting-plane mechanism, so that the number of dual variables s_{ij}^+ keeps changing based on the subset $\mathcal{E}^+$ of constraints currently in the problem.

We briefly describe the specific adjustments of this algorithm. Given a current primal iterate $\hat{X}$ for which $\hat{X}_{ij} < 0$ for some $\{i,j\} \notin \mathcal{E}^+$, i.e. a violated inequality not currently added to the problem, we add the corresponding inequality constraint as $X_{ij} - Y_{ij} = 0$ with an auxiliary variable $Y_{ij} \geq 0$ that temporarily replaces nonnegativity of X_{ij} and removes the need to adjust its current value. To ease this discussion, we temporarily denote the index set of newly added inequalities by $\mathcal{E}^+_{\text{add}}$. To maintain primal-dual symmetry, for each $(i,j) \in \mathcal{E}^+_{\text{add}}$ we also add a dual slack $z_{ij} \geq 0$ to the corresponding dual constraint $s_{ij}^+ - z_{ij} = 0$ which makes s_{ij}^+ a free variable that can be initialized to zero. Both computational experience and a theoretical analysis of this approach show that the slacks Y_{ij} are generally sufficiently decoupled from the other constraints to mitigate the impact of adding new violated cuts [16, 19]. In particular, it turns out that initial values for Y_{ij} and z_{ij} can be chosen much smaller than necessary when adjusting the original matrix $\hat{X}$ or its entry $\hat{X}_{ij}$ and s_{ij}^+ in order to avoid jamming, and we have found that $\hat{Y}_{ij} = \hat{z}_{ij} = \sqrt{\mu}$ is especially meaningful because it allows to preserve the current value of μ from the previous iteration:

$$\frac{\hat{X} \bullet \hat{S} + \sum_{\{i,j\} \in \mathcal{E}^+} \hat{X}_{ij} \hat{s}_{ij}^+ + \sum_{\{i,j\} \in \mathcal{E}^+_{\text{add}}} \hat{Y}_{ij} \hat{z}_{ij}}{n + |\mathcal{E}^+| + \left|\mathcal{E}^+_{\text{add}}\right|} = \frac{(n + |\mathcal{E}^+|)\mu + \left|\mathcal{E}^+_{\text{add}}\right|\mu}{n + |\mathcal{E}^+| + \left|\mathcal{E}^+_{\text{add}}\right|} = \mu. \tag{11}$$

After new constraints have been added, we remove inequalities cuts $X_{ij} = Y_{ij} \geq 0$ and their dual $s_{ij}^+ = z_{ij} \geq 0$ for $(i,j) \in \mathcal{E}^+$ if the remaining residuals $|X_{ij} - Y_{ij}|$ and $\left|s_{ij}^+ - z_{ij}\right|$ have become sufficiently small and the primal-dual ratio Y_{ij}/z_{ij} has grown sufficiently large. Using the theory of indicator functions, any strictly positive threshold $\delta > 0$ will eventually detect the convergence of vanishing z_{ij} and thereby allow to remove (i,j) from $\mathcal{E}^+$ and drop the dual variable and auxiliary primal-dual slacks $(s_{ij}^+, Y_{ij}, z_{ij})$ from the problem [14, 18].

For step 3 in Algorithm 3, note that the corresponding Newton system is not the same as in (8) as it is augmented by the new dual and auxiliary slack variables $(\hat{s}_{ij}^+, \hat{Y}_{ij}, \hat{z}_{ij})$ for $\{i,j\} \in \mathcal{E}^+$. As demonstrated in [16, 19], the simple structure of the additional constraints $X_{ij} - Y_{ij} = 0$ and $s_{ij}^+ - z_{ij} = 0$ allows the use of a slightly perturbed system of equal size as (8). Finally, the full termination criteria in step 6 also includes several other (possibly different) tolerance thresholds for both primal and dual infeasibility; writing down these details is straightforward.

1. Drop all nonnegativity constraints and initialize the set of inequalities $\mathcal{E}^+ = \emptyset$.
2. Compute an optimal solution $X^* \in \mathcal{P}^n$ for the SDP relaxation
$$\max\{E \bullet X : X \in \mathcal{P}^n,\ X_{ij} = 0 \text{ for all } (i,j) \in \mathcal{E},\ X_{ij} \geq 0 \text{ for all } (i,j) \in \mathcal{E}^+\}.$$
3. If $X^*_{ij} < 0$ for any $\{i,j\} \notin \mathcal{E}$, add $X_{ij} \geq 0$ as new inequality cut: $\mathcal{E}^+ \leftarrow \mathcal{E}^+ \cup \{(i,j)\}$.
4. If new constraints were added, remove redundant constrains and go back to step 2.
5. If no new constraints were added, stop: $X^* \in \mathcal{P}^n \cap \mathcal{N}^n$ is optimal, $\vartheta'(\mathcal{G}) = E \bullet X^*$.

