Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

VOLKER KAIBEL MATTHIAS PEINHARDT MARC E. PFETSCH

# Orbitopal Fixing

Supported by the DFG Research Center MATHEON "Mathematics for key technologies"
During the research of this work the first author was a visiting professor at the Technical University of Berlin.

ZIB-Report 06-48 (November 2006)

# ORBITOPAL FIXING

VOLKER KAIBEL, MATTHIAS PEINHARDT, AND MARC E. PFETSCH

Abstract. The topic of this paper are integer programming models in which a subset of 0/1-variables encode a partitioning of a set of objects into disjoint subsets. Such models can be surprisingly hard to solve by branch-and-cut algorithms if the permutation of the subsets of the partition is irrelevant. This kind of symmetry unnecessarily blows up the branch-and-cut tree.

We present a general tool, called orbitopal fixing, for enhancing the capabilities of branch-and-cut algorithms in solving this kind of symmetric integer programming models. We devise a linear time algorithm that, applied at each node of the branch-and-cut tree, removes redundant parts of the tree produced by the above mentioned permutations. The method relies on certain polyhedra, called orbitopes, which have been investigated in [11]. However, it does not add inequalities to the model, and thus, it does not increase the difficulty of solving the linear programming relaxations. We demonstrate the computational power of orbitopal fixing at the example of a graph partitioning problem motivated from frequency planning in mobile telecommunication networks.

## 1. Introduction

Being welcome in most other contexts, symmetry causes severe trouble in the solution of many integer programming (IP) models. This paper describes a method to enhance the capabilities of branch-and-cut algorithms with respect to handling symmetric models of a certain kind that frequently occurs in practice.

We illustrate this kind of symmetry by the example of a graph partitioning problem (another notorious example is the vertex coloring problem). Here, one is given a graph $G = (V, E)$ with nonnegative edge weights $w \in \mathbb{Q}^E_{\geq 0}$ and an integer $q \geq 2$. The task is to partition $V$ into $q$ disjoint subsets such that the sum of all weights of edges connecting nodes in the same subset is minimized.

A straight-forward IP model arises by introducing 0/1-variables $x_{ij}$ for all $i \in [p] := \{1, \ldots, p\}$ and $j \in [q]$ that indicate whether node $i$ is contained in subset $j$ (where we assume $V = [p]$). In order to model the objective function, we furthermore need 0/1-variables $y_{ik}$ for all edges $\{i, k\} \in E$

*Date*: 11/17/2006.

2000 *Mathematics Subject Classification.* Primary 90C10; Secondary 90C57, 52B12.

*Key words and phrases.* integer programming, symmetry breaking, variable fixing, orbitopes.

Supported by the DFG Research Center Matheon in Berlin.

During the research of this work the first author was a visiting professor at the Technical University of Berlin.

indicating whether both nodes $i$ and $k$ are contained in the same subset. This yields the following model (see, e.g., [5]):

$$
\begin{array}{lll}
\min & \displaystyle\sum_{\{i,k\}\in E} w_{ik}\, y_{ik} & \\
\text{s.t.} & \displaystyle\sum_{j=1}^{q} x_{ij} = 1 & \text{for all } i \in [p] \\
& x_{ij} + x_{kj} - y_{ik} \leq 1 & \text{for all } i \in [p],\ j \in [q] \\
& x_{ij} \in \{0,1\} & \text{for all } i \in [p],\ j \in [q] \\
& y_{ik} \in \{0,1\} & \text{for all } \{i,k\} \in E.
\end{array}
\tag{1}
$$

The $x$-variables form $0/1$-matrices of size $p \times q$ with exactly one 1-entry per row. They encode the assignment of the nodes to the subsets of the partition. The methods that we discuss in this paper do only rely on this structure and thus can be applied to many other models as well. We use the example of the graph partitioning problem as a prototype application and report on computational experiments in Sect. 5. Graph partitioning problems are discussed in [5, 3, 4], for instance as a relaxation of frequency assignment problems in mobile telecommunication networks. The maximization version is relevant as well [12, 6]. Also capacity bounds on the subsets of the partition (which can easily be incorporated into the model) are of interest, in particular the graph equipartitioning problem [15, 16, 7, 8]. For the closely related clique partitioning problem, see [9, 10].

As it is given above, the model is unnecessarily difficult for state-of-the-art IP solvers. Even small instances require enormous efforts (see Sect. 5). One reason is that every feasible solution $(x, y)$ to this model can be turned into $q!$ different ones by permuting the columns of $x$ in an arbitrary way, thereby not changing the structure of the solution (in particular: its objective function value). Phrased differently, the symmetric group of all permutations of the set $[q]$ operates on the solutions by permuting the columns of the $x$-variables in such a way that the objective function remains constant along each orbit. Therefore, when solving the model by a branch-and-cut algorithm, basically the same work will be done in the tree at many places. Thus there should be potential for reducing the running times significantly by exploiting the symmetry. A more subtle second point is that the barycenters of the orbits are responsible for quite weak linear programming (LP) bounds. However, we will not address this second point in this paper.

