An efficient semidefinite programming relaxation for the graph partition problem

R. Sotirov*

August 28, 2012

Abstract

We derive a new semidefinite programming relaxation for the general graph partition problem (GPP). Our relaxation is based on matrix lifting with matrix variable having order equal to the number of vertices of the graph. We show that this relaxation is equivalent to the Frieze-Jerrum relaxation [A. Frieze and M. Jerrum. Improved approximation algorithms for max k-cut and max bisection. Algorithmica, 18(1):67–81, 1997] for the maximum k-cut problem with an additional constraint that involves the restrictions on the subset sizes. Since the new relaxation does not depend on the number of subsets k into which the graph should be partitioned we are able to compute bounds for large k.

We compare theoretically and numerically the new relaxation with other SDP relaxations for the GPP. The results show that our relaxation provides competitive bounds and is solved significantly faster than any other known SDP bound for the general GPP.

Keywords: graph partition, graph equipartition, matrix lifting, vector lifting, semidefinite programming

1 Introduction

The general graph partition problem (GPP) is defined as follows. Let $G = (V, E)$ be an undirected graph with vertex set V, $|V| = n$ and edge set E, and $k \geq 2$ be a fixed number. The goal is to find a partition of the vertex set into k disjoint subsets $S_1, \ldots, S_k$ of specified sizes $m_1 \geq \ldots \geq m_k$, $\sum_{j=1}^{k} m_j = n$ such that the total weight of edges joining different sets S_j is minimized. Here we also refer to the described GPP problem as the *k-partition problem*. If there is a requirement that all m_j, $j = 1, \ldots, k$ are equal, then we refer to this as the *graph equipartition problem* (GEP). The case of the GPP with $k = 2$ is known as the *graph bisection problem* (GBP). The special case of the GBP with both m_j equal is usually called the *equicut problem*, see e.g., [35].

We denote by A the adjacency matrix of G. For a given partition of the graph into k subsets, let $X = (x_{ij})$ be the $n \times k$ matrix defined by

$$x_{ij} = \begin{cases} 1 & \text{if vertex } i \in S_j \\ 0 & \text{if vertex } i \notin S_j. \end{cases}$$

*Department of Econometrics and OR, Tilburg University, The Netherlands. r.sotirov@uvt.nl

Note that the jth column $X_{:,j}$ is the characteristic vector of S_j, and k-partitions are in one-to-one correspondence with the set

$$\mathcal{P}_k := \left\{ X \in \mathbb{R}^{n\times k} : \ Xu_k = u_n, \ X^{\mathrm{T}} u_n = m, \ x_{ij} \in \{0,1\}, \ \forall i,j \right\},$$

where $m = (m_1, \dots, m_k)^{\mathrm{T}}$, and u_k (u_n) is the all-ones k-vector (n-vector, respectively). For each $X \in \mathcal{P}_k$, it holds that

- $\operatorname{tr} X^{\mathrm{T}} D X = \operatorname{tr} D$, if D is diagonal.
- $\frac{1}{2} \operatorname{tr} X^{\mathrm{T}} A X$ gives the total weight of the edges within the subsets S_j.

Therefore, the total weight of edges cut by X, i.e., those joining different sets S_j is

$$w(E_{\mathrm{cut}}) := \frac{1}{2} \operatorname{tr}(X^{\mathrm{T}} L X),$$

where

$$L := \operatorname{Diag}(Au_n) - A$$

is the Laplacian matrix of the graph. Thus, the graph partition problem in a trace formulation is:

$$\text{(GPP)} \qquad \begin{array}{ll} \min & \frac{1}{2} \operatorname{tr}(X^{\mathrm{T}} L X) \\ \text{s.t.} & X \in \mathcal{P}_k. \end{array}$$

The graph partition problem has many applications such as VLSI design [36], parallel computing [6, 26, 44], network partitioning [19, 43], and floor planing [9]. The graph equipartition problem also plays a role in telecommunications, see e.g., [37].

The GPP is a NP-hard [22] combinatorial optimization problem. Nevertheless, it is a fundamental problem that is extensively studied. Many heuristics are suggested; see e.g., Kernighan and Lin [29], Fiduccia and Mattheyses [19], Battiti and Bertossi [5], Bui and Moon [7]. There are also known relaxations of the problem, some of which we list below.

In 1973, Donath and Hoffman [15] derived an eigenvalue-based bound for the general GPP that was further improved by Rendl and Wolkowicz [41] in 1995. Alizadeh [1] proved that the Donath-Hoffman bound is the dual of a semidefinite program (SDP). Also, Anstreicher and Wolkowicz [2] showed that the Donath-Hoffman bound can be obtained using the Lagrangian dual of an appropriate quadratically constrained problem.

In 1998, Karisch and Rendl [34] suggested two relaxations with increasing complexity for the graph equipartition problem that are stronger than the Donath-Hoffman eigenvalue-based bound and the Rendl-Wolkowicz bound. These relaxations are based on matrix lifting with matrix variables having order n. The strongest bound from [34] is currently the best known SDP bound for the GEP. In [11], a SDP relaxation for the GEP is derived from a SDP relaxation for the more general quadratic assignment problem. This bound can be computed efficiently for larger graphs that have suitable algebraic symmetry. For a comparison of the SDP bounds for the GEP, see [45].

While the GEP is well studied, there are very few SDP relaxations for the general GPP. In particular, besides the SDP formulation of the Donath-Hoffman bound we only know of the Wolkowicz-Zhao relaxation for the general GPP [47]. The Wolkowicz-Zhao relaxation is based on vector lifting and its matrix variable has order kn. Clearly, it is very hard to solve that relaxation for large k. The case of the GPP where $k = 2$, i.e., the graph bisection problem (GBP) is studied separately in the literature. For the GBP there is a SDP relaxation with matrix variable of order n. This SDP relaxation is introduced by Karisch, Rendl, and Clausen [35] and it is also used in [17, 25] to derive approximation algorithms for the GBP. Of course the Wolkowicz-Zhao relaxation [47] also provides a bound for the GBP.

Frieze and Jerrum [20] derived a SDP relaxation for the *maximum k-cut problem* whose matrix variable depends only on the number of the vertices of the graph. The max-k-cut problem partitions the vertex set into at most k subsets such that the total weight of edges joining different sets is maximized. Eisenblätter [16] proposed a SDP relaxation for the minimum k-partition problem using the approach similar to the one used in [20]. The minimum k-partition problem from [16] asks for a partition of the vertex set into at most k subsets such that the total weight of edges in the induced subgraphs is minimized. Since in the above mentioned two problems there is no restriction on the sizes of the subsets in partitions, they can be seen as generalizations of the graph partition problem that we analyze here. Moreover, the minimum k-partition problem from [16] is equivalent to finding a maximum k-cut. Karger, Motwani, and Sudan [32] derived a SDP relaxation for the graph coloring problem using the approach similar to the one used in [20] and [16].

Several researchers presented results on solving the GPP by incorporating semidefinite programming relaxations within a branch-and-bound framework or a branch-and-cut framework. Karisch, Rendl, and Clausen [35] reported on solving the graph bisection problem for problem instances with 80 to 90 vertices using a branch-and-bound algorithm, and they also obtained tight approximations for larger instances. Ghaddar, Anjos, and Liers [21] implemented a branch-and-cut algorithm based on SDP for a special case of the GPP in which there is no prespecified size of k subsets. They computed optimal solutions for dense graphs with up to 60 vertices, for grid graphs with up to 100 vertices, and for different values of k. Armbruster, Helmberg, Fügenschuh, and Martin [4] evaluated the strengths of a branch-and-cut framework for linear and semidefinite relaxations of the minimum graph bisection problem on large and sparse instances. They showed that in the majority of the cases the semidefinite approach is the clear winner. This is very encouraging since SDP relaxations are widely believed to be of use only for small dense instances.

Main results and outline

In this paper we propose a new SDP relaxation of the general GPP that is based on matrix lifting with matrix variable of order n. We show that this relaxation is equivalent to the well known Frieze-Jerrum relaxation [20] for the max-k-cut problem with an additional constraint that involves the restrictions of the subset sizes. To the best of our knowledge this is the only SDP relaxation of the general GPP whose size is independent of k. The computational experiments show that when $k > 2$ the new relaxation is solved significantly faster than any other known SDP relaxation for the GPP. The numerical tests also show that it is the only SDP relaxation for the general GPP that could be solved when $k > 5$ and $n > 50$.

The further set-up of the paper is as follows. In Section 2, we study vector lifting SDP relaxations of the GPP. In particular, in Section 2.1 we simplify the technical approach from the paper by Wolkowicz and Zhao [47] to derive the Wolkowicz-Zhao relaxation of the GPP. Then, the Wolkowicz-Zhao bound is improved by adding nonnegativity constraints (we refer further to this relaxation as the improved Wolkowicz-Zhao relaxation). In Section 2.2 we show that the GPP is a special case of the quadratic assignment problem (QAP).

Therefore, we suggest the well known SDP relaxation of the QAP [48] for a relaxation of the GPP. The same relaxation was used as the SDP relaxation of the GEP in [11, 45].

In Section 3, we study matrix lifting SDP relaxations of the GPP. In Section 3.1, we derive the new SDP relaxation and prove that it is dominated by the improved Wolkowicz-Zhao relaxation. Here we we also suggest possible improvements of the new SDP relaxation by adding triangle constraints and/or independent set type of constraints. In Section 3.2 we show that the new relaxation is equivalent to the Frieze-Jerrum relaxation [20] for the max-k-cut problem with an additional constraint.

In Section 4 we show that when restricted to the equipartition problem, the new SDP relaxation is equivalent to the improved Wolkowicz-Zhao relaxation, and in Section 5 we compare SDP relaxations of the graph bisection problem. We prove that when restricted to the bisection problem the improved Wolkowicz-Zhao relaxation is equivalent to the QAP relaxation from [48]. Further, we show that the improved Wolkowicz-Zhao relaxation dominates the SDP relaxation from [35] that is proven to be equivalent to our relaxation.

In Section 6, the new relaxation is numerically compared to all above mentioned relaxations. The numerical results include random graphs and graphs from the literature that are known to be hard. The results show that the bounds provided by the new relaxation are competitive and that these are computed significantly faster compared to the other relaxations.

Notation

The space of $p \times q$ real matrices is denoted by $\mathbb{R}^{p\times q}$, the space of $k \times k$ symmetric matrices is denoted by $\mathcal{S}_k$, and the space of $k \times k$ symmetric positive semidefinite matrices by $\mathcal{S}_k^+$. We will sometimes also use the notation $X \succeq 0$ instead of $X \in \mathcal{S}_k^+$, if the order of the matrix is clear from the context. For two matrices $X, Y \in \mathbb{R}^{n\times n}$, $X \geq Y$ means $x_{ij} \geq y_{ij}$, for all i, j.

For an index set $I \subset \{1, \dots, n\}$ the principal submatrix of A is abbreviated as $A_{I,I}$. To denote the ith column of the matrix X we write $X_{:,i}$. We use I_n to denote the identity matrix of order n, and e_i to denote the i-th standard basis vector. Similarly, J_n and u_n denote the $n \times n$ all-ones matrix and all-ones n-vector respectively. We will omit subscripts if the order is clear from the context. We set $E_{ij} = e_i e_j^{\mathrm{T}}$.

The 'vec' operator stacks the columns of a matrix, while the 'diag' operator maps an $n \times n$ matrix to the n-vector given by its diagonal. The adjoint operator of 'diag' we denote by 'Diag'. The trace operator is denoted by 'tr'.

The Kronecker product $A \otimes B$ of matrices $A \in \mathbb{R}^{p\times q}$ and $B \in \mathbb{R}^{r\times s}$ is defined as the $pr \times qs$ matrix composed of pq blocks of size $r \times s$, with block ij given by $a_{ij}B$, $i = 1, \dots, p$, $j = 1, \dots, q$, see e.g., [23]. We use the following property of the Kronecker product

$$(A \otimes B)\mathrm{vec}(X) = \mathrm{vec}\left(BXA^{\mathrm{T}}\right). \tag{1}$$

The Hadamard product of two matrices A and B of the same size is denoted by $A \circ B$ and defined as $(A \circ B)_{ij} = a_{ij} \cdot b_{ij}$ for all i, j.

2 Vector lifting SDP relaxations of the GPP

In this section we study the Wolkowicz-Zhao relaxation [47], the improved Wolkowicz-Zhao relaxation, and the Zhao-Karisch-Rendl-Wolkowicz relaxation [48].

2.1 The improved Wolkowicz-Zhao relaxation

We simplify the technical approach from [47] to derive the Wolkowicz-Zhao relaxation. Further, we impose on the derived SDP relaxation nonnegativity constraints and obtain the improved Wolkowicz-Zhao relaxation. In order to compare the bounds in later sections, we reformulate the improved Wolkowicz-Zhao relaxation.

