McMaster University

Advanced Optimization Laboratory

Title:

Semidefinite cut-and-price approaches for the maxcut problem

Authors:

Kartik Krishnan, John Mitchell

AdvOL-Report No. 2004/5

March 2004, Hamilton, Ontario, Canada

Semidefinite cut-and-price approaches for the maxcut problem[1] [2]

Kartik Krishnan
Department of Computing & Software
McMaster University
Hamilton, Ontario L8S 4K1
Canada
kartik@caam.rice.edu
http://optlab.cas.mcmaster.ca/~kartik

John E. Mitchell
Mathematical Sciences
Rensselaer Polytechnic Institute
Troy, NY 12180
mitchj@rpi.edu
http://www.rpi.edu/~mitchj

May 11, 2004

Abstract

We investigate solution of the maximum cut problem using an SDP cut-and-price approach. The dual of the well-known SDP relaxation of maxcut is formulated as a semi-infinite linear programming problem, which is solved within a cutting plane algorithm. Cutting planes based on the polyhedral theory of the maxcut problem are then added to the primal problem in order to improve the SDP relaxation. We provide computational results, and compare these results with a standard SDP cutting plane scheme.

Keywords: Semidefinite programming, column generation, cutting plane methods.

[1]Research supported in part by NSF grant numbers CCR–9901822 and DMS–0317323
[2]Work done as part of the first authors Ph.D. dissertation at RPI

1 Introduction

Let $G = (V, E)$ denote an edge weighted undirected graph without loops or multiple edges. Let $V = \{1, \ldots, n\}$, $E \subset \{\{i, j\} : 1 \leq i < j \leq n\}$, and $w \in I\!R^{|E|}$, with $\{i, j\}$ the edge with endpoints i and j, and weight w_{ij}. We assume that $n = |V|$, and $m = |E|$. For $S \subseteq V$, the set of edges $\{i, j\} \in E$ with one endpoint in S and the other in $V \backslash S$ form the *cut* denoted by $\delta(S)$. We define the weight of the cut as $w(\delta(S)) = \sum\limits_{\{i,j\} \in \delta(S)} w_{ij}$. The *maximum cut problem*, denoted as (MC), is the problem of finding a cut for which the total weight is maximum. (MC) can be formulated as

$$\max\{w(\delta(S)) | S \subseteq V\} \qquad (MC) \tag{1}$$

The problem finds numerous applications in VLSI and circuit board design, network design, and finally Ising spin glass problems in statistical physics. Ising spin glass problems are sources of the maxcut problem where the edge weights can be negative. The maximum cut problem is one of the original NP complete problems (Karp [23]). However there are several classes of graphs for which the maximum cut problem can be solved in polynomial time; for instance planar graphs, graphs with no K_5 minor etc. A good survey on the maxcut problem appears in Poljak and Tuza [40]. We discuss various approaches to solving the maxcut problem. The aim is to try and solve the maxcut problem down to optimality, using convex tractable relaxations of the problem, cutting planes, heuristics, and branch and bound.

Linear programming formulations of the maxcut problem introduce a variable for each edge. The value of the variable indicates whether the two endpoints of the edge are on the same side of the cut or on opposite sides. These variables can be regarded as the entries of a matrix which has a row and a column for each vertex, and this matrix should be rank one (and therefore positive semidefinite) for the values of the variables to be consistent. Cutting planes developed for the linear programming formulation can be extended to the semidefinite formulation.

The paper is organized in a manner that we have all the details at hand to describe our cut-and-price approach, which appears in §4. This paper is organized as follows: §2 deals with a linear programming LP formulation of the maxcut problem, together with an interior point LP cutting plane approach to solving the maxcut problem and §3 deals with semidefinite formulations of the maxcut problem. In §4 we present our SDP cutting plane approach for the maxcut problem, with further details given in §5. We present some computational results in §6, and our conclusions in §7.

1.1 Notation

A quick word on notation : we represent vectors and matrices, by lower and upper case letters respectively. For instance d_{ij} denotes the jth entry of the vector d_i in a collection, whereas X_{ij} denotes the entry in the ith row and jth column of matrix X. The requirement that a matrix X be positive semidefinite (psd) is expressed $X \succeq 0$, We also use MATLAB like notation frequently in this paper. For instance $y(n+1 : m)$ will denote components $n+1$ to m of vector y, while $A(n+1 : m, 1 : p)$ will denote the sub-matrix obtained by considering rows $n+1$ to m, and the first p columns of matrix A. Finally $d_j.^2$ refers to a *row* vector obtained by squaring all the components of d_j.

2 Linear programming formulations of the maxcut problem

Consider a variable x_{ij} for each edge $\{i,j\}$ in the graph. Let x_{ij} assume the value 1 if edge $\{i,j\}$ is in the cut, and 0 otherwise. The maxcut problem can then be modelled as the following integer programming problem.

$$\begin{array}{lcl} \max & \sum_{i=1}^{n} \sum_{\{i,j\}\in E} \frac{1}{2} w_{ij} x_{ij} & \\ \text{subject to} & x & \text{is the incidence vector a cut} \end{array} \qquad (2)$$

Here n is the number of vertices in the graph, and the factor $\frac{1}{2}$ appears since each edge in the graph is considered twice.

Let CUT(G) denote the convex hull of the incidence vectors of cuts. Since maximizing a linear function over a set of points is equivalent to maximizing it over the convex hull of this set of points, we can also write (2) as the following linear program.

$$\begin{array}{lc} \max & c^T x \\ \text{s.t.} & x \in \mathrm{CUT}(G) \end{array} \qquad (3)$$

Here c and $x \in I\!R^m$, where m is the number of edges in the graph. We usually index c and x with the subscript e to relate them to the edges in the graph. However from time to time we drop the subscript e as appropriate. If we had an oracle capable of solving the separation problem with respect to this polytope in polynomial time, we could use this oracle in conjunction with the ellipsoid algorithm to solve the maxcut problem in polynomial time (Grötschel et al [15]). Unfortunately we do not have polynomial time separation oracles for some of the inequalities that describe the maxcut polytope, owing to the *NP complete* nature of the problem.

It is instructive to study facets of the maxcut polytope, since these provide good separation inequalities in a cutting plane approach. Numerous facets of the maxcut polytope have been studied, but the ones widely investigated in connection with a cutting plane approach are the following inequalities

$$\begin{array}{rcll} x_e & \geq & 0, & \forall e \in E, \\ x_e & \leq & 1, & \forall e \in E, \\ x(F) - x(C \backslash F) & \leq & |F| - 1 & \forall \text{ circuits } C \subseteq E \\ & & & \text{and all } F \subseteq C \text{ with } |F| \text{ odd} \end{array} \tag{4}$$

Here $x(S) = \sum_{\{i,j\} \in S} x_{ij}$ for any $S \subseteq E$. It is easy to see that these are valid inequalities for the maxcut polytope. The first and second set of inequalities define facets of CUT(G), if e is not contained in a triangle. The third set defines facets for each $F \subseteq C$, with $|F|$ odd, if C has no chord. We will call a graph *chordal* if any cycle C of length ≥ 4 has a chord. The latter inequalities can be derived from the observation that every cycle and cut intersect in an even number of edges. Moreover, the integral vectors satisfying (4) are vertices of CUT(G). So we indeed have an integer programming formulation of the maximum cut problem, namely

$$\begin{array}{llcll} \max & c^T x & & & \\ \text{s.t.} & x(F) - x(C \backslash F) & \leq & |F| - 1 & \forall \text{ circuits } C \subseteq E \\ & & & & \text{and all } F \subseteq C \text{ with } |F| \text{ odd} \\ & x & \geq & 0 & \\ & x & \leq & e & \\ & x & \in & \{0,1\}^m & \end{array} \tag{5}$$

Incidentally, these inequalities define a polytope known as the *odd cycle polytope.* Barahona and Mahjoub [4] show that we can drop the integrality restriction in (5), and solve the maxcut problem simply as an LP, if G does not have any subgraph contractible to K_5. This includes planar graphs, so we can indeed solve these maxcut instances in polynomial time.