In order to remove symmetry, the above model for the graph partitioning problem in the literature is often replaced by models containing only edge variables [7]. However, for this to work the underlying graph has to be complete, which might introduce many unnecessary variables. Moreover, formulation (1) is sometimes favorable, e.g., if node-weighted capacity constraints should be incorporated.

One way to deal with symmetry is to restrict the feasible region in each of the orbits to a single representative, e.g., to the lexicographically maximal (with respect to the row-by-row ordering of the $x$-components) element in the orbit. In fact, this can be done by adding some inequalities to the model

that enforce the columns of $x$ to be sorted in a lexicographically decreasing way. This can be achieved by $\mathrm{O}(pq)$ many *column inequalities*. In [11] even a complete (and irredundant) linear description of the convex hull of all 0/1-matrices of size $p \times q$ with exactly one 1-entry per row and lexicographically decreasing columns (called an *orbitope*) is derived. The description basically consists of an exponentially large superclass of the column inequalities, called *shifted column inequalities*, for which there is a linear time separation algorithm available. We recall some of these results in Sect. 2.

Incorporating the inequalities from the orbitope description into the IP model removes symmetry. At each node of the branch-and-cut tree this ensures that the corresponding IP is infeasible as soon as there is no representative in the subtree rooted at that node. Of course, the column inequalities are sufficient for this purpose already.

In this paper, we investigate a way to utilize these inequalities (or the orbitope that they describe) without adding any of the inequalities to the models explicitly. The reason for doing this is the unpleasant effect that adding (shifted) column inequalities to the models results in more difficult LP relaxations. One way to avoid adding these inequalities to the LPs is to exploit them for deriving logical implications instead: If we are working in a branch-and-cut node at which the $x$-variables corresponding to index subsets $I_0$ and $I_1$ are fixed to zero and one, respectively, then there might be a (shifted) column inequality yielding implications for all representatives in the subtree rooted at the current node. For instance, it might be that for some $(i,j) \notin I_0 \cup I_1$ we have $x_{ij} = 0$ for all feasible solutions in the subtree. In this case, $x_{ij}$ can be fixed to zero at the current node, enlarging $I_0$. We call the iterated process of searching for such additional fixings *sequential fixing* with (shifted) column inequalities.

Let us mention at this point that deviating from parts of the literature, we do not distinguish between "fixing" and "setting" of variables in this paper.

Sequential fixing with (shifted) column inequalities is a special case of constraint propagation, which is well known from constraint logic programming. Modern IP solvers like SCIP [1] use such strategies also in branch-and-cut algorithms. However, with orbitopes we can aim at something better: Consider a branch-and-cut node identified by fixing the variables corresponding to sets $I_0$ and $I_1$ to zero and one, respectively. Denote by $W(I_0, I_1)$ the set of all vertices $x$ of the orbitope with $x_{ij} = 0$ for all $(i,j) \in I_0$ and $x_{ij} = 1$ for all $(i,j) \in I_1$. Define the sets $I_0^\star$ and $I_1^\star$ of indices of *all* variables, for which no $x$ in $W(I_0, I_1)$ satisfies $x_{ij} = 1$ for some $(i,j) \in I_0^\star$ or $x_{ij} = 0$ for some $(i,j) \in I_1^\star$. Fixing of the corresponding variables is called *simultaneous fixing* at the branch-and-cut node. Simultaneous fixing is always at least as strong as sequential fixing.

Investigations of sequential and simultaneous fixing for orbitopes are the central topic of the paper. The main contributions and results are the following:

- We give a linear time algorithm for *orbitopal fixing*, i.e., for solving the problem to compute simultaneous fixings for orbitopes (Theorem 12).
- While, for general 0/1-polytopes, sequential fixing even with complete and irredundant linear descriptions is weaker than simultaneous fixing

(Theorem 3), we prove that sequential fixing with the whole set of shifted column inequalities is as strong as orbitopal fixing (Theorem 11). However, in general, sequential fixing with only column inequalities is weaker (Theorem 10).

- We report on computer experiments (Sect. 5) with the graph partitioning problem described above showing that orbitopal fixing leads to significant performance improvements for branch-and-cut algorithms. These improvements remain significant if compared to the improvements that commercial IP-solvers like CPLEX 10.01 achieve by their symmetry breaking tools.

To summarize, orbitopal fixing is a "plug-and-play tool" for significantly enhancing the performance of branch-and-cut based IP solvers on models that exhibit the partitioning-type symmetry described above. We believe that it will be as valuable for several different problems exhibiting that kind of symmetry as it seems to be for the graph partitioning problem.

Margot [13, 14] considers a related method for symmetry handling. His approach works for more general types of symmetries than ours. Similarly to our approach, the basic idea is to assure that only (partial) solutions which are lexicographical maximal in their orbit are explored in the branch-and-cut tree. This is guaranteed by an appropriate fixing rule. The fixing and pruning decisions are done by means of a Schreier-Sims table for representing the group action. While Margot's approach is much more generally applicable than orbitopal fixing, the latter seems to be more powerful in the special situation of partitioning-type symmetries. One reason is that Margot's method requires to choose the branching variables according to an ordering that is chosen globally for the entire branch-and-cut tree. We will elaborate on the relation between the two approaches in more detail in the full version of the paper.

