SEMIDEFINITE PROGRAMMING RELAXATIONS for SET PARTITIONING PROBLEMS [*]

Henry Wolkowicz[†] and Qing Zhao[‡]

October 14, 1996
University of Waterloo
CORR Report 96-??

Abstract

We present a relaxation for the set partitioning problem that combines the standard linear programming relaxation with a semidefinite programming relaxation. We include numerical results that illustrate the strength and efficiency of this relaxation.

Contents

[*]This report is available by anonymous ftp at orion.uwaterloo.ca in the directory pub/henry/reports; or over WWW with URL ftp://orion.uwaterloo.ca/pub/henry/reports/ABSTRACTS.html

[†]University of Waterloo, Department of Combinatorics and Optimization, Faculty of Mathematics, Waterloo, Ontario, N2L 3G1 Canada

1 INTRODUCTION

We present a relaxation for the set partitioning problem (denoted SP) that combines the standard linear programming relaxation with a semidefinite programming (denoted SDP) relaxation. We include numerical results that illustrate the strength and efficiency of this relaxation.

1.1 Background

The set partitioning problem, SP, can be described as follows.

> Suppose we are given a set M with m elements; and let
>
> $$\mathcal{M} = \{M_j : j \in N := \{1, 2, \ldots, n\}\}$$
>
> be a given collection of subsets of M such that the union contains M, i.e. $\cup_{j \in N} M_j = M$. For each M_j, there is an associated cost $c_j \geq 0$. We want to find a subset F of the index set N such that:
>
> 1. the union still contains M, $\cup_{j \in F} M_j = M$;
> 2. the sets are pairwise disjoint, $M_k \cap M_j = \phi$, for $k \neq j \in F$;
> 3. and the sum of the costs $\sum_{i \in F} c_j$ is minimized.

Let $A = (a_{ij})$ be the $m \times n$ matrix with

$$a_{ij} = \begin{cases} 1 & \text{if element } \ i \in M_j \\ 0 & \text{otherwise.} \end{cases}$$

This matrix A is called the *incidence matrix* of the collection $\mathcal{M}$; each column of A is the indicator vector for the set M_j. Each subset $F \subset N$, for which the collection of sets $\{M_j, j \in F\}$ satisfies conditions 1 and 2, is called a *set partition* of the set M. For a given set partition, we let $x \in \{0,1\}^n$ be defined by

$$x_j = \begin{cases} 1 & \text{if } \ j \in F \\ 0 & \text{otherwise.} \end{cases}$$

Such an x represents the set partition.

The set partitioning problem can now be formulated as the following 0-1 integer programming problem

$$(SP) \qquad \begin{array}{rcl} \mu^* := & \min & c^t x \\ & \text{subject to} & Ax = e \\ & & x \in \{0,1\}^n, \end{array}$$

where e is the vector of ones. Without loss generality, we assume that the feasible set of (SP) is nonempty and that A has full row rank. For each $i \in \{1, 2, \ldots, m\}$, we let

$$a_i := (a_{i1}, a_{i2}, \ldots, a_{in}).$$

The i-th constraint, $a_i x = 1$, guarantees that the i-th element is in exactly one set.

The set partitioning problem has been extensively investigated because of its special structure and its numerous practical applications. The best known application is airline crew scheduling, see e.g. the recent reference [14]. Other applications include: truck scheduling; bus scheduling; facility location; circuit design and capital investment. (For applications and algorithms see e.g Garfinkel and Nemhauser [11], Marsten [16], Balas and Padberg [4], Balas [3], Nemhauser and Weber [17], Fisher and Kedia [10] Chan and Yano [6] and Hoffman and Padberg [15].)

Since the set partitioning problem is well-known to be NP-hard, many current approaches focus on finding a "near optimal" solution using various heuristic techniques. A natural candidate for generating a lower bound is the linear programming relaxation.

$$(SPLP) \qquad \begin{array}{rll} \mu^*_{LP} := & \min & c^t x \\ & \text{subject to} & Ax = e \\ & & x \geq 0. \end{array}$$

To improve the approximate solution for (SP), one can use cutting planes and/or branch-and-bound techniques in conjunction with various bound improvement techniques. In addition, various heuristics have been tried. (See Chu and Beasley [8] for a literature survey on exact and heuristic algorithms for (SP).) We include the following related papers in the bibliography [1, 2, 5, 12, 13, 18]

The latex bib file can be obtained over WWW or with anonymous ftp using URL: ftp://orion.uwaterloo.ca/pub/

In this paper, we develop an SDP relaxation for the set partitioning problem. Our approach is similar to that in [21, 22, 20]; i.e.: we derive a semidefinite relaxation from the dual of the dual of a quadratic constrained quadratic program formulation of (SP); we employ a "gangster operator" to efficiently model the 0-1 constraints in the relaxation; we project the feasible set onto the minimal face of the semidefinite cone in order to guarantee a constraint qualification; and we apply a primal-dual interior-point (p-d i-p) algorithm with an incomplete conjugate gradient method to solve the SDP relaxation. In addition, we combine the SDP relaxation with the standard LP relaxation and take advantage of block structures in the data.

2 SDP RELAXATION

To derive an SDP relaxation for SP, we reformulate the 0-1 integer programming model as a quadratically constrained quadratic programming problem. Since the variables x_i are restricted to 0-1, we have $x_i = x_i^2$, i.e.

$$x = x \circ x,$$

where $\circ$ denotes the Hadamard, or elementwise, product. In addition, since $a_i x = 1$, for each $i \in \{1, \ldots, m\}$, we have

$$\{k \neq j,\ a_{ik} = 1,\ a_{ij} = 1\} \Rightarrow x_k x_j = 0. \tag{2.1}$$

Therefore (SP) is equivalent to the following.

$$(SPQP) \qquad \begin{array}{rll} \mu^* = & \min & c^t(x \circ x) \\ & \text{subject to} & A(x \circ x) = e \\ & & (a_i x - 1)^2 = 0, \quad \text{for} \ \ i \in \{1, 2, \ldots, m\} \\ & & (x \circ x) - x = 0 \\ & & x_k x_j = 0, \ \text{if} \ k \neq j, \ \text{and} \ a_{ik} = a_{ij} = 1, \ \text{for some} \ i. \end{array}$$