Although there are an exponential number of linear constraints in (5), Barahona and Mahjoub [4] describe a polynomial time separation oracle for the inequalities (4) that involves solving n shortest path problems on an auxiliary graph with twice the number of nodes, and four times the number of edges. Thus it is possible to optimize a linear function over the odd cycle polytope in polynomial time. This exact algorithm, together with separation heuristics, has been used in cutting plane algorithms for the maxcut problem by Mitchell [35] and De Simone et al. [8, 9], among others. The initial relaxation used in these approaches is

$$\begin{array}{lrcccl} \max & & & c^T x & & \\ \text{s.t.} & 0 & \leq & x & \leq & e \end{array} \tag{6}$$

where $c, x \in I\!R^m$, where m is the number of edges in the graph.

3 Semidefinite formulations of the maxcut problem

In this section we consider semidefinite programming (SDP) formulations of the maxcut problem. In particular we consider Goemans and Williamson's celebrated 0.878 approximation algorithm for the maxcut problem based on this SDP formulation. We will later consider a semidefinite cutting plane approach for the maxcut problem.

The maxcut problem can be modelled as an integer program using cut vectors $x \in \{-1,1\}^n$ with $x_i = 1$ if $i \in S$, and $x_i = -1$ for $i \in V \setminus S$. Consider the following problem

$$\max_{x \in \{-1,1\}^n} \quad \sum_{i,j=1}^{n} w_{ij} \frac{1 - x_i x_j}{4} \tag{7}$$

A factor of $\frac{1}{2}$ accounts for the fact that each edge is considered twice. Moreover the expression $\frac{(1-x_i x_j)}{2}$ is 0 if $x_i = x_j$, i.e. if i and j are in the same set, and 1 if $x_i = -x_j$. Thus $\frac{(1-x_i x_j)}{2}$ yields the *incidence vector* of a cut associated with a cut vector x, evaluating to 1 if and only if edge $\{i,j\}$ is in the cut.

The Laplacian matrix of the graph G is $L := \text{Diag}(Ae) - A$. The following semidefinite programming relaxation of the maxcut problem can be derived (Helmberg [18], Krishnan & Terlaky [30] and Laurent & Rendl [34]):

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{s.t.} & \text{diag}(X) & = & e \\ & X & \succeq & 0 \end{array} \tag{8}$$

and its dual

$$\begin{array}{lrcl} \min & e^T y & & \\ \text{s.t.} & S & = & \text{Diag}(y) - \frac{L}{4} \\ & S & \succeq & 0 \end{array} \tag{9}$$

We will refer to the feasible region of (8) as the *elliptope*. In other words the elliptope is the set of symmetric positive semidefinite matrices, whose diagonal entries are all one. A point that must be emphasized is that the elliptope is no longer a polytope. Thus (8) is actually a non-polyhedral relaxation of the maxcut problem.

These semidefinite programs satisfy strong duality, since $X = I$ is strictly feasible in the primal, and we can generate a strictly feasible dual solution by assigning y an arbitrary positive value. In fact setting $y_i = 1 + \sum_{j=1}^{n} |\frac{L_{ij}}{4}|$ and $S = \mathrm{Diag}(y) - \frac{L}{4}$ should suffice.

In a semidefinite cutting plane scheme for the maxcut problem, we would begin with (8) as our starting SDP relaxation. It is interesting to relate (8) with (6) for the LP cutting plane approach. The constraints $\mathrm{diag}(X) = e$, and $X \succeq 0$ together imply that $-1 \leq X_{ij} \leq 1$. We now transform the problem into $\{0, 1\}$ variables using the transformation $y_{ij} = \frac{1-X_{ij}}{2}$. This gives $\frac{L}{4} \bullet X = \frac{1}{2} \sum_{i=1}^{n} \sum_{\{i,j\} \in E} w_{ij} y_{ij}$. Also we have $0 \leq y_{ij} \leq 1$, $\forall \{i, j\} \in E$. Thus it is easy to see that (8) is actually tighter than (6).

Goemans and Williamson [13] developed a 0.878 approximation algorithm for the maxcut problem when all the edge weights are nonnegative. Their algorithm uses the solution X to the SDP relaxation (8), followed by an ingenious randomized rounding procedure to generate the incidence vector x of the cut. This is a tremendous improvement over the LP relaxation, where the ratio of the maxcut value to that of the LP relaxation over the odd cycle polytope could be $\frac{1}{2}$ for various sparse random graphs (Poljak and Tuza [41]). This ratio improves to $\frac{3}{4}$ if one considers dense random graphs, where it can be further improved by considering k-gonal inequalities; in fact one has a PTAS for dense graphs (Avis & Umemoto [2]). On the negative side, Håstad [17] showed that it is NP-hard to approximate the maxcut problem to within a factor of 0.9412.

If we include negative edge weights and still have the Laplacian matrix $L \succeq 0$, then Nesterov [36] showed that the Goemans and Williamson rounding procedure gives an $\frac{2}{\pi}$ approximation algorithm for the maxcut problem.

The solution obtained by the randomized rounding procedure can be further improved using a Kernighan-Lin [25] local search heuristic.

We can further improve the relaxation (8) using the following linear inequalities.

1. **The chord-less odd cycle inequalities** :

   In the $\{-1, 1\}$ setting the odd cycle inequalities (4) are

$$\begin{array}{rcl} X(C\backslash F) - X(F) & \leq & |C| - 2 \\ & & \text{for each cycle } C, F \subset C, |F| \text{ odd} \end{array} \quad (10)$$

These include among others the triangle inequalities. Since $\mathrm{diag}(X) = e$, and $X \succeq 0$ imply $-1 \leq X_{ij} \leq 1$, the feasible region of the initial GW SDP relaxation (8) intersected with the odd cycle inequalities (10) is contained within the odd cycle polytope. We should thus get tighter upper bounds on the maxcut value.