## 2. Orbitopes

Throughout the paper, let $p$ and $q$ be positive integers with $p \geq q \geq 2$. The *orbitope* $\mathrm{O}^{=}_{p,q}$ is the convex hull of all 0/1-matrices $x \in \{0,1\}^{[p]\times[q]}$ with exactly one 1-entry per row whose columns are in decreasing lexicographical order (i.e., they satisfy $\sum_{i=1}^{p} 2^{p-i} x_{ij} > \sum_{i=1}^{p} 2^{p-i} x_{i,j+1}$ for all $j \in [q-1]$). As these 0/1-matrices have $x_{ij} = 0$ for all $(i,j)$ with $i < j$, we drop these components and consider $\mathrm{O}^{=}_{p,q}$ as a subset of the space $\mathbb{R}^{\mathcal{I}_{p,q}}$ with $\mathcal{I}_{p,q} := \{(i,j) \in \{0,1\}^{[p]\times[q]} : i \geq j\}$. Thus, we consider matrices, in which the $i$-th row has only $q(i) := \min\{i, q\}$ components.

Let the symmetric group of size $q$ act on $\{0,1\}^{\mathcal{I}_{p,q}}$ via permuting the columns of all $x \in \{0,1\}^{\mathcal{I}_{p,q}}$. Then the vertices of $\mathrm{O}^{=}_{p,q}$ are exactly the lexicographically maximal matrices (with respect to the row-by-row ordering of the components) in those orbits whose elements are matrices with exactly one 1-entry per row. In [11], in the context of more general orbitopes, $\mathrm{O}^{=}_{p,q}$ is referred to as the *partitioning orbitope with respect to the symmetric group*. As we will confine ourselves with this one type of orbitopes in this paper, we will simply call it *orbitope*.

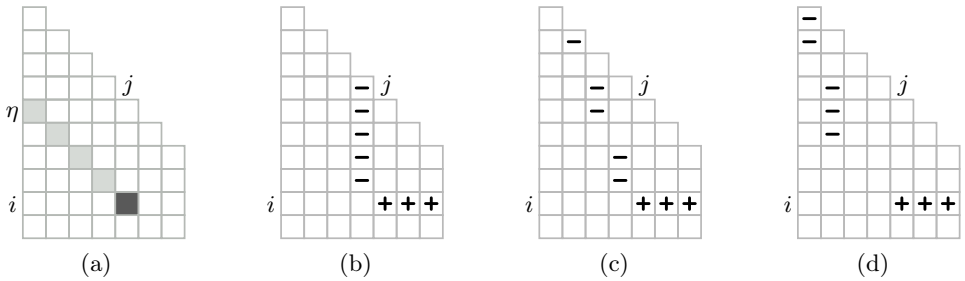



**Figure 1:** (a) Example for coordinates $(9, 5) = \langle 5, 5\rangle$. (b), (c), (d) Three shifted column inequalities, the left one of which is a column inequality

The main result in [11] is a complete linear description of $\mathrm{O}^{=}_{p,q}$. In order to describe the result, it will be convenient to address the elements in $\mathcal{I}_{p,q}$ via a different "system of coordinates":

$$\langle \eta, j\rangle := (j + \eta - 1, j) \text{ for } j \in [q],\ 1 \le \eta \le p - j + 1.$$

Thus (as before) $i$ and $j$ denote the row and the column, respectively, while $\eta$ is the index of the diagonal (counted from above) containing the respective element; see Figure 1 (a) for an example.

A set $S = \{\langle 1, c_1\rangle, \langle 2, c_2\rangle, \ldots, \langle \eta, c_\eta\rangle\} \subset \mathcal{I}_{p,q}$ with $c_1 \le c_2 \le \cdots \le c_\eta$ and $\eta \ge 1$ is called a *shifted column*. For $(i, j) = \langle \eta, j\rangle \in \mathcal{I}_{p,q}$, the set $B = \{(i, j), (i, j + 1), \ldots, (i, q(i))\}$, and a shifted column

$$S = \{\langle 1, c_1\rangle, \langle 2, c_2\rangle, \ldots, \langle \eta, c_\eta\rangle\} \subset \mathcal{I}_{p,q}$$

with $c_\eta < j$, we call $x(B) - x(S) \le 0$ a *shifted column inequality*. The set $B$ is called its *bar*. In case of $c_1 = \cdots = c_\eta = j-1$ the shifted column inequality is called a *column inequality*. See Figure 1 for examples.

Finally, a bit more notation is needed. For each $i \in [p]$, we define $\mathrm{row}_i := \{(i, j) : j \in [q(i)]\}$. For $A \subset \mathcal{I}_{p,q}$ and $x \in \mathbb{R}^{\mathcal{I}_{p,q}}$, as usual, $x(A)$ denotes the sum $\sum_{(i,j)\in A} x_{ij}$.

