Solving Lift-and-Project Relaxations of Binary Integer Programs

Samuel Burer* Dieter Vandenbussche†

June 7, 2004

Abstract

We propose a method for optimizing the lift-and-project relaxations of binary integer programs introduced by Lovász and Schrijver. In particular, we study both linear and semidefinite relaxations. The key idea is a restructuring of the relaxations, which isolates the complicating constraints and allows for a Lagrangian approach. We detail an enhanced subgradient method and discuss its efficient implementation. Computational results illustrate that our algorithm produces tight bounds more quickly than state-of-the-art linear and semidefinite solvers.

Keywords: Integer programming, lift-and-project relaxations, semidefinite programming, augmented Lagrangian.

1 Introduction

In the field of optimization, binary integer programs have proven to be an excellent source of challenging problems, and the successful solution of larger and larger problems over the past few decades has required significant theoretical and computational advances. One of the fundamental issues is how to obtain a "good" description of the convex hull of integer solutions, and many specific classes of integer programs have been solved by finding problem-specific ways to address this issue.

Researchers have also developed techniques for approximating the convex hull of integer solutions without any specific knowledge of the problem, i.e., techniques that apply to arbitrary binary integer programs. Some of the earliest work done in this direction was by Gomory (1963) in generating linear inequalities that tighten the basic linear relaxation. A different idea, which has been advocated by several authors, is to approximate the convex hull as the projection of some polyhedron lying in a space of higher dimension. We refer the reader to Balas (1979); Sherali and Adams (1990); Lovász and Schrijver (1991); Balas et al. (1993); Lasserre (2001). Connections between these works are explored in Laurent and Rendl (2002); Laurent (2003).

Although these so-called *lift-and-project* methods are quite powerful theoretically, they present great computational challenges because one typically must optimize in the space of the lifting, i.e.,

*Department of Management Sciences, University of Iowa, Iowa City, IA 52242-1000, USA. (Email: samuel-burer@uiowa.edu). This author was supported in part by NSF Grant CCR-0203426.

†Department of Mechanical and Industrial Engineering, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, USA. (Email: dieterv@uiuc.edu).

the space of higher dimension. Computational issues are detailed in Balas et al. (1993); Sherali and Adams (1997); Ceria and Pataki (1998); Karisch et al. (1999); Dash (2001).

In this paper, we focus on the techniques proposed by Lovász and Schrijver (LS), including both linear and semidefinite relaxations. In particular, our main goal is to present improved computational methods for optimizing over the first-level LS relaxations. We are aware of only one study (Dash, 2001), which investigates the strength of these relaxations computationally. This shortage of computational experience is due to the dramatic size of these relaxations. For example, one specific semidefinite relaxation that has been considered by Dash (and which we also consider in this paper) has well over 1.7 million constraints.

Since it is unlikely that these relaxations can be solved using direct algorithms, we instead adopt the paradigm of decomposition, which is common in large-scale optimization methods. (Dash (2001) also considers a decomposition approach.) The main idea here is a clever decomposition of the LS relaxations, which allows for a Lagrangian approach. Instead of using the common subgradient algorithm, however, we propose to use an augmented Lagrangian algorithm, which is, in some sense, an enhanced subgradient method and which also has connections with the bundle method for convex optimization. We provide a theoretical explanation of the benefits of the augmented Lagrangian algorithm and give a detailed explanation of our implementation, which is demonstrated to outperform state-of-the-art subgradient, linear, and semidefinite solvers.

We remark that, while the idea of using the augmented Lagrangian method for linear programs is not new (see Poljak and Tret′jakov, 1972), very little work has been done on the computational aspects of such a method. In this paper, we fill this gap concerning large-scale linear programs and also present an augmented Lagrangian method for semidefinite programs for the first time.

The paper is organized as follows. In Section 2, we give background on the Lovász-Schrijver lift-and-project relaxations as well as propose the decomposition technique that will become the basis of our augmented Lagrangian algorithm. Then in Section 3, we discuss the augmented Lagrangian algorithm, including its theoretical benefits and specialization in the current context. Next, in Section 4, we present the details of our implementation and computational results. We also discuss the strength of the LS relaxations on various problem classes, with one highlight being that, in practice, the LS semidefinite relaxation provides the strongest known bounds for a collection of problems in the Quadratic Assignment Problem Library (QAPLIB). Finally, we conclude with a few final remarks and suggestions for future research in Section 5.

1.1 Notation and Terminology

In this section, we introduce some of the notation that will be used throughout the paper. $\mathbb{R}^n$ will refer to n-dimensional Euclidean space. The norm of a vector $x \in \mathbb{R}^n$ is denoted by $\|x\| := \sqrt{x^T x}$. We let $e_i \in \mathbb{R}^n$ represent the ith unit vector, and e is the vector of all ones. $\mathbb{R}^{n\times n}$ is the set of real, $n \times n$ matrices, $\mathcal{S}^n$ is the set of symmetric matrices in $\mathbb{R}^{n\times n}$, while $\mathcal{S}^n_+$ is the set of positive semidefinite symmetric matrices. The special notation $\mathbb{R}^{1+n}$ and $\mathcal{S}^{1+n}$ is used to denote the spaces $\mathbb{R}^n$ and $\mathcal{S}^n$ with an additional "0-th" entry prefixed or an additional 0-th row and 0-th column prefixed, respectively. The inner product of two matrices $A, B \in \mathbb{R}^{n\times n}$ is defined as $A \bullet B := \mathrm{tr}(A^T B)$, where $\mathrm{tr}(\cdot)$ denotes the sum of the diagonal entries of a matrix. The Frobenius norm of a matrix $A \in \mathbb{R}^{n\times n}$ is defined as $\|A\|_F := \sqrt{A \bullet A}$. $\mathrm{diag}(A)$ is defined as the vector with the diagonal of A as its entries.

2 The Lift-and-Project Operators of Lovász and Schrijver

When solving a 0-1 integer program of the form

$$\min \left\{ c^T x \mid Ax \leq b,\ x \in \{0,1\}^n \right\} \tag{IP}$$

we are often interested in relaxations of the convex hull of integer solutions

$$P := \operatorname{conv} \left\{ x \in \{0,1\}^n \mid Ax \leq b \right\}.$$

Optimization over such relaxations provides lower bounds that can be used within branch-and-bound methods or allow one to asses the quality of feasible solutions of (IP). The trivial linear programming (LP) relaxation is obtained by replacing $x \in \{0,1\}^n$ with $x \in [0,1]^n$. In an effort to develop relaxations that are stronger than the LP relaxation, Lovász and Schrijver (1991) introduced the lifted matrix variable

$$Y = \begin{pmatrix} 1 \\ x \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix}^T = \begin{pmatrix} 1 & x^T \\ x & xx^T \end{pmatrix}.$$

Given this definition relating Y and $x \in \{0,1\}^n$, we can observe a number of interesting properties of Y:

- Y is symmetric and positive semidefinite, i.e., $Y \in \mathcal{S}_+^{1+n}$;
- the diagonal of Y equals the 0-th column of Y, i.e., $\operatorname{diag}(Y) = Ye_0$;
- if we multiply the constraints $Ax \leq b$ of P by some x_i, we obtain the set of nonlinear inequalities $bx_i - Axx_i \geq 0$, which are valid for P; these inequalities can be written in terms of Y as
$$\left(b \middle| A\right) Ye_i \geq 0 \quad \forall \quad i = 1, \ldots, n.$$
- analogously, multiplying $Ax \leq b$ by $1 - x_i$ yields
$$\left(b \middle| A\right) Y(e_0 - e_i) \geq 0 \quad \forall \quad i = 1, \ldots, n.$$

Lovász and Schrijver observed that these properties could be used to obtain relaxations of P. In particular, they homogenized the standard LP relaxation of (IP) by defining

$$K := \left\{ \begin{pmatrix} x_0 \\ x \end{pmatrix} \in \mathbb{R}^{1+n} \mid Ax \leq x_0 b,\ 0 \leq x \leq x_0 e \right\}.$$

We remark that enforcing $x_0 = 1$ in K yields the LP relaxation and that the third and fourth properties above can be written as $Ye_i \in K$ and $Y(e_0 - e_i) \in K$. They then proposed the following sets:

$$\begin{aligned} M(K) &:= \left\{ Y \in \mathcal{S}^{1+n} \mid Ye_0 = \operatorname{diag}(Y), \quad Ye_i \in K,\ Y(e_0 - e_i) \in K \ \ \forall \ \ i = 1, \ldots, n \right\} \\ M_+(K) &:= \left\{ Y \in \mathcal{S}_+^{1+n} \mid Y \in M(K) \right\} \end{aligned}$$

Note that $M_+(K)$ differs from $M(K)$ only in that positive semidefiniteness is enforced on the Y variable. $M(K)$ now leads to a linear relaxation of P via the projected set

$$N(K) := \left\{ x \in \mathbb{R}^n \mid \begin{pmatrix} 1 \\ x \end{pmatrix} = \operatorname{diag}(Y) \ \text{ for some } \ Y \in M(K) \right\},$$

and $M_+(K)$ leads to an analogous semidefinite relaxation $N_+(K)$ of P. In particular, Lovász and Schrijver showed that $P \subseteq N_+(K) \subseteq N(K)$ and that $N(K)$ is contained in the LP relaxation of (IP). Further, they showed that applying these relaxation procedures iteratively n times yields P exactly.

We remark that our definitions of $N(K)$ and $N_+(K)$ are actually just slices (at $Y_{00} = 1$ and $x_0 = 1$) of the cones originally defined by Lovász and Schrijver.