2. **The hypermetric inequalities** :
   These are inequalities of the form (11)

$$bb^T \bullet X \geq 1 \qquad \text{where } \; b \in \mathcal{Z}^n, \qquad \sum_{i=1}^{n} b_i \;\text{ odd} \qquad \text{and} \min\{(b^T x)^2 : x \in \{-1,1\}^n\} = 1. \tag{11}$$

   For instance, the triangle inequality $X_{ij} + X_{ik} + X_{jk} \geq -1$ can be written as a hypermetric inequality by taking b to be the incidence vector of the triangle (i, j, k). On the other hand the other inequality $X_{ij} - X_{ik} - X_{jk} \geq -1$ can be written in a similar way, except that $b_k = -1$. Although there are a countably infinite number of them, these inequalities also form a polytope known as the *hypermetric polytope* (Deza et al [10] and Laurent [33]). Helmberg and Rendl [20] describe simple heuristics to detect violated hypermetric inequalities.

One can also employ lift and project cutting planes as discussed in Iyengar and Cezik [21]. This is an extension of the original lift and project approaches of Balas et al [3] for mixed integer SDPs, some of whose variables are constrained to be $\{\pm 1\}$.

Interestingly although the additional inequalities improve the SDP relaxation, they do not necessarily give rise to better approximation algorithms. On the negative side Karloff [22] exhibited a set of graphs for which the optimal solution to (8) satisfies all the triangle inequalities as well, so after the GW rounding procedure we are still left with a 0.878 approximation algorithm.

More recently Anjos and Wolkowicz [1] presented a strengthened semidefinite relaxation of the maxcut problem. Their SDP relaxation is tighter than the relaxation (8) together with all the triangle inequalities. To arrive at their relaxation they add certain redundant constraints of the maxcut problem to (8), and consider the Lagrangian dual of the Lagrangian dual of this problem. On the negative side though this strengthened SDP relaxation is fairly expensive to solve. Also, see Lasserre [31, 32] for a hierarchy of SDP relaxations for the maxcut problem.

4 An SDP method for the maxcut problem

This section deals with the major contribution of this paper, which is a SDP cut and price method for solving the maxcut problem.

The main tools for solving SDPs are interior point methods. Interior point methods have several advantages, chief among these their ability to solve and SDP to any degree of precision in a polynomial number of arithmetic operations. Problems with more than 300 constraints are considered quite hard. Besides, interior point methods do not exploit any structure in the problem. Recently developed large scale approaches (such as the one we use in the paper) can be used to try to overcome this drawback. For example, recently Helmberg [19] has incorporated the spectral bundle method in a cutting plane approach to solving the maxcut problem.

In this paper, we will solve each SDP relaxation in turn as an LP in an inner cutting plane scheme. Our method is based on the semi-infinite formulation of the dual SDP relaxation (9), namely

$$\begin{array}{lrcl} \min & e^T y & & \\ \text{s.t.} & \mathcal{A}^T y + S & = & C \\ & d^T S d & \geq & 0 \quad \forall d \in B \end{array} \tag{12}$$

where the linear constraints are represented by $\mathcal{A}^T y + S = C$ and where B is a compact set, typically $\{d : ||d||_2 \leq 1\}$ or $\{d : ||d||_\infty \leq 1\}$. We described a cutting plane approach to solving this relaxation in [26, 27, 28, 29]. This approach sets up a relaxation to the dual,

$$\begin{array}{lrcl} \max & b^T y & & \\ \text{subject to} & d_i d_i^T \bullet \mathcal{A}^T y & \leq & d_i d_i^T \bullet C \quad \text{for } i = 1, \ldots, m. \end{array} \tag{LDR}$$

with a corresponding constrained version of the primal problem,

$$\begin{array}{lrcl} \min & C \bullet (\sum_{i=1}^{m} x_i d_i d_i^T) & & \\ \text{subject to} & \mathcal{A}(\sum_{i=1}^{m} x_i d_i d_i^T) & = & b \\ & x & \geq & 0. \end{array} \tag{LPR}$$

We have chosen to work with a relaxation of the dual (9) since it has fewer variables. This cutting plane approach will be used as a subroutine to solve the SDP relaxations which arise while solving the maxcut problem. The vectors d are generated during the course of the algorithm. We used two methods to generate them. First, they can be chosen explicitly to cut off a current indefinite matrix S. Secondly, any ± 1 incidence vector of a good solution to the maxcut problem can be used as a vector d. We discuss these options in greater detail later.

Our approach resembles a cut and price algorithm (Wolsey [45]) to solving the maxcut problem. Such an approach has been applied to solving large set-covering problems that arise in practice such as airline crew scheduling problems. We can motivate this as follows: In our primal LP formulation (LPR) for the maxcut problem, the matrix X is a convex combination of various rank one matrices; if all of these matrices were valid integer solutions, i.e., were of the form dd^T, where d was an incidence vector of a cut, then we would have exactly a cut and price formulation since we are pricing only the candidate optimal solutions. However, to only generate columns d which were incidence vectors of cuts would require solving the maxcut problem to optimality when doing the pricing. Therefore, we also allow the generation of other columns d in order to drive the process into the correct area of the problem space; in Step 3 of the algorithm (see below) we also run the Goemans Williamson rounding procedure on the primal matrix X to generate good incidence cut vectors. Finally, in order to solve the maxcut problem to optimality, we need a primal matrix X feasible in (8), that is also rank one. To do this, we add cutting planes that are facets of the maxcut polytope.

First, we must mention that Gruber and Rendl [16] have a similar approach, where they solve a semi-infinite formulation of the primal SDP relaxation (8). They do not motivate it in this way; their aim is rather to strengthen the LP cutting plane approach by adding cutting planes valid for the SDP approach. An advantage of their approach is that they deal exclusively with the primal problem, and not oscillate between the primal and dual as in our approach. If the graph is chordal then it suffices to force every submatrix of X corresponding to a maximal clique to be positive semidefinite, so it is not necessary to consider the other entries in the primal matrix X explicitly. Grone et al. [14] showed that values for the missing entries can be chosen to make X positive semidefinite provided all these submatrices are positive semidefinite.

The approach is advantageous only if the underlying graph is chordal and sparse (see Gruber & Rendl [16] for more details). If the graph is not chordal, then one needs to construct a chordal extension of it, by adding redundant edges of weight 0. This could lead to a fairly dense graph.

We now describe our entire algorithm in a nutshell. Each of these steps will be elaborated in more detail in §5.

1. **Initialize**

2. **Approximately solve the dual maxcut SDP as an LP** using the interior point cutting plane scheme described in [27]. Construct the primal matrix X.

3. **Generate good integer cut vectors** by running the Goemans-Williamson and Kernighan-Lin heuristics on the primal matrix X. Add these integer vectors into the LP relaxation, and resolve to find the new X.

4. **Check for termination :** If the difference between the upper bound, and the value of the best cut is small, STOP with optimality.

5. **Separation oracle :** Find violated odd cycle inequalities. Bucket sort the resulting violated inequalities, and add a subset of constraints to the relaxation. This changes the dual slack matrix S.

6. **Update the LP relaxation** by choosing the most important constraints in the LP relaxation including the best integer cut vector, constraints based on the spectral factorization of X, and the initial box constraints. Solve the LP relaxation.

