

**Solving semidefinite programming problems
using branch-and-bound**

John E. Mitchell

Department of Mathematical Sciences

Rensselaer Polytechnic Institute

Troy, NY 12180 USA

mitchj@rpi.edu

<http://www.math.rpi.edu/~mitchj>

IPM 2000 and EURO 2000,

Budapest,

July 2000

Abstract

Many combinatorial optimization problems have relaxations that are semidefinite programming problems. In principle, the combinatorial optimization problem can then be solved by using a branch-and-cut procedure, where the problems to be solved at the nodes of the tree are semidefinite programs. It is desirable that the solution to one node of the tree should be exploited at the child node in order to speed up the solution of the child. We discuss the incorporation of branch-and-cut into solution techniques for semidefinite programming problems. In particular, we show how the solution to the parent relaxation can be used as a warm start to construct an appropriate initial dual solution to the child problem. This restart method for SDP branch-and-cut can be regarded as analogous to the use of the dual simplex method in the branch-and-cut method for mixed integer linear programming problems.

Contents

1. Introduction.
2. Formulation of SDP problems.
3. Branch and bound, and our branching structure.
4. Finding a new strictly feasible dual solution.
5. A lower bound for the new dual.
6. Properties of well-centered iterates, and implications for branching rules, breadth first vs. depth first search, and fixing variables.
7. The child primal problem, and the Slater constraint qualification.
8. Conclusions.

1 Introduction

Many [combinatorial and quadratic optimization problems](#) can be written as:

$$\begin{aligned} \min \quad & v^T C v \\ \text{subject to} \quad & v^T A_i v \ \# \ b_i, \quad i = 1, \dots, m \quad (COP) \\ & v_j = \pm 1, \quad j = 1, \dots, n, \end{aligned}$$

where v is an n -vector, C and $A_i, i = 1, \dots, m$ are symmetric $n \times n$ matrices, b is an m -vector, and $\#$ denotes either $=$ or \geq or \leq , depending on the particular i th constraint.

Often, C is a [sparse](#) matrix and each A_i is a [rank one](#) matrix that may also be [sparse](#).

2 Examples

- The [maximum cut problem](#) in a graph with weighted edges requires dividing the vertices of a graph into two sets so that the total weight of the edges having one endpoint in each set is as large as possible. This can be expressed in the form (*COP*) by letting C_{ij} be the negative of the edge weight between vertices i and j , with no constraints of the form $v^T A_i v \# b_i$ necessary.
- The [equipartition problem](#) requires that the vertices of a graph be divided into two sets of equal cardinality with as few edges as possible having one endpoint in each set. This can be formulated by taking C to be the matrix of edge weights and taking $A_1 = ee^T$, where e denotes a vector of ones of appropriate dimension, and $b_1 = 0$. No other constraints are needed.
- For other examples, see [1, 2, 6, 7, 10].

3 Formulation as an SDP

A [semidefinite programming relaxation](#) of (COP) is

$$\begin{aligned}
 \min \quad & C \bullet V \\
 \text{subject to} \quad & V_{jj} = 1, \quad j = 1, \dots, n, \\
 & A_i \bullet V \# b_i, \quad i = 1, \dots, m \\
 & V \succeq 0.
 \end{aligned} \tag{SDP}$$

Notation:

1. \bullet denotes the Frobenius inner product, so $C \bullet V := \sum_{i=1}^n \sum_{j=1}^n C_{ij} V_{ij}$.
2. $V \succeq 0$ denotes that the matrix V must be positive semidefinite.

Derivation:

(SDP) is obtained from (COP) by noticing that the quadratic form $v^T C v$ can be written as $C \bullet V$ if we equate vv^T and V . It follows that the diagonal elements of V must equal one, and that V must be positive semidefinite. The condition that V is a rank one matrix is omitted, so (SDP) is a relaxation of (COP) .