Wolkowicz and Zhao obtain a tractable relaxation after linearizing the objective function by lifting a variable into the space of $(nk+1)\times(nk+1)$ matrices. They approximate the polytope

$$\hat{\mathcal{P}}_k := \operatorname{conv}\left\{ \begin{pmatrix} 1 \\ x \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix}^{\mathrm{T}} :\ x = \operatorname{vec}(X),\ X \in \mathcal{P}_k \right\} \subseteq S^{+}_{kn+1},$$

by a larger set that contains $\hat{\mathcal{P}}_k$. In the sequel we use the following block notation

$$Y = \begin{pmatrix} Y^{(00)} & Y^{(01)} & \dots & Y^{(0k)} \\ Y^{(10)} & Y^{(11)} & \dots & Y^{(1k)} \\ \vdots & \vdots & \ddots & \vdots \\ Y^{(k0)} & Y^{(k1)} & \dots & Y^{(kk)} \end{pmatrix}, \tag{2}$$

for matrices in $\mathcal{S}_{nk+1}$ where $Y^{(00)}$ is a scalar, $Y^{(10)}, \dots, Y^{(k0)} \in \mathbb{R}^n$ and $Y^{(ij)} \in \mathbb{R}^{n\times n}$ for $i, j = 1, \dots, k$. Note that we index elements from the space of symmetric matrices of order $nk+1$ from zero.

In order to derive the Wolkowicz-Zhao relaxation, we need the following lemma.

Lemma 1. *Let V_p, $p \in \{k, n\}$ be defined as*

$$V_p := \begin{pmatrix} I_{p-1} \\ -u_{p-1}^{\mathrm{T}} \end{pmatrix}, \tag{3}$$

and $m = (m_1, \dots, m_k)^{\mathrm{T}}$. Then,

$$\left\{ X \in \mathbb{R}^{n\times k} : Xu_k = u_n,\ X^{\mathrm{T}}u_n = m \right\} = \left\{ \frac{1}{n} u_n m^{\mathrm{T}} + V_n R V_k^{\mathrm{T}} : R \in \mathbb{R}^{(n-1)\times(k-1)} \right\}.$$

Proof. This follows from the fact that $V_p^{\mathrm{T}} u_p = 0$ and $\operatorname{rank}(V_p) = p-1$, for $p \in \{k, n\}$. □

The previous lemma follows from Lemma 3.1 [41]. Recall that matrix V_p that is used in Lemma 1 could be any basis of $u_p^{\perp}$. The following theorem gives us some more structure of the elements in $\hat{\mathcal{P}}_k$.

Theorem 2. *Let $Y \in \hat{\mathcal{P}}_k$ and*

$$\hat{V} := \begin{pmatrix} 1 & 0 \\ \frac{1}{n} m \otimes u_n & V_k \otimes V_n \end{pmatrix},$$

where V_k, V_n are of the form (3). Then there exists a symmetric matrix Z of order $(k-1)(n-1)+1$ (indexed from 0) such that

$$Z \succeq 0, \quad Z_{00} = 1 \ \text{ and } \ Y = \hat{V} Z \hat{V}^{\mathrm{T}}.$$

Proof. (See also [47].) First we look at the extreme points of $\hat{\mathcal{P}}_k$. Let Y be one of them i.e.,

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

where $x = \mathrm{vec}(X)$, $X \in \mathcal{P}_k$. It follows from Lemma 1 that for $X \in \mathcal{P}_k$ there exists a matrix $R \in \mathbb{R}^{(n-1)\times(k-1)}$ such that

$$X = \frac{1}{n} u_n m^{\mathrm{T}} + V_n R V_k^{\mathrm{T}}.$$

From (1) we have

$$x = \mathrm{vec}(X) = \frac{1}{n} m \otimes u_n + (V_k \otimes V_n)\mathrm{vec}(R) = W\bar{z},$$

where $\bar{z} = \begin{pmatrix} 1 \\ \mathrm{vec}(R) \end{pmatrix}$ and

$$W := \left(\ \tfrac{1}{n} m \otimes u_n, V_k \otimes V_n \ \right).$$

Now

$$Y = \begin{pmatrix} 1 & (W\bar{z})^{\mathrm{T}} \\ W\bar{z} & W\bar{z}\bar{z}^{\mathrm{T}}W^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} e_1^{\mathrm{T}} \\ W \end{pmatrix} \bar{z}\bar{z}^{\mathrm{T}} \begin{pmatrix} e_1^{\mathrm{T}} \\ W \end{pmatrix}^{\mathrm{T}} = \hat{V} Z \hat{V}^{\mathrm{T}},$$

with $Z = \bar{z}\bar{z}^{\mathrm{T}}$. Hence Z is a symmetric positive semidefinite matrix and $Z_{00} = 1$. Since the same holds for convex combinations of several extreme points the theorem is proved. □

Zhao and Wolkowicz [47] approximate $\hat{\mathcal{P}}_k$ by the larger set

$$\hat{P}_k := \left\{ Y \in \mathcal{S}_{kn+1} : \exists Z \in \mathcal{S}^{+}_{(k-1)(n-1)+1} \ \text{ s.t. } \ Z_{00} = 1, \ Y = \hat{V} Z \hat{V}^{\mathrm{T}} \right\}.$$

Further, they note that for $X \in \mathcal{P}_k$ one has

$$X_{:,i} \circ X_{:,j} = 0, \quad \forall i \neq j,$$

and therefore impose to the elements of $\hat{P}_k$ the constraints

$$\mathrm{tr}(E_{ll} Y^{(ij)}) = 0, \quad \forall i, j = 1, \ldots, k, \ i \neq j, \quad l = 1, \ldots, n. \tag{4}$$

We collect all these equalities in the constraint $\mathcal{G}(Y) = 0$. This sparsity pattern is sometimes called the Gangster constraint, see e.g., [47, 48]. Finally, the SDP relaxation of the GPP introduced in [47] is

$$(\mathrm{GPP_{ZW}}) \qquad \begin{array}{rl} \min & \mathrm{tr}(L_A \hat{V} Z \hat{V}^T) \\ \text{s.t.} & \mathcal{G}(\hat{V} Z \hat{V}^T) = 0 \\ & Z_{00} = 1, \quad Z \in \mathcal{S}^{+}_{(k-1)(n-1)+1}, \end{array}$$

where

$$L_A := \frac{1}{2} \begin{pmatrix} 0 & 0 \\ 0 & I_k \otimes L \end{pmatrix}.$$

The following theorem gives an explicit description of $\hat{P}_k$. This extends the lemma of Wolkowizc and Zhao [47], who only proved the sufficiency.

Theorem 3. *Let $Y \in \mathcal{S}^+_{kn+1}$ with the block form (2). Then $Y \in \hat{P}_k$ if and only if*

(i) $Y^{(00)} = 1$, $\sum_{i=1}^k Y^{(i0)} = u_n$, $u_n^{\mathrm{T}} Y^{(i0)} = m_i$, $i = 1, \ldots k$.

(ii) $m_i Y^{(0j)} = u_n^{\mathrm{T}} Y^{(ij)}$, $i, j = 1, \ldots, k$.

(iii) $\sum_{i=1}^k Y^{(ij)} = u_n Y^{(0j)}$, $j = 1, \ldots, k$.

(iv) $\sum_{i=1}^k \mathrm{diag}(Y^{ij}) = Y^{(j0)}$, $j = 1, \ldots, k$.

Proof. Let $Y \in \hat{P}_k$, then *(i)–(iv)* follow from the fact that $TY = 0$, where

$$T := \begin{pmatrix} -m & I_k \otimes u_n^{\mathrm{T}} \\ -u_n & u_k^{\mathrm{T}} \otimes I_n \end{pmatrix}, \tag{5}$$

see Lemma 4.1 in [47].

Conversely, let $Y \in \mathcal{S}^+_{nk+1}$ satisfies *(i)–(iv)*. Then $TY = 0$. From Theorem 3.1 [47] we know that $T\hat{V} = 0$ and that the columns of $\hat{V}$ are linearly independent. Therefore there exists a $U \in \mathbb{R}^{((k-1)(n-1)+1)\times(kn+1)}$ such that $Y = \hat{V}U$. Since Y is a symmetric matrix, it follows that $Y = Y^{\mathrm{T}} = U^{\mathrm{T}}\hat{V}^{\mathrm{T}}$ and therefore $TU^{\mathrm{T}}\hat{V}^{\mathrm{T}} = 0$. From the same reasoning as before, there exists a $Z \in \mathbb{R}^{((k-1)(n-1)+1)\times((k-1)(n-1)+1)}$ such that $U^{\mathrm{T}} = \hat{V}Z$. Now $Y = U^{\mathrm{T}}\hat{V}^{\mathrm{T}} = \hat{V}Z\hat{V}^{\mathrm{T}}$ and $Y = \hat{V}U = \hat{V}Z^{\mathrm{T}}\hat{V}^{\mathrm{T}}$. Thus $Z = Z^{\mathrm{T}}$, and $Z \succeq 0$ since $Y \succeq 0$. From $Y_{00} = 1$ it follows that $Z_{00} = 1$. □

The following corollary is an immediate consequence of part *(iv)* of Theorem 3, see also [47].

Corollary 4. *Let $Y \in \hat{P}_k$ and $\mathcal{G}(Y) = 0$. Then* $\mathrm{diag}(Y) = Y_{:,0}$.

Note that the relaxation $\mathrm{GPP_{ZW}}$ can be strengthened by adding nonnegativity constraints. Although Zhao and Wolkowicz do not add nonnegativity constraints to their relaxation, they mentioned that it would be worth adding them. Here we do add them, and call the corresponding relaxation $\mathrm{GPP_{ZWN}}$, i.e.,

$$(\mathrm{GPP_{ZWN}}) \quad \begin{aligned} \min \quad & \mathrm{tr}(L_A \hat{V} Z \hat{V}^T) \\ \text{s.t.} \quad & \mathcal{G}(\hat{V} Z \hat{V}^T) = 0 \\ & \hat{V} Z \hat{V}^T \geq 0 \\ & Z_{00} = 1, \quad Z \in \mathcal{S}^+_{(k-1)(n-1)+1}. \end{aligned}$$

In this paper we also refer to $\mathrm{GPP_{ZWN}}$ as the improved Wolkowicz-Zhao relaxation. Our numerical results show that $\mathrm{GPP_{ZWN}}$ provides much stronger bounds than $\mathrm{GPP_{ZW}}$, see Section 6. Since $\mathrm{GPP_{ZWN}}$ contains $O(n^4)$ constraints it is difficult to solve the relaxation for larger graphs and/or larger k.

Remark 5. *If the nonnegativity constraints are added to* $\mathrm{GPP_{ZW}}$*, then the Gangster constraint in the so obtained relaxation* $\mathrm{GPP_{ZWN}}$ *can be replaced by*

$$\mathrm{tr}((J_k - I_k) \otimes I_n)(\hat{V} Z \hat{V}^T)_{1:kn,1:kn} = 0.$$

The improved Wolkowicz-Zhao relaxation can also be derived in the following way. For $X \in \mathcal{P}_k$ we define $y := \mathrm{vec}(X)$ and $Y := yy^{\mathrm{T}}$. We write $Y \in \mathcal{S}^+_{nk}$ in block form (2) where the first row and column are excluded, and

$$Y^{(ij)} := X_{:,i} X_{:,j}^{\mathrm{T}} \in \mathbb{R}^{n \times n}, \quad i, j = 1, \ldots, k.$$

We associate $X \in \mathcal{P}_k$ with a rank-one matrix $Y_X \in \mathcal{S}^+_{nk+1}$ as follows:

$$Y_X := \begin{pmatrix} 1 \\ \mathrm{vec}(X) \end{pmatrix} \begin{pmatrix} 1 \\ \mathrm{vec}(X) \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & y^{\mathrm{T}} \\ y & Y \end{pmatrix}, \tag{6}$$

which has block form (2) where $Y^{(i0)} = X_{:,i}$, $i = 1, \dots, k$. The matrix Y_X has the sparsity pattern (4), and any $Y \in \mathcal{S}_{nk}$, $Y \geq 0$ has the same sparsity pattern if and only if

$$\mathrm{tr}((J_k - I_k) \otimes I_n)Y = 0. \tag{7}$$

The constraints $Xu_k = u_n$ and $X^{\mathrm{T}}u_n = mu_k$ are equivalent to

$$T \begin{pmatrix} 1 \\ \mathrm{vec}(X) \end{pmatrix} = 0,$$

where T is defined in (5). This constraint may be rewritten as

$$\mathrm{tr}(T^{\mathrm{T}}TY_X) = 0. \tag{8}$$

For any $Y \in \mathcal{S}_{nk}$, $Y \geq 0$ that satisfies (7), one has

$$\mathrm{tr}\left(T^{\mathrm{T}}T \begin{pmatrix} 1 & y^{\mathrm{T}} \\ y & Y \end{pmatrix}\right) = (\sum_{i=1}^{k} m_i^2 + n) - 2((m^{\mathrm{T}} + u_k^{\mathrm{T}}) \otimes u_n^{\mathrm{T}})y + \mathrm{tr}(I_k \otimes J_n)Y + \mathrm{tr}(Y).$$