7. **Loop :** return to step 2

Assuming that we do not resort to branch and bound, at the termination of the cutting plane scheme we are solving the following problem (13)

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{s.t.} & \operatorname{diag}(X) & = & e, \\ & \displaystyle\sum_{ij \in C \setminus F} X_{ij} - \sum_{ij \in F} X_{ij} & \leq & |C| - 2, \\ & \text{C a cycle,} & F \subseteq C, & |F| \text{ odd,} \\ & X & \succeq & 0 \end{array} \tag{13}$$

An important point needs to be emphasized here: We use a primal-dual IPM to solve the LP relaxations approximately; the query points at which we call the SDP separation oracle are closer to the analytic center of the feasible region of the current LP relaxation, and this results in better cuts. One could also use the simplex method to solve the LP relaxations (this leads to the classical Kelley LP cutting plane scheme [24] for convex optimization); here the query points are extreme points of the LP feasible region. It is known that the Kelley cutting plane method suffers from tailing effects, and we observed in Krishnan [26] (see also Elhedhli & Goffin [12]) that a simplex implementation performed very badly.

However, our choice of a primal-dual IPM for an LP relaxation presents its own share of difficulties: Since we are using an interior point solver for the LP relaxations, it is imperative that we use warm start information so that re-optimization can be carried out in reasonable time, after the addition of cutting planes. This is a difficulty with IPMs and is in sharp contrast to the Kelley cutting plane method, where one could re-optimize using the dual-simplex method. Since the oracle returned *deep* cutting planes, the current iterate $X^{current}$ is infeasible after the addition of cutting planes. Therefore we have to construct a new feasible point X^{start} for restarting the method.

In the following we assume that we add m odd cycle inequalities (10) to the initial relaxation (8). We express these linear constraints as $\mathcal{A}X + s = b$, where $b \in I\!R^m$, and $s \in I\!R^m_+$ is a vector of slack variables. Then the new semidefinite relaxation reads as

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{s.t.} & \operatorname{diag}(X) & = & e \\ & \mathcal{A}(X) + s & = & b \\ & X & \succeq & 0 \\ & s & \geq & 0 \end{array} \tag{14}$$

with dual

$$\begin{array}{lrcl} \min & \sum_{i=1}^{n} y_i + \sum_{i=1}^{m} b_i y_{n+i} & & \\ \text{s.t.} & S & = & \operatorname{Diag}(y(1:n)) + \mathcal{A}^T y(n+1:n+m) - \frac{L}{4} \\ & S & \succeq & 0 \\ & y(n+1:n+m) & \geq & 0 \end{array} \tag{15}$$

We first discuss a strategies for restarting the primal (14) with a strictly feasible primal iterate. It is easy to see that the identity matrix I is the analytic center of the feasible region of (8). The idea then is to backtrack towards I along the straight line between the last iterate X^{prev} and I. Thus we choose $X^{start} = (\lambda X^{prev} + (1-\lambda)I)$ for some $\lambda \in [0,1)$. The matrix I is some sort of a center for the maxcut polytope. To see this consider rank one matrices $X = xx^T$ where x is a ± 1 vector; averaging over all such matrices gives the matrix I. So it is guaranteed that the procedure will terminate with a strictly feasible primal iterate. A similar idea was employed in Helmberg and Rendl [20].

Restarting the dual (15) is relatively straightforward, since we can get into the dual SDP cone $S \succeq 0$, by assigning arbitrarily large values to the first n components of y. In our approach, we assign all the new components of y the value 1, i.e. $y(n+1 : n+m) = e$ (note that the new components of y are required to be positive, whereas the first n components are unrestricted in sign). Let $y^{prev} \in I\!R^n$ be the previous dual solution. Compute $\bar{S} = \text{Diag}(y^{prev}) + \mathcal{A}^T e - \frac{L}{4}$, and estimate the magnitude of its most negative eigenvalue λ. $S^{start} = \bar{S} + \alpha\lambda I$, where $\alpha > 1$ is positive definite, and

$$y^{start} = \left[\begin{array}{c} y^{prev} + \lambda e(1:n) \\ e(n+1:n+m) \end{array} \right]$$

in the usual MATLAB notation.

Since the separation oracle generates more violated inequalities than we are willing to include in our SDP relaxations, it is important that we select the promising ones from the vast set of violated inequalities in Step 5 of the algorithm. Our metric is the *amount* of violation. With regard to dropping constraints in Step 6, we drop those constraints whose dual variables are small. Another criterion is to use the ratio of the dual variable to the slack in the constraint, and drop constraints for which the ratio is small.

5 Details of the algorithm

In this section, we describe the steps of our algorithm in more detail.

We note first that the initial dual LP relaxation is

$$\begin{array}{llll} \min & \sum_{i=1}^{n} y_i & (LDR_0) & \\ \text{s.t.} & y_i & \geq & \frac{L_{ii}}{4}, \quad i = 1, \ldots, n \end{array}$$

An optimal solution is $(x^0, y^0, z^0) = (e, \text{diag}(\frac{L}{4}), 0) \in I\!R^n \times I\!R^n \times I\!R^n$. This corresponds to initial matrices $X^0 = I$, and $S^0 = \frac{L}{4} - \text{Diag}(\frac{L}{4})$. We refer to the constraints $y \geq \text{diag}(\frac{L}{4})$ as *box constraints.*

The current dual maxcut SDP relaxation does not need to be solved to optimality using our LP cutting plane approach in step 2. We use a dynamically altered tolerance on the required accuracy of the solution (for specifics, see §6). If we do not solve the SDP accurately enough, then the objective value of the LP relaxation is not guaranteed to be an upper bound on the maxcut value. In fact we typically have the following inequalities on the objective values of the various relaxations. The SDP here is (9), the dual relaxation for the maxcut problem. Typically, we find that

$$\begin{array}{rcl} \text{Cheap LP relaxation to SDP} & \leq & \text{Optimal Maxcut value} \quad \leq \\ \text{Good LP relaxation to SDP} & \leq & \text{SDP objective value} \end{array}$$

The accuracy to which to which the SDP is solved as an LP affects the performance of the GW randomized rounding procedure. Using the same analysis as Goemans and Williamson [13], we can show that the value of the cut generated by the procedure is at least 0.878 times the value of the current LP relaxation, if all the edge weights are nonnegative. However, as mentioned in point (1) we cannot always guarantee that the value of our LP relaxation is an upper bound on the maxcut value. Thus we do not have a performance guarantee out here. Nevertheless, we can say in a weak sense that the more accurately we solve the SDP as an LP in the cutting plane scheme, the more likely that a better cut is produced.

It should be noted that we add cutting planes in both the primal as well as the dual. We add cutting planes (10) based on the polyhedral structure of the maxcut polytope in the primal, while the cutting planes added in the dual are sometimes low dimensional faces of the dual SDP cone (in y space), and not always facets (see also [29]).

As regards the primal, we cannot guarantee that our primal LP feasible region always contains the entire maxcut polytope; in most cases it does not. The primal LP feasible region increases, when we solve the dual SDP as an LP, (i.e. add cutting planes in the dual), and decreases, when we add cutting planes in the primal.