By adding a scalar x_0, we can eliminate the linear terms (homogenize) in the existing constraints of the above problem.

$$
(SPQPH) \qquad
\begin{array}{rl}
\mu^* = \min & c^t(x \circ x) \\
\text{subject to} & A(x \circ x) = e \\
& (-1, a_i)(x_0, x^t)^t(x_0, x^t)(-1, a_i)^t = 0, \quad \text{for } \; i \in \{1, 2, \dots, m\} \\
& (x \circ x) - x_0 x = 0 \\
& x_k x_j = 0, \text{ if } k \neq j, \text{ and } a_{ik} = a_{ij} = 1, \text{ for some } i \\
& x_0^2 = 1.
\end{array}
$$

We now replace the quadratic terms with a matrix, i.e.we replace the rank one matrix $(x_0, x^t)^t(x_0, x^t)$ by the positive semidefinite matrix $Y \succeq 0$ with $Y \in \mathcal{S}_{n+1}$, the space of $n+1 \times n+1$ symmetric matrices. We get the following SDP relaxation.

$$
(PSDP) \qquad
\begin{array}{rll}
\mu^*_{SDP} := \min & \operatorname{trace} CY & \\
\text{subject to} & \operatorname{trace}\left(\operatorname{Diag}\left([0, a_i]\right)Y\right) = 1, & i = 1, \dots, m \\
& (-1, a_i)Y(-1, a_i)^t = 0, & i = 1, \dots, m \\
& \operatorname{arrow}(Y) = e_0 & \\
& \mathcal{G}_J(Y) = 0 & \\
& Y_{00} = 1 & \\
& Y \succeq 0, &
\end{array}
$$

where $C = \operatorname{Diag}\left([0, c^t]\right)$ is the diagonal matrix formed from the vector $[0, c^t]$, and the operator $\mathcal{G}_J$ is a *gangster operator*, i.e. $\mathcal{G}_J : \mathcal{S}_{n+1} \to \mathcal{S}_{n+1}$ shoots "holes" (or zeros) in a matrix. The ij component is defined as

$$
(\mathcal{G}_J(Y))_{ij} := \begin{cases} Y_{ij} & \text{if } (i,j) \text{ or } (j,i) \in J \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}
$$

where the set

$$
J := \{(k, j) : \text{if } k \neq j \text{ and } a_{ik} = a_{ij} = 1 \text{ for some } i\};
$$

the gangster operator is self-adjoint, $\mathcal{G}_J = \mathcal{G}_J^*$. The *arrow operator*, acting on the $(n+1) \times (n+1)$ matrix Y, is defined as

$$
\operatorname{arrow}(Y) := \operatorname{diag}(Y) - (0, (Y_{0,1:n^2})^t, \tag{2.3}
$$

where $Y_{0,1:n^2}$ is the vector formed from the last n^2 components of the first, or 0, row of Y and diag denotes the vector formed from the diagonal elements; e_0 is the first unit vector. The arrow constraint represents the 0,1 constraints by guaranteeing that the diagonal and 0-th row (or column) are identical; the gangster operator constraint represents the constraints in (2.1); and, finally, the assignment constraints $Ax = e$ are represented by the first two sets of constraints in (PSDP).

Define the $m \times (n+1)$ *assignment constraint matrix*

$$
T := [-e, A].
$$

Each feasible Y satisfies $Y \succeq 0$ and

$$
(-1, a_i)Y(-1, a_i)^t = 0, \; i = 1, \dots, m.
$$

Therefore the range space and null space satisfy

$$\mathcal{R}(T^t) \subset \mathcal{N}(Y) \text{ or alternatively } \mathcal{R}(Y) \subset \mathcal{N}(T).$$

Now let the null space of T be spanned by the columns of a $(n+1) \times (n-m+1)$ matrix V, i.e. let

$$\mathcal{N}(T) = \mathcal{R}(V).$$

This implies that $Y = VZV^t$ for some $Z = Z^t \succeq 0$, i.e. we are able to express each feasible Y as VZV^t. In order to solve large scale problems, a sparse representation of the null space of T is useful. We use a simple technique, called *Wolfe's variable-reduction* technique [19]. (For a "sparsest" representation, see e.g. [9].)

Without loss generality, we assume that

$$T = [T_B, T_N],$$

where T_B is a $m \times m$ matrix with full rank and T_N is a $m \times (n-m+1)$ matrix. Then, the matrix

$$V = \begin{bmatrix} -T_B^{-1}T_N \\ I_{n-m+1} \end{bmatrix}$$

satisfies $\mathcal{N}(T) = \mathcal{R}(V)$.

We now take a look at the following interesting properties of the matrix VZV^t.

Lemma 2.1 *For any arbitrary $(n-m+1) \times (n-m+1)$ symmetric matrix*

$$Z = \left[\begin{array}{c|ccc} Z_{00} & Z_{01} & \dots & Z_{0(n-m)} \\ \hline Z_{10} & Z_{11} & \dots & Z_{1(n-m)} \\ \vdots & \vdots & \ddots & \vdots \\ Z_{(n-m)0} & Z_{(n-m)1} & \dots & Z_{(n-m)(n-m)} \end{array}\right],$$

let $Y = VZV^t$ and write Y as

$$Y = \left[\begin{array}{c|ccc} Y_{00} & Y_{01} & \dots & Y_{0n} \\ \hline Y_{10} & Y_{11} & \dots & Y_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{n0} & Y_{n1} & \dots & Y_{nn} \end{array}\right].$$

Then:

a)

$$a_i Y_{1:n,0} = Y_{00}, \quad \textit{for} \quad i = 1, \dots, m;$$

b)

$$Y_{0j} = a_i Y_{1:n,j}, \ \textit{for} \ \ i = 1, \dots, m, \quad j = 1, \dots, n.$$

Proof. Since

$$TY = TVZV^t = 0,$$

we have, $(-1, a_i)Y = 0$, for each $1 \leq i \leq m$.

$\square$

This shows that the first two sets of constraints are redundant. Before we write our final SDP relaxation, we present another lemma which helps get rid of more redundant constraints.