Algorithm 1: Standard Cutting-Plane Method for the Computation of $\vartheta'(\mathcal{G})$

1. Given the primal-dual pair (6,7), initialize $(\hat{X}, \hat{S}, \hat{t}, \hat{s}, \hat{s}^+)$ with $(\hat{X}, \hat{S}) \succ 0$ and $(\hat{X}_{ij}, \hat{s}^+_{ij}) > 0$ for all $\{i,j\} \in \mathcal{E}^+$; select $\gamma \in [0,1)$ and a small tolerance value $\varepsilon > 0$.
2. Set or update the barrier parameter μ according to the reduction formula in (10).
3. If $\max\{\mu, \left|I \bullet \hat{X} - 1\right|, \left|A \bullet \hat{X}\right|, \left|\hat{X}_{ij}\right| : \{i,j\} \in \mathcal{E}^+\} \leq \varepsilon$, stop with ε-optimal solution.
4. Compute a new direction, step size, and iterate from (8,9) and go back to step 2.

Algorithm 2: Standard Primal-Dual Path-Following IPM for Step 2 in Algorithm 1

1. Drop all nonnegativity constraints and set $\mathcal{E}^+ = \emptyset$; initialize $(\hat{X}, \hat{S}, \hat{t}, \hat{s})$ with $(\hat{X}, \hat{S}) \succ 0$; set $\gamma \in [0,1)$, tolerance $\varepsilon > 0$, indicator threshold $\delta > 0$, and separation frequency κ.
2. Set or update the barrier parameter μ according to the reduction formula in (11).
3. Compute directions, step sizes, and iterates from an amended Newton system (8).
4. If $\hat{X}_{ij} < 0$ for any $\{i,j\} \notin \mathcal{E}$, add $X_{ij} \geq 0$ as new inequality cut: $\mathcal{E}^+ \leftarrow \mathcal{E}^+ \cup \{(i,j)\}$; add the associated dual variable and primal-dual slacks $(\hat{s}^+_{ij}, \hat{Y}_{ij}, \hat{z}_{ij}) = (0, \sqrt{\mu}, \sqrt{\mu})$.
5. If $\left|\hat{X}_{ij} - \hat{Y}_{ij}\right| < \varepsilon$ and $\hat{Y}_{ij}/\hat{z}_{ij} > \delta$, drop cut and $(s^+_{ij}, Y_{ij}, z_{ij})$: $\mathcal{E}^+ \leftarrow \mathcal{E}^+ \setminus \{(i,j)\}$.
6. If $\mu < \varepsilon$ and $\hat{X} \geq 0$, stop: $\vartheta'(\mathcal{G}) = E \bullet \hat{X}$ is optimal; otherwise go back to step 2..

Algorithm 3: New Cutting-Plane Interior-Point Algorithm for the Computation of $\vartheta'(\mathcal{G})$

4 Computational Experiments

We implemented a standard interior-point CPM as outlined in Algorithms 1 and 2 and our new method in Algorithm 3 using MATLAB. All computations were carried out on a 2.0 GHz AMD Dual Opteron Processor. For the solution of the relaxations in Step 2 of Algorithm 1, we used the solver SDPT3 [51] with default settings and default initial points, which implements a primal-dual path-following predictor-corrector method similar to Algorithm 2. Furthermore, we removed previous inequality cuts if the corresponding feasibility residual fell below $\varepsilon = 10^{-4}$ and its primal-dual ratio exceeded the threshold value $\delta = 10^2$. While there may be different values of ε and δ that can further improve the method, these particular choices delivered good practical performance.

We then conducted a series of computational experiments using different variants of Algorithm 3. To ensure comparability, we used the same combination of Algorithms 1 and 2 as before but now restricted the maximum number of Newton iterations taken during each call to the SDP solver to κ. We supplied the previous iterate as our own starting point, and disabled all preprocessing. As initial point we choose the scaled identity matrix nI for both the primal and dual matrix, and zero for all other dual variables.. The separation, addition, and removal of inequalities were done by complete enumeration and based on the two criteria on feasibility residual and primal-dual ratio..

Our experiments were carried out on randomly generated graphs as well as on two classes of instances from the literature. In particular, we investigated whether there is a noticeable difference from restricting the number of new constraints per iteration, or allowing intermediate steps before new constraints are added to possibly accelerate convergence but risk a larger number of deeper cuts at a later iterate. For the first variation, we decided to distinguish between *full adds* that add all new constraints associated with negative entries $\hat{X}_{ij} < 0$, and *sparse adds* for which we initially sort all current values in increasing order and then select at most n of the most negative entries so that no two newly added cuts $X_{ij} = 0$ or $X_{ij} = Y_{ij}$ and $Y_{ij} \geq 0$ share a common row or column index i or j, respectively. For the second, we specify the additional separation frequency κ that allows to compare the algorithm when adding or removing constraints after every iteration (if $\kappa = 1$) with variants that also allow few regular intermediate steps (if $\kappa > 1$).

4.1 Results on Randomly Generated Graphs

The first set of problem instances were created using the commonly used random graph generator RUDY written by G. Rinaldi that is available at

`http://www-user.tu-chemnitz.de/~helmberg/rudy.tar.gz.`

Specifically, for every size $n \in \{100, 200, \ldots, 600\}$ we generated five graphs using the command

`rudy -rnd_graph n dens rnd_seed`

where the edge densities `dens` $\in \{40, 10, 4, 2.5, 1.6, 1\}$ percent are chosen so to achieve graphs with no more than $2,000$ edges, and the random seed `rnd_seed` is varied between $1, 2, 3, 4, 5$. The number of associated nonnegativity constraints accordingly ranges between 4,950 for $n = 100$ and 179,700 for $n = 600$ so that their separation can still be done reasonably quickly using complete enumeration.