Thus constraint (8) becomes

$$\mathrm{tr}(I_k \otimes J_n)Y + \mathrm{tr}(Y) = -(\sum_{i=1}^{k} m_i^2 + n) + 2((m^{\mathrm{T}} + u_k^{\mathrm{T}}) \otimes u_n^{\mathrm{T}})y.$$

Finally, by relaxing the rank-one condition on Y_X to $Y_X \in \mathcal{S}^+_{nk+1}$ we obtain the following reformulation of $\mathrm{GPP_{ZWN}}$:

$$(\mathrm{GPP_{ZWN}}) \qquad \begin{array}{ll} \min & \frac{1}{2}\,\mathrm{tr}(I_k \otimes L)Y \\ \text{s.t.} & \mathrm{tr}((J_k - I_k) \otimes I_n)Y = 0 \\ & \mathrm{tr}(I_k \otimes J_n)Y + \mathrm{tr}(Y) = -(\sum_{i=1}^{k} m_i^2 + n) + 2y^{\mathrm{T}}((m + u_k) \otimes u_n) \\ & \begin{pmatrix} 1 & y^{\mathrm{T}} \\ y & Y \end{pmatrix} \in \mathcal{S}^+_{nk+1}, \quad Y \geq 0. \end{array}$$

Due to its simplicity, we use the above reformulation of $\mathrm{GPP_{ZWN}}$ to compare the bounds in Section 5.

2.2 The Zhao, Karisch, Rendl, and Wolkowicz relaxation

It is known that the GPP is a special case of the quadratic assignment problem. To show this, we recall that the set Π_n of all permutation matrices and $\mathcal{P}_k$ are related in the following way (see e.g., [33]). If $Z \in \Pi_n$ then $X = ZU \in \mathcal{P}_k$ where

$$U = \begin{pmatrix} u_{m_1} & 0 & \dots & 0 \\ 0 & u_{m_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & u_{m_k} \end{pmatrix}.$$

Conversely, each $X \in \mathcal{P}_k$ can be written as $X = ZU$ with $Z \in \Pi_n$. For such a related pair (Z, X) it follows that

$$\operatorname{tr}(X^{\mathrm{T}}LX) = \operatorname{tr}(Z^{\mathrm{T}}AZU(J_k - I_k)U^{\mathrm{T}}) = \operatorname{tr}(Z^{\mathrm{T}}AZB), \tag{9}$$

where

$$B := \begin{pmatrix} 0_{m_1\times m_1} & J_{m_1\times m_2} & \dots & J_{m_1\times m_k} \\ J_{m_2\times m_1} & 0_{m_2\times m_2} & \dots & J_{m_2\times m_k} \\ \vdots & \vdots & \ddots & \vdots \\ J_{m_k\times m_1} & J_{m_k\times m_2} & \dots & 0_{m_2\times m_k} \end{pmatrix} \in \mathcal{S}_n. \tag{10}$$

Note that in (9) we exploit the fact that for $X \in \mathcal{P}_k$ it follows

$$\operatorname{tr}(X^{\mathrm{T}}LX) = \operatorname{tr}(X^{\mathrm{T}}AX(J_k - I_k)), \tag{11}$$

where L is the Laplacian matrix of the graph. Now, the GPP may be formulated as the QAP

$$\min_{Z\in\Pi_n} \frac{1}{2}\operatorname{tr}(AZBZ^{\mathrm{T}}),$$

where A is the adjacency matrix of G, and B is of the form (10). Therefore, the following SDP relaxation of the QAP (see [48, 40]) is also a relaxation for the GPP:

$$(\mathrm{GPP}_{\mathrm{QAP}}) \qquad \begin{array}{ll} \min & \frac{1}{2}\operatorname{tr}(B\otimes A)Y \\ \text{s.t.} & \operatorname{tr}(I_n \otimes E_{jj})Y = 1, \quad \operatorname{tr}(E_{jj}\otimes I_n)Y = 1 \quad j = 1,\dots,n \\ & \operatorname{tr}(I_n \otimes (J_n - I_n)) + (J_n - I_n)\otimes I_n)Y = 0 \\ & \operatorname{tr}(J_{n^2}Y) = n^2 \\ & Y \succeq \mathcal{S}^+_{n^2}, \quad Y \geq 0. \end{array}$$

One may easily verify that $\mathrm{GPP}_{\mathrm{QAP}}$ is indeed a relaxation of the QAP by noting that

$$Y := \operatorname{vec}(Z)\operatorname{vec}(Z)^{\mathrm{T}}$$

is a feasible point of $\mathrm{GPP}_{\mathrm{QAP}}$ for $Z \in \Pi_n$, and that the objective value of $\mathrm{GPP}_{\mathrm{QAP}}$ at this point Y is precisely $\operatorname{tr}(AZBZ^{\mathrm{T}})$. The constraints involving E_{jj} are generalizations of the assignment constraints $Zu_n = Z^{\mathrm{T}}u_n = u_n$, and the sparsity constraints, i.e., $\operatorname{tr}(I_n \otimes (J_n - I_n) + (J_n - I_n)\otimes I_n)Y = 0$ generalize the orthogonality conditions $ZZ^{\mathrm{T}} = Z^{\mathrm{T}}Z = I_n$.

Note that it is in general hard to solve $\mathrm{GPP}_{\mathrm{QAP}}$ since the relaxation has $O(n^4)$ sign constraints and $O(n^3)$ equality constraints. De Klerk et al. [11] show that in the case of the GEP and when the adjacency matrix of the graph has a large automorphism group, then $\mathrm{GPP}_{\mathrm{QAP}}$ can be solved efficiently. In Section 5 we analyze $\mathrm{GPP}_{\mathrm{QAP}}$ for another special case of the GPP, i.e., for the bisection problem.

3 Matrix lifting SDP relaxations of the GPP

In this section we derive a new SDP relaxation of the GPP that is based on matrix lifting and compare it with the SDP relaxation $\mathrm{GPP}_{\mathrm{ZWN}}$ that is based on vector lifting. Further, we show that our new relaxation is equivalent to the Frieze-Jerrum relaxation [20] for the max-k-cut problem with an additional constraint.

3.1 The new SDP relaxation

One way to obtain a relaxation of the graph partition problem is to linearize the objective function $\operatorname{tr}(X^{\mathrm{T}}LX) = \operatorname{tr}(LXX^{\mathrm{T}})$ by replacing XX^{T} by a new variable Y. This yields the following feasible set of the GPP:

$$P_k := \operatorname{conv}\{Y : \exists X \in \mathcal{P}_k \;\text{ s.t. }\; Y = XX^{\mathrm{T}}\}.$$

It is clear that for $Y \in P_k$ one has

$$\operatorname{diag}(Y) = u_n, \quad \operatorname{tr}(JY) = \sum_{i=1}^{k} m_i^2, \quad Y \geq 0, \quad Y \succeq 0.$$

The following proposition shows that one can impose a stronger positive semidefiniteness constraint on elements in P_k than $Y \succeq 0$.

Proposition 6. *Let $Y \in P_k$. Then $kY - J_n \succeq 0$.*

Proof. First we look at the extreme points of P_k. Let Y be one of them, i.e., $Y = XX^{\mathrm{T}}$ where $X \in \mathcal{P}_k$. Let $x_i = X_{:,i}$, $i = 1, \ldots, k$. Then,

$$kY - J_n = kXX^{\mathrm{T}} - u_n u_n^{\mathrm{T}} = k\sum_{i=1}^{k} x_i x_i^{\mathrm{T}} - (\sum_{i=1}^{k} x_i)(\sum_{i=1}^{k} x_i)^{\mathrm{T}} = \sum_{i>j}(x_i - x_j)(x_i - x_j)^{\mathrm{T}} \succeq 0.$$

Since the same holds for convex combinations of several extreme points the proposition is proved. □

By collecting all mentioned constraints we obtain the following SDP relaxation for the graph partition problem:

$$(\mathrm{GPP_{RS}}) \qquad \begin{array}{rl} \min & \frac{1}{2}\operatorname{tr}(LY) \\ \text{s.t.} & \operatorname{diag}(Y) = u_n \\ & \operatorname{tr}(JY) = \sum_{i=1}^{k} m_i^2 \\ & kY - J_n \in \mathcal{S}_n^+, \quad Y \geq 0. \end{array}$$

Note that this semidefinite program has a matrix variable of order n, independent of k. To the best of our knowledge this relatively simple model has not been previously investigated.

It is easy to see that $\mathrm{GPP_{RS}}$ has a strictly feasible point. Indeed, the following point is in the interior of the feasible set of $\mathrm{GPP_{RS}}$:

$$\widetilde{Y} = \frac{\sum_{i=1}^{k} m_i^2}{n^2} J_n + \frac{n^2 - \sum_{i=1}^{k} m_i^2}{n^2(n-1)}(nI_n - J_n). \tag{12}$$

Note that $\widetilde{Y}$ can be derived from the barycenter point of $\mathrm{GPP_{ZWN}}$, see [47].

In order to strengthen $\mathrm{GPP_{RS}}$ one can add the triangle constraints

$$y_{ij} + y_{ik} \leq 1 + y_{jk}, \quad \forall\ (i, j, k). \tag{13}$$

Constraints (13) explore the following property of P_k: if i, j and i, k belong to the same set of the partition, then i, j and k must be in the same set. Note that there are

$3\binom{n}{3}$ inequalities of type (13). One can also add to $\mathrm{GPP_{RS}}$ the independent set type of constraints

$$\sum_{i<j,\ i,j\in\mathcal{I}} y_{ij} \geq 1, \text{ for all } \mathcal{I} \text{ with } |\mathcal{I}| = k+1. \tag{14}$$

These constraints insure that if $Y \in P_k$, then the graph with adjacency matrix Y has no independent set of size $k+1$. There are $\binom{n}{k+1}$ inequalities of type (14). Constraints (13) and (14) are also used in [34] to strengthen the SDP relaxation for the graph equipartition problem, and in [8] for a polyhedral setting of the GPP. By adding constraints (13) and/or (14) to $\mathrm{GPP_{RS}}$, we obtain stronger relaxations that are more computationally demanding than $\mathrm{GPP_{RS}}$. In the section with numerical results we will show the trade-off between the strengths of the $\mathrm{GPP_{RS}}$ bound without, and with (13) and/or (14), and the computational effort required to compute these relaxations.

The following result relates the new relaxation $\mathrm{GPP_{RS}}$ with $\mathrm{GPP_{ZWN}}$.

Theorem 7. *The SDP relaxation* $\mathrm{GPP_{ZWN}}$ *dominates the SDP relaxation* $\mathrm{GPP_{RS}}$*.*

Proof. Let (Y, y) be feasible for $\mathrm{GPP_{ZWN}}$ and of block form (2). We construct from Y a feasible point $\widetilde{Y} \in \mathcal{S}_n$ for $\mathrm{GPP_{RS}}$ in the following way:

$$\widetilde{Y} := \sum_{j=1}^{k} Y^{(jj)}.$$

Clearly, $\widetilde{Y} \geq 0$. From Theorem 3 and Corollary 4 it follows that

$$\operatorname{diag}(\widetilde{Y}) = \operatorname{diag}(\sum_{j=1}^{k} Y^{(jj)}) = u_n,$$

and

$$\operatorname{tr}(J\widetilde{Y}) = \sum_{j=1}^{k} \operatorname{tr}(JY^{(jj)}) = \sum_{j=1}^{k} m_j^2.$$

To prove that $k\widetilde{Y} - J_n \succeq 0$ we use

$$\begin{pmatrix} 1 & y^{\mathrm{T}} \\ y & Y \end{pmatrix} \in \mathcal{S}^+_{nk+1},$$

which implies that

$$\sum_{i=1}^{k} \begin{pmatrix} 1 & Y^{(0i)} \\ Y^{(i0)} & Y^{(ii)} \end{pmatrix} = \begin{pmatrix} k & u_n^{\mathrm{T}} \\ u_n & \widetilde{Y} \end{pmatrix} \succeq 0.$$

Now, the positive semidefinite constraint follows from the Schur complement theorem. It remains to show that the objectives coincide. Indeed,

$$\operatorname{tr}(I_k \otimes L)Y = \sum_{j=1}^{k} \operatorname{tr}(LY^{(jj)}) = \operatorname{tr}(L\widetilde{Y}).$$

□

Numerical experiments show that there are graphs for which relaxations $\mathrm{GPP_{ZWN}}$ and $\mathrm{GPP_{RS}}$ provide the same optimal values. On the other hand, our numerical results show that for $k \geq 3$, $\mathrm{GPP_{RS}}$ provides stronger bounds than $\mathrm{GPP_{ZW}}$ (for details see Section 6).