Lemma 2.2 *Let $Y = VZV^t$ with $Y_{00} = 1$. Then*

$$\mathcal{G}_J(Y) = 0 \Rightarrow \text{arrow}\,(Y) = e_0.$$

Proof. Suppose $Y = VZV^t$ and $\mathcal{G}_J(Y) = 0$. Let $j \in \{1, 2, \ldots, n\}$. Then there exists $i \in \{1, 2, \ldots, m\}$ such that $a_{ij} = 1$. By Lemma 2.1, we have $(a_{i1}, \ldots, a_{in})Y_{1:n,j} = Y_{0j}$. This implies that

$$Y_{jj} + \sum_{\substack{k \\ k \neq j, a_{ik}=1}} Y_{kj} = Y_{0j}.$$

From the definition of the gangster operator, we have

$$\sum_{\substack{k \\ k \neq j, a_{ik}=1}} Y_{kj} = 0.$$

Therefore $Y_{jj} = Y_{0j}$.

$\square$

Now replacing Y by VZV^t in (PSDP) and getting rid of the redundant constraints, we have the following final SDP relaxation for SP. We let $J^0 = J \cup \{(0, 0)\}$.

$$(PSDPF) \quad \begin{array}{rll} \mu^*_{SDP} = & \min & \text{trace}\, V^tCVZ \\ & \text{subject to} & \mathcal{G}_{J^0}(VZV^t) = E_{00} \\ & & Z \succeq 0, \end{array}$$

where $Z \in \mathcal{S}_{n-m+1}$ and $C = \text{Diag}\,(0, c^t)$. The dual is

$$(DSDPF) \quad \begin{array}{ll} \max & W_{00} \\ \text{subject to} & V^t\mathcal{G}^*_{J^0}(W)V \preceq V^tCV. \end{array}$$

From Lemma 2.1 and Lemma 2.2, we can immediately derive the following.

Theorem 2.1 *Let Z be any feasible solution of (PSDPF). Then* $(\text{diag}\,(VZV^t))_{1:n}$ *(the last n diagonal elements of the matrix VZV^t), is a feasible solution of the linear programming relaxation (SPLP).*

Proof. Let Z satisfy the hypothesis and $Y = VZV^t$. Then $\mathcal{G}_{J^0}(Y) = E_{00}$ and

$$Y_{jj} \geq 0, \quad \text{for} \quad i \in \{1, \ldots, n\}.$$

From Lemma 2.1 and Lemma 2.2, we have $Y_{:,0} = \operatorname{diag}(Y)$ and $Y_{00} = 1$, and thus for each $i \in \{1, \dots, m\}$,

$$a_i(Y_{11}, \dots, Y_{nn})^t = a_i Y_{0j}^t = Y_{00} = 1.$$

$\square$

Based on the theorem above and the fact that the objective value of the SDP relaxation is $(0, c^t)\operatorname{diag}(VZV^t)$, the following corollary follows.

Corollary 2.1 *The lower bound given by the SDP relaxation $(PSDPF)$ is greater than or equal to the one given by the LP relaxation, i.e. $\mu^*_{SDP} \geq \mu^*_{LP}$.*

In addition, we now see that there is no duality gap between (PSDPF) and (DSDPF).

Theorem 2.2 *Problem (DSDPF) is strictly feasible.*

Proof. From Lemma 2.2, we have

$$\mathcal{G}_{J^0}(VZV^t) = 0 \;\Rightarrow\; \operatorname{arrow}(VZV^t) = 0.$$

Therefore

$$\mathcal{N}(\mathcal{G}_{J^0}(V \cdot V^t)) \subset \mathcal{N}(\operatorname{arrow}(V \cdot V^t)).$$

In other words, their adjoint operators satisfy

$$\mathcal{R}(V^t \operatorname{Arrow}(\cdot)V) \subset \mathcal{R}(V^t \mathcal{G}_{J^0}(\cdot)V).$$

Therfore for $y = -e \in \mathcal{R}^{n+1}$, there exists W such that

$$V^t \operatorname{Arrow}(-e_n)V = V^t \mathcal{G}_{J^0}(W)V$$

and, by using Schur complements, we see that

$$V^t \mathcal{G}_{J^0}(-\alpha E_{00} + W)V = V^t(-\alpha E_{00} - \operatorname{Arrow}(e_n))V \prec 0,$$

for α big enough. Therefore $\beta(-\alpha E_{00} + W)$ is strictly feasible for large enough β.

$\square$

From Theorem 2.2, we know that the dual problem satisfies the Slater condition. Therefore, there is no duality gap between the primal problem (PSDPF) and the dual problem (DSDPF) and, moreover, the primal optimal value is attained. However, the primal problem may not be strictly feasible.

Example 2.1 *Consider SP with constraints*

$$\begin{array}{llll} x_1 & & & = 1 \\ x_1 & +x_2 & +x_3 & +x_4 = 1 \\ x_1, & x_2, & x_3, & x_4 \geq 0. \end{array}$$

Observe that the feasible set is a singleton $(1,0,0,0)^t$. Note that for this problem $n = 4$ and $m = 2$, so V is a 5×3 matrix. Thus, for any feasible solution of its final SDP relaxation $Z \in \mathcal{P}_3$, the diagonal of VZV^t is $(1,1,0,0,0)^t$. This means that $\operatorname{rank}(VZV^t) \leq 2$, *which implies that* $\operatorname{rank}(Z) \leq 2$. *Therefore, the final SDP relaxation is not strictly feasible.*

	nrow	ncol	nzero	LP	SDP
small01	14	34	108	1864	1864*
small02	16	46	139	2259	2259*
small03	27	97	234	17327	18324
small04	33	192	584	4503	4503*
small05	44	277	770	21706	21706*
tiny04	6	27	72	1035	1091
tiny01	3	6	9	17.5	25.00
tiny05	7	35	70	1215	1257

Table 1: Numerical Results

3 NUMERICAL TESTS

The algorithm (a p-d i-p approach) we use to solve the SDP relaxation is very similar to the one in [21, 22, 20] for the quadratic assignment and graph partitioning problems. An incomplete conjugate gradient method is used to solve the large Newton equations that arise at each iteration of the algorithm.