We first solved each of the above problems using our version of Algorithms 1 and 2 with both a full add and a sparse add. The average results of the five instances for each graph size are given in Table 1 and show running times (in cpu seconds), the total number of iterations (summed over all individual relaxations), and the total number of inequalities that remain added at optimality (with the number of removals indicated in parenthesis). On the one hand, these results suggest that sparse adds are typically more efficient in identifying a smaller set of relevant cuts because smaller numbers of inequalities remain added at optimality or are removed in previous stages of the algorithm. On the other hand, sparse adds also lead to additional iterations because it becomes necessary to solve more yet smaller-sized relaxations until a relevant set of cuts has been identified and added to the problem. In particular, the trade off between a full and sparse add, or equivalently between the size and number of relaxations that need to be solved, suggests a full add if the initial problem is relatively small ($n \leq 400$) but switches to sparse adds for the larger and sparser instances with 500 and 600 nodes.

We then looked at the computational impact of using a full add versus a sparse add, and of varying the number κ of regular intermediate steps in unit increments between 1 and 10 for Algorithm 3. The corresponding results are given in Tables 2 to 7 separated by problem size for better readability; a comparison

Table 1: Running times (in cpu seconds), number of iterations, and number of added (and removed) inequality cuts at optimality when computing $\vartheta'(\mathcal{G})$ for random graphs using the *standard interior-point CPM in Algorithms 1 and 2* with full adds and sparse adds.

			full add				sparse add			
n	m	dens	time	iter	inequalities		time	iter	inequalities	
100	1980	40%	125	77	216	(± 36)	257	165	203	(± 10)
200	1990	10%	264	97	644	(± 158)	618	274	518	(± 22)
300	1794	4%	326	110	848	(± 159)	671	315	552	(± 17)
400	1995	2.5%	442	111	902	(± 110)	811	279	525	(± 9)
500	1996	1.6%	897	186	903	(± 70)	872	264	444	(± 5)
600	1797	1%	1298	258	960	(± 39)	1016	282	399	(± 8)

Tables 2 to 7: Running times (in cpu seconds), number of iterations, and number of added (and removed) inequality cuts at optimality when computing $\vartheta'(\mathcal{G})$ for random graphs using the *integrated interior-point CPM in Algorithm 3* with full adds and sparse adds, and different separation frequencies κ.

Table 2: Results on graphs with $n = 100$.

	full add				sparse add			
κ	time	iter	inequalities		time	iter	inequalities	
1	83	34	244	(± 8)	89	38	236	(± 7)
2	68	32	242	(± 6)	81	38	232	(± 4)
3	59	28	234	(± 5)	89	44	222	(± 3)
4	59	30	229	(± 5)	92	48	221	(± 0)
5	68	36	232	(± 5)	107	58	221	(± 0)
6	53	29	223	(± 0)	111	62	216	(± 0)
7	70	39	232	(± 6)	118	68	212	(± 0)
8	74	41	246	(± 1)	131	76	214	(± 0)
9	89	50	245	(± 8)	148	87	217	(± 0)
10	98	55	252	(± 0)	165	98	219	(± 0)

Table 3: Results on graphs with $n = 200$.

	full add				sparse add			
κ	time	iter	inequalities		time	iter	inequalities	
1	219	69	777	(± 0)	402	164	545	(± 0)
2	237	74	780	(± 21)	410	162	558	(± 5)
3	248	77	771	(± 0)	387	146	577	(± 0)
4	242	74	810	(± 2)	386	150	566	(± 4)
5	219	67	770	(± 38)	328	124	576	(± 7)
6	222	69	768	(± 40)	293	111	573	(± 0)
7	199	66	688	(± 29)	289	109	568	(± 0)
8	170	53	796	(± 0)	317	123	560	(± 0)
9	175	54	806	(± 0)	344	136	554	(± 0)
10	188	59	796	(± 9)	382	152	555	(± 0)

Table 4: Results on graphs with $n = 300$.

	full add				sparse add			
κ	time	iter	inequalities		time	iter	inequalities	
1	243	79	957	(± 0)	394	172	585	(± 0)
2	272	83	970	(± 0)	429	186	592	(± 0)
3	283	88	962	(± 0)	463	201	589	(± 0)
4	282	90	929	(± 27)	471	200	600	(± 1)
5	281	83	990	(± 14)	435	181	608	(± 0)
6	295	89	972	(± 8)	429	180	608	(± 1)
7	307	91	956	(± 37)	357	147	627	(± 1)
8	209	63	934	(± 0)	351	139	622	(± 0)
9	203	60	994	(± 0)	405	169	610	(± 0)
10	225	67	967	(± 40)	380	158	610	(± 0)

Table 5: Results on graphs with $n = 400$.

	full add				sparse add			
κ	time	iter	inequalities		time	iter	inequalities	
1	370	90	971	(± 13)	502	160	596	(± 0)
2	375	89	963	(± 0)	491	160	566	(± 0)
3	370	88	970	(± 0)	556	181	560	(± 0)
4	453	100	1008	(± 1)	532	171	565	(± 0)
5	396	91	968	(± 5)	537	171	563	(± 4)
6	361	85	961	(± 0)	546	173	567	(± 0)
7	345	80	980	(± 15)	537	167	583	(± 0)
8	369	85	993	(± 18)	485	151	601	(± 0)
9	318	72	960	(± 2)	425	128	597	(± 0)
10	356	80	1005	(± 0)	449	141	587	(± 0)