3.2 The Frieze-Jerrum relaxation

In [20], Frieze and Jerrum derived a SDP relaxation for the max-k-cut problem. The max-k-cut problem partitions the vertex set into at most k subsets such that the total weight of edges joining different sets is maximized. Note that here there is no restriction on the sizes of the subsets. The relaxation from [20] takes the form

$$
(\text{k–MC}_{\text{FJ}}) \qquad \begin{array}{ll} \max & \frac{k-1}{2k}\operatorname{tr}(LY) \\ \text{s.t.} & \operatorname{diag}(Y) = u_n \\ & Y_{ij} \geq -\frac{1}{k-1}, \quad i \neq j \\ & Y \succeq 0. \end{array}
$$

The SDP relaxation k–MC$_{\text{FJ}}$ was used to derive an approximation algorithm for the max-k-cut, see [20]. This form of the relaxation was also used in [16] to partition a vertex set of a graph into at most k subsets such that the total weight of edges in the induced subgraphs is minimized.

Since in the max-k-cut problem there is no restriction on the sizes of the subsets, it differs from the GPP of our interest. Nevertheless, we compare bounds for these two problems in the following way. For a given graph and k, we compute the SDP bound k–MC$_{\text{FJ}}$ and compare it with the SDP bounds GPP$_{\text{RS}}$ for the maximum k-partitions that are computed for all combinations of $(m_1, \ldots, m_p)^{\mathrm{T}}$ such that $m_1 + \ldots + m_p = n$, $p = 2, \ldots, k$, see Section 6.

Note that one can also add a constraint to k–MC$_{\text{FJ}}$ that involves the restrictions of the subset sizes. This results in the following SDP relaxation

$$
(\text{GPP}_{\text{FJ}}) \qquad \begin{array}{ll} \min & \frac{k-1}{2k}\operatorname{tr}(LY) \\ \text{s.t.} & \operatorname{diag}(Y) = u_n \\ & \operatorname{tr}(JY) = \frac{1}{k-1}(k\sum\limits_{i=1}^{k} m_i^2 - n^2) \\ & Y_{ij} \geq -\frac{1}{k-1}, \quad i \neq j \\ & Y \succeq 0. \end{array}
$$

Although this relaxation has not been considered in the literature, we use the abbreviation FJ to emphasize that this model is motivated by k–MC$_{\text{FJ}}$ due to Frieze and Jerrum [20]. It turns out that GPP$_{\text{FJ}}$ is equivalent to GPP$_{\text{RS}}$.

Theorem 8. *The SDP relaxations GPP*$_{\text{RS}}$ *and GPP*$_{\text{FJ}}$ *are equivalent.*

Proof. Let $Y \in \mathcal{S}_n^+$ be feasible for GPP$_{\text{FJ}}$. We construct from Y a feasible point $Z \in \mathcal{S}_n$ for GPP$_{\text{RS}}$. Namely, we set $Z := ((k-1)Y + J)/k$. By direct verification it follows that $\operatorname{diag}(Z) = u_n$, $\operatorname{tr}(JZ) = \sum_{i=1}^{k} m_i^2$, and $Z \geq 0$. Also,

$$
kZ - J = (k-1)Y + J - J = (k-1)Y \succeq 0,
$$

and

$$
\operatorname{tr}(LZ) = \frac{k-1}{k}\operatorname{tr}(LY) + \frac{1}{k}\operatorname{tr}(JL) = \frac{k-1}{k}\operatorname{tr}(LY).
$$

Conversely, let $Z \in \mathcal{S}_n$ be feasible for GPP$_{\text{RS}}$, and set $Y := (kZ - J)/(k-1)$. It follows by direct verification that Z is feasible for GPP$_{\text{FJ}}$. It is also easy to see that the two objectives coincide. $\square$

4 The graph equipartition problem

The graph equipartition is a special case of the GPP where the vertices of the graph are partitioned into subsets of the same size. This problem is studied in detail in [45]. Here, we relate the known relaxations for the GEP with GPP_{RS}.

For the graph equipartition problem Karisch and Rendl [34] derive the following SDP relaxation with matrix variable also in $\mathcal{S}_n$:

$$
\text{(k–GEP}_{\text{KR}}) \qquad \begin{aligned} \min \quad & \tfrac{1}{2}\operatorname{tr}(LY) \\ \text{s.t.} \quad & \operatorname{diag}(Y) = u_n, \quad Yu_n = \bar{m}u_n \\ & Y \geq 0, \; Y \succeq 0, \end{aligned}
$$

where $\bar{m} = \frac{n}{k}$. Note that k–GEP_{KR} is *not* a relaxation for the general GPP. In [45] it is proven that, when restricted to the GEP case, relaxations GPP_{ZWN}, GPP_{QAP}, and k–GEP_{KR} are equivalent. We show here that also k–GEP_{KR} and GPP_{RS} are equivalent.

Theorem 9. *Let $n = mk$. When restricted to the equipartition case, the SDP relaxation* GPP_{RS} *is equivalent to* k–GEP_{KR}.

Proof. Let $Z \in \mathcal{S}_n^+$ be feasible for k–GEP_{KR}. Then, $\operatorname{diag}(Z) = u_n$ and

$$
\operatorname{tr}(JZ) = u_n^{\mathrm{T}} Z u_n = \frac{n}{k} u_n^{\mathrm{T}} u_n = \frac{n^2}{k}.
$$

To finish this part of the proof we need to show that $kZ - J_n \succeq 0$. Since u_n is an eigenvector of Z with corresponding eigenvalue n/k, the eigenvalue decomposition of Z is

$$
Z = \frac{1}{k} J_n + \sum_{i=2}^{n} \lambda_i q_i q_i^{\mathrm{T}},
$$

where λ_i and q_i are the eigenvalues and eigenvectors of Z, respectively. It follows that $kZ - J_n \succeq 0$.

Conversely, let $Y \in \mathcal{S}_n$ be feasible for GPP_{RS}. Then from $kY - J_n \succeq 0$ it follows that $Y \succeq \frac{1}{k} J_n \succeq 0$. It remains only to show that $Yu_n = \frac{n}{k} u_n$. This follows from

$$
kY - J \succeq 0 \quad \text{and} \quad u_n^{\mathrm{T}} (kY - J) u_n = 0.
$$

□

By collecting all results from this paper and [45], it follows that for the graph equipartition problem the following relaxations are equivalent: k–GEP_{KR}, GPP_{RS}, GPP_{FJ}, GPP_{ZWN}, and GPP_{QAP}.

5 The graph bisection problem

The graph bisection problem is a special case of the GPP where $k = 2$, $m_1 \geq m_2$, $m_1 + m_2 = n$. We restrict here to $m_1 > m_2$. In this section we compare several SDP relaxations for the GBP. In particular, we compare the improved Wolkowicz-Zhao relaxation and the Zhao-Karisch-Rendl-Wolkowicz relaxation [48] with the relaxation by Karisch, Rendl and Clausen [35], and then our new relaxation with the latter.

In Section 2.2 we showed that the GPP is a special case of the QAP. Here we prove that when $k = 2$, the SDP relaxation $\mathrm{GPP}_{\mathrm{ZWN}}$ is equivalent to $\mathrm{GPP}_{\mathrm{QAP}}$ where

$$B := \begin{pmatrix} 0_{m_1 \times m_1} & J_{m_1 \times m_2} \\ J_{m_2 \times m_1} & 0_{m_2 \times m_2} \end{pmatrix} \in \mathcal{S}_n.$$

To show this, we use the result that the SDP relaxation $\mathrm{GPP}_{\mathrm{QAP}}$ for the GBP reduces to the following SDP relaxation:

$$\begin{array}{ll} \min & \frac{1}{2} \operatorname{tr} A(X_3 + X_4) \\ \text{s.t.} & X_1 + X_5 = I_n \\ & \sum_{i=1}^{6} X_i = J_n \\ & \operatorname{tr}(JX_i) = s_i, \quad X_i \geq 0, \quad i = 1, \ldots, 6 \\ & X_1 - \frac{1}{m_1 - 1} X_2 \succeq 0, \quad X_5 - \frac{1}{m_2 - 1} X_6 \succeq 0 \\ & \begin{pmatrix} \frac{1}{m_1}(X_1 + X_2) & \frac{1}{\sqrt{m_1 m_2}} X_3 \\ \frac{1}{\sqrt{m_1 m_2}} X_4 & \frac{1}{m_2}(X_5 + X_6) \end{pmatrix} \succeq 0, \\ & X_3 = X_4^{\mathrm{T}}, \; X_1, X_2, X_5, X_6 \in \mathcal{S}_n, \end{array} \tag{15}$$

where $s_1 = m_1$, $s_2 = m_1(m_1 - 1)$, $s_3 = m_1 m_2$, $s_4 = m_1 m_2$, $s_5 = m_2$, and $s_6 = m_2(m_2 - 1)$, see [10].

Theorem 10. *Let $m_1 > m_2$, $m_1 + m_2 = n$. Then the SDP relaxations $\mathrm{GPP}_{\mathrm{QAP}}$ and $\mathrm{GPP}_{\mathrm{ZWN}}$ are equivalent.*

Proof. Let $Y \in \mathcal{S}^+_{n^2}$ be feasible for $\mathrm{GPP}_{\mathrm{QAP}}$ with block form (2) where $Y^{(ij)} \in \mathbb{R}^{n \times n}$, $i, j = 1, \ldots, n$. We construct from $Y \in \mathcal{S}^+_{n^2}$ a feasible point (W, w) with $W \in \mathcal{S}_{2n}$ for $\mathrm{GPP}_{\mathrm{ZWN}}$ in the following way. First, define blocks

$$W^{(11)} := \sum_{i,j=1}^{m_1} Y^{(ij)}, \quad W^{(12)} := \sum_{i=1}^{m_1} \sum_{j=m_1+1}^{n} Y^{(ij)}, \quad W^{(22)} := \sum_{i,j=m_1+1}^{n} Y^{(ij)}, \tag{16}$$

and then collect all four blocks into the matrix

$$W = \begin{pmatrix} W^{(11)} & W^{(12)} \\ W^{(21)} & W^{(22)} \end{pmatrix}, \tag{17}$$

where $W^{(21)} = (W^{(12)})^{\mathrm{T}}$, and $w := \operatorname{diag}(W)$. The sparsity pattern $\operatorname{tr}((J_2 - I_2) \otimes I_n)W = 0$ follows from the sparsity pattern of Y, i.e., from $\operatorname{tr}((J_n - I_n) \otimes I_n)Y = 0$. By direct verification it follows that

$$\operatorname{tr}(I_2 \otimes J_n)W + \operatorname{tr}(W) = -(m_1^2 + m_2^2 + n) + 2((m_1 + 1, m_2 + 1) \otimes u_n^{\mathrm{T}})w.$$

It remains only to prove that

$$\begin{pmatrix} 1 & w^T \\ w & W \end{pmatrix} \succeq 0. \tag{18}$$

In [24] it is proven that if $W \in \mathcal{S}_{2n}$ such that $\mathrm{diag}(W) = cWu_{2n}$ for some $c \in \mathbb{R}$, then (18) holds if and only if $\mathrm{tr}(JW) \geq \mathrm{tr}(W)^2$ and $W \succeq 0$. From the valid equalities for $\mathrm{GPP}_{\mathrm{QAP}}$ (see [48]) it follows that

$$Wu_{2n} = n\,\mathrm{diag}(W), \quad \mathrm{tr}(JW) = n^2, \quad \mathrm{tr}(W) = n.$$

To finish this part of the proof let $\widetilde{x} \in \mathbb{R}^{n^2}$ be defined by

$$\widetilde{x}^{\mathrm{T}} := \left(u_{m_1}^{\mathrm{T}} \otimes x_{1:n}^{\mathrm{T}}, u_{m_2}^{\mathrm{T}} \otimes x_{n+1:2n}^{\mathrm{T}}\right)$$

for any $x \in \mathbb{R}^{2n}$. Then

$$x^{\mathrm{T}}Wx = \widetilde{x}^{\mathrm{T}}Y\widetilde{x} \geq 0,$$

since $Y \succeq 0$, and we can apply the result from [24].