3.1 Add slack variables

With the addition of slack variables, if necessary, (*SDP*) can be written equivalently as

$$\begin{aligned}
 \min \quad & C \bullet V + c^T x && (SDPE) \\
 \text{subject to} \quad & V_{jj} = 1, && j = 1, \dots, n \\
 & A_i \bullet V + a_i^T x = b_i, && i = 1, \dots, m \\
 & V \succeq 0, \quad 0 \leq x \leq u,
 \end{aligned}$$

where c , x , u and $a_i, i = 1, \dots, m$ are l -vectors. Since $V \succeq 0$ and the diagonal elements of V are all equal to one, we have that $-1 \leq V_{ij} \leq 1$ for any element. Therefore, it is easy to calculate an upper bound u_i for any slack variable x_i .

3.2 Dual problem

The dual problem to (SDPE) is

$$\begin{aligned}
 & \max \quad \text{trace}(W) + b^T y - u^T w && (SDPD) \\
 \text{subject to} \quad & W + \sum_{i=1}^m y_i A_i + S = C \\
 & A^T y + z - w = c \\
 & S \succeq 0, \quad z \geq 0, \quad w \geq 0,
 \end{aligned}$$

where A is an $m \times l$ matrix whose i th row is a_i , W is a diagonal matrix, y is an m -vector, S is an $n \times n$ matrix, and z and w are l -vectors.

4 Branch and bound

- Helmberg and Rendl [7] proposed branching on whether v_i and v_j should be the same or different.
- For the MAX-CUT problem, this corresponds to deciding whether **two vertices** i and j should be on the **same side** of the cut or on **opposite sides**.
- With this branching rule, V_{ki} and V_{kj} are then **also constrained** to be either the same or different (depending on the branch) for each index k .
- This means that the problem can be replaced by an equivalent semidefinite program of **dimension one less**.
- Without loss of generality, in what follows we assume that the indices i and j are the **last two indices**.
- Thus, we are branching on whether $v_{n-1} = v_n$ or $v_{n-1} \neq v_n$.
- For simplicity, we now restrict attention to the branch where $v_{n-1} = v_n$. (The other branch is very similar.)

4.1 The child formulation: objective function

Write the objective function matrix C as

$$C = \begin{bmatrix} \bar{C} & p_1 & p_2 \\ p_1^T & \alpha & \beta \\ p_2^T & \beta & \gamma \end{bmatrix}, \quad (1)$$

where \bar{C} is an $(n - 2) \times (n - 2)$ matrix, p_1 and p_2 are $(n - 2)$ -vectors, and α , β , and γ are scalars, then we obtain two different matrices depending on the branch.

If $v_{n-1} = v_n$, we obtain:

$$C_S = \begin{bmatrix} \bar{C} & p_1 + p_2 \\ p_1^T + p_2^T & \alpha + 2\beta + \gamma \end{bmatrix}. \quad (2)$$

4.2 The child formulation: constraints

Similarly, write the constraint matrices A_i as

$$A_i = \begin{bmatrix} \bar{A}_i & q_{i1} & q_{i2} \\ q_{i1}^T & \alpha_i & \beta_i \\ q_{i2}^T & \beta_i & \gamma_i \end{bmatrix}, \quad (3)$$

where \bar{A}_i is an $(n - 2) \times (n - 2)$ matrix, q_{i1} and q_{i2} are $(n - 2)$ -vectors, and α_i , β_i , and γ_i are scalars.

If $v_{n-1} = v_n$, we obtain:

$$A_{iS} = \begin{bmatrix} \bar{A}_i & q_{i1} + q_{i2} \\ q_{i1}^T + q_{i2}^T & \alpha_i + 2\beta_i + \gamma_i \end{bmatrix}. \quad (4)$$

4.3 Low rank matrices

With the dual scaling algorithm of Benson *et al.* [2], it is useful to work with low rank matrices, if possible. The transformation defined above leaves a rank one matrix as a rank one matrix, as the following lemma shows.