When we are adding cutting planes in the dual, we update the primal matrix $X = \sum_{i=1}^{m} x_i d_i d_i^T$. On the other hand, when we add cutting planes in the primal we are updating the dual slack matrix $S = \text{Diag}(y) + \sum_{i=1}^{p} y_{n+i} A_i - \frac{L}{4}$. At each point in the algorithm, we have an LP relaxation of the dual maxcut SDP of the form

$$
\begin{array}{lc}
\min & \sum_{i=1}^{n} y_i + \sum_{i=1}^{p} (|C_i| - 2) y_{n+i} \\
\text{s.t.} & \begin{bmatrix} I & 0 & \dots & 0 \\ d_{n+1}.^2 & d_{n+1}^T A_1 d_{n+1} & \dots & d_{n+1}^T A_p d_{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ d_m.^2 & d_m^T A_1 d_m & \dots & d_m^T A_p d_m \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ y_{n+1} \\ \vdots \\ y_{n+p} \end{bmatrix} \\
& \geq \begin{bmatrix} \text{diag}(\frac{L}{4}) \\ d_{n+1}^T \frac{L}{4} d_{n+1} \\ \vdots \\ d_m^T \frac{L}{4} d_m \end{bmatrix} \qquad (LDRC) \\
& y_{n+i} \ \geq \ 0, \quad i = 1, \dots, p
\end{array}
$$

and the primal relaxation $(LPRC)$ is

$$\begin{array}{llcll} \max & \frac{L}{4} \bullet (\sum_{j=1}^{m} x_j d_j d_j^T) & & & \\ \text{s.t.} & \operatorname{diag}(\sum_{j=1}^{m} x_j d_j d_j^T) & = & e & (LPRC) \\ & A_i \bullet (\sum_{j=1}^{m} x_j d_j d_j^T) & \leq & (|C_i| - 2), & i = 1, \ldots, p \\ & x_j & \geq & 0, & j = 1, \ldots, m \end{array}$$

An odd cycle inequality (10) is written as $A \bullet X \leq (|C| - 2)$, where A is a symmetric matrix, and $|C|$ is the cardinality of the cycle. Also x_j is the primal variable corresponding to the jth dual constraint. The current solution is $(\hat{x}, \hat{y})$, where $\hat{x} \in I\!R^m$ and $\hat{y} \in I\!R^{n+p}$.

In step 2 of the algorithm, we improve the LP relaxation of the SDP using a cutting plane LP approach as detailed in [26, 27, 28]. The LP relaxations are solved approximately using a primal-dual LP interior point method. Cutting planes are added to the dual $(LDRC)$, which try and force the dual slack matrix $S = \operatorname{Diag}(y(1:n)) + \sum_{i=1}^{p} A_i y_{n+i} - \frac{L}{4}$ to be psd. Appropriate cuts can be found by looking for eigenvectors of the current dual slack matrix with negative eigenvalues.

The relaxation $(LDRC)$ can be improved further in step 3 by adding good cut vectors. Compute $\hat{X} = \sum_{j=1}^{m^k} {x_j}^k d_j d_j^T$. This matrix is feasible in the primal SDP. In other words $\hat{X}$ is psd with diagonal entries one, and strictly satisfies the current odd cycle inequalities. Run the Goemans and Williamson randomized rounding procedure on $\hat{X}$, and generate a number of good cut vectors. As we remarked earlier, the strength of these cuts depends on how accurately we solve the current dual SDP as an LP in the cutting plane scheme. We then improve the cuts using the Kernighan and Lin (KL) heuristic. If the objective value of these cuts is better than the current LP relaxation, then add the 5 best cuts as cutting planes in the dual $(LDRC)$.

The value of the best cut found so far gives a lower bound on the optimal value of the maxcut problem. The value of any dual feasible vector y for which the corresponding slack matrix S is positive semidefinite gives an upper bound on this optimal value. Such a vector can be constructed from any vector $\hat{y}$ feasible in $(LDRC)$ by increasing the first n components of $\hat{y}$ appropriately. We construct an upper bound in this way using any dual feasible $\hat{y}$ for which the matrix S is close to being positive semidefinite. The best such upper bound is stored, and when the upper bound and lower bound are close enough the algorithm is terminated in step 4.

Cutting planes for $(LPRC)$ are found in step 5. These odd cycle inequalities are violated by the current solution. The matrix $\hat{X}$ is fed to the Barahona-Mahjoub separation oracle which returns violated odd cycle inequalities (10), which are facets of the maxcut polytope. These odd cycle inequalities are bucket sorted by violation, and the most violated ones are added to $(LPRC)$.

In order to keep the size of the relaxations manageable, the constraint matrix is modified in step 6. The rule is to drop those constraints whose primal variables x_j are small. Since trace$(\hat{X}) = n$, we have $\sum_{j=1}^{m} \hat{x}_j d_j^T d_j = n$. Constraints with a small value of $\hat{x}_j d_j^T d_j$ in relation to n can be safely dropped without adversely affecting the LP relaxation. In most instances, the value of the LP relaxation is determined entirely by the value of the best cut. In this case, the LP is degenerate, and all the constraints with zero $\hat{x}_j$ values can be dropped. We sort constraints $n+1, \ldots, n+p$ based on increasing values of their dual variables $\hat{x}_j$, $i = n+1, \ldots, n+p$. We retain the q (an upper bound is given below) best constraints ensuring that the best integer vector is among these constraints. The modified relaxation $(LDRC)$ has the following constraints :

- The box constraints, i.e. the first n constraints in the earlier relaxation $(LDRC)$.
- The q best constraints from the earlier relaxation $(LDRC)$.
- The eigenvectors v_i, $i = 1, \ldots, r$ corresponding to nonzero eigenvalues of $\hat{X}$ (r here is the rank of $\hat{X}$), found using a spectral factorization $\hat{X} = V \Lambda V^T$. In a sense, the eigenvectors corresponding to the nonzero eigenvalues *aggregate* the important information contained in the constraints added so far.

The number of constraints in (2) and (3) is no more than $\sqrt{2n}$, since this is an upper bound for the rank of at least one extreme point optimal solution for (13) with p odd cycle inequalities (see Pataki [38]). When the algorithm loops back to step 2 after adding odd cycle inequalities, warm start information should be exploited in solving the new $(LDRC)$ and $(LPRC)$. Since the odd cycle inequalities returned in step 5 are designed to cut off the primal matrix $\hat{X}$, the old solution $(\hat{x}, \hat{y})$ is not strictly feasible in the modified $(LDRC)$ and $(LPRC)$. We consider the primal LP relaxation first.

A new primal matrix $\bar{X}$ is constructed as

$$\bar{X} \;:=\; \tfrac{1}{2}\{\sum_{j=1}^{q} \hat{x}_j d_{n+j} d_{n+j}^T + \sum_{i=1}^{r} \lambda_i v_i v_i^T\}$$

This matrix is positive semidefinite, but it may not satisfy the linear constraints of (8). The old feasible X however differs from $\bar{X}$ by a positive sum of outer products of d_j vectors, and this difference has a non-negative diagonal.

Thus, one can add positive linear combinations of $e_i e_i^T$, $i = 1, \ldots, n$ to $\bar{X}$ until

$$\tilde{X} := \sum_{i=1}^{n} \alpha_i e_i e_i^T + \bar{X}$$

satisfies the equality constraints, i.e. $\text{diag}(\tilde{X}) = e$. To strictly satisfy the odd cycle inequalities $A_i \bullet X \leq (|C_i| - 2)$, $i = 1, \ldots, p$, we backtrack towards I along the straight line between $\tilde{X}$ and I. Since I is the analytic center of the feasible region $\{X : \text{diag}(X) = e, X \succeq 0\}$, which contains the max cut polytope, a line search in this direction should enable us to strictly satisfy the violated odd cycle inequalities. The final primal matrix X enables us to find a strictly primal feasible solution $\bar{x}$ in $(LPRC)$.