Conversely, let (W, w) be feasible for $\mathrm{GPP}_{\mathrm{ZWN}}$ and suppose that W has the block form (17). By exploiting the fact that relaxations (15) and $\mathrm{GPP}_{\mathrm{QAP}}$ are equivalent, we define:

$$\begin{aligned} X_1 &:= \mathrm{Diag}(\mathrm{diag}(W^{(11)})), & X_2 &:= W^{(11)} - X_1, \\ X_5 &:= \mathrm{Diag}(\mathrm{diag}(W^{(22)})), & X_6 &:= W^{(22)} - X_5, \\ X_3 &:= W^{(12)}, & X_4 &:= W^{(21)}. \end{aligned}$$

From Theorem 3, it follows that

$$\sum_{i,j=1}^{2} W^{(ij)} = \sum_{i=1}^{6} X_i = J_n, \quad X_1 + X_5 = I, \quad \text{and} \quad \mathrm{tr}(JX_i) = s_i, \quad \forall i.$$

It only remains to show the linear matrix inequalities. Note that

$$\begin{aligned} X_1 - \tfrac{1}{m_1-1}X_2 &= \tfrac{1}{m_1-1}(m_1\,\mathrm{Diag}(\mathrm{diag}(W^{(11)})) - W^{(11)}) \\ &= \tfrac{1}{m_1-1}(\mathrm{Diag}(W^{(11)}u_n) - W^{(11)}), \end{aligned}$$

where the last equality follows from Theorem 3. Now, for any $x \in \mathbb{R}^n$ we have

$$x^{\mathrm{T}}(\mathrm{Diag}(W^{(11)}u_n) - W^{(11)})x = \sum_{i \neq j} W_{ij}^{(11)}(x_i - x_j)^2 \geq 0,$$

which shows that $X_1 - \frac{1}{m_1-1}X_2 \succeq 0$. Similarly, one can show that $X_5 - \frac{1}{m_2-1}X_6 \succeq 0$. Finally, for any $x \in \mathbb{R}^{2n}$ let $\tilde{x} \in \mathbb{R}^{2n}$ be defined by

$$\tilde{x} := \left[\frac{1}{\sqrt{m_1}}x_{1:n}^{\mathrm{T}}, \frac{1}{\sqrt{m_2}}x_{n+1:2n}^{\mathrm{T}}\right],$$

then

$$x^{\mathrm{T}} \begin{pmatrix} \frac{1}{m_1}(X_1 + X_2) & \frac{1}{\sqrt{m_1 m_2}}X_3 \\ \frac{1}{\sqrt{m_1 m_2}}X_4 & \frac{1}{m_2}(X_5 + X_6) \end{pmatrix} x = \tilde{x}^{\mathrm{T}}W\tilde{x} \geq 0.$$

It remains to show that the objective values coincide for any pair of feasible solutions $(Y, (W, w))$ that are related as described. This follows trivially from (11) and (16). $\square$

Note that for the graph equipartition problem it is also known that the relaxations $\mathrm{GPP_{ZWN}}$ and $\mathrm{GPP_{QAP}}$ are equivalent, see [45].

Karisch, Rendl, and Clausen [35] consider the following SDP relaxation for the GBP

$$
(\mathrm{GBP_{KRC}}) \qquad \begin{array}{rl} \min & \frac{1}{4}\operatorname{tr}(A(J-X)) \\ \text{s.t.} & \operatorname{diag}(X) = u_n \\ & \operatorname{tr}(JX) = (m_1 - m_2)^2 \\ & X \in \mathcal{S}_n^+. \end{array}
$$

The relaxation $\mathrm{GBP_{KRC}}$ with added triangle inequalities (13) was implemented within a branch-and-bound framework to solve instances of the GBP with 80 to 90 vertices, see [35]. The same relaxation was later used to derive approximation algorithms for the maximum bisection problem, see [17, 25].

In the sequel we compare $\mathrm{GBP_{KRC}}$ with $\mathrm{GPP_{RS}}$ and $\mathrm{GPP_{ZWN}}$. In [10] De Klerk et al. show that the optimal value of (15) is at least that of $\mathrm{GBP_{KRC}}$, and can be strictly greater for some instances. Therefore, the following result follows trivially.

Corollary 11. *Let $m_1 > m_2$, $m_1+m_2 = n$. Then the SDP relaxation* $\mathrm{GPP_{ZWN}}$ *dominates the SDP relaxation* $\mathrm{GBP_{KRC}}$.

In the following theorem we relate $\mathrm{GBP_{KRC}}$ and $\mathrm{GPP_{RS}}$ (or equivalently $\mathrm{GPP_{FJ}}$).

Theorem 12. *Let $m_1 > m_2$, $m_1 + m_2 = n$. Then the SDP relaxations* $\mathrm{GBP_{KRC}}$ *and* $\mathrm{GPP_{RS}}$ *are equivalent.*

Proof. Let $X \in \mathcal{S}_n^+$ be feasible for $\mathrm{GBP_{KRC}}$, and set $Y := (J_n + X)/2$. Now the result follows by direct verification.

Conversely, let $Y \in \mathcal{S}_n^+$ be feasible for $\mathrm{GBP_{RS}}$, and set $X = 2Y - J_n$. It is clear that $\operatorname{diag}(X) = u_n$ and $X \succeq 0$. By direct verification we have $\operatorname{tr}(J_nY) = (m_1 - m_2)^2$ and $\frac{1}{2}\operatorname{tr}(LX) = \frac{1}{4}\operatorname{tr}(A(J-X))$. □

We summarize the relations between the presented SDP relaxations of the GBP in Figure 1. In the diagram the arrow points from a weaker to a stronger relaxation. We also indicate where one can find a proof of the relation between the two relaxations. Although numerical experiments show that $\mathrm{GPP_{RS}}$ (equivalently $\mathrm{GBP_{KRC}}$) and $\mathrm{GPP_{ZW}}$ provide the same bounds for *all* test instances, we could not prove that these two relaxations are equivalent.

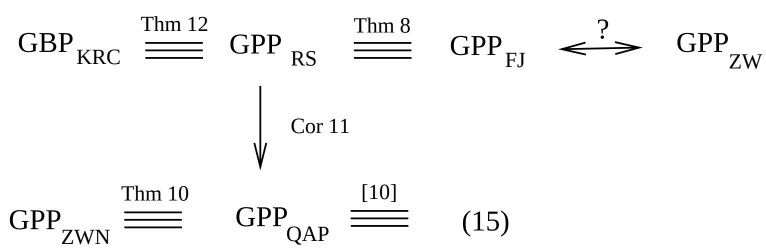


Figure 1: Relations between the SDP relaxations of the bisection problem.

6 Numerical results

In this section we present numerical results. All relaxations are solved with SeDuMi [46] using the Yalmip interface [38] on an Intel Xeon X5680, 3.33 GHz dual-core processor with 32 GB memory.

6.1 The GPP with more than two subsets

6.1.1 Random graphs

We first compare the SDP relaxations GPP_{ZWN}, GPP_{ZW}, and GPP_{RS} on randomly generated graphs with 30 vertices for the 3, 4, and 5-partition problem. Each edge in a graph is generated independently of other edges with probability $p = 0.5$. For any given p, a graph formulated in the described way is known as the Erdös-Rényi random graph $G_p(|V|)$. The Erdös-Rényi random graph was initiated by Erdös and Rényi in 1959, see [30, 31].

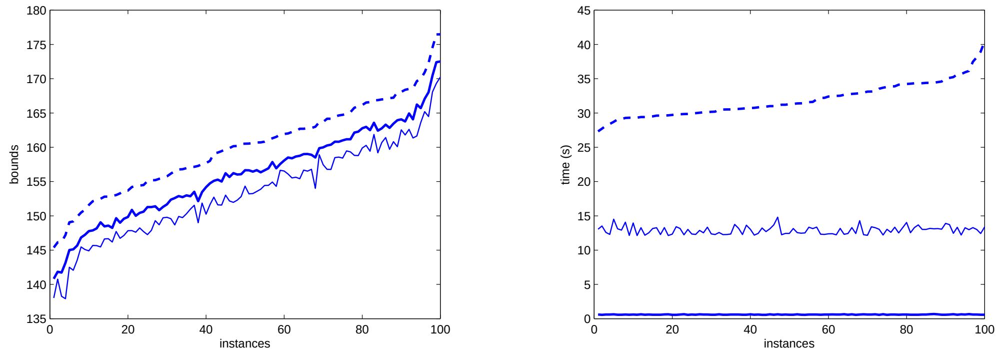
In Figure 2 (a) the bounds GPP_{ZWN}, GPP_{ZW}, and GPP_{RS} are plotted for 100 graphs $G_{0.5}(30)$ in the case of the 3-partition problem where $m = (15, 10, 5)^{\mathrm{T}}$. These lower bounds are sorted w.r.t. increasing values of GPP_{ZWN}. The dashed line represents GPP_{ZWN}, the thin line GPP_{ZW}, and the thick line GPP_{RS}. From Figure 2 (a) it is clear that GPP_{ZWN} dominates the other two bounds, and that GPP_{RS} dominates GPP_{ZW}. (It is interesting that for the 3-partition problem we were able to find randomly generated graphs with 15 vertices for which GPP_{RS} and GPP_{ZWN} provide the same bounds.) Figure 2 (b) contains the computation times required for solving the relaxations. Here, the times are sorted w.r.t. increasing computation times required to solve GPP_{ZWN}. While the average time for solving the strongest relaxation is 32 seconds, and for GPP_{ZW} it is 13 seconds, for the new relaxation it is only 0.6 seconds.

We did similar experiments on 100 Erdös-Rényi random graphs $G_{0.5}(30)$ for the 4-partition problem where $m = (15, 10, 3, 2)^{\mathrm{T}}$ (see Figure 3), and the 5-partition problem where $m = (10, 10, 5, 3, 2)^{\mathrm{T}}$ (see Figure 4). Clearly, our bound dominates GPP_{ZW} in all instances. However, the computation times to solve our relaxation remains below one second, independent of k, while the computation times to solve GPP_{ZW} and GPP_{ZWN} significantly increase with k (see Figure 3 (b) and Figure 4 (b)).

The three relaxations are also compared on 50 Erdös-Rényi random graphs $G_{0.75}(20)$ for the 6-partition problem. The outcome of the computations is that in *all* instances our relaxation and GPP_{ZWN} provide the same bound, while GPP_{ZW} is significantly weaker, see Figure 5. To solve GPP_{ZWN} one requires 5GB memory, while to solve GPP_{RS} the computational effort is negligible.

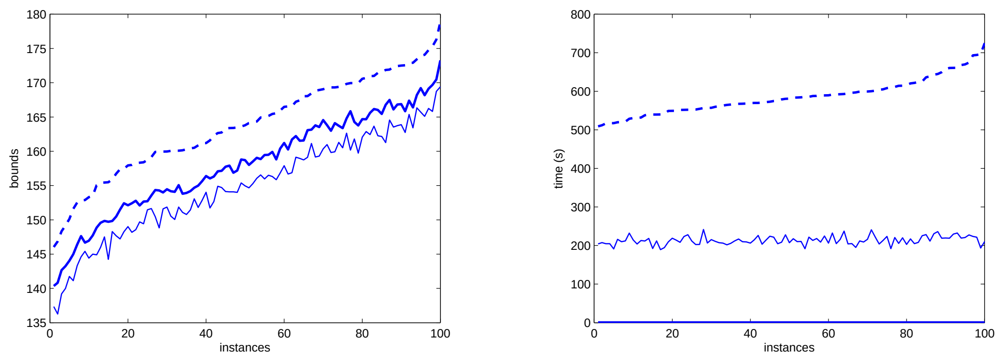
Besides this we have conducted all described tests for different values of the edge probability p, and for randomly generated weighted graphs. (We derive a randomly generated weighted graph in a similar way as the Erdös-Rényi random graph, but assign random numbers from the open interval (0,1) as weights to the edges.) The results show that the quality of GPP_{RS} does *not* depend on the density of the graph, or on whether the graph is weighted or not.

Our bound can be improved by adding triangle constraints (13) and/or independent set type of constraints (14). In Table 1 we present results obtained for $G_{0.5}(100)$ and the 3-partition problem where $m = (60, 30, 10)^{\mathrm{T}}$. Here we compare bounds for all three SDP relaxations with the bounds that are obtained by iteratively adding the most violated inequalities of type (13) and/or (14) to GPP_{RS}, see the second row of the table.