Lemma 1 *If A_i is a rank one matrix then A_{iS} is a rank one matrix.*

Proof: If A_i is a rank one matrix, we can write

$$A_i = \begin{bmatrix} v_A \\ \Gamma \\ \Phi \end{bmatrix} \begin{bmatrix} v_A^T & \Gamma & \Phi \end{bmatrix},$$

where v_A is an $(n - 2)$ -vector, and Γ and Φ are scalars. It is can then be verified that A_{iS} is given by

$$A_{iS} = \begin{bmatrix} v_A \\ \Gamma + \Phi \end{bmatrix} \begin{bmatrix} v_A^T & \Gamma + \Phi \end{bmatrix},$$

as required. □

Example

For the [equipartition](#) problem, on the $v_{n-1} = v_n$ branch, the constraint matrix $A_i = ee^T$ becomes

$$A_{iS} = \begin{bmatrix} 1 & \dots & 1 & 2 \\ \vdots & \ddots & \vdots & \vdots \\ 1 & \dots & 1 & 2 \\ 2 & \dots & 2 & 4 \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & \dots & 1 & 2 \end{bmatrix},$$

a rank one matrix.

4.4 The new relaxation

After branching, the relaxation takes the following form, where V is now an $(n - 1) \times (n - 1)$ symmetric matrix:

$$\begin{aligned}
 \min \quad & C_S \bullet V + c^T x && (SDPES) \\
 \text{subject to} \quad & V_{jj} = 1, && j = 1, \dots, n - 1, \\
 & A_{iS} \bullet V + a_i^T x = b_i, && i = 1, \dots, m \\
 & V \succeq 0, \quad 0 \leq x \leq u.
 \end{aligned}$$

Theorem 1 (*Helmberg and Rendl [7].*) *Let v be a feasible solution to (COP). Define $v^{n-1} := [v_1, \dots, v_{n-1}]^T$ and $V^{n-1} := v^{n-1}(v^{n-1})^T$. If $v_{n-1} = v_n$ then V^{n-1} is feasible in (SDPES) for some choice of $x \geq 0$.*

5 Finding a new strictly feasible dual solution

Branch-and-cut for [mixed integer linear programming problems](#):

- fixing a primal variable at zero or one corresponds to dropping a dual constraint
- so the old dual solution is still feasible
- so the subproblems obtained after branching are solved using the [dual simplex method](#).

We show that in the [SDP](#) case that it is still possible to construct a [dual strictly feasible solution](#) to the new subproblems using the old solution in a straightforward manner. A dual-scaling algorithm (Benson *et al.* [2, 3]) could then be used to solve the resulting subproblems.

5.1 Branch and bound outcomes

At each iteration of an interior point method for solving $(SDPE)$, the matrix of dual slacks S is positive definite. At optimality, this matrix will be positive semidefinite. There are [four possible outcomes](#) at a node of the branch-and-cut tree:

- The optimal solution to the relaxation corresponds to a [feasible solution](#) to (COP) . In this case, we can [fathom](#) the node through feasibility.
- The problem $(SDPE)$ is [infeasible](#), so again we can fathom the node.
- The optimal solution to $(SDPE)$ has value [worse than a known feasible solution](#) to (COP) , so we can fathom by bounds.
- The optimal solution to $(SDPE)$ has value better than the best known feasible solution to (COP) and it does not correspond to a feasible solution to (COP) . In this case, it is [necessary to branch](#).

5.2 Approximate solutions

Notice that in every case except the first, an approximate optimal solution to $(SDPE)$ will suffice, enabling us to determine the status of the node. In the first case, there is no need to branch. Thus, we may assume that when we branch we know a feasible dual solution where the [matrix of dual slacks is positive definite](#).

If, for numerical reasons, it is desired to use an [earlier dual iterate](#), that is also possible with the restart procedure we describe below, again provided that the matrix of dual slacks is positive definite.