**Theorem 1** (see [11])**.** *The orbitope $\mathrm{O}^{=}_{p,q}$ is completely described by the non-negativity constraints $x_{ij} \ge 0$, the row-sum equations $x(\mathrm{row}_i) = 1$, and the shifted column inequalities.*

In fact, in [11] it is also shown that, up to a few exceptions, the inequalities in this description define facets of $\mathrm{O}^{=}_{p,q}$. Furthermore, a linear time separation algorithm for the exponentially large class of shifted column inequalities is given.

## 3. The Geometry of Fixing Variables

In this section, we deal with general 0/1-integer programs and, in particular, their associated polytopes. We will define some basic terminology used later in the special treatment of orbitopes and we are going to shed some light on the geometric situation of fixing variables.

We denote by $[d]$ the set of indices of variables, and by $\mathrm{C}^d = \{x \in \mathbb{R}^d : 0 \le x_i \le 1 \text{ for all } i \in [d]\}$ the corresponding 0/1-cube. For two disjoint subsets $I_0, I_1 \subseteq [d]$ (with $I_0 \cap I_1 = \varnothing$) we call

$$\{x \in \mathrm{C}^d : x_i = 0 \text{ for all } i \in I_0,\ x_i = 1 \text{ for all } i \in I_1\}$$

the *face of* $\mathrm{C}^d$ *defined by* $(I_0, I_1)$. All faces of $\mathrm{C}^d$ are of this type.

For a polytope $P \subseteq \mathrm{C}^d$ and for a face $F$ of $\mathrm{C}^d$ defined by $(I_0, I_1)$, we denote by $\mathrm{Fix}_F(P)$ the smallest face of $\mathrm{C}^d$ that contains $P \cap F \cap \{0,1\}^d$ (i.e., $\mathrm{Fix}_F(P)$ is the intersection of all faces of $\mathrm{C}^d$ that contain $P \cap F \cap \{0,1\}^d$). If $\mathrm{Fix}_F(P)$ is the cube face defined by $(I_0^\star, I_1^\star)$, then $I_0^\star$ and $I_1^\star$ consist of all $i \in [d]$ for which $x_i = 0$ and $x_i = 1$, respectively, holds for all $x \in P \cap F \cap \{0,1\}^d$. In particular, we have $I_0 \subseteq I_0^\star$ and $I_1 \subseteq I_1^\star$. Thus, if $I_0$ and $I_1$ are the indices of the variables fixed to zero and one, respectively, in the current branch-and-cut node (with respect to an IP with feasible points $P \cap \{0,1\}^d$), the sets $I_0^\star$ and $I_1^\star$ yield the maximal sets of variables that can be fixed to zero and one, respectively, for the whole subtree rooted at this node. We call $(I_0^\star, I_1^\star)$ the *fixing of* $P$ *at* $(I_0, I_1)$. Similarly, we use the notion of *fixing* for the corresponding cube faces.

**Remark 2.** *If* $P, P' \subseteq \mathrm{C}^d$ *are two polytopes with* $P \subseteq P'$ *and* $F$ *and* $F'$ *are two faces of* $\mathrm{C}^d$ *with* $F \subseteq F'$*, then* $\mathrm{Fix}_F(P) \subseteq \mathrm{Fix}_{F'}(P')$ *holds.*

In general, it is not clear how to compute fixings efficiently. Indeed, computing the fixing of $P$ at $(\varnothing, \varnothing)$ includes deciding whether $P \cap \{0,1\}^d = \varnothing$, which, of course, is NP-hard in general. Instead, one can try to derive as large as possible subsets of $I_0^\star$ and $I_1^\star$ by looking at relaxations of $P$. In case of an IP that is based on an intersection with an orbitope, one might use the orbitope as such a relaxation. We will deal with the fixing problem for orbitopes in Sect. 4.

If $P$ is given via an inequality description, one possibility is to use the knapsack relaxations obtained from single inequalities out of the description. For each of these relaxations, the fixing can easily be computed. If the inequality system describing $P$ is exponentially large, and the inequalities are only accessible via a separation routine, it might still be possible to decide efficiently whether any of the exponentially many knapsack relaxations allows to fix some variable (see Sect. 4.2).

Suppose, $P = \{x \in \mathrm{C}^d : Ax \leq b\}$ and $P_1, \ldots, P_m$ are the knapsack relaxations of $P$ obtained from the $m$ inequalities in $Ax \leq b$. Let $F$ be some face of $\mathrm{C}^d$. The face $G$ of $\mathrm{C}^d$ obtained by setting $G := F$ and then iteratively replacing $G$ by $\mathrm{Fix}_G(P_r)$ as long as there is some $r \in [m]$ with $\mathrm{Fix}_G(P_r) \subsetneq G$, is denoted by $\mathrm{Fix}_F(Ax \leq b)$. Note that the outcome of this procedure is independent of the choices made for $r$ due to Remark 2. We call the pair $(\tilde{I}_0, \tilde{I}_1)$ defining the cube face $\mathrm{Fix}_F(Ax \leq b)$ the *sequential fixing of* $Ax \leq b$ *at* $(I_0, I_1)$. In the context of sequential fixing we often refer to $\mathrm{Fix}_F(P)$ by the notion of *simultaneous fixing.*

Due to Remark 2 it is clear that $\mathrm{Fix}_F(P) \subseteq \mathrm{Fix}_F(Ax \leq b)$ holds.