As we have seen from the geometrical discussion above, the algorithm may have to deal with those problems whose primal SDP relaxation are not strictly feasible and whose dual SDP relaxation can not attain their optimal value. Since the main purpose of our algorithm is to find good lower bounds, we apply an infeasible primal-dual interior-point algorithm. Because the dual problem is strictly feasible and only has inequality constraints, the linesearch can easily maintain dual feasibility. Therefore, a lower bound can always be obtained from the dual objective value.

The purpose of our numerical tests is to illustrate that the lower bound given by our algorithm for the SDP relaxation is better than the one given by LP relaxation. In addition, after solving the relaxation, the diagonal of the matrix Y from the SDP relaxation satisfies the constraints of the linear programming relaxation.

Our numerical tests for small problems are based on real data for bus scheduling ????ref!!!??? problems. The results are summarized in Table 1. The columns under nrow, ncol and nzero are for the number of rows, columns and nonzero elements, respectively. The last two columns show the lower bounds by LP and SDP relaxations, respectively. A lower bounds marked with a star means that the lower bound is equal to the optimal objective value.

4 SDP RELAXATION FOR LARGE SPARSE PROBLEMS

4.1 An SDP Relaxation with Block Structure

As we see from the introduction, the set partitioning problems are usually derived from real world problems such as scheduling problems. These problems can be of very large size ($> 10,000$) and very sparse.

Currently, an approximate solution for a large size set partitioning problem can be obtained by solving a corresponding large sparse linear programming relaxation and the information from the primal and dual optimal solutions are used to decide which columns, or sets M_j, should be chosen for the partition. Since the diagonal of an SDP solution is a feasible solution of the LP relaxation, we expect that this solution can help in making the choices. On the other hand, it is hard to solve SDP problem of large size, e.g. over 10,000. In order to make SDP relaxation more competitive with LP to solve the large sparse problem, we have to find a way to exploit the sparsity of the set partitioning problem. In this section, we relax part of the variables of the set partitioning by SDP, while we treat the others with an LP relaxation.

Consider a large sparse set partitioning problem

$$\begin{array}{llll} & \mu^* = & \min & c^t x \\ (SPL) & & \text{subject to} & Ax = e \\ & & & x \in \{0,1\}^n. \end{array}$$

By permuting the rows and columns of A, we can rewrite A as

$$A = \left[\begin{array}{cccc|c} F_1 & 0 & \dots & 0 & 0 \\ 0 & F_2 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & F_k & 0 \\ \hline G_1 & G_2 & \dots & G_k & H \end{array}\right], \tag{4.4}$$

where for each $i \in \{1, \dots, k\}$, F_i is $m_i \times n_i$, G_i is $m_G \times n_i$, H is $m_G \times n_H$, and

$$m_1 + \ldots + m_k + m_G = m;\ n_1 + \ldots + n_k + n_H = n.$$

The sparsity pattern of the matrix A is illustrated in Figure 2.

Corresponding to each submatrix F_i, $i \in \{1, \dots, k\}$, we define

$$x_{B_i} = (x_{B_i}^1, \dots, x_{B_i}^{n_i})^t \text{ and } x_N = (x_N^1, \dots, x_N^{n_H})^t$$

such that

$$x = \left(\begin{array}{c} x_{B_1} \dots x_{B_k} x_N \end{array} \right)^t.$$

Similarly, we define

$$c_{B_i} = (c_{B_i}^1, \dots, c_{B_i}^{n_i})^t, \text{ and } c_N = (c_N^1, \dots, c_N^{n_H})^t$$

such that

$$c = \left(\begin{array}{c} c_{B_1} \dots c_{B_k} c_N \end{array} \right).$$

For each $i \in \{1, \dots, k\}$, we write

$$F_i = \begin{bmatrix} F_i^1 \\ \vdots \\ F_i^{m_i} \end{bmatrix} = \begin{bmatrix} F_i^{11} & \dots & F_i^{1n_i} \\ \vdots & \ddots & \vdots \\ F_i^{m_i 1} & \dots & F_i^{m_i n_i} \end{bmatrix}.$$

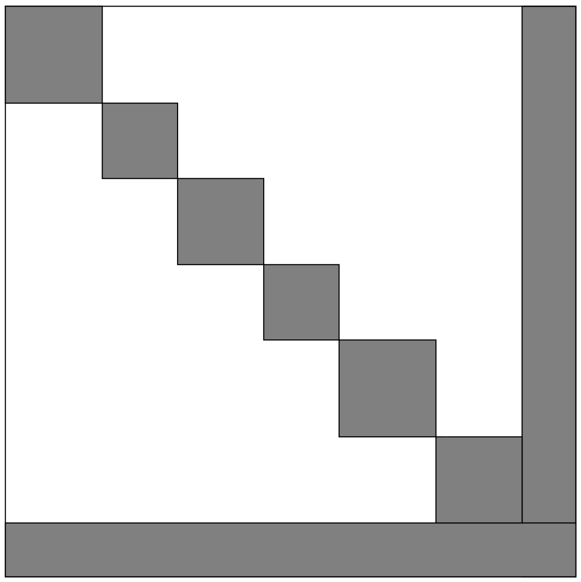

Figure 1: Sparsity Pattern of Matrix A

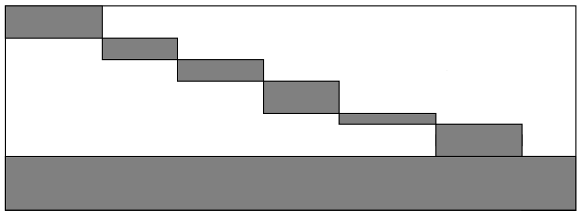

Figure 2: Sparsity Pattern

Similarly, for each $i \in \{1, \dots, k\}$, we write

$$G_i = \begin{bmatrix} G_i^1 \\ \vdots \\ G_i^{m_G} \end{bmatrix} = \begin{bmatrix} G_i^{11} & \dots & G_i^{1n_i} \\ \vdots & \ddots & \vdots \\ G_i^{m_G 1} & \dots & G_i^{m_G n_i} \end{bmatrix},$$

$$H = \begin{bmatrix} H^1 \\ \vdots \\ H^{m_G} \end{bmatrix} = \begin{bmatrix} H^{11} & \dots & H^{1n_H} \\ \vdots & \ddots & \vdots \\ H^{m_G 1} & \dots & H^{m_G n_H} \end{bmatrix}.$$

and define an index sets for gangster operators

$$J_i := \left\{ (p,q) : \ p < q \text{ for some } j \ \begin{array}{l} F_i^{jp} = F_i^{jq} = 1 \text{ or} \\ G_i^{jp} = G_i^{jq} = 1 \end{array} \right\}.$$