5.3 Dual restart

The dual problem to $(SDPES)$ is

$$\begin{aligned}
 & \max \quad \text{trace}(W) + b^T y - u^T w && (SDPDS) \\
 \text{subject to} \quad & W + \sum_{i=1}^m y_i A_{iS} + S = C_S \\
 & A^T y + z - w = c \\
 & S \succeq 0, \quad z \geq 0, \quad w \geq 0
 \end{aligned}$$

where now W is a $(n-1) \times (n-1)$ diagonal matrix, and S is a $(n-1) \times (n-1)$ symmetric matrix.

Let (W', y', S', z', w') be the [known feasible solution](#) at the parent node, where S' is a positive definite matrix.

We will restart with:

$$W_{ii} = \begin{cases} W'_{ii} & i = 1, \dots, n-2 \\ W'_{(n-1),(n-1)} + W'_{nn} & i = n-1 \end{cases} \quad (5)$$

$$y = y' \quad (6)$$

$$z = z' \quad (7)$$

$$w = w', \quad (8)$$

and S_S defined to be the [resulting dual slacks](#).

Is S_S positive definite?

5.4 Factorization of S'

We write S' as:

$$S' = \begin{bmatrix} M & r_1 & r_2 \\ r_1^T & \tau & \nu \\ r_2^T & \nu & \xi \end{bmatrix}, \quad (9)$$

where M is an $(n - 2) \times (n - 2)$ matrix, r_1 and r_2 are $(n - 2)$ -vectors, and τ , ν , and ξ are scalars. Since this is a positive definite matrix, it has a [Cholesky factorization](#):

$$S' = L'L'^T := \begin{bmatrix} L & 0 & 0 \\ v_1^T & \phi & 0 \\ v_2^T & \zeta & \chi \end{bmatrix} \begin{bmatrix} L^T & v_1 & v_2 \\ 0 & \phi & \zeta \\ 0 & 0 & \chi \end{bmatrix}, \quad (10)$$

where L is a lower triangular matrix, v_1 and v_2 are $(n - 2)$ -vectors, and ϕ , ζ , and χ are real numbers.

5.5 Feasibility of new dual solution

Theorem 2 *If S' is positive definite then S_S is positive definite.*

Proof: The matrix S_S can be factored as

$$S_S = \begin{bmatrix} L & 0 \\ (v_1 + v_2)^T & \sigma_S \end{bmatrix} \begin{bmatrix} L^T & v_1 + v_2 \\ 0 & \sigma_S \end{bmatrix} \quad (11)$$

where

$$\sigma_S^2 > 0.$$

It follows that S_S is positive definite. □

6 A lower bound for the new dual

The value of any feasible solution for the new dual subproblem (*SDPDS*) provides a lower bound on the optimal value of the corresponding subproblem. This can be used to **prune** the subproblem if the bound is large enough. Define Ξ to be the value of the known feasible solution to dual of the parent subproblem.

Theorem 3 *A valid lower bound for (*SDPDS*) is $\Xi + (\phi + \zeta)^2 + \chi^2$.*

This theorem can be used to help determine an appropriate **branching variable**: choose a pair of vertices where the increase in the lower bound is as large as possible.

Drawback: the examination of any pair of vertices requires a Cholesky factorization of the matrix, after re-ordering the columns.

The bounds given in Theorem 3 are exactly those obtained after fixing the last two variables in the tree defined in Goemans [4]. In general, if we modified our algorithm to take no iterations at subsequent nodes of the tree and if at level k we fix the relationship of vertices $n - k$ and $n - k + 1$, then we would again obtain the same bounds as in [4].

7 Properties of well centered iterates

For this section, we assume that the primal and dual solutions for the parent problem are [perfectly centered](#), namely,

$$V'S' = \mu I \quad (12)$$

for some positive scalar μ , where V' is the primal iterate for the parent problem. This is equivalent to

$$L'^T V' L' = \mu I, \quad (13)$$

where L' is defined in equation (10). Examining the (n, n) position of this equality shows that

$$\chi^2 = \mu, \quad (14)$$

since $V'_{ii} = 1$ for each i .