Applying these ideas to the stable set problem, Lovász and Schrijver proved that some classes of inequalities for the stable set polytope are satisfied by all the points in $N(K)$, while other classes are only valid for $N_+(K)$. In turn, these results have significant implications for the complexity of finding maximum stable sets in various classes of graphs. Further theoretical results concerning the strength of $N(K)$ and $N_+(K)$, as well as the higher-order liftings, have also been established; see Stephen and Tunçel (1999), Cook and Dash (2001), Goemans and Tunçel (2001), Laurent (2001), Lipták and Tunçel (2003).

To compute lower bounds available from the relaxations $N(K)$ and $N_+(K)$, one must solve the LP

$$\min \left\{ c^T x \mid x \in N(K) \right\} \tag{1}$$

or the semidefinite program (SDP)

$$\min \left\{ c^T x \mid x \in N_+(K) \right\}. \tag{2}$$

Defining $\tilde{c} := \begin{pmatrix} 0 \\ c \end{pmatrix}$ and using the above definitions, (1) can be written explicitly as

$$\begin{aligned}
\min \quad & \tilde{c}^T Y e_0 && && (3)\\
\text{s.t.} \quad & Y = Y^T && && (4)\\
& Y e_0 = \operatorname{diag}(Y) && && (5)\\
& Y e_i \in K && \forall \quad i = 1, \ldots, n && (6)\\
& Y(e_0 - e_i) \in K && \forall \quad i = 1, \ldots, n && (7)\\
& Y_{00} = 1. && && (8)
\end{aligned}$$

Likewise, (2) can be written as

$$\begin{aligned}
\min \quad & \tilde{c}^T Y e_0 \\
\text{s.t.} \quad & Y \text{ positive semidefinite} \\
& (4)\text{–}(8).
\end{aligned}$$

A couple of comments concerning (1) and (2) are in order. First, the constraints (6) and (7) imply $Y e_0 \in K$. Combined with (8), this in turn implies that each component of the 0-th column of Y is in $[0, 1]$. By (4), the same holds for the 0-th row of Y, which implies by (6) that all other components of Y are in $[0, 1]$. Hence, we may replace K with $\hat{K} := K \cap [0, 1]^{1+n}$ without affecting the optimal solution sets of (1) and (2).

Second, if K is defined by m constraints, including upper and lower bounds, then the LP described by (3)–(8) has $\mathcal{O}(n^2 + nm)$ constraints and $\mathcal{O}(n^2)$ variables. Consequently, solving this LP using, say, the simplex method or interior-point methods becomes too cumbersome even for problems with moderate n and m. This situation is further exacerbated when solving (2). In fact, using standard SDP methods, (2) will only be solvable for very small n and m. As a result, very

little research has been done on actually solving (1) and (2). Some computations of (2) for various 0-1 polytopes can be found in Dash (2001).

This second observation motivates us to investigate new optimization techniques for solving (1) and (2), and in particular, we are interested in applying decomposition methods for large-scale optimization. We first show how this can be done for (1). Unfortunately, all the constraints (4)–(8) are tightly linked, and so the problem does not immediately lend itself to decomposition. To partially overcome this obstacle, however, we introduce the matrix variable

$$Z = YQ \in \mathbb{R}^{(1+n)\times n} \qquad \text{where} \quad Q := \left(e_0 - e_1 \middle| e_0 - e_2 \middle| \cdots \middle| e_0 - e_n\right)$$

and reformulate (3)–(8) as

$$\begin{aligned} \min \quad & \tilde{c}^T Y e_0 \\ \text{s.t.} \quad & Y = Y^T, \quad Ye_0 = \operatorname{diag}(Y), \quad Z = YQ && (9) \\ & Ye_i \in \hat{K}, \quad Ze_i \in \hat{K} \quad \forall \quad i = 1, \ldots, n && (10) \\ & Y_{00} = 1. && (11) \end{aligned}$$

Note that K has been replaced with $\hat{K}$ in accordance with the first comment above. Furthermore, it is clear that the constraints (10) are separable over the columns of Y and Z but that these same columns are linked via the constraints (9).

A reasonable idea is to apply Lagrangian relaxation to the constraints (9), and in order to simplify notation, we denote (9) by the collection of linear equations $h(Y, Z) = 0$. Letting λ denote the vector of unconstrained dual multipliers for $h(Y, Z) = 0$, we obtain the Lagrangian relaxation

$$\begin{aligned} L(\lambda) := \min \quad & \tilde{c}^T Y e_0 + \lambda^T h(Z, Y) \\ \text{s.t.} \quad & Ye_i \in \hat{K} \quad \forall \quad i = 0, 1, \ldots, n && (12) \\ & Ze_i \in \hat{K} \quad \forall \quad i = 1, \ldots, n && (13) \\ & Y_{00} = 1. && (14) \end{aligned}$$

Note that we have added the constraint $Ye_0 \in K$, which is redundant for (9) and (10) but is included here in order to properly constrain the 0-th column of Y. It is now clear that $L(\lambda)$ is separable over the columns of Y and Z, and so to evaluate $L(\lambda)$ for any λ, we can simply solve $2n + 1$ separate linear optimizations over $\hat{K}$ (while respecting the simple constraint $Y_{00} = 1$). Furthermore, from standard LP theory, we know that the optimal value of

$$\max_{\lambda} \quad L(\lambda) \tag{15}$$

equals the optimal value of (1).

The semidefinite optimization (2) can be approached in a similar fashion, i.e., by introducing the auxiliary variable Z and then relaxing the linking constraints. However, we must also introduce a dual multiplier $S \in \mathcal{S}_+^{1+n}$ for the constraint that keeps Y positive semidefinite, which modifies the Lagrangian relaxation to read

$$\begin{aligned} L(\lambda, S) := \min \quad & \tilde{c}^T Y e_0 + \lambda^T h(Z, Y) - S \bullet Y \\ \text{s.t.} \quad & (12)\text{–}(14). \end{aligned}$$

so that the resulting Lagrangian optimization is

$$\sup_{\lambda,S} \left\{ L(\lambda, S) \mid S \in \mathcal{S}_+^{1+n} \right\}. \tag{16}$$

It is well-known that the dual SDP of (2) has an interior-point, i.e., it satisfies Slater's condition, which implies that there is no duality gap and that optimality is attained in (2) — although optimality may not be attained in the dual of (2). As a result, it is not difficult to see that the optimal value of (16) equals that of (2).

Theoretically, one can solve both (15) and (16) using a subgradient method. Our initial experiments, however, indicated that convergence of subgradient methods, including the volume algorithm of Barahona and Anbil (2000), was often slow. This motivated us to examine an augmented Lagrangian method, which proved to be more robust while still allowing us to exploit the structure inherent in (1) and (2). We discuss these issues in detail in Sections 3 and 4.

3 The Augmented Lagrangian Method for Linear Conic Programs

In this section, we discuss the specialization of the augmented Lagrangian method — a standard tool of nonlinear programming (NLP) — to the case of linear optimization over linear and conic constraints, of which problems (1) and (2) are particular examples. More specifically, let $C \subseteq \mathbb{R}^q$ be a closed, convex cone, and let

$$X := \{y \in \mathbb{R}^q \mid Ey = f,\ y \in C\}. \tag{17}$$

We consider the following generic problem throughout this section:

$$\min \left\{ c^T y \mid Ay = b,\ y \in X \right\}. \tag{18}$$

Here, $y \in \mathbb{R}^q$ is the optimization variable and $c \in \mathbb{R}^q$, $A \in \mathbb{R}^{p \times q}$, and $b \in \mathbb{R}^p$ are the data. We assume that an optimal solution exists and denote the optimal value by v^*.

We acknowledge that initially it may seem counterintuitive to apply a NLP algorithm to linear conic problems. In fact, we are aware of only one study (Poljak and Tret′jakov, 1972) which considers the augmented Lagrangian method in such a context. In this section, however, besides laying the groundwork for the augmented Lagrangian algorithm, we also hope to highlight the advantages of the method, which when combined with several computational ideas discussed in Section 4 make it a good choice for optimizing (1) and (2).

3.1 Lagrangian and Penalty Methods

The constraint structure of (18) is a convenient description for optimization problems in which X has a simple structure and the constraints $Ay = b$ are difficult to enforce. A typical tool for optimizing (18) in such a case is Lagrangian relaxation. One introduces dual multipliers $\lambda \in \mathbb{R}^p$ and relaxes the constraints $Ay = b$ into the objective, obtaining the Lagrangian dual optimization

$$\sup_{\lambda \in \mathbb{R}^p} \min \left\{ c^T y + \lambda^T (b - Ay) \mid y \in X \right\}. \tag{19}$$

For each λ, the optimal value of the inner minimization is a valid lower bound on the optimal value of (18), and we assume further that the optimal value of (19) actually equals v^*. The appeal

of (19) is immediate; instead of handling $Ay = b$ directly, it is only necessary to optimize a linear function over X. Of course, one must optimize over X repeatedly while searching for an optimal λ. In practice, Lagrangian relaxation works well when optimization over X is especially simple and quick relative to (18).