Table 6: Results on graphs with $n = 500$.

	full add				sparse add			
κ	time	iter	inequalities		time	iter	inequalities	
1	347	78	922	(± 0)	481	138	497	(± 0)
2	406	89	939	(± 0)	638	181	511	(± 5)
3	390	84	913	(± 36)	587	167	492	(± 0)
4	416	90	916	(± 1)	555	160	474	(± 0)
5	433	91	968	(± 1)	612	169	518	(± 1)
6	406	84	967	(± 0)	604	170	480	(± 0)
7	349	73	941	(± 0)	592	166	492	(± 1)
8	333	70	961	(± 3)	499	137	494	(± 2)
9	317	72	780	(± 0)	450	123	501	(± 0)
10	366	77	917	(± 6)	460	124	514	(± 0)

Table 7: Results on graphs with $n = 600$.

	full add				sparse add			
κ	time	iter	inequalities		time	iter	inequalities	
1	371	77	979	(± 0)	595	155	481	(± 2)
2	475	95	982	(± 2)	615	155	518	(± 2)
3	430	85	912	(± 53)	589	149	500	(± 2)
4	486	95	940	(± 45)	616	157	508	(± 5)
5	463	92	970	(± 15)	585	147	519	(± 2)
6	539	103	986	(± 11)	616	155	505	(± 9)
7	461	91	905	(± 4)	549	141	471	(± 4)
8	445	85	917	(± 70)	548	140	462	(± 4)
9	494	92	921	(± 66)	502	125	488	(± 0)
10	465	87	989	(± 10)	521	130	504	(± 5)

across problem sizes is facilitated using Figures 1 and 2 that show the different running times, number of iterations, and number of inequalities at optimality for increasing problem size n and different separation frequencies κ.

First, it is evident that our new approach is a significant improvement over the standard cutting-plane method. We also see that a full add leads to larger relaxations than a sparse add but finds the optimal solution more quickly and using fewer iterations. In contrast with Table 1, however, this holds now for all problem sizes. As is to be expected, the running times increase slightly with problem dimension for both a full add and sparse add, and the same trend can be observed with respect to iterations and inequalities for a full add, in principle. Best seen from the plots in Figures 1 and 2 is the almost identical number of about 1,000 inequalities for all larger instances when using a full add, and the (possibly surprising) observation that the number of iterations and inequalities does not seem to correlate with the problem size for a sparse add

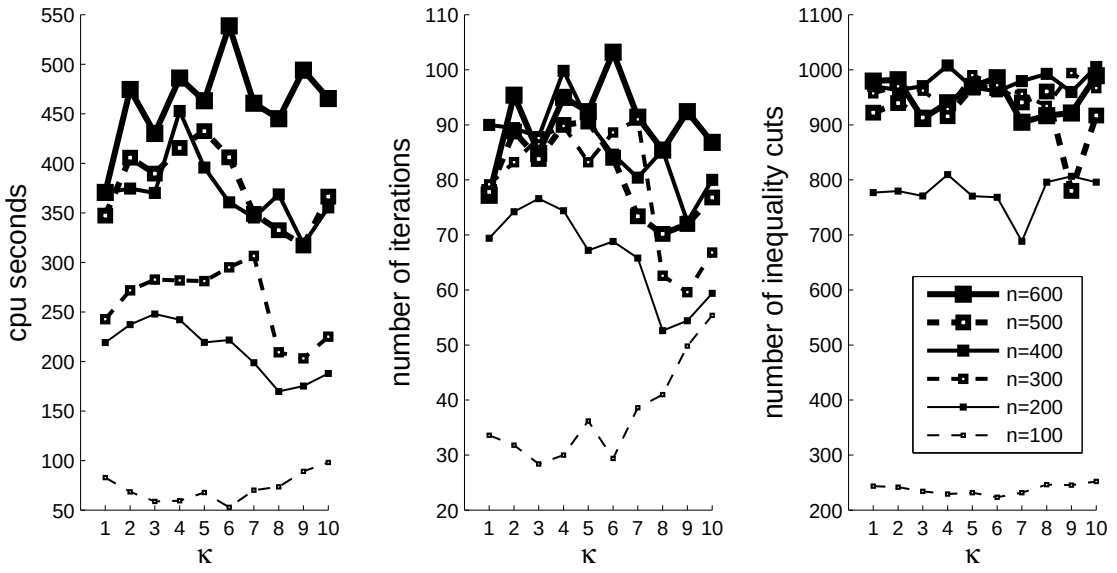

Figure 1: Running times (in cpu seconds), number of iterations, and number of added (and removed) inequality cuts at optimality when computing $\vartheta'(\mathcal{G})$ for random graphs using the integrated interior-point CPM in Algorithm 3 with *full adds* and different separation frequencies κ.