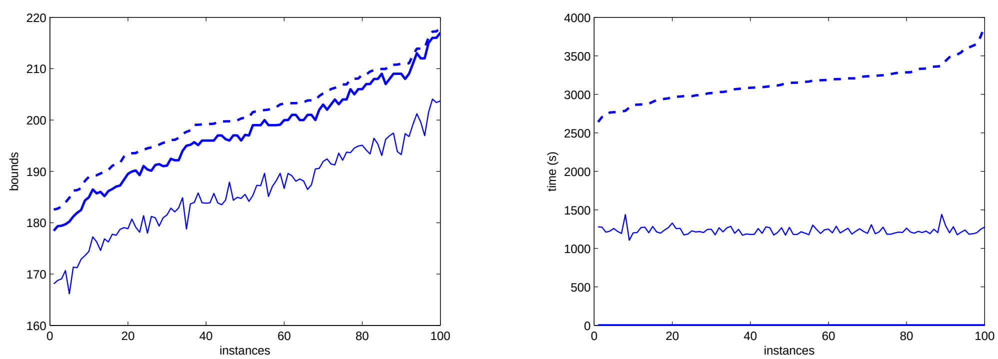
(a) bounds (b) computation times (s)

Figure 2: 3-partition: dashed line is GPP_{ZWN}, thin line GPP_{ZW}, and thick line GPP_{RS}



(a) bounds (b) computation times (s)

Figure 3: 4-partition: dashed line is GPP_{ZWN}, thin line GPP_{ZW}, and thick line GPP_{RS}



(a) bounds (b) computation times (s)

Figure 4: 5-partition: dashed line is GPP_{ZWN}, thin line GPP_{ZW}, and thick line GPP_{RS}

The numerical experiments show that the triangle inequalities are stronger than the independent set inequalities, in the sense that adding only the triangle inequalities leads to better bounds than adding only the independent set inequalities. Here the cutting plane scheme adds at most 5000 most violated cuts in each iteration and iterates until no more inequalities are violated. In the third row of Table 1 the computational times required for

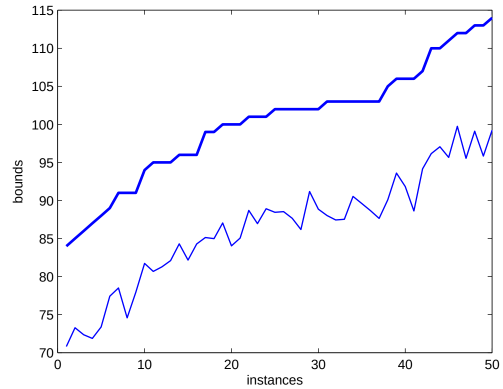


Figure 5: 6-partition: thick line represents $\mathrm{GPP_{RS}}$ and $\mathrm{GPP_{ZWN}}$, thin line $\mathrm{GPP_{ZW}}$

solving the relaxations are given. The results show that solving $\mathrm{GPP_{ZWN}}$ requires much more time than adding 14359 triangle inequalities (13) to $\mathrm{GPP_{RS}}$. After solving $\mathrm{GPP_{RS}}$ there are only 754 violated constraints of type (14). Therefore the computational time to solve $\mathrm{GPP_{RS}}$ with added violated constraints of type (14) is not large.

We have also tested other strategies of adding cuts. Our experiments show that the best strategy for graphs with 100 vertices is to add (up till) the 500 most violated cuts at once, and restrict to 5 rounds of adding cuts. By doing so, we computed the bound 1648 in one hour and six minutes for the graph whose results are reported in Table 1. It is clear that there is a trade-off between the quality of the bound obtained by adding cuts and the computational time. Since in cutting plane approaches it is common to have a tailing-off effect, it is reasonable to stop earlier iterations of the cutting plane algorithm. Similar results are also reported in [34].

	$\mathrm{GPP_{ZWN}}$	$\mathrm{GPP_{ZW}}$	$\mathrm{GPP_{RS}}$	$\mathrm{GPP_{RS}}$+(13)	$\mathrm{GPP_{RS}}$+(14)	$\mathrm{GPP_{RS}}$+(13, 14)
bound	1660	1627	1629	1649	1631	1650
time	27:58:59	09:00:37	00:11:40	01:58:12	00:33:53	03:12:20

Table 1: Bounds for $G_{0.5}(100)$, $m = (60, 30, 10)^{\mathrm{T}}$. The time is given in hr:min:s.

In Table 2 we have $\mathrm{GPP_{ZWN}}$, $\mathrm{GPP_{ZW}}$, and $\mathrm{GPP_{RS}}$ for $k \in \{4, 5, 6\}$ of $G_{0.5}(|V|)$, where $|V| \in \{50, 60, 100\}$. The table shows that we couldn't solve $\mathrm{GPP_{ZWN}}$ and $\mathrm{GPP_{ZW}}$ relaxations for the 4-partition (resp. 6-partition) problem when $|V| = 100$ (resp. $|V| = 50$), and also $\mathrm{GPP_{ZWN}}$ for the 5-partition problem when $|V| = 60$. We managed to compute $\mathrm{GPP_{ZW}}$ for the 5-partition problem on a graph with 60 vertices. This computation took more than 2 days, but the obtained bound is weaker than the bound $\mathrm{GPP_{RS}}$ that was computed in 20 seconds. This table also shows that $\mathrm{GPP_{RS}}$ is solved easily for all test instances and partitions.

	m^{T}	$\mathrm{GPP}_{\mathrm{ZWN}}$	time	$\mathrm{GPP}_{\mathrm{ZW}}$	time	$\mathrm{GPP}_{\mathrm{RS}}$	time
$G_{0.5}(60)$	(20,20,15,5)	787	12:17:37	764	4:20:35	780	00:00:21
$G_{0.5}(100)$	(40,30,20,10)	n.a.		n.a.		2149	00:11:14
$G_{0.5}(60)$	(20,15,10,10,5)	n.a.		823	50:15:35	852	00:00:20
$G_{0.5}(50)$	(15,10,10,5,5,5)	n.a.		n.a.		625	00:00:10

Table 2: Bounds for $G_{0.5}(|V|)$. The time is given in hr:min:s.

6.1.2 Rudy instances

We compare here the SDP relaxations $\mathrm{GPP}_{\mathrm{ZWN}}$, $\mathrm{GPP}_{\mathrm{ZW}}$, and $\mathrm{GPP}_{\mathrm{RS}}$ on the following types of graphs, generated by the rudy graph generator [42] (and of which most were also used in [21]).

- `clique`: Complete graphs with the edge weight of edge (i, j) being $|i - j|$.
- `grid_2D`: Planar unweighted grid graphs, where $|V| = (\#\text{ rows}) \times (\#\text{ columns})$.
- `spinglass2pm`: Toroidal two-dimensional grid graphs with ± 1 weights, where $|V| = (\#\text{ rows}) \times (\#\text{ columns})$. The percentage of negative weights is 50%.
- `spinglass3pm`: Toroidal three-dimensional grid graphs with ± 1 weights, where $|V| = (\#\text{ rows}) \times (\#\text{ columns}) \times (\#\text{ layers})$. The percentage of negative weights is 50%.

Table 3 shows the computational results for `grid_2D` instances where $k = 3, 4, 5, 6$. Table 4 presents the computational results for `clique` instances where $k = 3$. Table 5 (resp. Table 6) shows the computational results for `spinglass2pm` and `spinglass3pm` where $k = 3$ (resp. $k = 4$). Results presented in Table 3 are obtained by solving the GPP as a minimization problem, whereas the other tables present results for the GPP as a maximization problem. We assign arbitrary values for the subset sizes. Also, we round up (resp. down) the bounds to the closest integer for the minimization (resp. maximization) problems. The computational results lead to the following observations:

- The SDP relaxation $\mathrm{GPP}_{\mathrm{ZWN}}$ provides the best bounds with significant computational effort.
- The SDP relaxation $\mathrm{GPP}_{\mathrm{RS}}$ (equivalently $\mathrm{GPP}_{\mathrm{FJ}}$) provides good bounds and requires considerable less computational effort than $\mathrm{GPP}_{\mathrm{ZWN}}$.
- The relaxations $\mathrm{GPP}_{\mathrm{RS}}$ and $\mathrm{GPP}_{\mathrm{ZWN}}$ provide the same bounds for several `grid_2D` instances and $k = 3, 4, 5, 6$.
- The SDP bounds $\mathrm{GPP}_{\mathrm{RS}}$ and $\mathrm{GPP}_{\mathrm{ZWN}}$ differ when $k = 6$. Note that for the random instances of size 20 and $k = 6$ we obtained the same bounds for both relaxations, see Section 6.1.1.
- The numerical results indicate that it is harder to solve the above mentioned structured instances than the random ones of Section 6.1.1.

$\lvert V \rvert$	m^{T}	$\mathrm{GPP}_{\mathrm{ZWN}}$	time	$\mathrm{GPP}_{\mathrm{ZW}}$	time	$\mathrm{GPP}_{\mathrm{RS}}$	time
3×3	$(4,3,2)$	5	00:00:00	4	00:00:00	5	00:00:00
4×4	$(6,5,5)$	6	00:00:01	5	00:00:01	6	00:00:00
5×5	$(10,10,5)$	7	00:00:09	5	00:00:05	6	00:00:00
6×6	$(14,12,10)$	8	00:01:51	5	00:00:54	7	00:00:02
7×7	$(18,16,15)$	8	00:15:25	6	00:07:09	7	00:00:06
8×8	$(26,22,16)$	8	01:23:28	5	00:37:47	7	00:00:35
9×9	$(35,30,16)$	9	06:22:55	5	02:32:34	6	00:03:18
10×10	$(50,25,25)$	8	26:10:51	5	09:30:50	6	00:13:19
3×3	$(3,3,2,1)$	7	00:00:01	5	00:00:00	7	00:00:00
4×4	$(5,4,4,3)$	9	00:00:08	5	00:00:03	8	00:00:00
5×5	$(10,5,5,5)$	10	00:01:56	5	00:01:10	8	00:00:00
6×6	$(10,10,8,8)$	11	00:25:17	6	00:11:24	10	00:00:01
7×7	$(30,10,5,4)$	11	02:55:17	4	01:18:35	5	00:00:05
8×8	$(30,20,10,4)$	11	20:00:09	5	07:05:07	7	00:00:36
3×3	$(3,2,2,1,1)$	8	00:00:02	5	00:00:01	8	00:00:00
4×4	$(4,4,4,2,2)$	10	00:00:30	5	00:00:15	10	00:00:00
5×5	$(8,6,6,3,2)$	12	00:13:19	6	00:06:17	10	00:00:00
6×6	$(10,10,5,5,6)$	14	02:18:03	7	01:06:41	12	00:00:02
7×7	$(20,10,10,5,4)$	14	17:22:13	6	08:01:12	10	00:00:08
3×3	$(2,2,2,1,1,1)$	9	00:00:03	5	00:00:03	9	00:00:00
4×4	$(4,4,3,2,2,1)$	12	00:02:43	6	00:01:29	12	00:00:00
5×5	$(7,6,5,3,2,2)$	14	00:55:38	6	00:25:09	12	00:00:00
6×6	$(10,8,5,5,6,2)$	16	08:19:42	6	04:23:12	14	00:00:02

Table 3: Computational results for the GPP as a minimization problem for `grid_2D` instances where $k = 3,4,5,6$. The time is given in hr:min:s.

6.2 The minimum bisection problem

6.2.1 Random graphs

In Figure 6 we compare all known SDP relaxations for the bisection problem on 100 randomly generated weighted graphs with 60 vertices and $m = (40,20)^{\mathrm{T}}$. In Figure 6 (a) the dashed line represents $\mathrm{GPP}_{\mathrm{ZWN}}$, and the thick line $\mathrm{GPP}_{\mathrm{RS}}$ that is proven to be equivalent to $\mathrm{GBP}_{\mathrm{KRC}}$ and $\mathrm{GPP}_{\mathrm{FJ}}$, see Theorem 12 and Theorem 8 respectively. These lower bounds are sorted w.r.t. increasing values of $\mathrm{GPP}_{\mathrm{ZWN}}$ bounds. The numerical results suggest that the relaxations $\mathrm{GPP}_{\mathrm{ZW}}$ and $\mathrm{GPP}_{\mathrm{RS}}$ are equivalent since we obtain the same bounds for *all* test instances. (We have compared also these two relaxations on 100 instances for $n \in \{40,50,80\}$ and always obtained the same optimal values for both relaxations.) Unfortunately, we were not able to theoretically prove the conjecture that the relaxations $\mathrm{GPP}_{\mathrm{ZW}}$ and $\mathrm{GPP}_{\mathrm{RS}}$ are equivalent. Figure 6 (b) contains the computation times required for solving the relaxations. Here, the times are sorted w.r.t. increasing computation times required to solve $\mathrm{GPP}_{\mathrm{ZWN}}$. The results show that among all relaxations that provide

$\|V\|$	m^{T}	$\mathrm{GPP}_{\mathrm{ZWN}}$	time	$\mathrm{GPP}_{\mathrm{ZW}}$	time	$\mathrm{GPP}_{\mathrm{RS}}$	time
20	$(10, 5, 5)$	1,128	00:00:03	1,286	00:00:01	1,153	00:00:00
30	$(15, 10, 5)$	3,757	00:00:30	4,238	00:00:16	3,845	00:00:01
40	$(20, 10, 10)$	9,029	00:04:34	10,294	00:02:09	9,228	00:00:03
50	$(20, 20, 10)$	18,046	00:27:39	20,222	00:08:23	18,244	00:00:10
60	$(40, 10, 10)$	25,000	00:34:09	29,282	00:28:15	27,308	00:00:29
70	$(30, 20, 20)$	50,132	02:21:37	56,669	01:14:34	50,534	00:01:53
80	$(50, 20, 10)$	63,000	04:35:18	72,498	02:42:05	67,207	00:05:06
90	$(40, 30, 20)$	105,304	11:42:47	118,838	05:48:14	106,568	00:10:21
100	$(60, 25, 15)$	127,500	26:08:57	146,944	11:38:51	134,732	00:22:49

Table 4: Computational results for the GPP as a maximization problem and for `clique` instances where $k = 3$. The time is given in hr:min:s.