As regards the dual, perturb $\hat{S}$ until it is psd as follows. Set the new dual variables corresponding to the newly added odd cycle inequalities to 1. Calculate the dual slack matrix $\hat{S}$ with the old values of $\hat{y}$ together with these new components, so

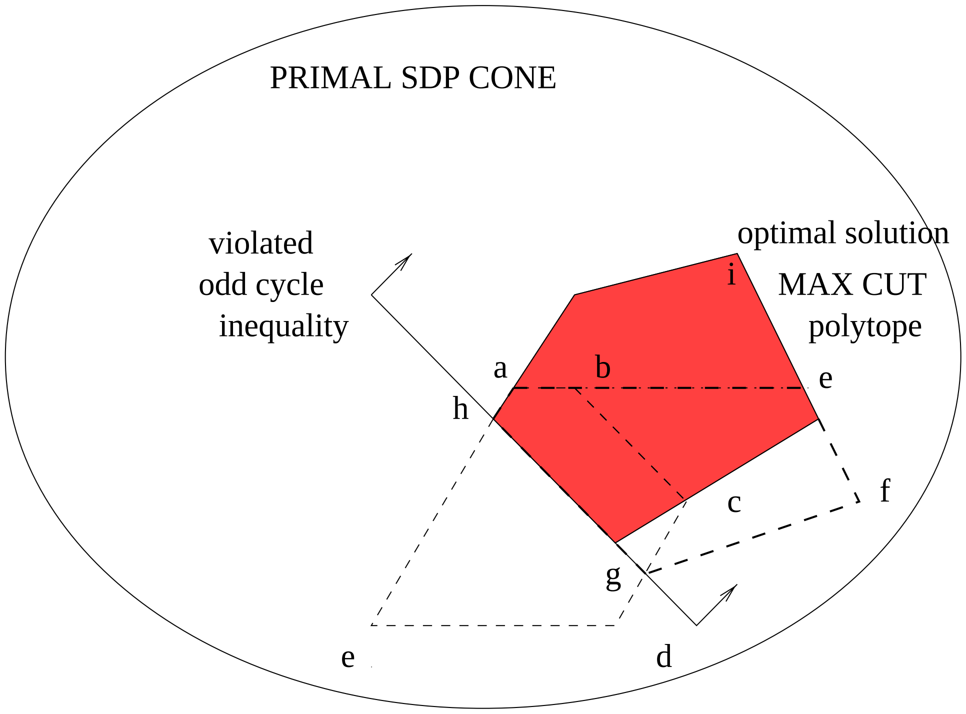
$$\hat{S} = \text{Diag}(y(1:n)) + \sum_{i=1}^{p} A_i y_{n+i} - \frac{L}{4}$$

where p has been updated to reflect the added primal constraints. Let λ be the magnitude of the most negative eigenvalue of $\hat{S}$. Compute

$$\bar{y}_i \quad = \quad \bar{y}_i + \lambda \quad \text{for } i = 1, \ldots, n$$

and leave the other dual variables at their previous values. This gives a strictly feasible dual solution, since the $(n+i)$th dual constraint is equivalent to requiring $d_{n+i}^T S d_{n+i}$ be nonnegative. The desired strictly feasible starting point is $(\bar{x}, \bar{y})$.

The overall SDP cutting plane method for maxcut is best summarized in Figure 1. The figure illustrates how the primal LP feasible region evolves during the course of the algorithm. Since we are solving a *constrained* version of the primal SDP relaxation, the LP feasible region is always within the corresponding primal SDP cone. The polytope shaded in red is the maxcut polytope. Suppose we are in the kth iteration, and the LP feasible region is the polytope *abcdea*. Since we have solved the current dual SDP only approximately as an LP, the polytope *abcdea* does not contain the entire maxcut polytope. The Barahona separation oracle returns an odd cycle inequality, that is violated by the current LP solution x^{k+1}. The new LP feasible region is now the polytope *abcgha*. We then try and grow this LP feasible region, by adding cuts in the dual to try and force the new dual slack matrix S^{k+1} to be positive semidefinite. At the end of the process, let $abefgh$ be the new LP feasible region. Although, $abefgha$ does not contain the maxcut polytope either, it is a better approximation than the earlier polytope $abcdea$. Proceeding in this manner, we eventually hit the optimal vertex i.



abcde : Initial LP relaxation
abcgh : LP region after adding violated odd cycles
abefgh : after the interior point sdp cutting plane scheme
Max Cut Polytope : Shaded in red
i : Optimal integer cut vector

Figure 1: The cutting plane SDP method for the maxcut problem

6 Computational results

In this section we report our preliminary computational experience. All results were obtained on a *Sun Ultra 5.6*, *440 MHz* machine with *128 MB* of memory. We carried out our tests on a number of maxcut problems from *SDPLIB* [6], the *DIMACS Implementation Challenge* [39], and some randomly generated Ising spin glass problems, i.e. maxcut problems with ± 1 edge weights from Mitchell [35]. We compare our results with a traditional SDP cutting plane scheme that uses SDPT3 (Toh et al [43]) to solve the SDP relaxations.

We implemented our entire approach within the *MATLAB* environment. The LP solver was Zhang's *LIPSOL* [46]; we suitably modified the algorithm to restart the LP relaxations with a strictly feasible starting point. The SDP separation oracle used in step 2 uses the 2 norm, and employs MATLABs Lanczos solver. We also experimented with the ∞ norm using LOQO by Vanderbei [44] for this purpose. The ∞ norm oracle however took too much time on the larger problems, and we have chosen not to mention these results here. Finally, we used the Barahona-Mahjoub separation oracle to generate valid cutting planes in the primal; our implementation was based on the *CPLEX* [7] network optimization solver to solve the shortest path problem as an LP. In practice, one uses heuristics (see Berry and Goldberg [5]) to generate the violated cuts, but we used the exact separation oracle in our experiments.

The computational results are summarized in Table 1. The columns in the table represent

Name:	Problem name
n:	Number of nodes
m:	Number of edges
Maxcut:	Objective value of the best incidence vector
SDP1:	Objective value over the elliptope (8)
SDP2:	Objective value over the elliptope and odd cycles (13)
UB:	The upper bound returned by the cutting plane scheme

We ran 10 cutting plane iterations in all. In our LP subroutine for the SDP, we add 5 cutting planes in each iteration, our starting tolerance is 1, and this is lowered by 0.9 in each iteration. Initially, we solve our SDP relaxations very cheaply, i.e. perform only 5 LP cutting plane iterations, and we increase this number if we are unable to improve the best incidence cut vector. Also, we add the $\frac{n}{4}$ most violated odd cycle inequalities returned by the Barahona-Mahjoub approach in each iteration.