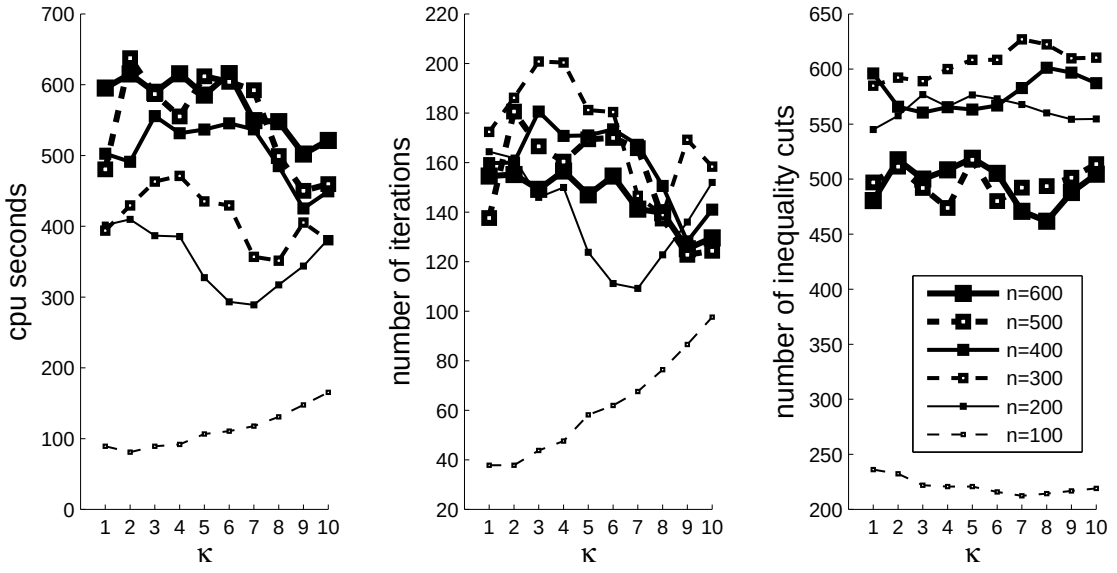

Figure 2: Running times (in cpu seconds), number of iterations, and number of added (and removed) inequality cuts at optimality when computing $\vartheta'(\mathcal{G})$ for random graphs using the integrated interior-point CPM in Algorithm 3 with *sparse adds* and different separation frequencies κ.

(excluding the smallest instances with $n = 100$ at the far bottom). Of course, it is important to recall that in spite of the increasing number of nodes we always maintain about $2,000$ edges and therefore compare graphs with very different sparsity structures. In summary, these results confirm the previously observed trade off between larger relaxation sizes yet smaller running times due to fewer iterations for a full add, and the more accurate identification of a relevant set of cutting planes at the cost of a larger running time for a sparse add.

Regarding the separation frequencies, we note that there is no clear trend as to an optimal parameter value especially for the running times for a full add, and the number of inequalities for a full add and a

sparse add. Excluding again the smallest instances with $n = 100$, the running times for a sparse add and the number of iterations for both a full and a sparse add seem to show an overall decrease for larger values of κ, although still subject to quite a bit of variation. Most notably, and contrasting with our earlier experience with a similar approach for max-cut in which we observed that κ values of 2 or 3 gave the most competitive running times [17, 18], the new features implemented for our current method seem to further improve the overall robustness of the algorithm and lead to more consistent results over a wider spectrum of parameter choices.

Finally, we note that any tested variant of the integrated interior-point CPM in Algorithm 3 improves the performance of the standard CPM both in terms of running times and total number of iterations, while finding very similar sets and practically the same number of cuts at optimality. This last aspect is particularly important as it convincingly demonstrates the general validity of adding and dropping inequalities at intermediate iterates without compromising the progress of the algorithm.

4.2 Results on DIMACS and Sloane Test Graph Instances

We applied Algorithm 3 to some of the standard benchmark problems from the Second DIMACS Implementation Challenge on Maximum Clique, Graph Coloring, and Satisfiability [31] that are available on the ftp site

`ftp://dimacs.rutgers.edu/pub/challenge/`

and to some independent set problems arising in coding theory [47] provided on the web site

`http://www2.research.att.com/~njas/doc/graphs.html`

maintained by Sloane. The DIMACS collection also includes problems based on coding theory (johnson and hamming instances) as well as various other applications including fault diagnosis, randomized vertex covers (sanr instances), and some specific set covering instances of the Steiner Triple problem (MANN instances). As all the DIMACS instances are formulated as maximum-clique problems, we converted them to equivalent stable set problems on the complement graphs (for simplicity we refer to these problems by the same name) and then chose all of those problems with no more than 2,000 nodes and 5,000 edges. The representative results in Tables 8 and 9 give for each graph $\mathcal{G}$ the corresponding problem dimensions n and m, the true stability (or clique) number $\alpha(\mathcal{G})$, the computed integer bound from $\vartheta'(\mathcal{G})$ using problem (4), the computed integer bound from $\vartheta(\mathcal{G})$ from (3), and the running time (in seconds), number of iterations, and number of remaining and removed inequalities at optimality when using Algorithm 3 with a full add and $\kappa = 1$. Note that the values for $\vartheta(\mathcal{G})$ and $\vartheta'(\mathcal{G})$ reported here are not new and consist of the computed values rounded down to the nearest integer.