We rewrite $(SPLP)$ as

$$\begin{array}{rll} \mu^* = & \min & \sum_{i=1}^k c_{B_i}^t x_{B_i} + c_N^t x_N \\ & \text{subject to} & F_i x_{B_i} = e_{m_i}, \quad i \in \{1, \dots, k\} \\ & & G_1 x_{B_1} + \dots + G_k x_{B_k} + H x_N = e_{m_G} \\ & & x_{B_1}, \dots, x_{B_k}, x_N \in \{0,1\}^n. \end{array}$$

An equivalent quadratically constrained quadratic programming formulation can then be expressed as follows

$$\begin{array}{rll} \mu^* = & \min & \sum_{i=1}^k c_{B_i}^t x_{B_i} \circ x_{B_i} + c_N^t x_N \circ x_N \\ & \text{subject to} & F_i x_{B_i} \circ x_{B_i} = e_{m_i}, \\ & & (F_i^j x_{B_i} - 1)^2 = 0, \quad \text{for } \ j \in \{1, 2, \dots, m_i\}, \\ & & x_{B_i} \circ x_{B_i} - x_{B_i} = 0, \\ & & x_{B_i}^p x_{B_i}^q = 0, \quad \text{for any pair } (p,q) \in J_i, \\ & & \text{for } \ i \in \{1, \dots, k\} \\ & & G_1 x_{B_1} \circ x_{B_1} + \dots + G_k x_{B_k} \circ x_{B_k} + H x_N \circ x_N = e. \end{array}$$

By adding, for each $i \in \{1, \dots, k\}$, a scalar $x_{B_i}^0$, we homogenize the above problem as follows

$$\begin{array}{rll} \mu^* = & \min & \sum_{i=1}^k c_{B_i}^t x_{B_i} \circ x_{B_i} + c_N^t x_N \circ x_N \\ & \text{subject to} & F_i x_{B_i} \circ x_{B_i} = e_{m_i}, \\ & & (-1, F_i^j)(x_{B_i}^0, x_{B_i}^t)^t (x_{B_i}^0, x_{B_i}^t)(-1, F_i^j)^t = 0, \\ & & \text{for } \ j \in \{1, 2, \dots, m_i\}, \\ & & x_{B_i} \circ x_{B_i} - x_{B_i}^0 x_{B_i} = 0, \\ & & (x_{B_i}^0)^2 = 1, \\ & & x_{B_i}^p x_{B_i}^q = 0, \quad \text{for any pair } (p,q) \in J_i, \\ & & \text{for } \ i \in \{1, \dots, k\} \\ & & G_1 x_{B_1} \circ x_{B_1} + \dots + G_k x_{B_k} \circ x_{B_k} + H x_N \circ x_N = e. \end{array}$$

In the above quadratically constrained quadratic programming, we replace the rank-one matrix $(x^0_{B_i}, x^t_{B_i})^t(x^0_{B_i}, x^t_{B_i})$ by the matrix Y_i for each $i \in \{1, \dots, k\}$, and also $X_N X_N^t$ by Y_N. Then we obtain an SDP relaxation as follows

$$\begin{array}{rcl} \mu^*_{LR} := & \min & \sum_{i=1}^k c^t_{B_i}(\operatorname{diag}(Y_i))_{1:n_i} + c^t_N \operatorname{diag}(Y_N) \\ & \text{subject to} & F_i(\operatorname{diag}(Y_i))_{1:n_i} = e_{m_i}, \\ & & (-1, F_i^j)Y_i(-1, F_i^j)^t = 0, \quad \text{for } \; j \in \{1, 2, \dots, m_i\}, \\ & & \operatorname{arrow}(Y_i) = 0, \\ & & (Y_i)_{00} = 1, \\ & & \mathcal{G}_{J_i}(Y_i) = 0, \\ & & \text{for } \; i \in \{1, \dots, k\} \\ & & \sum_{i=1}^k G_i(\operatorname{diag}(Y_i))_{1:n_i} + H \operatorname{diag}(Y_N) = e, \\ & & Y_1 \succeq 0, \dots, Y_k \succeq 0, Y_N \succeq 0, \end{array}$$

where $Y_i \in \mathcal{P}_{n_i+1}$ for $i \in \{1, \dots, k\}$ and $Y_N \in \mathcal{P}_{n_H}$. Since the coefficient matrices for Y_N are all diagonal, we can always write $Y_N = \operatorname{Diag}(x)$, where $x \in \mathcal{R}^{n_H}, x \geq 0$. For each $i \in \{1, \dots, k\}$, we define an operator $\mathcal{A}_i : \mathcal{P}_{n_i+1} \to \mathcal{R}^{m_G}$ such that

$$\mathcal{A}_i(Y_i) := G_i(\operatorname{diag}(Y_i))_{1:n_i}.$$

Then we have the following equivalent problem.

$$\begin{array}{rcl} \mu^*_{LR} = & \min & \sum_{i=1}^k c^t_{B_i}(\operatorname{diag}(Y_i))_{1:n_i} + c^t_N x \\ & \text{subject to} & F_i(\operatorname{diag}(Y_i))_{1:n_i} = e_{m_i}, \\ & & (-1, F_i^j)Y_i(-1, F_i^j)^t = 0, \quad \text{for } \; j \in \{1, 2, \dots, m_i\}, \\ & & \operatorname{arrow}(Y_i) = 0, \\ & & (Y_i)_{00} = 1, \\ & & \mathcal{G}_{J_i}(Y_i) = 0, \\ & & \text{for } \; i \in \{1, \dots, k\} \\ & & \sum_{i=1}^k \mathcal{A}_i(Y_i) + Hx = e, \\ & & Y_{B_1} \succeq 0, \dots, Y_{B_k} \succeq 0, x \geq 0. \end{array}$$