The primary difficulty in optimizing (19) is that it is a non-differentiable optimization problem since the inner minimization generally has multiple optimal solutions. The most common approach for dealing with this lack of differentiability is the subgradient method (Polyak, 1967), which is briefly outlined for (19) as Algorithm 1. Hopefully the sequence of lower bounds associated with

Algorithm 1 Subgradient algorithm

Set $\lambda^1 = 0$
for $k = 1, 2, 3, \ldots$ **do**
  Calculate some $y^k \in \text{Argmin}\{c^T y + (\lambda^k)^T (b - Ay) : y \in X\}$
  Calculate the subgradient $d^k := b - Ay^k$
  Choose a step-size $\alpha_k > 0$
  Calculate $\lambda^{k+1} = \lambda^k + \alpha_k d^k$
end for

$\{\lambda^k\}$ will converge to v^*, and indeed, there exist rules for choosing the step-sizes α_k that guarantee this (Polyak, 1967). However, in practice, these step-size rules are usually abandoned for heuristic step-size strategies that improve the convergence rate and/or the quality of the lower bounds. Whatever the strategy for choosing α_k, the convergence of the subgradient method can often be slow and erratic. Another downside of the subgradient method is that in practice there is usually no clear stopping criterion, which ideally would be based on the duality gap between an iterate λ^k and a feasible y for the primal problem (18). Unfortunately, the auxiliary iterates $\{y^k\}$ generally do not satisfy the equation $Ay = b$, not even in the limit, and hence the algorithm does not have a primal feasible y at its disposal. There has been some work on producing primal information in subgradient-type algorithms; see Shor (1985); Sherali and Choi (1996); Barahona and Anbil (2000). Regardless of any disadvantages, the subgradient method is widely used because of its simplicity.

Another method, which is often used for optimizing nonlinear problems with hard constraints, is the quadratic penalty method. When applied to (18), the quadratic penalty method solves a sequence of problems, which are parameterized by a scalar $\sigma > 0$, having the general form

$$\min \left\{ c^T y + \frac{\sigma}{2} \|b - Ay\|^2 \mid y \in X \right\}. \tag{20}$$

The parameter $\sigma > 0$ is referred to as a penalty parameter because higher values of σ place larger penalties on infeasible points $y \in X$. The quadratic penalty method in the context of (18) is stated as Algorithm 2. It is a standard exercise (see, e.g., proposition 4.2.1 of Bertsekas (1995)) to show that every limit point of the sequence $\{y^k\}$ is a global minimum of (18), and it is moreover not difficult to see that the penalty subproblem optimal value is a valid lower bound on v^*. In addition, approximate dual solutions of (19) can be gotten from the optimality conditions of the subproblems. In this way, the quadratic penalty shares some of the nice properties of the subgradient method. An advantage of the quadratic penalty method over the subgradient method is that it produces a feasible y in the limit. On the other hand, in practice, it is usually necessary to increase the penalty parameter σ to levels that make the quadratic optimization over X highly ill-conditioned. In general NLPs, this can make calculating y^k a very slow and difficult task, and the same holds true for the quadratic problem (20).

Algorithm 2 Quadratic penalty algorithm

Set $\sigma_1 = 1$
for $k = 1, 2, 3, \ldots$ **do**
  Calculate some $y^k \in \operatorname{Argmin} \left\{ c^T y + \frac{\sigma_k}{2} \|b - Ay\|^2 : y \in X \right\}$
  Choose $\eta_k \gg 1$
  Calculate $\sigma_{k+1} = \eta_k \sigma_k$
end for

3.2 The Augmented Lagrangian Method

The augmented Lagrangian method can be seen as a combination of the subgradient and quadratic penalty methods. It is based on the following function, which is specified for fixed $\lambda \in \mathbb{R}^p$ and $\sigma \geq 0$ (typically $\sigma > 0$):

$$L_{\lambda,\sigma}(y) = c^T y + \lambda^T (b - Ay) + \frac{\sigma}{2} \|b - Ay\|^2 . \tag{21}$$

Note that $L_{\lambda,0}$ and $L_{0,\sigma}$ appear in (19) and (20), respectively. The augmented Lagrangian method is then stated as Algorithm 3. Roughly speaking, the augmented Lagrangian method runs the

Algorithm 3 Augmented Lagrangian algorithm

Set $\lambda^1 = 0$, $\sigma_1 = 1$
for $k = 1, 2, 3, \ldots$ **do**
  Calculate some $y^k \in \operatorname{Argmin} \left\{ L_{\lambda^k,\sigma_k}(y) : y \in X \right\}$
  Calculate the subgradient $d^k = b - Ay^k$
  Choose either ($\alpha_k = \sigma_k$ and $\eta_k = 1$) or ($\alpha_k = 0$ and $\eta_k \gg 1$)
  Calculate $\lambda^{k+1} = \lambda^k + \alpha_k d^k$ and $\sigma_{k+1} = \eta_k \sigma_k$
end for

subgradient and penalty methods at the same time by optimizing a combined function and then alternating between the update of λ and σ (typically using some pre-defined update strategy, such as performing the non-trivial update of σ every 10 iterations). Some of the main advantages of the augmented Lagrangian algorithm are that it yields both primal and dual solutions, as well as dual bounds, for (18).

There are also some additional, less obvious advantages of the augmented Lagrangian method over the subgradient and penalty methods. First, an important feature of the augmented Lagrangian, in contrast with the subgradient method, is that there is a definitive choice of step-size α_k in each iteration. This choice is dictated by convergence results for augmented Lagrangian methods (Bertsekas, 1995) and also seems to work well in practice. Second, it is well-known in the study of NLP that the introduction of explicit dual variables into the quadratic penalty method (as in Algorithm 3) tends to lessen the ill-conditioning encountered in penalty methods. In particular, it can be proved that if the iterates λ^k are sufficiently close to an optimal dual solution of (19), then there is a finite value of σ that still guarantees convergence (Bertsekas, 1995).

In fact, in the specific case of (18), where X is given by (17), it is possible to show a much stronger result, namely that Algorithm 3 converges even if σ_k is held constant at an arbitrary initial value $\sigma_1 > 0$. In order to state the result, we point out that the dual of (18) can be written explicitly as

$$\max \left\{ b^T \lambda + f^T \eta \mid A^T \lambda + E^T \eta + s = c,\ s \in C^* \right\}, \tag{22}$$

where

$$C^* := \left\{ s \in \mathbb{R}^q : s^T y \geq 0 \ \ \forall \ \ y \in C \right\}$$

is the dual cone of C, $\eta \in \mathbb{R}^m$ is the dual variable associated with the constraint $Ey = f$, and $s \in \mathbb{R}^q$ is the dual variable associated with the constraint $x \in C$. We also make the following assumptions: A has full row rank, and both (18) and (22) have Slater points, i.e., feasible points in the interior of C and C^*, respectively. In addition, we let η^k and s^k denote the optimal multipliers associated with $Ey = f$ and $y \in C$ gotten from the solution of the k-th augmented Lagrangian subproblem. The result is then as follows:

Theorem 3.1 *Let X be given by (17), and suppose that Algorithm 3 is executed so that $\sigma_k = \sigma_1$ for all $k \geq 1$, i.e., the dual multiplier is updated non-trivially in each iteration. Then any accumulation point of the combined sequence $\{(y^k, \lambda^{k+1}, \eta^k, s^k)\}$ constitutes a primal-dual optimal pair of (18) and (22).*

This result is proven by Poljak and Tret′jakov (1972) for the specific case of $C = \mathbb{R}^q_+$, and the essence of their proof carries over to general C. Although it is not difficult to extend the result based on their ideas, we include a proof in Section 3.5 for completeness.

We finish this subsection with a few observations. First, Theorem 3.1 shows that (1) and (2) can theoretically be solved without ever increasing σ, and computational experiments support this. In practice, however, it still may be advantageous to increase σ to moderate levels in order to facilitate convergence of the method, as we demonstrate in Section 4.

Second, even with the theoretical and practical benefits of augmented Lagrangian methods, it is still important to keep in mind that the inner optimization over $y \in X$ in each iteration of the algorithm utilizes a convex quadratic objective, instead of a linear one as in the subgradient method. In some applications, this is a disadvantage of the augmented Lagrangian method that may preclude its use, but we will show that, in the case of (1) and (2), the use of the augmented Lagrangian method is indeed beneficial.

Third, in practice it is rarely the case that the inner minimization of Algorithm 3 is computed exactly. Nonetheless, convergence is typically observed in practice even if y^k is a "nearly" optimal solution (Bertsekas, 1995). This behavior will guide some of our implementation decisions, which we describe in Section 4.

3.3 Variations on the Augmented Lagrangian Method

Typically the augmented Lagrangian method is stated in terms of handling difficult equality constraints, such as the constraints $Ay = b$ in (18). Although there exist variations which can handle inequality constraints directly (see Nocedal and Wright (1999)), a standard approach for handling inequalities is to simply add slack variables and then to revert to the equality case. For example, consider the problem

$$\min \left\{ c^T y \mid Ay \leq b, \ y \in X \right\}.$$

By introducing the variable $z \in \mathbb{R}^p_+$, we have the equivalent problem

$$\min \left\{ c^T y \mid Ay + s = b, \ (y, z) \in X \times \mathbb{R}^p_+ \right\}.$$

The augmented Lagrangian can now be applied directly to the second formulation. Of course, the inner optimization of the augmented Lagrangian algorithm is now slightly more complicated due to the addition of slack variables, but this complication is usually worth the trouble.

Further extensions of this idea can also be considered. If $Z \subseteq \mathbb{R}^p$ is an arbitrary set, a problem of the form

$$\min \left\{ c^T y \mid b - Ay \in Z,\ y \in X \right\}$$

can then be converted to

$$\min \left\{ c^T y : Ay + z = b,\ (y, z) \in X \times Z \right\},$$

after which the augmented Lagrangian method can be applied. Here again, the simplicity of the inner optimization over $(y, z) \in X \times Z$ is the key to the overall efficiency of the augmented Lagrangian algorithm.