$\|V\|$	m^{T}	$\mathrm{GPP}_{\mathrm{ZWN}}$	time	$\mathrm{GPP}_{\mathrm{ZW}}$	time	$\mathrm{GPP}_{\mathrm{RS}}$	time
4×4	$(8, 4, 4)$	12	00:00:01	15	00:00:01	13	00:00:00
5×5	$(10, 10, 5)$	20	00:00:12	24	00:00:05	22	00:00:00
6×6	$(20, 10, 6)$	29	00:01:51	36	00:00:54	31	00:00:02
7×7	$(25, 15, 9)$	40	00:15:10	47	00:07:10	43	00:00:10
8×8	$(22, 22, 20)$	56	01:08:06	65	00:37:49	56	00:00:40
9×9	$(40, 30, 11)$	69	08:06:28	78	02:44:07	72	00:03:43
$2 \times 3 \times 4$	$(12, 8, 4)$	21	00:00:11	25	00:00:04	23	00:00:00
$2 \times 4 \times 4$	$(12, 12, 8)$	33	00:00:48	38	00:00:33	34	00:00:01
$3 \times 3 \times 3$	$(15, 10, 2)$	26	00:00:18	28	00:00:08	28	00:00:00
$3 \times 3 \times 4$	$(15, 15, 6)$	37	00:02:06	42	00:00:58	40	00:00:01
$3 \times 4 \times 4$	$(25, 15, 8)$	48	00:12:47	55	00:06:13	53	00:00:06
$4 \times 4 \times 4$	$(30, 20, 14)$	71	01:27:00	81	00:38:00	74	00:00:35

Table 5: Computational results for the GPP as a maximization problem and for `spinglass2pm` and `spinglass3pm` instances where $k = 3$. The time is given in hr:min:s.

the same bounds, i.e., $\mathrm{GPP}_{\mathrm{ZW}}$, $\mathrm{GPP}_{\mathrm{RS}}$, and $\mathrm{GBP}_{\mathrm{KRC}}$, one can solve $\mathrm{GBP}_{\mathrm{KRC}}$ with the smallest effort.

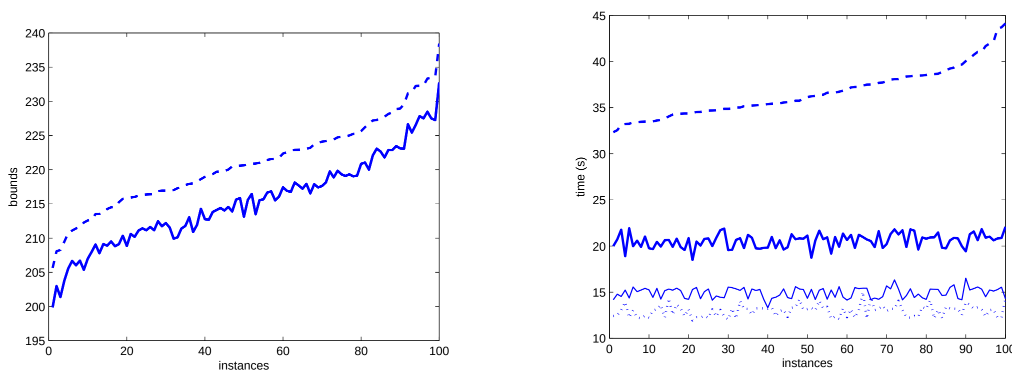
6.2.2 Graphs from the literature

We now compare SDP relaxations for the bisection problem on various classes of graphs from the literature. We consider instances from the following types of graphs that have less than 200 vertices.

- `compiler design instances`: These instances were introduced by Johnson, Mehorta, and Nemhauser [28]. They were also used in [3, 35] for solving the graph equipartition problem. We denote them with the initials `cd.xx.yy`, where `xx` is the number of vertices and `yy` the number of edges in the graph.

$\|V\|$	m^{T}	$\mathrm{GPP}_{\mathrm{ZWN}}$	time	$\mathrm{GPP}_{\mathrm{ZW}}$	time	$\mathrm{GPP}_{\mathrm{RS}}$	time
4×4	$(8, 4, 2, 2)$	12	00:00:09	16	00:00:04	13	00:00:00
5×5	$(10, 5, 5, 5)$	21	00:02:32	28	00:01:12	22	00:00:00
6×6	$(16, 10, 6, 4)$	30	00:33:13	40	00:12:53	32	00:00:02
7×7	$(20, 15, 10, 4)$	42	03:39:38	52	01:23:38	44	00:00:08
$2 \times 3 \times 4$	$(8, 8, 4, 4)$	22	00:01:48	28	00:00:54	23	00:00:00
$2 \times 4 \times 4$	$(10, 10, 8, 4)$	34	00:17:08	43	00:05:59	35	00:00:01
$3 \times 3 \times 3$	$(10, 10, 5, 2)$	28	00:04:47	34	00:02:09	30	00:00:00
$3 \times 3 \times 4$	$(10, 10, 10, 6)$	39	03:46:44	48	00:12:27	40	00:00:02
$3 \times 4 \times 4$	$(20, 15, 10, 3)$	52	03:17:44	61	01:13:23	55	00:00:06

Table 6: Computational results for the GPP as a maximization problem and for `spinglass2pm` and `spinglass3pm` instances where $k = 4$. The time is given in hr:min:s.



(a) bounds ($\mathrm{GBP}_{\mathrm{RS}}$ and $\mathrm{GPP}_{\mathrm{ZW}}$ coincide)

(b) computation times (s)

Figure 6: bisection: dashed line is $\mathrm{GPP}_{\mathrm{ZWN}}$, thin line $\mathrm{GPP}_{\mathrm{ZW}}$, thick line $\mathrm{GPP}_{\mathrm{RS}}$, and dotted line $\mathrm{GBP}_{\mathrm{KRC}}$

- `kkt instances`: These instances originate from nested dissection approaches for solving sparse symmetric linear systems. Each instance consists of a graph that represents the support structure of a sparse symmetric linear system, for details see [27]. These instances were also considered in [3, 4]. We denote them with the initials `kkt_name`.

- `mesh instances`: These instances arise from an application of the finite element methods [12]. They were solved as equipartition problems in [3, 35]. We denote them with the initials `mesh.xx.yy`, where `xx` is the number of vertices and `yy` the number of edges in the graph.

- `VLSI design instances`: These instances were created from data arising in the layout of electronic circuits. For details see Ferreira et al. [18]. They were also used in computations in [3, 4]. We denote them with the initials `vlsi.xx.yy` where `xx` is the number of vertices and `yy` the number of edges in the graph.

The computational results for the bisection problem that involve the above mentioned instances are presented in Table 7. We partition vertices of graphs into subsets of arbitrary sizes. Also, we round up the bounds to the closest integer. The results lead to the following observations:

- GPP_{ZW} and GBP_{KRC} (equivalently GPP_{RS}) provide the same bounds for all test instances.
- The SDP relaxation GPP_{ZWN} dominates GBP_{KRC} in all presented instances, except for kkt_lowt01 where all relaxations provide the same bound.
- There is only a marginal time difference for solving GPP_{ZW} and GBP_{KRC} for graphs up to 100 vertices.

Problem	$\|V\|$	m^T	GPP_{ZWN}	time	GPP_{ZW}	time	GBP_{KRC}	time
cd.30.47	30	$(20, 10)$	114	00:00:01	110	00:00:01	110	00:00:01
cd.30.56	30	$(20, 10)$	169	00:00:01	156	00:00:01	156	00:00:01
cd.45.98	45	$(25, 20)$	631	00:00:09	576	00:00:04	576	00:00:04
cd.47.99	47	$(25, 22)$	514	00:00:11	471	00:00:05	471	00:00:05
cd.47.101	47	$(25, 22)$	361	00:00:13	326	00:00:05	326	00:00:04
cd.61.187	61	$(40, 21)$	798	00:00:38	774	00:00:22	774	00:00:21
kkt_lowt01	82	$(42, 40)$	5	00:07:29	5	00:02:33	5	00:02:32
kkt_putt01	115	$(59, 56)$	22	01:07:48	20	00:19:26	20	00:19:22
mesh.35.54	35	$(22, 13)$	4	00:00:03	2	00:00:02	2	00:00:01
mesh.69.212	69	$(40, 29)$	2	00:02:16	2	00:00:45	2	00:00:36
mesh.70.120	70	$(50, 20)$	4	00:02:27	2	00:00:50	2	00:00:42
mesh.74.129	74	$(70, 4)$	4	00:01:58	1	00:01:10	1	00:01:00
mesh.137.231	137	$(100, 37)$	3	03:16:48	1	01:00:57	1	00:48:09
mesh.148.265	148	$(120, 28)$	5	05:31:32	1	01:32:01	1	01:21:00
vlsi.15.29	15	$(10, 5)$	16	00:00:00	11	00:00:00	11	00:00:00
vlsi.34.71	34	$(22, 12)$	6	00:00:02	4	00:00:01	4	00:00:01
vlsi.37.92	37	$(30, 7)$	6	00:00:03	3	00:00:02	3	00:00:02
vlsi.38.105	38	$(20, 18)$	86	00:00:04	84	00:00:02	84	00:00:02
vlsi.42.132	42	$(20, 22)$	99	00:00:06	97	00:00:03	97	00:00:03
vlsi.48.81	48	$(40, 8)$	12	00:00:15	4	00:00:05	4	00:00:04
vlsi.166.504	166	$(100, 66)$	23	15:03:55	12	03:10:46	12	03:03:23
vlsi.170.424	170	$(100, 70)$	37	16:19:17	35	03:31:16	35	03:30:42

Table 7: Computational results for the minimum bisection problem. The time is given in hr:min:s.

6.3 The maximum k-cut problem and the maximum k-partition problem

In the sequel we compare the max-k-cut problem with the max-k-partition problem. Since in the max-k-cut problem there is no restriction on the size of the subsets in the partitions, in order to compare relaxations k–MC_{FJ} and GPP_{RS} (where minimization is replaced by maximization) we do the following. For a given $k \geq 2$, we compute GPP_{RS} for all combinations of $m_1, \ldots, m_k$ such that $m_1 + \ldots + m_p = n$, $p = 2, \ldots, k$.

In Figure 7 the bounds GPP_{RS} (as a maximization problem) are plotted for 100 graphs $G_{0.8}(37)$ in case of $k = 2$ and different $(m_1, m_2)^{\mathrm{T}}$. Our numerical results show that among all possible combinations of $(m_1, m_2)^{\mathrm{T}}$ such that $m_1 + m_2 = 37$, the upper bounds GPP_{RS} and 2–MC_{FJ} differ only marginally when $(m_1, m_2) = (19, 18)$. Therefore we do not plot 2–MC_{FJ} in Figure 7. The upper bounds on the same figure are sorted w.r.t. increasing values of GPP_{RS} where $m = (19, 18)^{\mathrm{T}}$. The thick line represents GPP_{RS} where $m = (19, 18)^{\mathrm{T}}$ and 2–MC_{FJ}, the dashed line GPP_{RS} where $m = (25, 12)^{\mathrm{T}}$, the dotted line GPP_{RS} where $m = (28, 9)^{\mathrm{T}}$, and the thin line GPP_{RS} where $m = (32, 5)^{\mathrm{T}}$. It is clear from Figure 7 that, in general, the relaxation 2–MC_{FJ} does not provide a tight bound for the maximum bisection problem. The computational time to compute GPP_{RS} or 2–MC_{FJ} bound for an instance with 37 vertices is about two seconds.