For each $i \in \{1, \dots, k\}$, we construct a $(n_i + 1) \times (n_i - m_i + 1)$ matrix V_i such that the null space of $[-e_{m_i}, F_i]$ is spanned by the columns of V_i. We follow the same procedure as that in the above section, i.e., for $i \in \{1, \dots, k\}$, we replace Y_i by $V_i X_i V_i^t$ and get rid of the redundant constraints. We denote $C_i := \operatorname{Diag}(0, c^t_{B_i})$. Note that $c^t_{B_i}(\operatorname{diag}(Y_i))_{1:n_i} = \operatorname{trace}(\operatorname{Diag}(0, c^t_B)Y_i)$. Then we have the following final SDP relaxation.

$$(LPSDPF) \qquad \begin{array}{rcl} \mu^*_{LR} & = & \min & \sum_{i=1}^k \operatorname{trace} V_i^t C_i V_i X_i + c^t_N x \\ & & \text{subject to} & \sum_{i=1}^k \mathcal{A}_i(V_i X_i V_i^t) + Hx = e_{m_G} \\ & & & \mathcal{G}_{J_i^0}(X_i) = E^i_{00}, \quad \text{for } \; i \in \{1, \dots, k\} \\ & & & X_1 \succeq 0, \dots, X_k \succeq 0, \quad x \geq 0, \end{array}$$

where, for $i \in \{1, \dots, k\}$, $X_i \in \mathcal{P}_{n_i - m_i + 1}$ and the operator $\mathcal{G}_{J_i^0}$ is a gangster operator with

$$J_i^0 := \left\{ (p, q) : p < q \text{ for some } j \; \begin{array}{c} F_i^{jp} = F_i^{jq} = 1 \text{ or} \\ G_i^{jp} = G_i^{jq} = 1 \end{array} \right\} \cup (0, 0).$$

Observe that in the final SDP relaxation $(LSPRP)$ there are semidefinite matrix variables and nonnegative vector variables as well. Thus, we call the final SDP relaxation a *mixed LP-SDP* relaxation.

Its dual is

$$(LSPRD) \qquad \begin{array}{rl} \max & \sum_{i=1}^{m_G} \lambda_i + \sum_{i=1}^{k} (Y_i)_{00} \\ \text{subject to} & V_i^t(\operatorname{Diag}(0, \lambda^t G_i) + Y_i)V_i \preceq V_i^t C_i V_i, \\ & Y_i \in \mathcal{S}_{J_i^0}, \\ & \text{for } \ i \in \{1, \ldots, k\} \\ & H^t \lambda \leq c, \end{array}$$

where for $i \in \{1, \ldots, k\}$, Y_i and λ_i are dual variables.

For each feasible solution of $(LSPRP)$ $(X_1, \ldots, X_k, x)$, we construct a $n \times 1$ vector

$$y = \begin{pmatrix} y_1 \\ \vdots \\ y_k \\ x \end{pmatrix}, \tag{4.5}$$

where $y_i = (\operatorname{diag}(V_i X_i) V_i^t)_{1:n_i}$, for $i = 1, \ldots, k$. Applying the Theorem 2.1 to each block, we have $F_i y_i = e_{m_i}$ for $i = 1, \ldots, k$. Also note that $\sum_{i=1}^{k} G_i y_i + Hx = e_{m_G}$. Therefore, we have the following results.

Theorem 4.1 *Let $(X_1, \ldots, X_k, x)$ be any feasible solution of $(LSPRP)$. Then the vector*

$$\begin{pmatrix} (\operatorname{diag}(V_1 X_1) V_1^t)_{1:n_1} \\ \vdots \\ (\operatorname{diag}(V_k X_k) V_k^t)_{1:n_k} \\ x \end{pmatrix}$$

is a feasible solution of the linear programming relaxation $(SPLP)$.

Based on the above theorem and the fact that C_i, for $i = 1, \ldots, k$, are all diagonal matrices, the following corollary is straightforward.

Corollary 4.1 *The lower bound given by the SDP relaxation $(LSPRP)$ is great that or equal to the one given by the LP relaxation $(SPLP)$, i.e., $\mu_{LR}^* \geq \mu_{LP}^*$.*

4.2 An Infeasible Primal-Dual Interior-Point Method

We rewrite the dual $(LSPRD)$ by introducing a slack matrix Z_i for each $i \in \{1, \ldots, k\}$ and a slack vector z.

$$(LSPRD) \qquad \begin{array}{rl} \max & \sum_{i=1}^{m_G} \lambda_i + \sum_{i=1}^{k} (Y_i)_{00} \\ \text{subject to} & V_i^t(\operatorname{Diag}(0, \lambda^t G_i) + Y_i)V_i + Z_i = V_i^t C_i V_i, \\ & Y_i \in \mathcal{S}_{J_i^0}, \\ & \text{for } \ i \in \{1, \ldots, k\} \\ & H^t \lambda + z = c \\ & Z_1 \succeq 0, \ldots, Z_k \succeq 0, z \geq 0. \end{array}$$

The Karush-Kuhn-Tucker conditions of the dual log-barrier problem are

$$\begin{array}{rcccl}
\sum_{i=1}^k \mathcal{A}_i(V_iX_iV_i^t) + Hx - e_{m_G} & = & F_P^0 & = & 0 \\
\mathcal{G}_{J_i^0}(V_iX_iV_i^t) - E_{00}^i & = & F_{P1}^i & = & 0, \\
\text{for } \ i \in \{1,\ldots,k\} & & & & \\
H^t\lambda + z - c & = & F_D^0 & = & 0 \\
V_i^t(\mathrm{Diag}\,(0,\lambda^tG_i) + Y_i - C_i)V_i + Z_i & = & F_D^i & = & 0, \\
\text{for } \ i \in \{1,\ldots,k\} & & & & \\
z \circ x - \mu u & = & F_{ZX}^0 & = & 0 \\
Z_iX_i - \mu I & = & F_{ZX}^i & = & 0, \\
\text{for } \ i \in \{1,\ldots,k\}. & & & &
\end{array}$$

The first two equations are primal feasibility conditions, while the third and fourth are the dual feasibility conditions and the last two takes cares of complimentary slackness for X_i and Z_i and x and z, respectively. We solve this system of equations with a variant of Newton's method. We apply operators $\mathcal{A}_i$ and $\mathcal{G}_{J_i^0}$ to nonsymmetric matrices and then we linearize the above system as follows.