3.4 Relationship with the Bundle Method

In Section 3.2, we have presented the augmented Lagrangian algorithm as an alternative to the standard subgradient algorithm. When the set X is convex, another well-known alternative to the subgradient algorithm is the bundle method (see Lemaréchal (1978), Kiwiel (1985)). Like the subgradient method, the bundle method uses subgradient information to produce a sequence $\{\lambda^k\}$ of dual multipliers whose corresponding Lagrangian objective values converge to v^*. However, the bundle method differs from the subgradient method in the precise way that the subgradient information is used.

The bundle method is initialized with $\lambda^1 = 0$ and, at the k-th iteration, the basic idea is to calculate λ^{k+1} by solving an approximation of (19). More specifically, the bundle method assumes that a current best point $\bar{\lambda}$ has been calculated (note that $\bar{\lambda}$ does not necessarily equal λ^k) and that a finite collection of dual points $\{\lambda^j \mid j \in J^k\}$ is available, for some finite index set J^k. For example, one may take $J^k = \{1, \ldots, k\}$ so that the dual points correspond to the dual solutions λ^j already produced by the algorithm in the first $k-1$ iterations, though other choices are possible. Defining $\tilde{X} := \text{conv}\{y^j : j \in J^k\}$, the approximation of (19) is then given as

$$\sup_{\lambda \in \mathbb{R}^p} \min \left\{ c^T y + \lambda^T (b - Ay) \mid y \in \tilde{X} \right\}. \tag{23}$$

Generally speaking, the inner minimization of (23) (viewed as a function of λ) is only considered a reliable approximation of the inner minimization of (19) for those λ relatively close to $\bar{\lambda}$. Hence, the next iterate λ^{k+1} is not actually chosen as an optimal solution of (23), but rather as an optimal solution of

$$\max_{\lambda \in \mathbb{R}^p} \min \left\{ c^T y + \lambda^T (b - Ay) \mid y \in \tilde{X} \right\} - \frac{\rho}{2} \left\| \lambda - \bar{\lambda} \right\|^2, \tag{24}$$

where $\rho > 0$ is a proximity parameter. In other words, (24) is similar to (23) except that it penalizes points that are too far from the current best iterate $\bar{\lambda}$, and the parameter ρ controls the precise amount of penalization. Once λ^{k+1} has been calculated, the bundle method calculates the value of the Lagrangian function at λ^{k+1} and decides whether or not λ^{k+1} should become the new best iterate $\bar{\lambda}$.

With this description of the bundle method, it is not difficult to see that the bundle method's procedure for computing λ^{k+1} is similar to that of the augmented Lagrangian algorithm. Indeed, (24) can be rearranged as

$$\min \left\{ \max_{\lambda \in \mathbb{R}^p} \ c^T y + \lambda^T (b - Ay) - \frac{\rho}{2} \left\| \lambda - \bar{\lambda} \right\|^2 \ \mid y \in \tilde{X} \right\}. \tag{25}$$

Since the inner optimization of (25) is a concave maximization over $\lambda \in \mathbb{R}^p$, its optimal solution is given by $\bar{\lambda} + \rho^{-1}(b - Ay)$, which further simplifies (25) to

$$\min \left\{ c^T y + \bar{\lambda}^T (b - Ay) + \frac{1}{2\rho} \| b - Ay \|^2 \mid y \in \tilde{X} \right\}. \tag{26}$$

Letting $\sigma = \rho^{-1}$, we now easily see that (26) is similar in form to the k-th augmented Lagrangian subproblem except that (26) approximates X by $\tilde{X}$. Next, once the bundle method calculates an optimal solution y^k of (26), λ^{k+1} is calculated by the formula

$$\lambda^{k+1} := \lambda^k + \rho^{-1} \left(b - Ay^k \right),$$

which matches the update formula used by the augmented Lagrangian algorithm.

As described above, in addition to the approximation $\tilde{X}$ of X, the bundle method differs from the augmented Lagrangian method in that it selectively keeps a current best iterate $\bar{\lambda}$ (which affects each stage of the algorithm), whereas the augmented Lagrangian algorithm simply generates each λ^k in succession. It is further interesting to note that the bundle method is known to converge for fixed ρ.

3.5 Proof of Theorem 3.1

In this subsection, we give the proof of Theorem 3.1, which has been stated in Section 3.2. We remark that our proof is an extension of the proof given by Poljak and Tret′jakov (1972) for the case $C = \mathbb{R}^q_+$.

The main idea of the theorem is that, when X is given by (17), the augmented Lagrangian algorithm converges without ever increasing the penalty parameter σ. For this, we consider the value $\sigma > 0$ to be fixed throughout the execution of Algorithm 3, i.e., $\sigma_k = \sigma$ for all $k \geq 1$. Also recall the following assumptions: A has full row rank, and (18) and (22) have Slater points. We will investigate three sequences produced by the algorithm:

(i) the sequence $\{y^k\}$ of primal estimates;

(ii) the shifted sequence $\{\lambda^{k+1}\}$ of dual multipliers; note that λ^{k+1} is calculated as a result of the k-th augmented Lagrangian subproblem;

(iii) the sequence $\{(\eta^k, s^k)\}$ of optimal multipliers for the constraints $Ey = f$ and $y \in C$ in the sequence of augmented Lagrangian subproblems.

Because (18) and (22) each have a Slater point, strong duality holds and there exists a primal-dual solution (y, λ, η, s) that satisfies $s^T y = 0$. The k-th augmented Lagrangian problem (with fixed σ) is

$$\min \left\{ c^T y + (\lambda^k)^T (b - Ay) + \frac{\sigma}{2} \|b - Ay\|^2 \mid Ey = f,\ y \in C \right\}, \tag{27}$$

and its dual (see Dorn (1960/1961) for QP duality in the case of $\mathbb{R}^q_+$) can be stated as

$$\begin{aligned} \max \quad & b^T \lambda^k + f^T \eta + \frac{\sigma}{2} \left(b^T b - v^T A^T A v \right) \\ \text{s.t.} \quad & A^T \left(\lambda^k + \sigma(b - Av) \right) + E^T \eta + s = c \\ & s \in C^*. \end{aligned} \tag{28}$$

Since (18) has a Slater point, so does (27). Furthermore, since A has full row rank, we can use a Slater point from (22) to construct such a point for (28). As a result, strong duality also holds between (27) and (28), and there exists a primal-dual solution (y, v, η, s) such that $v = y$ and $s^T y = 0$.

We first show that $Ay^k \to b$ via two lemmas and a proposition.

Lemma 3.1 *Let $\bar{\lambda}$ and $\hat{\lambda}$ be arbitrary multipliers for $Ax = b$, and let $\bar{y}$ and $\hat{y}$ be optimal solutions of the corresponding augmented Lagrangian subproblems. Then*

$$\left(\bar{\lambda} - \hat{\lambda}\right)^T (A\bar{y} - A\hat{y}) \geq \sigma \|A\bar{y} - A\hat{y}\|^2.$$

Proof. Optimality of $\bar{y}$ with respect to $\bar{\lambda}$ implies

$$(c - A^T(\bar{\lambda} + \sigma(b - A\bar{y})))^T(y - \bar{y}) \geq 0$$

for all y such that $Ey = f, y \in C$. Likewise for $\hat{y}$ and $\hat{\lambda}$:

$$(c - A^T(\hat{\lambda} + \sigma(b - A\hat{y})))^T(y - \hat{y}) \geq 0.$$

Applying these results with $y = \hat{y}$ and $y = \bar{y}$, respectively, summing the two resultant inequalities, and rearranging terms, we achieve the result. ∎

Lemma 3.2 *Let (λ^*, η^*, s^*) be an optimal solution of (22). Then any $y \in X$ is optimal for the augmented Lagrangian subproblem corresponding to λ^* if and only if y is optimal for (18).*

Proof. Using dual feasibility, we have that

$$c^T y + (\lambda^*)^T(b - Ay) + \frac{\sigma}{2}\|b - Ay\|^2 = (\lambda^*)^T b + f^T \eta^* + (s^*)^T y + \frac{\sigma}{2}\|b - Ay\|^2$$

Hence, ignoring the constant terms $b^T\lambda^* + f^T\eta^*$, the minimum value attainable by the augmented Lagrangian function is clearly bounded below by 0. Moreover, 0 is attained if and only if $(s^*)^T y = 0$ and $Ay = b$, which proves the result. ∎

Proposition 3.1 *The sequence $\{Ay^k\}$ converges to b.*

Proof. Let (λ^*, η^*, s^*) be any optimal solution of (22), and let y^* be any optimal solution of (18). For all $k \geq 1$,

$$\begin{aligned}
\|\lambda^{k+1} - \lambda^*\|^2 &= \|\lambda^k - \lambda^*\|^2 + 2\sigma(b - Ay^k)^T(\lambda^k - \lambda^*) + \sigma^2\|b - Ay^k\|^2 \\
&\leq \|\lambda^k - \lambda^*\|^2 - 2\sigma^2\|Ay^* - Ay^k\|^2 + \sigma^2\|b - Ay^k\|^2 \qquad \text{(by Lemmas 3.1 and 3.2)} \\
&= \|\lambda^k - \lambda^*\|^2 - \sigma^2\|b - Ay^k\|^2.
\end{aligned}$$

For arbitrary N, summing this inequality for $k = 1, \ldots, N$, we have

$$\begin{aligned}
0 &\leq \sum_{k=1}^{N} \left(\|\lambda^k - \lambda^*\|^2 - \sigma^2\|b - Ay^k\|^2 - \|\lambda^{k+1} - \lambda^*\|^2\right) \\
&= \|\lambda^1 - \lambda^*\|^2 - \|\lambda^{N+1} - \lambda^*\|^2 - \sigma^2 \sum_{k=1}^{N} \|b - Ay^k\|^2,
\end{aligned}$$

which implies $\sigma^2 \sum_{k=1}^{N} \|b - Ay^k\|^2 \leq \|\lambda^1 - \lambda^*\|^2$. Hence, because N is arbitrary, Ay^k must converge to b. ∎

Now, with Proposition 3.1 and an additional lemma, we prove Theorem 3.1.