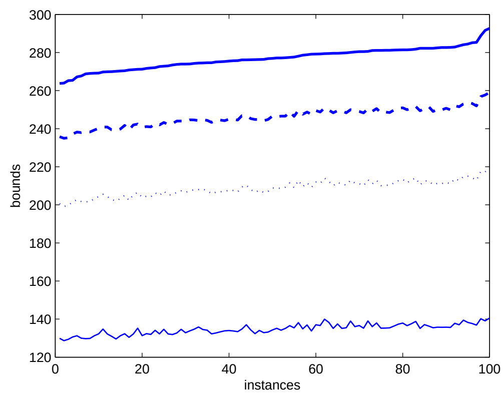


Figure 7: max GPP_{RS} for different (m_1, m_2): thick line for $(19, 18)$, dashed line for $(25, 12)$, dotted line for $(28, 9)$, and thin line for $(32, 5)$

Finally, in Figure 8 the bounds GPP_{RS} (as a maximization problem) and k–MC_{FJ} are plotted for 100 graphs $G_{0.5}(21)$ in case of $k = 3$. The thick line represents 3–MC_{FJ} and GPP_{RS} where $m = (7, 7, 7)^{\mathrm{T}}$, and all other lines represent GPP_{RS} for different $(m_1, m_2, m_3)^{\mathrm{T}}$ such that $m_1 + m_2 + m_3 = 21$. We do not plot GPP_{RS} for the maximum bisection problem although we also compute them. If plotted, these bounds would be in the lower part of the figure. Figure 8 demonstrates differences between the max-k-cut problem and different max k-partition problems. The computational time to compute GPP_{RS} or 3–MC_{FJ} for an instance with 21 vertices is less than one second.

7 Conclusion

In this paper, we derive a new SDP relaxation for the general graph partition problem, i.e., not restricted to the equipartition problem. We show that the new relaxation is equivalent to the well know Frieze-Jerrum relaxation [20] for the max-k-cut problem with an additional constraint on the sizes of the partition subsets. The new relaxation is based on matrix lifting and it is the only known SDP relaxation of the GPP whose size does not increase with the number of subsets k in which the graph should be partitioned. Therefore this is, to the best of our knowledge, the only known SDP relaxation for the GPP that provides bounds for graphs with more than 50 vertices and when the number of subsets is larger than five.

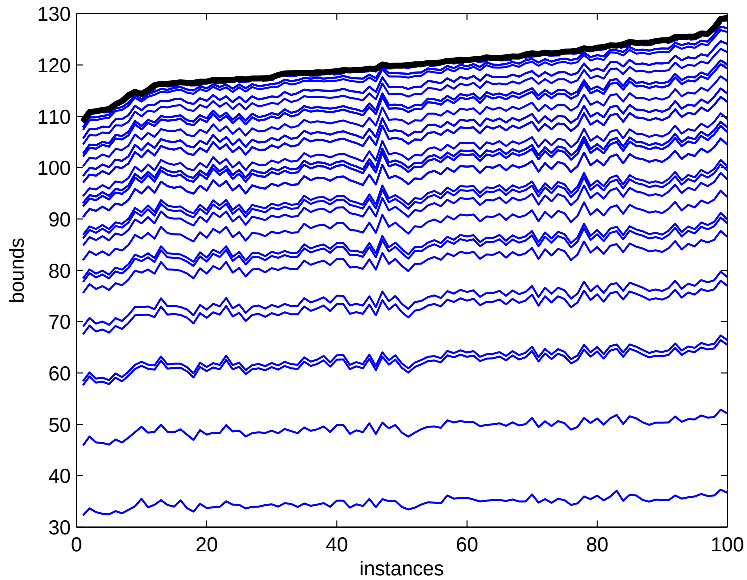


Figure 8: thick line is 3–MC_{FJ} and GPP_{RS} where $m = (7,7,7)^T$, other lines represent GPP_{RS} for different $(m_1, m_2, m_3)^T$

We prove here that our relaxation is dominated by the best known SDP relaxation of the GPP that is based on vector lifting, i.e., the improved Wolkowicz-Zhao relaxation. However, the computational effort to compute our bound is negligible in comparison with the computational effort required to solve the improved Wolkowicz-Zhao relaxation. Due to the mentioned quality of the new relaxation, we believe that it is suitable for implementation within a branch and bound framework.

In this paper, [13], [14], [39], and [45], it is shown that for some combinatorial optimization problems one can derive matrix lifting based SDP relaxations that turn out to be competitive with vector lifting based SDP relaxations. In particular, relaxations obtained by using matrix lifting can be solved with less computational effort than those obtained by using vector lifting. Besides this, the results show that the relaxations from a lower dimensional space have bounds that are close, and for some problems even equal, to the bounds of relaxations from a higher dimensional space. To conclude, we believe that there is a large potential in matrix lifting obtained SDP relaxations that should be investigated also for other combinatorial optimization problems.

Acknowledgments. The author would like to thank Edwin van Dam for valuable discussions and careful reading of this manuscript. The author would also like to thank the Associate Editor and two anonymous referees for suggestions that led to an improvement of this paper.

References

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

[2] K. Anstreicher and H. Wolkowicz. On Lagrangian relaxation of quadratic matrix constraints. *SIAM. J. Matrix Anal. and Appl.*, 22(1):41–55, 2000.

[3] M. Armbruster. *Branch-and-cut for a semidefinite relaxation of large-scale minimum bisection problems*. PhD thesis, Technische Universität Chemnitz, Germany, 2007.

[4] M. Armbruster, C. Helmberg, M. Fügenschuh, and A. Martin. LP and SDP branch-and-cut algorithms for the minimum graph bisection problem: a computational comparison. Preprint 2011-6, Technische Universität Chemnitz, Fakultät für Mathematik, March 2011.

[5] R. Battiti and A. Bertossi. Greedy, prohibition, and reactive heuristics for graph partitioning. *IEEE Trans. Comput.*, 48(4):361–385, 1999.

[6] R. Biswas, B. Hendrickson, and G. Karypis. Graph partitioning and parallel computing. *Parallel Comput.*, 26(12):1515-1517, 2000.

[7] T.N. Bui and B.R. Moon. Genetic algorithm and graph partitioning. *IEEE Trans. Comput.*, 45:814–855, 1995.

[8] S. Chopra and M.R. Rao. The partition problem. *Math. Program.*, 59(1/3):87–115, 1993.

[9] W. Dai and E. Kuh. Simultaneous floor planning and global routing for hierarchical building-block layout. *IEEE Trans. Comput.-Aided Des. Integrated Circuits* & *Syst.*, CAD-6, 5:828–837, 1987.

[10] E. De Klerk, F.M. de Oliveira Filho, and D.V. Pasechnik. Relaxations of combinatorial problems via association schemes. In: *Handbook of Semidefinite, Conic and Polynomial Optimization: Theory, Algorithms, Software and Applications*, M.F. Anjos and J. B. Lasserre (eds.). International Series in Operational Research and Management Science. Volume 166:171–200, 2012.

[11] E. De Klerk, D.V. Pasechnik, R. Sotirov, and C. Dobre. On semidefnite programming relaxations of maximum k-section. *Math. Program. Ser. B.* (to appear)

[12] C.C. de Souza, R. Keunings, L.A. Wolsey, O. Zone. A new approach to minimizing the frontwidth in finite element calculations. *Computer Methods in Applied Mechanics and Engineering*, 111:323334, 1994.

[13] Y. Ding and H. Wolkowicz. A low dimensional semidefinite relaxation for the quadratic assignment problem. *Math. Oper. Res.*, 34(4):1008–1022, 2009.

[14] Y. Ding, D. Ge, and H. Wolkowicz. On equivalence of semidefinite relaxations for quadratic matrix programming. *Math. Oper. Res.*, 36(1):88-104, 2011.

[15] W.E. Donath and A.J. Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17:420–425, 1973.

[16] A. Eisenblätter. *Frequency assignment in GSM networks*. PhD thesis, Technische Universität Berlin, Germany, 2001.

[17] U. Feige and M. Langberg. Approximation algorithms for maximization problems arising in graph partitioning. *J. Algorithm*, 41:174–211, 2001.

[18] C. E. Ferreira, A. Martin, C. C. de Souza, R. Weismantel, and L. A. Wolsey. The node capacitated graph partitioning problem: a computational study. *Math. Program.*, 81:229–256, 1998.

[19] C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. *Proceedings of the 19th Design Automation Conference*, 175–181, 1982.

[20] A. Frieze and M. Jerrum. Improved approximation algorithms for max k-cut and max bisection. *Algorithmica*, 18(1):67–81, 1997.

[21] B. Ghaddar, M.F. Anjos, and F. Liers. A branch-and-cut algorithm based on semidefinite programming for the minimum k-partition problem. *Ann. Oper. Res.*, 188(1):155-174, 2011.

[22] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoret. Comput. Sci.*, **1**(3), 237–267, 1976.

[23] A. Graham. *Kronecker products and matrix calculus with applications.* Ellis Horwood Limited, Chichester, 1981.

[24] D. Gijswijt. *Matrix algebras and semidefinite programming techniques for codes.* PhD thesis. University of Amsterdam, The Netherlands, 2005.

[25] Q. Han, Y. Ye, and J. Zhang. An improved rounding method and semidefinite relaxation for graph partitioning. *Math. Program.*, 92:509–535, 2002.

[26] B. Hendrickson and T.G. Kolda. Partitioning rectangular and structurally nonsymmetric sparse matrices for parallel processing. *SIAM J. Sci. Comput.*, 21(6):2048–2072, 2000.

[27] C. Helmberg. A cutting plane algorithm for large scale semidefinite relaxations. In: *Padberg Festschrift The Sharpest Cut*, M. Grötschel (Ed.). MPS-SIAM 233–256, 2004.

[28] E. Johnson, A. Mehrotra, G. Nemhauser. Min-cut clustering. *Math. Program.*, 62:133–152,1993.

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

[30] P. Erdös and A. Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

[31] P. Erdös and A. Rényi. The revolution on random graphs. *Magyar. Tud. Akad. Mat. Kutato Int. Kozl.*, 5:17–61, 1960.

[32] D. Karger, R. Motwani and M. Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45(2):246–265, 1998.

[33] S.E. Karisch. *Nonlinear approaches for quadratic assignment and graph partition problems.* PhD thesis. Technical University Graz, Austria, 1995.

[34] S.E. Karisch and F. Rendl. Semidefnite programming and graph equipartition. In *Topics in Semidefinite and Interior–Point Methods*, volume 18 of *The Fields Institute fir research in Mathematical Sciences, Communications Series*, Providence, Rhode Island, 1998. American Mathematical Society.

[35] S. E. Karisch, F. Rendl, and J. Clausen. Solving graph bisection problems with semidefinite programming. *INFORMS J. Comput.*, 12:177-191, 2000.

[36] T. Lengauer. *Combinatorial algorithms for integrated circuit layout.* Wiley, Chicester, 1990.

[37] A. Lisser and F. Rendl. Graph partitioning using linear and semidefinite programming. *Math. Program. Ser. B*, 95(1):91–101, 2003.

[38] J. Löfberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In: Proceeding of the CACSD Conference, Taipei, Taiwan, 2004. `http://control.ee.ethz.ch/~joloef/yalmip.php`

[39] A. Mobasher, R. Sotirov, and A.K. Khandani. Matrix-lifting SDP for detection in multiple antenna systems, *IEEE Transactions on Signal Processing*, 58(10):5178–5185, 2010.

[40] J. Povh and F. Rendl. Copositive and semidefinite relaxations of the quadratic assignment problem. *Discrete Optim.*, 6(3):231–241, 2009.

[41] F. Rendl and H. Wolkowicz. A projection technique for partitioning nodes of a graph. *Ann. Oper. Res.*, 58:155–179, 1995.

[42] G. Rinaldi. Rudy, 1996. `http://www-user.tu-chemnitz.de/~helmberg/rudy.tar.gz`

[43] L. Sanchis. Multiple-way network partitioning. *IEEE Trans. Comput.*, 38:62–81, 1989.

[44] H. D. Simon. Partitioning of unstructured problems for parallel processing. *Comput. Syst. Eng.*, 2:35–148, 1991.

[45] R. Sotirov. SDP relaxations for some combinatorial optimization problems. In: *Handbook of Semidefinite, Conic and Polynomial Optimization: Theory, Algorithms, Software and Applications*, M.F. Anjos and J. B. Lasserre (eds.). International Series in Operational Research and Management Science. Volume 166:795–820, 2012.

[46] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Methods Softw.*, 11-12:625–653, 1999.

[47] H. Wolkowicz and Q. Zhao. Semidefinite programming relaxations for the graph partitioning problem. *Discrete Appl. Math.*, 96/97:461–479, 1999.

[48] Q. Zhao, S.E. Karisch, F. Rendl, and H. Wolkowicz. Semidefinite programming relaxations for the quadratic assignment problem. *J. Comb. Optim.*, 2:71–109, 1998.