**Theorem 3.** *In general, even for a system of facet-defining inequalities describing a 0/1-polytope, sequential fixing is weaker than simultaneous fixing.*

*Proof.* The following example shows this. Let $P \subset \mathrm{C}^4$ be the 4-dimensional polytope defined by the trivial inequalities $x_i \geq 0$ for $i \in \{1,2,3\}$, $x_i \leq 1$ for $i \in \{1,2,4\}$, the inequality $-x_1 + x_2 + x_3 - x_4 \leq 0$ and $x_1 - x_2 + x_3 - x_4 \leq 0$. Let $F$ be the cube face defined by $(\{4\}, \varnothing)$. Then, sequential fixing does not lead to fixing any further variable, although simultaneous fixing yields

$I_0^\star = \{3,4\}$ (and $I_1^\star = \varnothing$). Note that $P$ has only 0/1-vertices, and all the inequalities are facet defining ($x_4 \geq 0$ and $x_3 \leq 1$ are implied). □

## 4. Fixing Variables for Orbitopes

For this section, suppose that $I_0, I_1 \subseteq \mathcal{I}_{p,q}$ are subsets of indices of orbitope variables with the following properties:

(P1) $|I_0 \cap \mathrm{row}_i| \leq q(i) - 1$ for all $i \in [p]$

(P2) For all $(i,j) \in I_1$, we have $(i,\ell) \in I_0$ for all $\ell \in [q(i)] \setminus \{j\}$.

Let $F$ be the face of the 0/1-cube $\mathrm{C}^{\mathcal{I}_{p,q}}$ defined by $(I_0, I_1)$. Note that if P1 is not fulfilled, then $\mathrm{O}^{=}_{p,q} \cap F = \varnothing$. The following statement follows immediately from Property P2.

**Remark 4.** *If a vertex $x$ of $\mathrm{O}^{=}_{p,q}$ satisfies $x_{ij} = 0$ for all $(i,j) \in I_0$, then $x \in F$.*

We assume that the face $\mathrm{Fix}_F(\mathrm{O}^{=}_{p,q})$ is defined by $(I_0^\star, I_1^\star)$. *Orbitopal fixing* is the problem to compute the simultaneous fixing $(I_0^\star, I_1^\star)$ from $(I_0, I_1)$.

**Remark 5.** *It is enough to determine $I_0^\star$, as we have $(i,j) \in I_1^\star$ if and only if $(i,\ell) \in I_0^\star$ holds for for all $\ell \in [q(i)] \setminus \{j\}$.*

### 4.1. The Walls of Orbitopes

Since $\mathrm{O}^{=}_{p,q} \subset \mathrm{C}^{\mathcal{I}_{p,q}}$ is a 0/1-polytope (i.e., it is integral), we have

$$\mathrm{conv}(\mathrm{O}^{=}_{p,q} \cap F \cap \{0,1\}^{\mathcal{I}_{p,q}}) = \mathrm{O}^{=}_{p,q} \cap F.$$

Thus, $\mathrm{Fix}_F(\mathrm{O}^{=}_{p,q})$ is the smallest cube face that contains the face $\mathrm{O}^{=}_{p,q} \cap F$ of the orbitope $\mathrm{O}^{=}_{p,q}$. Such faces of $\mathrm{O}^{=}_{p,q}$ arising from intersections of $\mathrm{O}^{=}_{p,q}$ with cube faces are called *walls* of $\mathrm{O}^{=}_{p,q}$.

Let us, for $i \in [p]$, define values $\alpha_i := \alpha_i(I_0) \in [q(i)]$ recursively by setting $\alpha_1 := 1$ and, for all $i \in [p]$ with $i \geq 2$,

$$\alpha_i \;:=\; \begin{cases} \alpha_{i-1} & \text{if } \alpha_{i-1} = q(i) \text{ or } (i, \alpha_{i-1}+1) \in I_0 \\ \alpha_{i-1} + 1 & \text{otherwise.} \end{cases}$$

The set of all indices of rows, in which the $\alpha$-value increases, is denoted by

$$\Gamma(I_0) \;:=\; \{i \in [p] \,:\, i \geq 2,\ \alpha_i = \alpha_{i-1} + 1\} \cup \{1\}$$

(where, for technical reasons 1 is included).

The following observation follows readily from the definitions.