Lemma 3.3 *For all $k \geq 1$, y^k and $(\lambda^{k+1}, \eta^k, s^k)$ are primal-dual optimal solutions of*

$$\min \left\{ c^T y \mid Ay = Ay^k,\ Ey = f,\ y \in C \right\}, \tag{29}$$

$$\max \left\{ (Ay^k)^T \lambda + f^T \eta \mid A^T \lambda + E^T \eta + s = c,\ s \in C^* \right\}. \tag{30}$$

Proof. Clearly, y^k is feasible for (29). Moreover, strong duality between (27) and (28) implies that (λ^k, η^k, s^k) is feasible for (28) such that $(s^k)^T y^k = 0$. Combining this with the definition $\lambda^{k+1} := \lambda^k + \sigma(b - Ay^k)$, we see that $(\lambda^{k+1}, \eta^k, s^k)$ is feasible for (30) and that strong duality holds between (29) and (30). This proves the result. ∎

Proof of Theorem 3.1. By Lemma 3.3, for each $k \geq 1$, $(y^k, \lambda^{k+1}, \eta^k, s^k)$ is a solution of the nonlinear system

$$\begin{aligned} &Ay = Ay^k,\ Ey = f,\ y \in C \\ &A^T \lambda + E^T \eta + s = c,\ s \in C^* \\ &y^T s = 0. \end{aligned}$$

By continuity, any accumulation point $(\bar{y}, \bar{\lambda}, \bar{\eta}, \bar{s})$ of $\{(y^k, \lambda^{k+1}, \eta^k, s^k)\}$ satisfies the above system with Ay^k replaced by its limit b (due to Proposition 3.1). In other words, $(\bar{y}, \bar{\lambda}, \bar{\eta}, \bar{s})$ is a primal-dual optimal solution for (18) and (22). ∎

4 Computational Issues and Results

In this section, we discuss the implementation details of the augmented Lagrangian algorithm used to solve (1) and (2). We then demonstrate the effectiveness of this approach on various problem classes and also illustrate the advantages of the augmented Lagrangian approach over other methods.

4.1 Optimizing over $N(K)$ and $N_+(K)$

We have suggested in Section 2 that one could consider a purely Lagrangian approach for solving the linear relaxation (1) since the calculation of $L(\lambda)$ is separable into $2n+1$ LPs over the columns of the variables Y and Z. We have argued in Section 3, however, that the augmented Lagrangian method has several advantages over the Lagrangian approach. In the case of (1), the k-th augmented Lagrangian subproblem can be stated as

$$\min \quad \tilde{c}^T Y e_0 + (\lambda^k)^T h(Z,Y) + \frac{\sigma_k}{2} \|h(Z,Y)\|^2 \tag{31}$$

$$\text{s.t.} \quad Ye_i \in \hat{K} \quad \forall \quad i = 0, 1, \ldots, n \tag{32}$$

$$Ze_i \in \hat{K} \quad \forall \quad i = 1, \ldots, n \tag{33}$$

$$Y_{00} = 1. \tag{34}$$

An important observation is that, in contrast with $L(\lambda)$, (31)–(34) is a nonseparable convex QP. More precisely, the quadratic term in the objective (31) is the sole cause of the nonseparability.

Even with this complication, we still advocate the use of the augmented Lagrangian method. To exploit the structure inherent in the constraints (32)–(34), we propose to employ block coordinate descent for solving the subproblem, iteratively taking a descent step over a particular column of Y

or Z, while keeping all other columns fixed. Block coordinate descent is known to be a convergent method; see proposition 2.7.1 in Bertsekas (1995).

When the amount of coupling between the blocks is small, block coordinate descent can be expected to converge quickly. One can observe that, because of the particular structure of $h(Y, Z)$, the amount of coupling between the columns of Y and Z is relatively small. In particular, the greatest amount of coupling is between Ye_i, Ze_i, and Ye_0 for $i = 1, \ldots, n$. As a result, we expect block coordinate descent to be a good choice for optimizing (31)–(34).

For optimizing over $N_+(K)$, we also advocate the augmented Lagrangian method, but at first glance, it is not clear how to handle the constraint that Y be positive semidefinite. This constraint is difficult not only because it involves positive semidefiniteness but also because it links the columns of Y. To handle this constraint, we follow the suggestions laid out in Section 3.3. In particular, we introduce an "excess" variable U, which is required to be symmetric positive semidefinite and must also satisfy $U = Y$, and simultaneously drop the positive semidefiniteness constraint on Y. After introducing a matrix S of Lagrange multipliers for the constraint $U = Y$, the resulting k-th augmented Lagrangian subproblem becomes

$$\begin{array}{ll} \min & \tilde{c}^T Y e_0 + (\lambda^k)^T h(Z, Y) + \dfrac{\sigma_k}{2}\|h(Y, Z)\|^2 + S^k \bullet (U - Y) + \dfrac{\sigma_k}{2}\|U - Y\|_F^2 \\ \text{s.t.} & (32)\text{–}(34), \;\; U \succeq 0. \end{array}$$

Again we propose to solve this problem using block coordinate descent. For example, we first fix U and solve a series of QPs as we did in the case (31)–(34), one for each column of Y and Z. We then proceed by fixing Y and Z and solving the subproblem over U. By completing the square, it is not difficult to see that the subproblem over U is equivalent to solving

$$\min \left\{ \left\| S^k + \sigma_k(U - Y) \right\|_F \;\; | \;\; U \succeq 0 \right\},$$

which in turn is equivalent to projecting $Y - \sigma_k^{-1} S$ onto the positive semidefinite cone. It is well-known that calculating the projection M_+ of a matrix M onto the positive semidefinite cone can be done by calculating the spectral decomposition $(M + M^T)/2 = QDQ^T$ and then forming the matrix $M_+ = QD_+Q^T$, where D_+ is derived from D by replacing all negative diagonal entries with 0.

Note that since S^k is the dual multiplier for the equality $U = Y$, it is unrestricted. However, from basic duality, we may restrict S^k to be symmetric positive semidefinite, enforcing this by projection onto the positive semidefinite cone after each dual update.

4.2 Implementation Details

The augmented Lagrangian algorithm for (1) and (2) has been implemented in ANSI C under the Linux operating system on a Pentium 4 having a 2.4 GHz processor and 1 GB of RAM. The pivoting algorithm of CPLEX 8.1 for convex QP (ILOG, Inc., 2002) has been employed for solving the $2n + 1$ quadratic subproblems encountered during block coordinate descent, and LAPACK (Anderson et al., 1999) has been utilized for the spectral decompositions required when projecting onto the positive semidefinite cone.

The choice of a pivoting algorithm for the quadratic subproblems — as opposed to an interior-point algorithm — was motivated by the warm-start capabilities of pivoting algorithms. In particular, it is not difficult to see that the objective functions of the $2n + 1$ quadratic subproblems

change only slightly between loops of block coordinate descent or between iterations of the augmented Lagrangian algorithm. As a result, the ability to warm-start from an advanced basis has proven to be invaluable for speeding up the overall algorithm.

Although Theorem 3.1 indicates that it is theoretically not necessary to increase the penalty parameter σ during the course of the algorithm, we have found that it is indeed advantageous to increase σ in order to enhance convergence. Our update rule is as follows:

> *Penalty update rule.* Every 500 iterations, σ is increased by a factor of 10.

We consider this to be a fairly conservative update rule.

Practically speaking, during the course of the algorithm, one can expect the norm of the constraint violation — $\|h(Y^k, Z^k)\|$ in the case of (1) and $\left(\|h(Y^k, Z^k)\|^2 + \|Y^k - U^k\|_F^2\right)^{1/2}$ in the case of (2) — to decrease towards 0, which is an indication that the algorithm is converging. As a result, we implement the following overall stopping criterion:

> *Stopping criterion.* The augmented Lagrangian algorithm is terminated once primal iterates are calculated such that the corresponding constraint violation is less than 10^{-6}.

We remark that, during early experiments on a handful of instances, this criterion was not achieved due to numerical difficulties caused by a large value of σ — typically around 10^8 or higher. As a result, we have also implemented the following:

> *Alternate stopping criterion.* The augmented Lagrangian algorithm is terminated once σ grows larger than 10^8.

Another implementation detail is how accurately the augmented subproblem is solved in each iteration. Recall from Section 3 that it is not theoretically necessary to solve each subproblem exactly, and in fact, we have found in practice that it often suffices to solve them fairly loosely. In the computational results presented in Section 4.4, our goal is to highlight the quality and speed of dual bounds provided by (1) and (2), rather than to calculate optimal solutions of high precision. In light of this goal, we decided to "solve" each augmented Lagrangian subproblem by performing exactly one cycle of block coordinate descent.

4.3 Problems

We have chosen three classes of 0-1 integer programs to illustrate the performance of the augmented Lagrangian algorithm.

4.3.1 Maximum stable set

Given an undirected, simple graph G with vertex set $V = \{1, \ldots, n\}$ and edge set $E \subseteq V \times V$, the (unweighted) maximum stable set problem is

$$\max \left\{ e^T x \mid x_i + x_j \leq 1,\ (i,j) \in E,\ x \in \{0,1\}^n \right\}.$$

As mentioned in Section 2, the maximum stable set problem has been studied extensively by Lovász and Schrijver (1991), and a number of theoretical results are known which illustrate the strength of (1) and (2).