To fix notation, we have

$$V' = \left[\begin{array}{c|c} \cdots & \vdots \\ \hline & 1 \ \epsilon \\ \cdots & \epsilon \ 1 \end{array} \right] \quad (15)$$

and

$$L' = \left[\begin{array}{c|c} \cdots & 0 \\ \hline & \phi \ 0 \\ \cdots & \zeta \ \chi \end{array} \right]. \quad (16)$$

7.1 Branching rules

At the optimal solution to the underlying integer programming problem, V should be a rank one matrix. This suggests that it may be good to branch on columns of V that are close to [orthogonal](#) to one another. One situation in which the final two columns of V will not be parallel is if $\epsilon = 0$. This can be identified from the dual solution:

Lemma 2 . *If $V'_{(n-1),n} = 0$ then $\phi^2 = \mu$ and $\zeta = 0$, provided the iterates are perfectly centered.*

The next lemma relates to large values of ϵ .

Lemma 3 *If the iterates are perfectly centered then $-1 < \epsilon < 1$.*

We can describe the relationship of ϵ and ζ :

Lemma 4 *If the iterates are perfectly centered then $\phi\epsilon + \zeta = 0$, so ϵ and ζ have opposite signs.*

7.2 Fixing variables

Let Υ be the value of the best known feasible solution for (COP) . If the bound given in Theorem 3 is larger than Υ then we must have $v_{n-1} \neq v_n$ in **any optimal solution**.

If the iterates are perfectly centered then $\phi^2 = \frac{\mu}{1-\epsilon^2}$, so if ϵ is **close enough to -1** then the bound will be larger than Υ , since ζ is then negative.

Of course, in practice the iterates will only be approximately centered. Nonetheless, for any entry V_{ij} that is close enough to -1 , it may be worthwhile performing a Cholesky factorization of S' , with the columns ordered so that these two columns are last, in order to confirm that one of the bounds is large enough to fix the variables.

A different method for fixing variables is proposed in Helmberg [5], which also uses the matrix S' of dual slacks.

7.3 The Determinant of S_S

Theorem 4 *Assume the iterates are perfectly centered. Let $\epsilon = V'_{(n-1),n}$.*

1. *If $\epsilon = 0$ then $\det(S_S) = \det(S_O) = \frac{2}{\mu} \det(S')$.*
2. *For general ϵ ,*

$$\det(S_S) = \frac{(1 - \epsilon)^2 + 1 - \epsilon^2}{\mu} \det(S')$$

$$\det(S_O) = \frac{(1 + \epsilon)^2 + 1 - \epsilon^2}{\mu} \det(S')$$

Thus, if $\epsilon \approx \pm 1$ then one of the child problems should have a far smaller determinant than S' .

7.4 Breadth first vs. depth first search

The [bound](#) on one of the child subproblems can be increased dramatically if $\epsilon \approx \pm 1$.

Branching in this manner resembles a [depth first search](#), with it likely that one of the branches will lead to a feasible solution quickly, and it may be possible to prune the other branch effectively.

Branching on values of ϵ close to zero resembles a [breadth first search](#) approach, where both child subproblems are improved approximately equally.

8 The child primal problem

Slater constraint qualification:

If there exists a **strictly feasible solution** to $(SDPE)$ and if the optimal value of $(SDPD)$ is bounded then there exists an optimal solution to $(SDPE)$ and the optimal values of $(SDPD)$ and $(SDPE)$ agree.

Note that the constraint qualification does not state whether the optimal value to $(SDPD)$ is attained.

We discuss methods for trying to **fix up the relaxations** if they don't contain primal strictly feasible solutions.