Table 8: Exact stability numbers $\alpha(\mathcal{G})$ and known Lovász bounds $\vartheta(\mathcal{G})$ (rounded down to the nearest integer) with running times (in cpu seconds), number of iterations, and number of added (and removed) inequality cuts at optimality when computing $\vartheta'(\mathcal{G})$ for *DIMACS graphs* using the integrated interior-point CPM in Algorithm 3 with full adds and $\kappa = 1$.

Graph $\mathcal{G}$	n	m	$\alpha(\mathcal{G})$	$\vartheta'(\mathcal{G})$	$\vartheta(\mathcal{G})$	time	iter	inequalities	
hamming6-2	64	192	32	32	32	1	11	0	(± 0)
hamming6-4	64	1312	4	4	5	10	13	32	(± 0)
hamming8-2	256	1024	128	128	128	7	10	0	(± 0)
johnson8-2-4	28	168	4	4	4	0	8	0	(± 0)
johnson8-4-4	70	560	14	14	14	2	10	0	(± 0)
johnson16-2-4	120	1680	8	8	8	14	9	0	(± 0)
san200_0.9_1	200	1990	70	70	70	26	14	0	(± 0)
san200_0.9_2	200	1990	60	60	60	23	13	0	(± 0)
san200_0.9_3	200	1990	44	44	44	33	21	0	(± 0)
sanr200_0.9	200	2037	42	48	49	295	95	673	(± 114)
MANN_a9	45	72	16	17	17	0	12	0	(± 0)
MANN_a27	378	702	126	132	132	11	13	0	(± 0)
MANN_a45	1035	1980	345	356	356	188	17	0	(± 0)

Table 9: Exact stability numbers $\alpha(\mathcal{G})$ and known Lovász bounds $\vartheta(\mathcal{G})$ (rounded down to the nearest integer) with running times (in cpu seconds), number of iterations, and number of added (and removed) inequality cuts at optimality when computing $\vartheta'(\mathcal{G})$ for *Sloane graphs* using the integrated interior-point CPM in Algorithm 3 with full adds and $\kappa = 1$.

Graph $\mathcal{G}$	n	m	$\alpha(\mathcal{G})$	$\vartheta'(\mathcal{G})$	$\vartheta(\mathcal{G})$	time	iter	inequalities	
1dc.64	64	543	10	10	10	13	35	501	(± 0)
1dc.128	128	1471	16	16	16	498	76	2666	(± 6)
1dc.256	256	3839	30	30	30	262	26	1343	(± 0)
1et.64	64	264	18	18	18	1	12	0	(± 0)
1et.128	128	672	28	29	29	7	17	154	(± 0)
1et.256	256	1664	50	54	55	812	104	2261	(± 294)
1et.512	512	4032	100	103	104	1794	77	1512	(± 540)
1tc.64	64	192	20	20	20	1	15	0	(± 22)
1tc.128	128	512	38	38	38	3	15	0	(± 0)
1tc.256	256	1312	63	63	63	64	62	377	(± 0)
1tc.512	512	3264	110	112	113	3025	156	2430	(± 0)
1zc.128	128	1120	18	20	20	9	13	0	(± 21)
1zc.256	256	2816	36	37	38	138	32	56	(± 0)

The results in Table 8 first illustrate the advantage of a cutting-plane approach to compute $\vartheta'(\mathcal{G})$ compared to solving the full relaxation directly. Namely, these results show that the DNN relaxation does not always improve the SDP relaxation, and that the Lovász function $\vartheta(\mathcal{G})$ in all but two instances (`hamming6-4` and `sanr200_0.9`) already gives the exact stability number for an optimal matrix that is both positive semidefinite and nonnegative, without the need of explicit nonnegativity constraints. Consequently, in these cases the large majority of inequalities can remain dropped so that there is no difference between a full and a sparse add. Considering the two instances in which $\vartheta'(\mathcal{G})$ improves $\vartheta(\mathcal{G})$ so that several inequalities become necessary, however, we observe that specifically for the larger problem `sanr200_0.9` with 200 nodes and 2,037 edges the running time (295 sec), number of iterations (95), and number of inequalities (673 ± 114) fall within those for the random instances of similar size in Table 3, with $n = 200$ and either a full or sparse add (time: 219/402 sec, iter: 69/164, ineq: 777/545).

Finally, the selected results in Table 9 show that for the Sloane test graphs, $\vartheta'(\mathcal{G})$ frequently improves $\vartheta(\mathcal{G})$ and often requires a considerable number of additional inequalities. These results are obtained with the same parameter settings as before, using a full add and a separation frequency $\kappa = 1$. Although we found that specific instances can be solved faster, in fewer iterations, or with a smaller number of added inequalities for different parameter choices, these results suffice to demonstrate the general success and overall robustness of our proposed method to dynamically select and add the relevant inequalities.

5 Conclusions

We proposed an interior-point cutting-plane method to efficiently handle the large number of nonnegativity constraints in the DNN relaxation for instances of the stable set and maximum clique problems. The method views the nonnegativity constraints as cutting planes and applies an interior-point cutting-plane scheme with selective addition of inequalities that dynamically adds the necessary constraints within the interior-point method.. Supported by the theoretical convergence results presented in the recent technical report [20], the method is shown to deliver significant computational benefits for this class of challenging SDP relaxations.