We have collected a total of 13 graphs for testing; a basic description of these problems can be seen in Table 3. Five of the graphs were obtained from the Center for Discrete Mathematics and Theoretical Computer Science (DIMACS) and originated as test instances for the maximum clique problem in the Second DIMACS Implementation Challenge. As such, we actually use the complement graphs as instances of the maximum stable set problem. The remaining eight graphs were randomly generated with different numbers of vertices and edges.

4.3.2 Problem of Erdös and Turán

We consider a 0-1 integer programming formulation of a problem studied by Erdös and Turán: calculate the maximum size of a subset of numbers in $\{1, \ldots, n\}$ such that no three numbers are in arithmetic progression. This number is the optimal value of

$$\max \left\{ e^T x \mid x_i + x_j + x_k \leq 2,\ i + k = 2j,\ i < j < k,\ x \in \{0,1\}^n \right\}.$$

For a full discussion, we refer the reader to Dash (2001), which includes background on the problem as well as some active set approaches for approximately optimizing (2) in this case. In the computational results, we consider 10 instances for $n = 60, 70, \ldots, 150$. It is interesting to note that the number of constraints in $N(K)$ and $N_+(K)$ for $n = 150$ is approximately 1.7 million.

4.3.3 Quadratic assignment

Besides 0-1 linear integer programs, the lift-and-project relaxations provided by Lovász and Schrijver can easily be applied to 0-1 QPs with linear constraints, and so we also consider the quadratic assignment problem (QAP), which is a problem of this type arising in location theory.

Because of its difficulty, QAP has attracted a large amount of attention and study; see QAPLIB, Burkard et al. (1991) and the recent survey by Anstreicher (2003). One of the most common forms of the QAP is the Koopmans-Beckmann form: given an integer n and matrices $A, B \in \mathbb{R}^{n \times n}$, the QAP is

$$\begin{aligned}
\min \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} a_{ij} b_{kl} x_{ik} x_{jl} \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_{ik} = 1 \quad \forall \quad k = 1, \ldots, n \\
& \sum_{k=1}^{n} x_{ik} = 1 \quad \forall \quad i = 1, \ldots, n \\
& x_{ik} \in \{0,1\} \quad \forall \quad i, k = 1, \ldots, n.
\end{aligned}$$

We have taken 91 test instances from QAPLIB, and all instances in QAPLIB are in Koopmans-Beckmann form.

In the effort to solve QAP to optimality, a variety of dual bounds have been developed for QAP. In particular, we will compare with three bounds from the literature: (i) the Gilmore-Lawler bound (denoted GLB) (Gilmore, 1962; Lawler, 1963); the LP bound (denoted KCCEB) of Karisch et al. (1999); and the semidefinite programming bound (denoted RSB) of Rendl and Sotirov (2003). It is known that KCCEB is stronger than GLB, and it has been observed that RSB is stronger than KCCEB. As one might expect, however, RSB requires the most time to compute, while GLB takes the least.

The bound KCCEB is based on the first-level reformulation-linearization technique of Sherali and Adams (1990) applied to QAP. It is not difficult to see that this relaxation is equivalent to (1), and so the bound provided by (1) and KCCEB are theoretically equal, although slight differences are observed in practice due to computational differences such as the level of precision to which the relaxation is solved.

The derivation of RSB is based on similar lift-and-project ideas as (2). However, RSB selectively includes certain constraints that are implied by $N_+(K)$, while adding in additional constraints that are not implied by $N_+(K)$ (at least not implied explicitly). It is currently unclear whether one bound is theoretically stronger than the other, although our computational results in the next subsection show that the Lovász-Schrijver bound is stronger than RSB on all test instances.

One additional comment regarding QAP is in order. Since QAP has equality constraints and since upper bounds on the variables are redundant, it is possible to show that the constraints (7) of (1) and (2) are implied by (6). As a result, in this instance it is unnecessary to introduce the variable Z, which has the benefit of reducing the number of quadratic subproblems in the block coordinate descent from $2n + 1$ to $n + 1$. We have implemented these savings in the code and remark that the calculation of KCCEB has taken this into account as well.

4.4 Results

We first provide a comparison of our implementation with existing methods on a few problems selected from our test instances. We feel that these comparisons provide a fair indication of the advantages of our method, in terms of both bound quality and computation time. Next, we provide detailed information on the performance of our method on the three problem classes discussed above.

4.4.1 Comparison with linear and semidefinite solvers

To confirm the effectiveness of our approach, we solved some instances of (1) directly using the default LP algorithm available in CPLEX 8.1 (ILOG, Inc., 2002). Together with running times and bounds from our augmented Lagrangian approach, we show these results in Table 1. Note that we include stable set and Erdös-Turán instances, and that the instances are ordered roughly in terms of increasing size. Problems marked with a ($\star$) exceeded a preset time limit of 15,000 seconds. The bound shown in those cases is an upper bound since the dual simplex algorithm was used. We did not solve any larger instances with CPLEX as these often required more than the available memory. For very large problems, these results clearly indicate the ability of our method to obtain bounds from $N(K)$ in much less time than is required by standard LP solvers.

We also attempted to carry out some optimizations of $N_+(K)$ using standard semidefinite solvers, such as CSDP developed by Borchers (1999), but found that all but the smallest of instances would require more than the 1 GB of available memory on our computer.

4.4.2 Comparison with subgradient methods

In this subsection, we present some of the computational evidence which led us to investigate the use of an augmented Lagrangian method, as opposed to a subgradient technique. We found that subgradient methods are much more sensitive to the choice of initial dual multipliers than the augmented Lagrangian method. Some experimentation was required to find appropriate starting duals. We implemented a subgradient approach to optimize $N(K)$ using the Volume Algorithm

Table 1: Comparisons with CPLEX 8.1 Dual Simplex

Instance	Aug Lagrangian Bound	Aug Lagrangian Time (s)	Dual Simplex Bound	Dual Simplex Time (s)
johnson08-4-4	23.33	100.75	23.33	76.88
johnson16-2-4	40.00	852.11	40.00	1226.26
rand-050-05	28.00	7.69	28.00	0.81
rand-050-10	22.00	19.65	22.00	4.53
rand-050-50	16.67	49.08	16.67	4.59
rand-075-05	35.22	189.09	35.20	37.83
rand-075-50	25.00	271.45	25.00	66.84
rand-100-05	43.00	320.39	43.00	445.94
rand-200-05	75.11	5367.29	82.98	$\star$15000.00
Erdös-Turán ($n = 60$)	34.67	354.00	34.29	2349.68
Erdös-Turán ($n = 70$)	40.74	778.01	40.00	10625.88
Erdös-Turán ($n = 80$)	46.74	1685.59	45.85	$\star$15000.00
Erdös-Turán ($n = 90$)	53.07	2549.53	52.93	$\star$15000.00
Erdös-Turán ($n = 100$)	59.65	2809.58	60.38	$\star$15000.00

Table 2: Comparisons with Subgradient Method

Instance	Aug Lagrangian Bound	Aug Lagrangian Time (s)	Subgradient Bound	Subgradient Time (s)
brock200-1	67.18	2884.01	67.54	7461.49
c-fat200-1	66.78	12639.58	67.20	28068.53
johnson16-2-4	40.00	392.22	40.02	796.53
keller4	57.80	1291.71	58.08	4507.11
rand-200-05	75.49	878.57	75.49	6325.93

developed by Barahona and Anbil (2000), for which an open source framework is available from the COIN-OR repository (Lougee-Heimer, 2003, see `www.coin-or.org`). We report the results in Table 2. After some experimentation, we found some initial duals that seemed to work fairly well for the Volume Algorithm. We used these same starting duals for both algorithms. Note that the columns for subgradient report the final bound and total running time for the Volume Algorithm, whereas the columns for augmented Lagrangian report the first bound found that is better than the bound from the Volume Algorithm. The table demonstrates the ability of the augmented Lagrangian algorithm to obtain tighter bounds more quickly.

We also experimented with our own implementation of a subgradient method to optimize over $N_+(K)$ but found that standard stepsize strategies for subgradient methods did not seem appropriate for the positive semidefinite multiplier S. Consequently, convergence was difficult to achieve in this case. Motivated by these problems, we instead turned to the augmented Lagrangian approach.

Table 3: Results on Maximum Stable Set Problem

					Bound		Time (s)	
name	$\|V\|$	$\|E\|$	α	ϑ	N	N_+	N	N_+
brock200-1	200	5066	21	27.7	66.6668	27.9874	6,980	33,776
c-fat200-1	200	18366	12	12.3	66.6667	‡ 14.9735	29,204	118,845
johnson08-4-4	70	560	14	14.0	23.3333	14.0076	101	637
johnson16-2-4	120	1680	8	8.0	40.0001	‡ 10.2637	852	4,289
keller4	171	5100	11	14.1	57.0001	‡ 15.4119	5,838	27,983
rand-050-05	50	50	28	28.0	28.0000	28.0000	8	27
rand-050-10	50	107	22	22.1	22.0000	22.0000	20	32
rand-050-50	50	590	8	8.8	16.6668	8.2330	49	610
rand-075-05	75	134	35	35.8	35.2200	35.1163	189	237
rand-075-50	75	1364	9	9.5	25.0000	9.6441	271	2,075
rand-100-05	100	242	43	43.8	43.0000	43.1602	320	569
rand-200-05	200	982	64	71.0	75.1087	72.7410	5,367	5,185
rand-200-50	200	10071	11	14.7	66.6671	‡ 17.1173	17,554	77,321

4.4.3 Maximum stable set

In Table 3, we give the dual bounds obtained by the augmented Lagrangian algorithm for the maximum stable set instances. The bounds and times (in seconds) to achieve those bounds are listed under N for (1) and N_+ for (2). When prefixed to the bound, the symbol (‡) indicates that the alternate stopping criterion for our method was enforced, i.e., σ had grown too large before primal feasibility had been obtained.