As an aside, note that the **dual scaling method** does not require a new strictly feasible primal iterate in order to be restarted. All that is required is an upper bound \bar{z} on the value of the current relaxation. Such an upper bound can be obtained from any point v that is feasible in (COP) and which satisfies the branching restrictions. Thus, heuristics can be used to construct an upper bound, if an appropriate point is not known already.

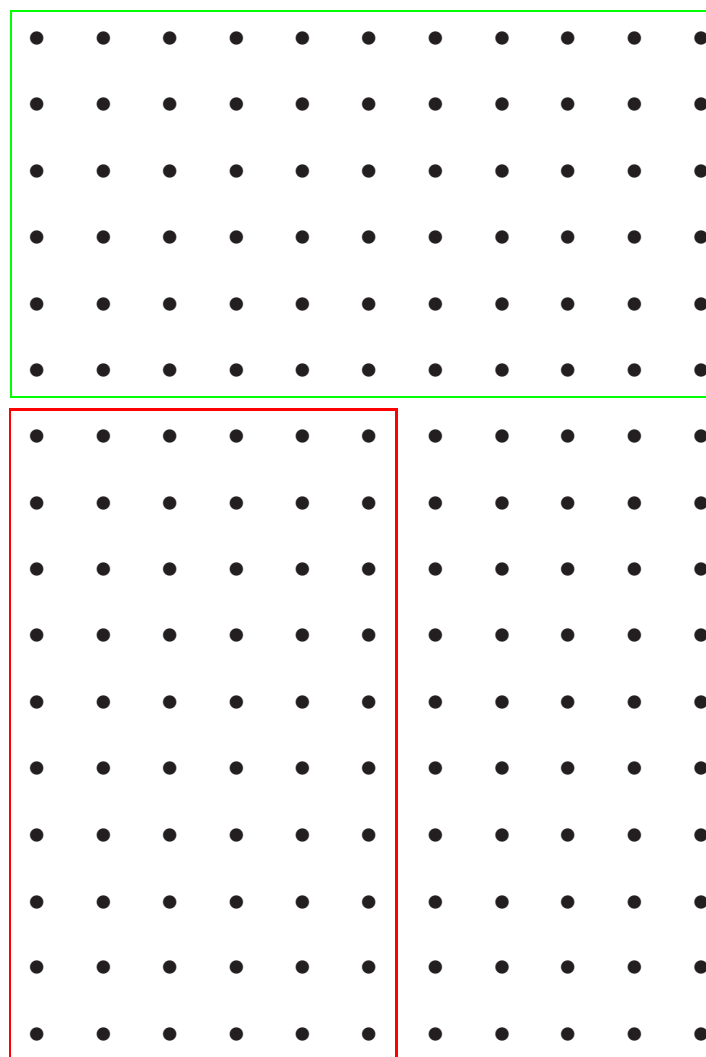
8.1 Ensuring the primal relaxation has an interior

It is possible that the child problem has **no feasible interior solutions** even if the parent problem has such solutions.

Example:

- An **equipartition** problem with n vertices.
- **Two subsets** of the vertices that have already been fixed, with the vertices in each subset required to appear on the same side of the partition as each other.
- We now branch to require that the two subsets appear on the **same side** of the partition.
- Assume each subset contains **less than half** the vertices.
- If **more than half** of the vertices are in the union of the two subsets then the subproblem will be infeasible.
- If **exactly half** of the vertices appear in the two subsets then the subproblem will have exactly one feasible solution, namely, that every other vertex should appear on the opposite side from the two subsets.

U_2 : $q < n/2$ vertices



U_1 : $p < n/2$ vertices

- All vertices in U_1 must be on same side of cut.
- All vertices in U_2 must be on same side of cut.
- Branch to force all vertices in $U_1 \cup U_2$ to be on same side of cut.