Our computational experience is summarized in two observations:

1. The computational comparison of full adds versus sparse adds reveals the characteristic trade off in CPMs between the aggressive addition of new constraints that reduce the number of iterations but also cause the underlying Newton systems to grow more rapidly, or a more conservative separation strategy that keeps the size of each relaxation smaller but typically requires more iterations until a relevant cut set can be determined. Although a best strategy is unlikely to exist, we find that *the ability to quickly drop seemingly redundant inequalities provides the integrated technique with some advantages when using a full add compared to both a sparse add and the use of a more standard technique.*

2. The variation of the frequency for adding and removing inequalities shows that a good performance can be achieved for a wide spectrum of strategies. This is in contrast to previous experience with triangle inequalities for max-cut where the best results were obtained for $\kappa = 2$ or 3. The reason may be either the structural differences in these two problems and the different classes of cutting planes used, or a consequence of our improved implementation of the detection and handling of cuts specifically for this new class of problems. It is important to note that the underlying interior-point code was the same in both cases.

In summary, these results support the paradigm of our method to add a limited number of cuts early on to steer the algorithm more quickly towards an optimal solution, but also indicate the general robustness and stability of this new method when taking few intermediate iterations or adding significantly larger numbers of inequalities.

References

[1] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optim.*, 5(1):13–51, 1995.

[2] M. F. Anjos and A. Vannelli. Computing globally optimal solutions for single-row layout problems using semidefinite programming and cutting planes. *INFORMS J. Comput.*, 20(4):611–617, 2008.

[3] E. Balas, S. Ceria, and G. Cornuéjols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Math. Programming*, 58(3, Ser. A):295–324, 1993.

[4] F. Barahona, R. Epstein, and A. Weintraub. Habitat dispersion in forest planning and the stable set problem. *Operations Research*, 40(1-Supplement-1):S14–S21, 1992.

[5] I. M. Bomze and E. De Klerk. Solving standard quadratic optimization problems via linear, semidefinite and copositive programming. *J. Global Optim.*, 24(2):163–185, 2002.

[6] I.M. Bomze. Copositive optimization – recent developments and applications. *Europ. J. Oper. Research*, page to appear, 2011.

[7] I.M. Bomze, W. Schachinger, and G. Uchida. Think co(mpletely)positive ! – properties, examples and a commented bibliography on copositive optimization. Technical Report 2011-02, ISOR, Univ. Vienna, 2011.

[8] S. Burer. Copositive programming. In Miguel F. Anjos and Jean Baptiste Lasserre, editors, *Handbook of Semidefinite, Cone and Polynomial Optimization: Theory, Algorithms, Software and Applications, to appear.*, International Series in Operations Research and Management Science. Springer, New York, 2011.

[9] T. Davi and F. Jarre. Solving large scale problems over the doubly nonnegative cone. Technical report, Institut für Informatik, Universität Düsseldorf, 2011.

[10] E. de Klerk and D. V. Pasechnik. Approximation of the stability number of a graph via copositive programming. *SIAM J. Optim.*, 12(4):875–892, 2002.

[11] I. Dukanovic and F. Rendl. Semidefinite programming relaxations for graph coloring and maximal clique problems. *Math. Program.*, 109(2-3, Ser. B):345–365, 2007.

[12] I. Dukanovic and F. Rendl. Copositive programming motivated bounds on the stability and the chromatic numbers. *Math. Program.*, 121(2, Ser. A):249–268, 2010.

[13] M. Dür. Copositive programming — a survey. In Moritz Diehl, Francois Glineur, Elias Jarlebring, and Wim Michiels, editors, *Recent Advances in Optimization and its Applications in Engineering*, pages 3–20. Springer, Berlin Heidelberg New York, 2010.

[14] A. S. El-Bakry, R. A. Tapia, and Y. Zhang. A study of indicators for identifying zero variables in interior-point methods. *SIAM Rev.*, 36(1):45–72, 1994.

[15] A. Engau. Recent progress in interior-point methods: Cutting plane methods and warm starts. In M. F. Anjos and J. B. Lasserre, editors, *Handbook of Semidefinite, Cone, and Polynomial Optimization*. Springer, in preparation.

[16] A. Engau, M. F. Anjos, and A. Vannelli. A primal-dual slack approach to warmstarting interior-point methods for linear programming. In J. W. Chinneck, B. Kristjansson, and M. J. Saltzman, editors, *Operations Research and Cyber-Infrastructure*, pages 195–217. Springer, 2009.

[17] A. Engau, M. F. Anjos, and A. Vannelli. An improved interior-point cutting-plane method for binary quadratic optimization. *Electronic Notes on Discrete Mathematics*, 36:743–750, 2010.

[18] A. Engau, M. F. Anjos, and A. Vannelli. On handling cutting-planes in interior-point methods for solving semidefinite relaxations of binary quadratic optimization problems. *Optim. Meth. Soft.*, 2010. forthcoming.

[19] A. Engau, M. F. Anjos, and A. Vannelli. On interior-point warmstarts for linear and combinatorial optimization. *SIAM J. Optim.*, 10(4):1828–1861, 2010.

[20] A. Engau and M.F. Anjos. A primal-dual interior-point algorithm for linear programming with selective addition of inequalities. Cahier du GERAD G-2011-44, GERAD, Montreal, QC, Canada, 2011.