**Remark 6.** *For each $i \in [p]$ with $i \geq 2$ and $\alpha_i(I_0) < q(i)$, the set*

$$S_i(I_0) \;:=\; \{(k, \alpha_k(I_0) + 1) \,:\, k \in [i] \setminus \Gamma(I_0)\}$$

*is a shifted column with $S_i(I_0) \subseteq I_0$.*

**Lemma 7.** *For each $i \in [p]$, no vertex of $\mathrm{O}^{=}_{p,q} \cap F$ has its 1-entry in row $i$ in a column $j \in [q(i)]$ with $j > \alpha_i(I_0)$.*

*Proof.* Let $i \in [p]$ and $j \in [q(i)]$. We may assume $\alpha_i(I_0) < q(i)$, because otherwise the statement trivially is true. Thus, $B := \{(i,j) \in \mathrm{row}_i \,:\, j > \alpha_i(I_0)\} \neq \varnothing$.

Let us first consider the case $i \in \Gamma(I_0)$. As we have $\alpha_i(I_0) < i$ and $\alpha_1(I_0) = 1$, there must be some $k < i$ such that $k \notin \Gamma(I_0)$. Let $k$ be maximal with this property, thus we have $k' \in \Gamma(I_0)$ for all $1 < k < k' \leq i$. According to Remark 6 $x(B) - x(S_k(I_0)) \leq 0$ is a shifted column inequality with $x(S_k(I_0)) = 0$, showing $x(B) = 0$ as claimed in the lemma.

Thus, let us suppose $i \in [p] \setminus \Gamma(I_0)$. If $\alpha_i(I_0) \geq q(i) - 1$, the the claim holds trivially. Otherwise, $B' := B \setminus \{(i, \alpha_i(I_0))\} \neq \varnothing$. Similarly to the first case, now the shifted column inequality $x(B') - x(S_{i-1}(I_0)) \leq 0$ proves the claim. □

For each $i \in [p]$ we define

$$\mu_i(I_0) \ := \ \min\{j \in [q(i)] \ : \ (i,j) \notin I_0\}.$$

Because of Property P1, the sets over which we take minima here are non-empty.

**Lemma 8.** *If $\mu_i(I_0) \leq \alpha_i(I_0)$ holds for all $i \in [p]$, then the point $x^\star = x^\star(I_0) \in \{0,1\}^{\mathcal{I}_{p,q}}$ with*

$$x^\star_{i,\alpha_i(I_0)} = 1 \text{ for all } i \in \Gamma(I_0),$$
$$x^\star_{i,\mu_i(I_0)} = 1 \text{ for all } i \in [p] \setminus \Gamma(I_0),$$

*and all other components being zero, is contained in $\mathrm{O}^{=}_{p,q} \cap F$.*

*Proof.* Due to $\alpha_i(I_0) \leq \alpha_{i-1}(I_0) + 1$ for all $i \in [p]$ with $i \geq 2$, the point $x^\star$ is contained in $\mathrm{O}^{=}_{p,q}$. It follows readily from the definitions that $x^\star$ does not have any 1-entry at a position in $I_0$. Thus, by Remark 4, we have $x^\star \in F$. □

Now we are prepared to characterize the case $\mathrm{O}^{=}_{p,q} \cap F = \varnothing$ (leading to pruning the corresponding node in the branch-and-cut tree) and to describe the set $I_0^\star$.

**Proposition 9.**

(1) *We have $\mathrm{O}^{=}_{p,q} \cap F = \varnothing$ if and only if there exists $i \in [p]$ with $\mu_i(I_0) > \alpha_i(I_0)$.*

(2) *If $\mu_i(I_0) \leq \alpha_i(I_0)$ holds for all $i \in [p]$, then the following is true.*

  (a) *For all $i \in [p] \setminus \Gamma(I_0)$, we have*

$$I_0^\star \cap \mathrm{row}_i \ = \ \{(i,j) \in \mathrm{row}_i \ : \ (i,j) \in I_0 \text{ or } j > \alpha_i(I_0)\}.$$

  (b) *For all $i \in [p]$ with $\mu_s(I_0) = \alpha_s(I_0)$, we have*

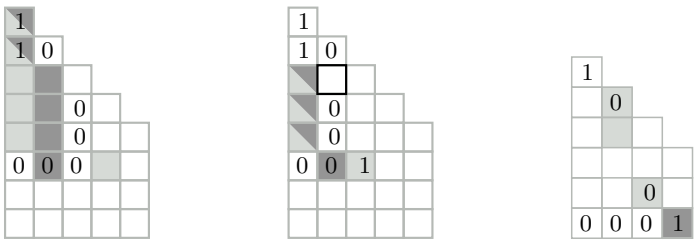
$$I_0^\star \cap \mathrm{row}_i = \mathrm{row}_i \setminus \{(i, \alpha_i(I_0))\}.$$

  (c) *For all $s \in \Gamma(I_0)$ with $\mu_s(I_0) < \alpha_s(I_0)$ the following holds: If there is some $i \geq s$ with $\mu_i(I_0) > \alpha_i(I_0 \cup \{(s, \alpha_s(I_0)\})$, then we have $I_0^\star \cap \mathrm{row}_s = \mathrm{row}_s \setminus \{(s, \alpha_s(I_0))\}$. Otherwise, we have*

$$I_0^\star \cap \mathrm{row}_s \ = \ \{(s,j) \in \mathrm{row}_s \ : \ (s,j) \in I_0 \text{ or } j > \alpha_s(I_0)\}.$$

*Proof.* Part 1 follows from Lemmas 7 and 8.