[2]All results for 10 iterations of the SDP scheme

[1]SDPLIB

[2]DIMACS

[3]Random Ising Spin glass problems (Mitchell)

[4]Best integer solution from the SDP scheme (Optimal solution)

[5]Over the elliptope

[6]Over the elliptope and odd cycles

[7]Upper bounds from our scheme

[8]Best results obtained using an interior point approach

Name	n	m	Maxcut(Opt) [4]	SDP1 [5]	SDP2 [6]	UB [7]
gpp100 [1]	100	269	210	221.69	211.27	212.56
gpp1241 [1]	124	149	137	141.91	137	137.45
gpp1242 [1]	124	318	256	269.64	257.66	258.60
gpp1243 [1]	124	620	446	467.68	458.35	458.90
gpp1244 [1]	124	1271	834	864.26	848.92	850.08
gpp2501 [1]	250	331	305	317.23	305	307.83
gpp2502 [1]	250	612	502	531.78	511.12	520.19
gpp5001 [1]	500	625	574	598.11	576.64 [8]	584.91
toruspm-8-50 [2]	512	1536	456	527.81	464.70 [8]	472.28
torusg3-8 [2]	512	1536	412.75 (416.84)	457.36	417.69 [8]	428.18
ising10 [3]	100	200	70	79.22	70	70.70
ising20_1 [3]	400	800	282	314.25	282	288.04
ising20_2 [3]	400	800	268	305.63	268	271.82
ising30 [3]	900	1800	630 (632)	709.00	632	

Table 1: Test Results on Maxcut

We must mention that our technique of estimating upper bounds does not performs very well. Therefore to obtain the upper bounds mentioned above, we solve the final primal SDP relaxation with the odd cycle inequalities to a tolerance of 1e-8 using SDPT3 (Toh et al [43]). Interestingly, we were able to obtain the optimal integer solution in most cases (except problems torusg3-8 and ising30). The former problem is not easy, since the edge weights and hence the Laplacian matrix are not integer, whereas the latter problem is currently as big as we can possibly handle. For most of the problems we do not have a proof of optimality, since the SDP relaxation over the elliptope and odd cycles is not enough, and there is yet some gap involved. For the Ising spin glass problems optimizing over the intersection of the elliptope and the odd cycle polytope is sufficient to give the optimal solution.

To give an estimate of the times involved we performed the following experiment:

1. We ran our code against a standard SDP cutting plane scheme that employed $SDPT3$ [43] version 2.3 to solve the SDP relaxations. We chose $SDPT3$ over any other solver since it offered the flexibility for a warm start procedure.

2. We chose eight problems from SDPLIB.

3. We ran 10 cutting plane iterations in all.

4. In our approach, we solve the SDP relaxations very cheaply (5 LP cutting plane iterations), and in every 5th iteration we solve this SDP relaxation more accurately (25 iterations). In $SDPT3$ we solve the SDP relaxations to a moderate tolerance $1e-2$, and in every 5th iteration we solve the SDP more accurately $(1e-6)$.

5. The other parameters for our LP cutting plane approach for the SDP are the same as mentioned above.

6. We add the $\frac{n}{4}$ most violated odd cycle inequalities returned by the Barahona-Mahjoub approach in each iteration.

The results can be found in table 2.

Problem	SDPT3			Our approach		
Name	UB	LB	Time	UB	LB	Time
gpp100	212.26	210	321	216.86	210	485
gpp1241	137	137	197 [1]	138.33	137	488
gpp1242	258.4	256	423	263.73	256	620
gpp1243	458.86	446	409	461.92	446	832
gpp1244	851.01	834	468	857.35	834	902
gpp2501	305	305	2327	309.95	304	1073
gpp2502	512.06	502	1739	521.06	502	1687
gpp5001	-	-	-	583.65	574	2893

Table 2: Comparing two SDP cutting plane schemes for maxcut

Table 2 shows how our cutting plane approach scales with the problem size, and the increase in runtime with problem size compares quite well with a traditional interior point SDP cutting plane approach. The interior point approach solved the problem gpp1241 to optimality in just 5 iterations. On the other hand on problem gpp5001, the interior point method was unable to complete the stipulated 10 iterations.

7 Conclusions

We have presented a cutting plane technique for approximating the maxcut problem. This linear approach allows one to potentially solve large scale problems. Here are

[1]Converged in 5 iterations

some conclusions based on our preliminary computational results

1. It is worth reiterating that our approach resembles a cut and price approach for the maxcut problem; it is also entirely a polyhedral approach. The SDP relaxations themselves are solved within a cutting plane scheme, so our scheme is really a cutting plane approach within another cutting plane approach.

2. The cutting plane approach is able to pick the optimal maxcut incidence vector quickly, but the certificate of optimality takes time.

3. Our technique for computing upper bounds performs poorly. We are currently working on a number of techniques to improve this upper bound using information from the eigenvector based on the most negative eigenvalue. Nevertheless we believe that if we are able to handle this difficulty, then our approach can be effective on large scale maxcut problems.

4. The main computational task in our approach is the time taken by the SDP separation oracle. Another difficulty is that the oracle returns cutting planes that lead to dense LP relaxations. One way is to use a variant of the *matrix completion* idea in Gruber and Rendl [16], and examine the positive semidefiniteness of S_K for any clique $K \subseteq V$ (here S_K is a sub-matrix of S corresponding to only those entries in the clique). This should speed up the separation oracle, which will also return sparser constraints, since any cutting plane for S_k can be lifted to $\mathcal{S}^n$ by assigning zero entries to all the components of the cutting plane not in K.

5. We are also trying to incorporate this approach in a branch and cut approach to solving the maxcut problem, i.e. we resort to branching when we are not able to improve our best cut values obtained so far, or our upper bounds. Again, the major issue is how to use warm start information from the parent node, to re-optimize the child nodes quickly.

6. Helmberg [19] shows that better SDP relaxations can be obtained by enlarging the number of cycles in the graph, i.e. adding edges of weights 0 between non-adjacent nodes. This gives rise to a complete graph, and the only odd cycles now are triangle inequalities, which can be inspected by complete enumeration. Again, this in sharp contrast to the LP case, where adding these redundant edges does not improve the relaxation. We hope to utilize this feature in our cutting plane approach, in the near future.

7. We hope to extend this approach to other combinatorial optimization problems such as min bisection, k way equipartition, and max stable set problems based on a Lovasz theta SDP formulation.

References

[1] M. F. Anjos and H. Wolkowicz. *Strengthened semidefinite relaxations via a second lifting for the max-cut problem.* Discrete Applied Mathematics, 119:79–106, 2002.

[2] D. Avis and J. Umemoto. *Stronger linear programming relaxations of max-cut.* Mathematical Programming, 97:451–469, 2003.

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

[4] F. Barahona and A. R. Mahjoub. *On the cut polytope.* Mathematical Programming, 36:157–173, 1986.

[5] J. Berry and M. Goldberg. *Path optimization for graph partitioning problems.* Discrete Applied Mathematics, 90:27–50, 1999.

[6] B. Borchers. *SDPLIB 1.2, a library of semidefinite programming test problems.* Optimization Methods and Software, 11:683–690, 1999.

[7] CPLEX Optimization Inc. CPLEX Linear Optimizer and Mixed Integer Optimizer. Suite 279, 930 Tahoe Blvd. Bldg 802, Incline Village, NV 89541.