[21] I. Fischer, G. Gruber, F. Rendl, and R. Sotirov. Computational experience with a bundle approach for semidefinite cutting plane relaxations of Max-Cut and equipartition. *Math. Program.*, 105(2-3, Ser. B):451–469, 2006.

[22] S. Fortunato. Community detection in graphs. *Phys. Rep.*, 486(3-5):75–174, 2010.

[23] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[24] M. Grötschel, L. Lovász, and A. Schrijver. Relaxations of vertex packing. *J. Combin. Theory Ser. B*, 40(3):330–343, 1986.

[25] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2 of *Algorithms and Combinatorics: Study and Research Texts*. Springer-Verlag, Berlin, 1988.

[26] G. Gruber and F. Rendl. Computational experience with stable set relaxations. *SIAM J. Optim.*, 13(4):1014–1028, 2003.

[27] I. Hamzaoglu and J.H. Patel. Test set compaction algorithms for combinational circuits. In *Proceedings of the 1998 IEEE/ACM international conference on Computer-aided design*, ICCAD '98, pages 283–289, New York, NY, USA, 1998. ACM.

[28] C. Helmberg. A cutting plane algorithm for large scale semidefinite relaxations. In *The sharpest cut*, pages 233–256. SIAM, Philadelphia, PA, 2004.

[29] C. Helmberg and F. Rendl. Solving quadratic $(0, 1)$-problems by semidefinite programs and cutting planes. *Math. Programming*, 82(3, Ser. A):291–315, 1998.

[30] F. Jarre and F. Rendl. An augmented primal-dual method for linear conic programs. *SIAM J. Optim.*, 19(2):808–823, 2008.

[31] D. S. Johnson and M. A. Trick, editors. *Cliques, coloring, and satisfiability*. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 26. American Mathematical Society, Providence, RI, 1996.

[32] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972)*, pages 85–103. Plenum, New York, 1972.

[33] M. Laurent and F. Rendl. Integer programming and semidefinite programming. In K. Aardal, G. L. Nemhauser, and R. Weismantel, editors, *Discrete optimization*, volume 12 of *Handbooks in Operations Research and Management Science*, pages 393–514. Elsevier Science B.V., Amsterdam, 2005.

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

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

[36] R. J. McEliece, E. R. Rodemich, and H. C. Rumsey, Jr. The Lovász bound and some generalizations. *J. Combin. Inform. System Sci.*, 3(3):134–152, 1978.

[37] J. E. Mitchell. Computational experience with an interior point cutting plane algorithm. *SIAM J. Optim.*, 10(4):1212–1227, 2000.

[38] J. E. Mitchell. Cutting plane methods and subgradient methods. In M. Oskoorouchi, editor, *Tutorials in Operations Research*, chapter 2, pages 34–61. INFORMS, 2009.

[39] J. E. Mitchell and B. Borchers. Solving real-world linear ordering problems using a primal-dual interior point cutting plane method. *Ann. Oper. Res.*, 62:253–276, 1996.

[40] Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13 of *SIAM Studies in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1994.

[41] L. Palagi, V. Piccialli, F. Rendl, G. Rinaldi, and A. Wiegele. Computational approaches to max-cut. In M.F. Anjos and J.B. Lasserre, editors, *Handbook of Semidefinite, Conic and Polynomial Optimization: Theory, Algorithms, Software and Applications*, International Series in Operations Research and Management Science. Springer, New York, 2011.

[42] J. Peña, J. Vera, and L. F. Zuluaga. Computing the stability number of a graph via linear and semidefinite programming. *SIAM J. Optim.*, 18(1):87–105, 2007.

[43] F. Rendl, G. Rinaldi, and A. Wiegele. Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. *Math. Program.*, 121(2, Ser. A):307–335, 2010.

[44] N. Rhodes, P. Willett, A. Calvet, J.B. Dunbar, and C. Humblet. Clip: Similarity searching of 3d databases using clique detection. *Journal of Chemical Information and Computer Sciences*, 43(2):443–448, 2003.

[45] A. Schrijver. A comparison of the Delsarte and Lovász bounds. *IEEE Trans. Inform. Theory*, 25(4):425–429, 1979.

[46] H. D. Sherali and W. P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM J. Discrete Math.*, 3(3):411–430, 1990.

[47] N.J.A. Sloane. Unsolved problems in graph theory arising from the study of codes. *Graph Theory Notes of New York*, 18:11–20, 1989.

[48] M. Szegedy. A note on the Theta number of Lovász and the generalized Delsarte bound. In *35th Annual Symposium on Foundations of Computer Science, 20-22 November 1994, Santa Fe, New Mexico, USA*, pages 36–39, 1994.

[49] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18(suppl 1):S136–S144, 2002.

[50] M. J. Todd. A study of search directions in primal-dual interior-point methods for semidefinite programming. *Optim. Methods Softw.*, 11/12(1-4):1–46, 1999.

[51] R. H. Tütüncü, K.-Ch. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Math. Program.*, 95(2, Ser. B):189–217, 2003.

[52] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented lagrangian methods for semidefinite programming. Technical report, IEOR, Columbia University, 2009.

[53] X.-Y. Zhao, D. Sun, and K.-C. Toh. A Newton-CG augmented Lagrangian method for semidefinite programming. *SIAM J. Optim.*, 20(4):1737–1765, 2010.