Full dimensional feasible regions

Lemma 5 *If the set of feasible solutions to (COP) is full-dimensional then the set of feasible solutions to (SDP) contains a positive definite matrix.*

Proof: Let $\{v^1, \dots, v^n\}$ be a set of linearly independent feasible solutions to (COP). Take $\hat{V} := [v^1, \dots, v^n]$. Let $V = \frac{1}{n}\hat{V}\hat{V}^T$. It follows that $V_{jj} = 1$ for $j = 1, \dots, n$. Consider the constraint $A_i \bullet V \# b_i$. We have

$$\begin{aligned} A_i \bullet V &= \text{trace}(A_i V) \\ &= \frac{1}{n} \text{trace}(A_i \hat{V} \hat{V}^T) \\ &= \frac{1}{n} \text{trace}(\hat{V}^T A_i \hat{V}) \\ &= \frac{1}{n} \sum_{k=1}^n (v^k)^T A_i v^k. \end{aligned}$$

Thus, if each v^k satisfies the constraint $(v^k)^T A_i v^k \# b_i$ then V satisfies $V^T A_i V \# b_i$. By construction, V is positive semidefinite and, further, $\text{rank}(V) = \text{rank}(\hat{V}) = n$. The result follows. \square

(Note that, independently, Tunçel [9] recently proved a similar result.)

The following corollary is an immediate consequence of this lemma.

Corollary 1 *If the set of solutions to (COP) with the additional restriction $(v_{n-1} \pm v_n = 0)$ still has dimension $n - 1$ then there is a positive definite feasible primal matrix for the child problem.*

If feasible region to (COP) not full dimensional:

There must be valid [equality constraints](#) for (COP) .

- One technique used in Benson *et al.* [2] for handling equality constraints is to include an [artificial variable](#) with a corresponding large objective function coefficient.
- When branching, an artificial variable could be added to all constraints that involved $V_{j,(n-1)}$ and/or V_{jn} for some $j \in \{1, \dots, n\}$, and this would ensure that the resulting child primal problem was full-dimensional if the parent problem was full-dimensional.
- Even if the child problem is infeasible, the problem with the artificial variable will still be feasible, albeit with a large optimal value and the artificial variable nonzero at optimality. It should not be necessary to solve such a subproblem to optimality within a branch-and-cut approach, as it is likely to be [pruned](#) before optimality is reached.
- By placing upper and lower bounds on the artificial variable, a strictly feasible interior point for the dual problem can be found easily.

8.2 Rank one inequalities

The particular representation that we have for the problem may not allow strictly feasible solutions because some additional variables have now become fixed or some constraints have become redundant.

Consider the triangle inequality

$$V_{j,(n-1)} - V_{j,n} - V_{(n-1),n} \geq -1.$$

This constraint can be written

$$dd^T \bullet V \geq 1 \tag{17}$$

where d is an n -vector given by

$$d_j = 1, \quad d_{n-1} = 1, \quad d_n = -1, \quad d_k = 0 \text{ otherwise.}$$

On the branch where we fix $v_{n-1} = v_n$, this constraint becomes $V_{jj} \geq 1$, which **forces** the corresponding primal slack variable to **equal zero**.

Thus, the primal problem no longer has a **strictly feasible point**.

Remedy: **drop** this constraint, which requires dropping the corresponding dual variable y_i .

Consequences:

- The matrix of dual slacks can be left unchanged if W_{jj} is increased by y_i .
- This leaves the dual objective value unchanged.
- The other constraint involving y_i is of the form $-y_i + z_i - w_i = c_i$, and this can also be dropped without affecting the dual problem, since z_i and w_i only appear in this constraint.

The other branch:

The triangle inequality becomes

$$V_{jj} + 4V_{j,(n-1)} + 4V_{(n-1),(n-1)} \geq 1.$$

Equivalently, $V_{j,(n-1)} \geq -1$, a **redundant** constraint.

If $y_i \geq 0$ then this constraint can be **dropped** and the resulting matrix of dual slacks will still be positive definite.

(Recall: $W + \sum_{i=1}^m y_i A_i + S = C$.)