In order to gauge the quality of the bounds in Table 3, we also include the size α of the maximum stable set (either gotten from the literature or computed using the IP solver of CPLEX) as well as the Lovász ϑ number of the graph (gotten by the algorithm of Burer and Monteiro (2003)). The number ϑ is a polynomial-time computable upper bound on α introduced by Lovász (1979), which can be computed by solving a certain semidefinite program. Theoretically, the N_+ bound is at least as strong as ϑ, but computationally, ϑ takes much less time to compute (a reasonable estimate is roughly one order of magnitude less).

The results indicate that, at least on a number of problems in this sample of graphs, the N_+ bound is significantly tighter than the N bound, but evidence indicates that N_+ is not much tighter than ϑ. Note that the computed N_+ bound is actually higher than ϑ on several instances, and this difference is most striking on those that were terminated according to the alternate stopping criterion. Overall, this seems to indicate that, in the specific case of the maximum stable set problem, it may not be worth the additional time to compute the N_+ bound instead of ϑ.

However, we stress that our overall intention is to show that (1) and (2) can be solved for general 0-1 integer programs. Accordingly, Table 3 serves to demonstrate that, given a specific problem, our method allows one to compute the N and N_+ bounds and hence to evaluate their quality.

4.4.4 Problem of Erdös and Turán

Table 4 lists the results of our algorithm on the Erdös-Turán instances, and the details of the table are the same as for Table 3. Note that we do not list the optimal values of the underlying IPs,

Table 4: Results on Problem of Erdös and Turán

	Bound		Time (s)	
n	N	N_+	N	N_+
60	34.6688	31.4183	354	653
70	40.7446	36.6120	778	1,062
80	46.7352	41.6060	1,686	1,676
90	53.0720	46.5339	2,550	3,164
100	59.6474	51.8541	2,810	4,295
110	65.8540	57.0635	5,387	6,773
120	71.1935	62.1836	10,932	9,659
130	77.2897	67.2867	11,023	13,459
140	82.9586	72.3553	21,164	18,239
150	89.5872	77.2238	34,180	23,450

since they are unknown for values of n greater than 60. As a result, it is difficult to directly asses the quality of these bounds. However, two things are interesting to note. First, the N_+ bounds are a significant improvement over the bounds reported by Dash (2001) (for example, Dash gives a bound of 87.6 for $n = 150$). Second, the times for computing the N and N_+ bounds are not dramatically different from one another, and in fact, on some of the largest problems, the N_+ bound actually takes less time to compute. Hence, on problems of this type, one might favor computing (2) over (1).

4.4.5 Quadratic assignment

Tables 5 and 6 list our results on the quadratic assignment instances, and the details of the table are similar to Table 3, except for the following: (i) the best-known feasible value for the QAP is listed such that, if the symbol (†) is *not* prefixed, then the feasible value is actually optimal, while (†) is present when the value is not known to be optimal; (ii) instead of listing dual bounds, we give optimality gaps, i.e.,

$$\text{gap} = \frac{\text{feas val} - \text{bound}}{\text{feas val}} \times 100\%.$$

A missing entry from the table (applicable only in the case of KCCEB and RSB) indicates that the gap was not available in the literature. In addition, note that the number contained in the names of the QAP instances is the size n of the QAP; for example, the instance *bur026a* has $n = 26$.

Though theory predicts that the KCCEB and N gaps should equal one another, we do see some discrepancies in Tables 5 and 6, probably because of numerical differences in the algorithms. Typically, N is slightly better than KCCEB, but in comparison with timings reported in Karisch et al. (1999), the calculation of N takes more time. We should point out, however, that the algorithm used to compute KCCEB exploits the structure of QAP to a great extent (more than just the reduction of $2n + 1$ subproblems to $n + 1$ mentioned previously) and does not appear to be generalizable to other 0-1 problems. On the other hand, the augmented Lagrangian method can be applied to any 0-1 problem.

The tables also indicate that the N_+ gap is significantly tighter than the RSB gap. In fact, on a number of relatively small problems, the N_+ gap is 0, indicating that (2) solves the QAP exactly. Previous to these results, RSB had provided the strongest known bounds for problems in

Table 5: Results on Quadratic Assignment Problem (I)

		Gap (%)					Time (s)	
NAME	FEAS VAL	GLB	KCCEB	RSB	N	N_+	N	N_+
bur026a	5,426,670	2.05	1.29		1.19	0.19	17,929	60,397
bur026b	3,817,852	2.70	1.69		1.57	0.22	18,248	53,749
bur026c	5,426,795	2.11	1.21		1.08	0.18	17,932	67,918
bur026d	3,821,225	2.87	1.64		1.51	0.21	18,064	69,804
bur026e	5,386,879	1.48	0.97		0.86	‡ 0.20	18,530	71,917
bur026f	3,782,044	1.99	1.27		1.14	0.10	18,195	78,748
bur026g	10,117,172	1.37	0.61		0.51	0.15	19,214	73,082
bur026h	7,098,658	1.77	0.75		0.63	0.09	18,385	70,939
chr012a	9,552	24.15	1.09		0.00	0.00	330	352
chr012b	9,742	26.65	0.00		0.00	0.00	293	363
chr012c	11,156	28.50	4.28		0.00	0.00	376	410
chr015a	9,896	43.16	11.65		4.35	0.14	1,415	1,461
chr015b	7,990	41.76	11.94		0.01	0.00	1,467	1,140
chr015c	9,504	35.13	3.80		0.00	0.00	901	1,188
chr018a	11,098	38.92	9.56		3.28	0.00	3,357	3,947
chr018b	1,534	0.00	0.00		0.00	0.13	1,277	6,239
chr020a	2,192	1.92	1.19		1.00	0.18	3,878	6,307
chr020b	2,298	4.44	1.70		0.87	0.13	4,030	9,778
chr020c	14,142	39.18	7.68		0.05	0.02	5,046	6,812
chr022a	6,156	3.77	0.58		0.24	0.03	6,762	12,711
chr022b	6,194	4.17	0.65		0.26	0.19	7,070	18,377
chr025a	3,796	27.16	4.98		0.53	0.37	13,901	33,402
els019	17,212,548	30.45	5.45		2.00	0.04	5,879	9,821
esc016a	68	44.12	39.71	13.24	29.41	5.88	452	1,195
esc016b	292	24.66	6.16	1.37	4.79	0.68	474	1,103
esc016c	160	48.13	43.13	11.25	26.25	3.75	538	1,981
esc016d	16	81.25	75.00	50.00	75.00	18.75	397	1,520
esc016e	28	57.14	57.14	17.86	50.00	3.57	349	1,318
esc016g	26	53.85	53.85	23.08	46.15	3.85	351	1,315
esc016h	996	37.25	29.32	2.61	29.32	1.91	618	1,369
esc016i	14	100.00	100.00	35.71	100.00	14.29	160	1,601
esc016j	8	87.50	75.00	12.50	75.00	0.00	345	1,331
esc032a	† 130	73.08			69.23	20.77	10,503	143,084
esc032b	† 168	42.86			42.86	21.43	5,308	130,393
esc032c	† 642	45.48			40.65	4.05	15,223	114,053
esc032d	† 200	47.00			44.00	4.50	10,767	117,556
esc032e	2	100.00			100.00	0.00	498	143,593
esc032f	2	100.00			100.00	0.00	496	144,820
esc032g	6	100.00			100.00	‡ 0.00	506	107,683
esc032h	† 438	41.32			33.79	3.20	15,031	140,406
had012	1,652	7.02	2.00	0.54	1.82	0.00	363	244
had014	2,724	8.52	2.31	0.33	2.13	0.00	818	1,092
had016	3,720	9.73	4.49	0.56	4.30	0.13	1,396	2,551
had018	5,358	10.86	5.23	0.77	5.06	0.11	2,518	4,966
had020	6,922	10.92	5.13	0.53	4.98	0.16	4,119	8,835

Table 6: Results on Quadratic Assignment Problem (II)