In order to prove Part 2, let us assume that $\mu_i(I_0) \leq \alpha_i(I_0)$ holds for all $i \in [p]$. For Part 2a, let $i \in [p] \setminus \Gamma(I_0)$ and $(i,j) \in \mathrm{row}_i$. Due to $I_0 \subset I_0^\star$, we only have to consider the case $(i,j) \notin I_0$. If $j > \alpha_i(I_0)$, then, by Lemma 7, we find $(i,j) \in I_0^\star$. Otherwise, the point that is obtained from $x^\star(I_0)$ (see




(a) (b) (c)

**Figure 2:** (a): Example for Prop. 9 (1). Light-gray entries indicate the entries $(i, \mu_i(I_0))$ and dark-gray entries indicate entries $(i, \alpha_i(I_0))$. (b): Example of fixing an entry to 1 for Prop. 9 (2c). As before light-gray entries indicate entries $(i, \mu_i(I_0))$. Dark-gray entries indicate entries $(i, \alpha_i(I_0 \cup \{(s, \alpha_s(I_0)\}))$ with $s = 3$. (c): gray entries show the SCI in the proof of Thm. 10.

Lemma 8) by moving the 1-entry in position $(i, \mu_i(I_0))$ to position $(i, j)$ is contained in $\mathrm{O}^{=}_{p,q} \cap F$, proving $(i, j) \notin I_0^\star$.

In the situation of Part 2b, the claim follows from Lemma 7 and the fact that $\mathrm{O}^{=}_{p,q} \cap F \neq \varnothing$ (due to Part 1).

For Part 2c, let $s \in \Gamma(I_0)$ with $\mu_s(I_0) < \alpha_s(I_0)$ and define $I_0' := I_0 \cup \{(s, \alpha_s(I_0))\}$. It follows that we have $\mu_i(I_0') = \mu_i(I_0)$ for all $i \in [p]$.

Let us first consider the case that there is some $i \geq s$ with $\mu_i(I_0) > \alpha_i(I_0')$. Part 1 (applied to $I_0'$ instead of $I_0$) implies that the $\mathrm{O}^{=}_{p,q} \cap F$ does not contain any vertex $x$ with $x_{s,\alpha_s(I_0)} = 0$. Therefore, we have $(s, \alpha_s(I_0)) \in I_1^\star$, and thus $I_0^\star \cap \mathrm{row}_s = \mathrm{row}_s \setminus \{(s, \alpha_s(I_0))\}$ holds (where for "$\subseteq$" we exploit $\mathrm{O}^{=}_{p,q} \cap F \neq \varnothing$ by Part 1, this time applied to $I_0$).

The other case of Part 2c follows from $s \notin \Gamma(I_0')$ and $\alpha_s(I_0') = \alpha_s(I_0) - 1$. Thus, Part 2a applied to $I_0'$ and $s$ instead of $I_0$ and $i$, respectively, yields the claim (because of $(s, \alpha_s(I_0)) \notin I_0^\star$ due to $s \in \Gamma(I_0)$ and $\mathrm{O}^{=}_{p,a} \cap F \neq \varnothing$). □

## 4.2. Sequential Fixing for Orbitopes

Let us, for some fixed $p \geq q \geq 2$, denote by $Ax \leq b$ the system of the inequalities $\mathbf{0} \leq x \leq \mathbf{1}$, the row-sum equations (each one written as two inequalities, in order to be formally correct) and all shifted column inequalities. Thus, as mentioned in Sect. 2, we have $\mathrm{O}^{=}_{p,q} = \{x \in \mathbb{R}^{\mathcal{I}_{p,q}} : Ax \leq b\}$. Let $A'x \leq b'$ be the subsystem of $Ax \leq b$ that does not contain any shifted column inequality that is not a column inequality.

**Theorem 10.** *For partitioning orbitopes $\mathrm{O}^{=}_{p,q}$, sequential fixing with column inequalities only is, in general, weaker than sequential fixing with all shifted column inequalities, i.e., there are examples with $\mathrm{Fix}_{Ax \leq b}(\mathrm{O}^{=}_{p,q}) \subsetneq \mathrm{Fix}_{A'x \leq b'}(\mathrm{O}^{=}_{p,q})$.*

*Proof.* The smallest example for this behavior is obtained from choosing $p = 6$, $q = 4$, $I_0 = \{(2,2), (5,3), (6,1), (6,2), (6,3)\}$ and $I_1 = \{(1,1), (6,4)\}$. The shifted column inequality with shifted column $\{(2,2), (3,2), (5,3)\}$ and bar $\{(6,4)\}$ allows to fix $x_{32}$ to one (see Fig. 2 (c)), while no column inequality (and no nonnegativity constraint and no row-sum equation) allows to fix any variable. □