If y_i is **negative**, it may be necessary to **decrease** W_{jj} and $W_{(n-1),(n-1)}$ in order to regain a positive definite matrix of dual slacks, when dropping the constraint.

The smallest total decrease in the components of W is to decrease W_{jj} by $3y_i$ and decrease $W_{(n-1),(n-1)}$ by $6y_i$. Resulting change in the j and $n - 1$ rows and columns of S_O is proportional to

$$\begin{bmatrix} -1 + 3 & -2 \\ -2 & -4 + 6 \end{bmatrix}.$$

Note that the constraint $-y_i + z_i - w_i \leq 0$ can also be dropped; the upper bound u_i is 8 ($=9-1$), and $w_i \geq -y_i$. Thus, the dual objective function value is not decreased by these modifications.

General rank one constraints:

$A_i = tt^T$ and $t^T = [\bar{t}^T, p_t, q_t]^T$, where \bar{t} is an $(n - 2)$ -vector and p_t and q_t are scalars.

Break into cases:

- If at least two components of \bar{t} are nonzero or at most one of p_t and q_t is nonzero, then the modified constraint is unlikely to be redundant.
- If both p_t and q_t are nonzero and \bar{t} is zero then the constraint can be deleted, with a possible adjustment in $W_{(n-1),(n-1)}$ and with the dropping of a constraint of the form $y_i + z_i - w_i = c_i$.
- If p_t and q_t are both nonzero and exactly one component of \bar{t} is nonzero, then the constraint can be handled as indicated above for the triangle inequalities. It may be that the resulting constraint requires that $V_{j,(n-1)}$ take a particular value, either 1 or -1. In this case, we can fix this element of V by combining columns j and $n - 1$.

9 Conclusions

Theorem 2 shows that an analogue of the dual simplex method can be used to solve semidefinite programming problems when using a branch-and-cut approach, because a strictly feasible dual solution can be found easily.

Investigation of the properties of this dual solution suggest that it may be useful to branch on elements of V' that are close to zero, since the determinants of the new dual slack matrices are well behaved and the bound on both branches of the tree can be increased.

It may be that there is no strictly feasible primal point after branching. We discussed how to handle some forms of redundant constraints and how to modify the primal problem in order to ensure the existence of a strictly feasible primal solution.

The results in this paper should make it possible to exploit a warm start in an SDP branch-and-cut algorithm, especially one that uses a dual-scaling algorithm.

References

- [1] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51, 1995.
- [2] S. J. Benson, Y. Ye, and X. Zhang. Mixed linear and semidefinite programming for combinatorial and quadratic optimization. *Optimization Methods and Software*, 11:515–544, 1999.
- [3] S. J. Benson, Y. Ye, and X. Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM Journal on Optimization*, 10(2):443–461, 2000.
- [4] M. X. Goemans. Semidefinite programming in combinatorial optimization. *Mathematical Programming*, 79:143–161, 1997.
- [5] C. Helmberg. Fixing variables in semidefinite relaxations. Technical Report SC-96-43, Konrad-Zuse-Zentrum fuer Informationstechnik, Berlin, December 1996.
- [6] C. Helmberg, S. Poljak, F. Rendl, and H. Wolkowicz. Combining semidefinite and polyhedral relaxations for integer programs. In E. Balas and J. Clausen, editors, *Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science*, volume 920, pages 124–134. Springer, 1995.
- [7] C. Helmberg and F. Rendl. Solving quadratic (0,1)-problems by semidefinite programs and cutting planes. *Mathematical Programming*, 82:291–315, 1998.
- [8] J. E. Mitchell. Restarting after branching in the SDP approach to MAX-CUT and similar combinatorial optimization problems. Technical report, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, June 1999. Accepted for publication in the *Journal of Combinatorial Optimization*.
- [9] L. Tunçel. On the Slater condition for the SDP relaxations of nonconvex sets. Technical Report CORR 2000–13, Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada, February 2000.

- [10] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.