		Gap (%)					Time (s)	
NAME	FEAS VAL	GLB	KCCEB	RSB	N	N_+	N	N_+
kra030a	88,900	23.10	15.00	12.86	14.51	2.50	30,618	128,883
kra030b	91,420	24.45	16.61	11.23	16.10	4.07	30,648	136,187
kra032	88,700	24.02		10.19	16.06	3.51	40,543	225,053
lipa020a	3,683	0.00	0.00		0.00	0.00	1,337	3,699
lipa020b	27,076	0.00			0.00	0.00	1,556	4,276
lipa030a	13,178	0.00	0.00		0.01	0.02	20,584	121,748
lipa030b	151,426	0.00			0.00	0.00	10,783	77,868
nug012	578	14.71	9.86	3.63	9.52	1.73	315	469
nug014	1,014	15.98		2.17	8.97	0.39	837	1,093
nug015	1,150	16.26	10.17	2.43	9.48	0.78	1,043	1,543
nug016a	1,610	18.39	11.86	2.48	11.49	0.75	1,456	2,421
nug016b	1,240	17.58	12.74	4.19	12.26	1.69	1,338	2,143
nug017	1,732	19.86	13.51	3.64	13.05	1.44	1,932	3,379
nug018	1,930	19.48	14.20	4.04	13.89	1.92	2,393	4,932
nug020	2,570	19.96	15.45	4.63	15.14	2.49	3,772	7,577
nug021	2,438	24.82	17.64	4.72	17.10	2.46	5,150	13,791
nug022	3,596	30.95	21.19	4.34	20.88	2.34	6,110	18,074
nug024	3,488	23.28	18.09	5.10	17.75	2.61	8,581	25,887
nug025	3,744	23.37	18.16	5.58	17.76	3.29	10,457	31,082
nug027	5,234	29.29		5.14	21.63	2.33	14,406	69,574
nug028	5,166	26.71		5.13	21.58	2.92	16,501	81,564
nug030	6,124	25.39	21.86	5.24	21.60	3.10	21,762	127,011
rou012	235,528	14.12	5.09	5.03	4.79	0.11	566	759
rou015	354,210	15.71	8.64	5.91	8.30	1.13	1,377	2,027
rou020	725,522	17.31	11.59	8.50	11.36	4.19	5,112	10,997
scr012	31,410	11.31	5.96	6.65	5.07	0.00	489	496
scr015	51,140	12.52	5.07	4.51	3.70	0.00	1,403	1,197
scr020	110,030	30.23	14.12	13.66	13.58	3.88	5,182	10,564
ste036a	9,526	25.22	17.49		16.80	5.31	68,877	427,884
ste036b	15,852	45.41			30.65	7.61	73,431	358,981
ste036c	8,239,110	22.40			14.70	‡ 3.92	89,618	377,109
tai012a	224,416	12.70	1.61	0.73	1.02	0.00	563	414
tai012b	39,464,925	75.20	22.66		‡ 20.04	‡ 1.01	845	1,039
tai015a	388,214	15.64	9.34	6.04	9.12	2.86	1,402	2,022
tai015b	51,765,268	78.28	0.52		0.53	0.35	2,070	3,199
tai017a	491,812	16.08	10.23	8.23	10.01	3.11	2,521	4,414
tai020a	703,482	17.46	12.34	9.41	12.11	4.52	5,056	10,418
tai020b	122,455,319	88.40	24.44		‡ 23.06	‡ 4.11	7,981	15,591
tai025a	1,167,256	17.55	13.82	10.79	14.30	4.66	13,382	39,565
tai025b	344,355,646	86.15	56.93		‡ 55.82	‡ 11.70	16,855	65,262
tai030a	1,818,146	17.24	13.91	9.13	13.74	6.12	27,299	155,797
tai030b	† 637,117,113	93.57	78.46		‡ 78.46	‡ 18.47	31,333	247,114
tai035a	† 2,422,002	19.44	16.66		16.52	8.48	60,604	329,608
tai035b	† 283,315,445	88.49	64.05		‡ 66.40	‡ 15.42	63,888	430,914
tho030	† 149,936	39.59	33.40	9.26	32.84	4.75	28,180	99,265

QAPLIB. Only partial timing results are given by Rendl and Sotirov (2003), and so we are unable to make precise timing comparisons with RSB.

5 Conclusions

In this paper, we propose a novel method to apply dual decomposition to the lift-and-project relaxations of binary integer programs introduced by Lovász and Schrijver (1991). We believe that this is some of the first work that focusses on developing effective tools for solving these very large relaxations. Rather than using subgradient techniques to solve the dual, we show how to use an augmented Lagrangian technique to obtain bounds from these relaxations in both the LP and semidefinite case. We extend a result by Poljak and Tret′jakov (1972) to show that in the case of linear, conic programs, the augmented Lagrangian approach can use a constant penalty parameter and still guarantee convergence. Through extensive computational testing, we demonstrate the ability of this technique to outperform standard LP, SDP, and subgradient methods for various classes of problems. For some instances, such as QAP, the bounds computed from these relaxations are the tightest known to date.

As part of our future work in this area, we will study the possibility of using special purpose algorithms to solve the QP subproblems, especially in cases such as QAP where the constraints of the subproblems are simply a homogenization of the assignment polytope. We also intend to examine how these techniques may be used to yield tight relaxations of problems with a mix of binary and continuous variables and of continuous nonconvex QP's.

In addition to introducing some of the first effective solution techniques for linear and positive semidefinite lift-and-project relaxations, the success of this approach also demonstrates the applicability of augmented Lagrangian techniques even for linear, conic problems. We believe it will be interesting to investigate how well this technique performs on other large-scale linear, conic problems with block-angular structure.

References

E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

K. M. Anstreicher. Recent advances in the solution of quadratic assignment problems. *Mathematical Programming (Series B)*, 97(1-2):27–42, 2003.

E. Balas. Disjunctive programming. *Annals of Discrete Mathematics*, 5:3–51, 1979.

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

F. Barahona and R. Anbil. The volume algorithm: producing primal solutions with a subgradient method. *Math. Program.*, 87:385–399, 2000.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, first edition, 1995.

B. Borchers. CSDP, a C library for semidefinite programming. *Optim. Methods Softw.*, 11/12(1-4): 613–623, 1999.

S. Burer and R.D.C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming (Series B)*, 95:329–357, 2003.

R. E. Burkard, S. Karisch, and F. Rendl. QAPLIB — a quadratic assignment problem library. *European Journal of Operational Research*, 55:115–119, 1991.

S. Ceria and G. Pataki. Solving integer and disjunctive programs by lift-and-project. In R. E. Bixby, E. A. Boyd, and R. Z. Rios-Mercato, editors, *Lecture Notes in Computer Science*, volume 1412, pages 271–283. IPCO VI, 1998.

W. Cook and S. Dash. On the matrix-cut rank of polyhedra. *Mathematics of Operations Research*, 26(1):19–30, 2001.

S. Dash. *On the matrix cuts of Lovász and Schrijver and their use in Integer Programming*. PhD thesis, Rice University, 2001.

DIMACS. http://dimacs.rutgers.edu/Challenges/.

W. S. Dorn. Duality in quadratic programming. *Quart. Appl. Math.*, 18:155–162, 1960/1961.

P. C. Gilmore. Optimal and suboptimal algorithms for the quadratic assignment problem. *SIAM Journal on Applied Mathematics*, 10:305–313, 1962.

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

R. E. Gomory. An algorithm for integer solutions to linear programs. In R. Graves and P. Wolfe, editors, *Recent Advances in Mathematical Programming*, pages 269–302. McGraw-Hill, 1963.

ILOG, Inc. *ILOG CPLEX 8.1, User Manual*, 2002.

S. E. Karisch, E. Çela, J. Clausen, and T. Espersen. A dual framework for lower bounds of the quadratic assignment problem based on linearization. *Computing*, 63:351–403, 1999.

K. C. Kiwiel. *Methods of descent for nondifferentiable optimization*. Springer, Berlin, 1985.

J. B. Lasserre. An explicit exact SDP relaxation for nonlinear 0-1 program. In K. Aardal and A. M. H. Gerards, editors, *Lecture Notes in Computer Science*, volume 2081, pages 293–303. 2001.

M. Laurent. Tighter linear and semidefinite relaxations for max-cut based on the Lovász-Schrijver lift-and-project procedure. *SIAM Journal on Optimization*, 12:345–375, 2001.

M. Laurent. A comparison of the Sherali-Adams, Lovász-Schijver and Lasserre relaxation for 0-1 programming. *SIAM Journal on Optimization*, 28(3):345–375, 2003.

M. Laurent and F. Rendl. Semidefinite programming and integer programming. Technical report PNA-R0210, CWI, Amsterdam, April 2002. To appear as chapter of the *Handbook on Discrete Optimization* edited by K. Aardal, G. Nemhauser and R. Weismantel.

E. L. Lawler. The quadratic assignment problem. *Management Science*, 9:586–599, 1963.

C. Lemaréchal. Nonsmooth optimization and descent methods. Technical report, International Institute for Applied Systems Analysis, 1978.

L. Lipták and L. Tunçel. The stable set problem and the lift-and-project rank of graphs. *Mathematical Programming (series B)*, 98:319–353, 2003.

R. Lougee-Heimer. The Common Optimization INterface for Operations Research. *IBM Journal of Research and Development*, 47(1):57–66, 2003. See www.coin-or.org.

L. Lovász. On the Shannon Capacity of a graph. *IEEE Transactions of Information Theory*, IT-25(1):1–7, January 1979.

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

J. Nocedal and S. Wright. *Numerical Optimization*. Springer-Verlag, 1999.

B. T. Poljak and N. V. Tret′jakov. A certain iteration method of linear programming and its economic interpretation. *Èkonom. i Mat. Metody*, 8:740–751, 1972.

B. T. Polyak. A general method for solving extremum problems. *Soviet Mathematics Doklady*, 8: 593–597, 1967.

QAPLIB. http://www.seas.upenn.edu/qaplib/.

F. Rendl and R. Sotirov. Bounds for the quadratic assignment problem using the bundle method. Manuscript, University of Klagenfurt, August 2003.

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.

H. D. Sherali and W. P. Adams. *A Reformulation-Linearization Technique (RLT) for Solving Discrete and Continuous Nonconvex Problems*. Kluwer, 1997.

H. D. Sherali and G. Choi. Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs. *Oper. Res. Lett.*, 19(3):105–113, 1996.

N. Z. Shor. *Minimization methods for nondifferentiable functions*, volume 3 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1985. Translated from the Russian by K. C. Kiwiel and A. Ruszczyński.

T. Stephen and L. Tunçel. On a representation of the matching polytope via semidefinite liftings. *Mathematics of Operations Research*, 24(1):1–7, 1999.