$$\begin{array}{rcl}
\sum_{i=1}^k \mathcal{A}_i(V_i\delta X_iV_i^t) + H\delta x & = & -F_P^0 \\
\mathcal{G}_{J_i^0}(V_i\delta X_iV_i^t) & = & -F_{P1}^i \\
\text{for } \ i \in \{1,\ldots,k\} & & \\
H^t\delta\lambda + \delta z & = & -F_D^0 \\
V_i^t(\mathrm{Diag}\,(0,\delta\lambda^tG_i) + \delta Y_i))V_i + \delta Z_i & = & -F_D^i \\
\text{for } \ i \in \{1,\ldots,k\} & & \\
\delta z \circ x + z \circ \delta x & = & -F_{ZX}^0 \\
\delta Z_iX_i + Z_i\delta X_i & = & -F_{ZX}^i \\
\text{for } \ i \in \{1,\ldots,k\}. & &
\end{array} \tag{4.6}$$

From the third and fourth equations, we have, for $i \in \{1,\ldots,k\}$,

$$\delta Z_i = -F_D^i - V_i^t(\mathrm{Diag}\,(0,\delta\lambda^tG_i) + \delta Y_i))V_i \tag{4.7}$$

and

$$\delta z = -F_D^0 - H^t\delta\lambda. \tag{4.8}$$

Substituting (4.7) and (4.8) into the last two equations, respectively, we have

$$\delta X_i = -Z_i^{-1}F_{ZX}^i + Z_i^{-1}F_D^iX_i + Z_i^{-1}V_i^t(\mathrm{Diag}\,(0,\delta\lambda^tG_i) + \delta Y_i))V_iX_i \tag{4.9}$$

and

$$\delta x = -z^{-1} \circ F_{ZX}^0 + z^{-1} \circ F_D^0 \circ x + z^{-1} \circ H^t\delta\lambda \circ x. \tag{4.10}$$

Substituting (4.9) and (4.10) into the first two equations, we have the following final normal equation.

$$\begin{array}{rcl}
\sum_{i=1}^k \mathcal{A}_i(V_iZ_i^{-1}V_i^t(\mathrm{Diag}\,(0,\delta\lambda^tG_i) + \delta Y_i)V_iX_iV_i^t) & & \\
+Hz^{-1} \circ H^t\delta\lambda \circ x & = & -F_P^0 + b_0 \\
\mathcal{G}_{J_i^0}(V_iZ_i^{-1}V_i^t(\mathrm{Diag}\,(0,\delta\lambda^tG_i) + \delta Y_i)V_iX_iV_i^t) & = & -F_{P1}^i + b_i \\
\text{for } \ i \in \{1,\ldots,k\}, & &
\end{array} \tag{4.11}$$

	nrow	ncol	nzero	LP	SDP	LP-SDP
small03	27	97	234	17327	18324	18320
tiny04	6	27	72	1037	1091	1066
tiny01	3	6	9	17.5	25	25
tiny05	7	35	70	1215	1257	1248

Table 2: Numerical Results

where

$$\begin{array}{ll} b_0 & = \sum_{i=1}^{k} \mathcal{A}_i(V_i(Z_i^{-1}F_{ZX}^i - Z_i^{-1}F_D^i X_i)V_i^t) + H(z^{-1} \circ F_{ZX}^0 - z^{-1} \circ F_D^0 \circ x), \\ b_i & = \mathcal{G}_{J_i^0}(V_i(Z_i^{-1}F_{ZX}^i - Z_i^{-1}F_D^1 X_i)V_i^t), \\ & \text{for } \ i \in \{1, \ldots, k\}. \end{array}$$

Denote the matrix representation of the left hand side of the normal equation by $\mathcal{K}$. The matrix $\mathcal{K}$ has a very nice sparsity structure similar to A in Figure 1, where the width of the long narrow bar is m_G which is much less than the size of the matrix.

We solve the normal equation by a preconditioned conjugate gradient method. Let $(\delta Y_1^*, \ldots, \delta Y_k^*, \delta\lambda^*)$ be the solution for the normal equation. By equations (4.7), (4.8), (4.9) and (4.10), we can obtain, for each $i \in \{1, \ldots, k\}$, δZ_i^*, δz_i^*,δX_i^* and δx_i^*, respectively. Finally, by symmetrizing δX_i^*, i.e.,

$$\delta X_i^* \leftarrow \frac{\delta X_i^* + (\delta X_i^*)^t}{2},$$

we obtain a search direction. We then do a linesearch and update the current point. Based on the duality gap, we update μ by using the following formula

$$\mu := \frac{\sum_{i=1}^{k} \text{trace}\,(Z_i X_i) + z^t x}{2(n - m + m_G + k)}.$$

4.3 Preliminary Numerical Tests and Future Work

In the previous subsections, we have developed an approach for solving problems with matrix structure (4.4). We did some preliminary numerical tests just to see how this SDP relaxation works for small problems. In our testing, we use the diagonal of the matrix representation $\mathcal{K}$ as the preconditioner. The infeasible primal-dual interior-point algorithm for the mixed LP-SDP relaxation is coded in C and Matlab. The results are summarized in Table 2. In Table 2, the columns under nrow, ncol and nzero are for the number of rows, columns and nonzero elements, respectively. The columns under LP and SDP show the lower bounds given by LP relaxation and SDP relaxation for a general dense problem, respectively, while the last column under LP-SDP shows the lower bounds given by our mixed LP-SDP relaxation.

It remains to try and use the mixed LP-SDP relaxation to derive an approach to solve general large sparse set partitioning problems. To achieve this, we propose the following:

- to have the same matrix sparsity pattern as described for the mixed LP-SDP relaxation, the matrix for the general problem need to be transformed into form like (4.4). This can be done by treating the 0-1 matrix A as an incidence matrix of a graph or netlist and applying graph partitioning and netlist partitioning techniques;
- because of the nice sparsity structure as shown in Figure 5.1, more sophisticated incomplete factorization preconditioners can be used to improve the performance of primal-dual interior-point solvers, see e.g. [7].