[8] C. De Simone, M. Diehl, M. Jünger, P. Mutzel, G. Reinelt, and G. Rinaldi. *Exact ground states of Ising spin glasses: New experimental results with a branch and cut algorithm.* Journal of Statistical Physics, 80:487–496, 1995.

[9] C. De Simone, M. Diehl, M. Jünger, P. Mutzel, G. Reinelt, and G. Rinaldi. *Exact ground states of two-dimensional $\pm J$ Ising spin glasses.* Journal of Statistical Physics, 84:1363–1371, 1996.

[10] M. M. Deza, V. P. Grishukhin, and M. Laurent. *The hypermetric cone is polyhedral.* Combinatorica, 13:397-411, 1993.

[11] M. M. Deza and M. Laurent. *Geomtry of cuts and metrics.* Springer Verlag, Berlin Heidelberg, 1997.

[12] S. Elhedhli and J. L. Goffin, *The integration of an interior-point cutting plane method within a branch-and-price algorithm.* to appear in Mathematical Programming, 2004.

[13] M. X. Goemans and D. P. Williamson. *Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming.* J. Assoc. Comput. Mach., 42:1115–1145, 1995.

[14] B. Grone, C.R. Johnson, E. Marques de Sa, and H. Wolkowicz. *Positive definite completions of partial Hermitian matrices.* Linear Algebra and its Applications, 58:109–124, 1984.

[15] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, Berlin, Germany, 1988.

[16] G. Gruber. *On semidefinite programming and applications in combinatorial optimization.* PhD thesis, University of Technology, Graz, Austria, April 2000.

[17] J. Håstad. *Some optimal inapproximability results.* Journal of the ACM, 48(4):798–859, 2001.

[18] C. Helmberg. *Semidefinite programming for combinatorial optimization.* Technical Report ZR-00-34, TU Berlin, Konrad-Zuse-Zentrum, Berlin, October 2000. Habilitationsschrift.

[19] C. Helmberg. *A cutting plane algorithm for large scale semidefinite relaxations.* Technical Report 01–26, TU Berlin, Konrad-Zuse-Zentrum, Berlin, October 2001. To appear in "The Sharpest Cut", Festschrift in honor of M. Padberg's 60th birthday, MPS-SIAM 2004.

[20] C. Helmberg and F. Rendl. *Solving quadratic (0,1)-problems by semidefinite programs and cutting planes.* Mathematical Programming, 82:291–315, 1998.

[21] G. Iyengar and M. T. Cezik. *Cutting planes for mixed 0-1 semidefinite programming.* In 8th International Conference on Integer Programming and Combinatorial Optimization (IPCO), Utrecht, The Netherlands, 2001.

[22] H. Karloff. *How good is the Goemans-Williamson MAX CUT algorithm?* SIAM Journal on Computing, 29:336–350, 1999.

[23] R. M. Karp. *Reducibility among combinatorial problems.* In Complexity of Computer Computations, R. E. Miller and J. W. Thatcher (editors), pages 85–103. Plenum Press, New York, 1972.

[24] J. E. Kelley. *The cutting plane method for solving convex programs.* Journal of SIAM, 8:703–712, 1960.

[25] B. W. Kernighan and S. Lin. *An efficient heuristic procedure for partitioning graphs.* Bell System Technical Journal, 49:291–307, 1970.

[26] K. Krishnan. *Linear programming approaches to semidefinite programming problems.* PhD thesis, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, July 2002.

[27] K. Krishnan and J. E. Mitchell. *Semi-infinite linear programming approaches to semidefinite programming (SDP) problems.* Fields Institute Communications Series, Volume 37, "Novel approaches to hard discrete optimization problems", edited by P. Pardalos and H. Wolkowicz, pages 121–140, 2003.

[28] K. Krishnan and J. E. Mitchell. *An unifying survey of existing cutting plane methods for semidefinite programming.* AdvOL-Report No. 2004/2, Advanced Optimization Laboratory, McMaster University, January 2004 (to appear in Optimization Methods & Software).

[29] K. Krishnan and J. E. Mitchell. *Properties of a cutting plane method for semidefinite programming.* Technical report, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, May 2003.

[30] K. Krishnan and T. Terlaky. *Interior point and semidefinite approaches in combinatorial optimization.* AdvOL-Report No. 2004/2, Advanced Optimization Laboratory, McMaster University, January 2004.

[31] J. B. Lasserre. *Global optimization with polynomials and the problem of moments.* SIAM Journal on Optimization, 11:796–817, 2001.

[32] J. B. Lasserre. An explicit equivalent semidefinite program for nonlinear 0-1 programs. SIAM Journal on Optimization, 12:756-769, 2002.

[33] M. Laurent. *A journey from some classical results to the Goemans-Williamson approximation algorithm for max-cut.* Lecture given at CORE, Louvain La Neuve, September 1-12 2003. Available from http://homepages.cwl.nl/˜monique/

[34] M. Laurent and F. Rendl. *Semidefinite programming and integer programming.* Technical Report PNA-R0210, CWI, Amsterdam, April 2003.

[35] J. E. Mitchell. *Computational experience with an interior point cutting plane algorithm.* SIAM Journal on Optimization, 10:1212–1227, 2000.

[36] Y. E. Nesterov. *Semidefinite relaxation and nonconvex quadratic optimization.* Optimization Methods and Software, 9:141–160, 1998.

[37] G. Pataki. *Cone-LP's and semidefinite programs: Geometry and a simplex-type method.* In Proceedings of the Fifth IPCO Conference, W. H. Cunningham, S. T. McCormick, and M. Queyranne (editors), volume 1084 of Lecture Notes in Computer Science, Springer Verlag Berlin, 1996.

[38] G. Pataki. *On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues.* Mathematics of Operations Research, 23:339–358, 1998.

[39] G. Pataki and S. H. Schmieta. *The DIMACS library of mixed semidefinite-quadratic-linear programs.* Computational Optimization Research Center, Columbia University, 2002. Also available at http://dimacs.rutgers.edu/Challenges/Seventh/Instances/

[40] S. Poljak and Z. Tuza. *Maximum cuts and large bipartite subgraphs.* In Combinatorial Optimization, volume 20 of DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pages 181–244. AMS/DIMACS, 1995.

[41] S. Poljak and Z. Tuza. *The expected relative error of the polyhedral approximation of the maxcut problem.* Operations Research Letters, 16:191-198, 1994.

[42] H. D. Sherali and W. P. Adams. *A hierarchy of relaxation between the continuous and convex hull representations for zero-one programming problems.* SIAM Journal Discrete Mathematics, 3:411–430, 1990.

[43] K. C. Toh, M. J. Todd, and R. Tutuncu. *SDPT3– a Matlab software package for semidefinite programming.* Optimization Methods and Software, 11:545–581, 1999.

[44] R. J. Vanderbei. *LOQO user's manual - version 3.10.* Optimization Methods and Software, 11:485–514, 1999.

[45] L. A. Wolsey. *Integer Programming.* John Wiley & Sons, Inc., New York 1998.

[46] Y. Zhang. *User's guide to LIPSOL: Linear programming interior point solvers v0.4*. Optimization Methods and Software, 11:385–396, 1999.