A APPENDIX-Notation

SP	set partitioning problem
SDP	semidefinite programming problem
$\mathcal{M})$	given collection of subsets of M
$A = (a_{ij})$	incidence matrix of the collection $\mathcal{M}$
e	the vector of ones
e_k	the k-th unit vector
a_i	the i-th row of incidence matrix A
SPLP	the linear programming relaxation for SP
p-d i-p	primal-dual interior-point (algorithm)
$B \circ C$	the Hadamard product of B and C
$Q \preceq R$	$R - Q$ is positive semidefinite
$\mathcal{S}_n$	the space of symmetric $n \times n$ matrices
PSDP	the semidefinite relaxation primal problem
Diag	the diagonal matrix formed from the vector
$\mathcal{P}_n$ **or** $\mathcal{P}$	the cone of positive semidefinite matrices in $\mathcal{S}_n$
T	$[-e, A]$, the assignment constraint matrix
Slater CQ	the Slater constraint qualification; strict feasiblity
p-d i-p	primal-dual interior-point method
$G = (\mathcal{V}, \mathcal{E})$	graph with node set $\mathcal{V}$ and edge set $\mathcal{E}$
cut edge	an edge connecting nodes in different subsets of a partition

$X = (x_{ij})$	partition matrix
$w(E_{cut})$	total weight of cut edges of the partition
$w^*(E_{cut})$	minimal total weight of cut edges over all partitions
L	Laplace matrix of the graph
Y_X	partition matrix lifted into higher dimensional matrix space
$\bar{m}$	$(m_1, \ldots, m_k)^t$
$A \otimes B$	the Kronecker product of A and B
$\mathrm{vec}\,(X)$	the vector formed from the columns of the matrix A
$Y_{0,1:n^2}$	the n^2 vector from the first row of Y
diag	the vector formed from the diagonal elements
arrow	the arrow operator $\mathrm{diag}\,(Y) - (0, (Y_{0,1:n^2})^t$
$\mathcal{G}_J$	the gangster operator $\mathcal{G}_J$ shoots "holes" in a matrix
$\mathcal{R}(B)$	range space of B
$\mathcal{N}(B)$	null space of B
J^0	$J^0 = J \cup \{(0,0)\}$
$K \lhd C$	K is a face of C
relint	relative interior
$\mathrm{diag}\,(A)$	the vector formed from the diagonal of the matrix A
$\mathrm{Diag}\,(v)$	the diagonal matrix formed from the vector v
E_n	the matrix of ones in $\mathcal{S}_n$
E_{ij}	the ij unit matrix in $\mathcal{S}_n$

References

[1] J. ARABEYRE, J. FEARNLEY, F. STEIGER, and W. TEATHER. The airline crew scheduling problem: a survey. *Transportation Science*, 2:140–163, 1969.

[2] E. BAKER and M. FISHER. Computational results for very large air crew scheduling problems. *OMEGA*, 9(6):613–618, 1981.

[3] E. BALAS. Some valid inequalities for the set partitioning problems. *Annals of Discrete Mathematics*, 1:13–47, 1977.

[4] E. BALAS and M.W. PADBERG. Set partitioning: A survey. *SIAM Review*, 18:710–760, 1976.

[5] J. BARUTT and T. HULL. Airline crew scheduling: aupercomputers and algorithms. *SIAM News*, 23(6), 1990.

[6] T. J. CHAN and C. A. YANO. A multiplier adjustment approach for the set partitioning problem. *Operations Research*, 40:S40–S47, 1992.

[7] PAULINA CHIN. *Iterative Algorithm for Solving Linear Programming from Engineering Applications.* PhD thesis, University of Waterloo, 1995.

[8] P. C. CHU and J. E. BEASLEY. A genetic algorithm for the set partitioning problem. Technical report, Imperial College, The Management School, London, England, 1995. URL: http:/mscmga.ms.ic.ac.uk/pchu/pchu.html.

[9] T.F. COLEMAN and A. POTHEN. The null space problem 1. complexity. *SIAM J. ALG. DISC. METH.*, 7:527–537, 1986.

[10] M. L. FISHER and P. KEDIA. Optimal solution of set covering/partitioning problems using dual heuristics. *Management Science*, 36:674–688, 1990.

[11] R.S. GARFINKEL and G.L. NEMHAUSER. The set partitioning problem: Set covering with equality constraints. *Operations Research*, 17:848–856, 1969.

[12] I. GERSHKOFF. Optimizaing flight crew schedules. *Interfaces*, 19:29–43, 1989.

[13] F. HARCHE and G.L. THOMPSON. The column subtraction algorithm: an exact method for solving weighted set covering, packing and partitioning problems. *Computers & operations Research*, 21(6):689–705, 1994.

[14] K. L. HOFFMAN and M. PADBERG. Solving airline crew-scheduling problems by branch-and-cut. *Management Science*, 6:657–682, 1993.

[15] K. L. HOFFMAN and M. PADBERG. Solving large set-partitioning problems with side constraints. *ORSA/TIMS Joint National Meeting, San Francisco*, November 2-4, 1992.

[16] R.E. MARSTEN. An algorithm for large set partitioning problems. *Management Science*, 20:774–787, 1974.

[17] G.L. NEMHAUSER and G. M. WEBER. Optimal set partitioning, matchings and lagrangean duality. *Naval Research Logistics Quarterly*, 26:553–563, 1979.

[18] D.M. RYAN and J.C. FALKNER. On the integer properties of scheduling and set partitioning models. *European Journal of Operational Research*, 35:422–456, 1988.

[19] P. WOLFE. The reduced gradient method. Unpublished manuscript, 1962.

[20] H. WOLKOWICZ and Q. ZHAO. Semidefinite relaxations for the graph partitioning problem. Research report corr 96-16, University of Waterloo, Waterloo, Ontario. URL: ftp://orion.uwaterloo.ca/pub/henry/reports/graphpart.ps.gz.

[21] Q. ZHAO. *Semidefinite Programming for Assignment and Partitioning Problems.* PhD thesis, University of Waterloo, 1996. URL: ftp://orion.uwaterloo.ca/pub/henry/software/qap.d/zhaophdthesis.ps.gz.

[22] Q. ZHAO, S. KARISCH, F. RENDL, and H. WOLKOWICZ. Semidefinite programming relaxations for the quadratic assignment problem. Research report, University of Waterloo, Waterloo, Ontario, 1995. CORR 95-27, URL: ftp://orion.uwaterloo.ca/pub/henry/reports/qapsdp.ps.gz.