**Algorithm 1** Orbitopal Fixing

1: Set $I_0^\star \leftarrow I_0$, $I_1^\star \leftarrow I_1$, $\mu_1 \leftarrow 1$, $\alpha_1 \leftarrow 1$, and $\Gamma = \varnothing$.
2: **for** $i = 2, \ldots, p$ **do**
3: compute $\mu_i \leftarrow \min\{j : (i,j) \notin I_0\}$.
4: **if** $\alpha_{i-1} = q(i)$ or $(i, \alpha_{i-1}+1) \in I_0$ **then**
5: $\alpha_i \leftarrow \alpha_{i-1}$
6: **else**
7: $\alpha_i \leftarrow \alpha_{i-1}+1$, $\Gamma \leftarrow \Gamma \cup \{i\}$
8: **if** $\mu_i > \alpha_i$ **then**
9: return "branch-and-cut node infeasible"
10: Set $I_0^\star \leftarrow I_0^\star \cup \{(i,j) : j > \alpha_i\}$.
11: **if** $|I_0^\star \cap \mathrm{row}_i| = q(i) - 1$ **then**
12: set $I_1^\star \leftarrow I_1^\star \cup (\mathrm{row}_i \setminus I_0^\star)$.
13: **for all** $s \in \Gamma$ with $(s, \alpha_s) \notin I_1^\star$ **do**
14: Set $\beta_s \leftarrow \alpha_s - 1$.
15: **for** $i = s+1, \ldots, p$ **do**
16: **if** $\beta_{i-1} = q(i)$ or $(i, \beta_{i-1}+1) \in I_0$ **then**
17: $\beta_i \leftarrow \beta_{i-1}$
18: **else**
19: $\beta_i \leftarrow \beta_{i-1}+1$
20: **if** $\mu_i > \beta_i$ **then**
21: $I_1^\star \leftarrow I_1^\star \cup \{(s, \alpha_s)\}$ and $I_0^\star \leftarrow \mathrm{row}_s \setminus \{(s, \alpha_s)\}$.
22: Proceed with the next $s$ in Step 13.

At first sight, it is not clear whether sequential fixing with the exponentially large system $Ax \leq b$ can be done efficiently. A closer look at the problem reveals, however, that one can utilize the linear time separation algorithm for shifted column inequalities (mentioned in Sect. 2) in order to devise an algorithm for this sequential fixing whose running time is bounded by $\mathrm{O}(\varrho pq)$, where $\varrho$ is the number of variables that are fixed by the procedure.

However, it turns out that this is not the preferable choice for doing the fixing. In fact, we will describe below a linear time algorithm for the problem to solve the orbitopal fixing problem, i.e., to compute, for a given cube face $F$, the simultaneous fixing $\mathrm{Fix}_F(\mathrm{O}^=_{p,q})$ rather than the sequential fixing $\mathrm{Fix}_F(Ax \leq b)$. While this is very efficient, the next result shows that (despite the situation for 0/1-polytopes in general, see Proposition 3) it does not yield a stronger fixing than the sequential fixing with shifted column inequalities does.

**Theorem 11.** *For partitioning orbitopes $\mathrm{O}^=_{p,q}$, sequential fixing with shifted column inequalities (and row-sum equations and nonnegativity constraints) is as strong as orbitopal fixing, i.e., we have $\mathrm{Fix}_F(\mathrm{O}^=_{p,q}) = \mathrm{Fix}_{Ax \leq b}(\mathrm{O}^=_{p,q})$ if $F$ is the cube face defined by $(I_0)$.*

We omit the proof, which uses similar arguments as above.

### 4.3. An Algorithm for Orbitopal Fixing

Algorithm 1 describes a method to compute the simultaneous fixing $(I_0^\star, I_1^\star)$ from $(I_0, I_1)$ (which are assumed to satisfy Properties P1 and P2). Note that we use $\beta_i$ for $\alpha_i(I_0 \cup \{\alpha_s(I_0)\})$.

**Theorem 12.** *A slight modification of Algorithm 1 can be implemented in such a way that it solves the orbitopal fixing problem in time $\mathrm{O}(pq)$.*

*Proof.* The correctness of the algorithm follows by the structural results given in Proposition 9.

In order to prove the statement on the running time, let us assume that the data structures for the sets $I_0$, $I_1$, $I_0^\star$, and $I_1^\star$ allow both membership testing and addition of single elements in constant time (e.g., the sets can be stored as bit vectors).

As none of the Steps 3 to 12 needs more time than $\mathrm{O}(q)$, we only have to take care of the second part of the algorithm starting in Step 13. (In fact, used verbatim as described above, the algorithm might need time $\Omega(p^2)$.)

For $s, s' \in \Gamma$ with $s < s'$ denote the corresponding $\beta$-values by $\beta_i$ $(i \geq s)$ and by $\beta'_i$ $(i \geq s')$, respectively. We have $\beta_i \leq \beta'_i$ for all $i \geq s'$, and furthermore, if equality holds for one of these $i$, we can deduce $\beta_k = \beta'_k$ for all $k \geq i$. Thus, as soon as a pair $(i, \beta_i)$ is used a second time in Step 20, we can break the for-loop in Step 15 and reuse the information that we have obtained earlier.

This can, for instance, be organized by introducing, for each $(i, j) \in \mathcal{I}_{p,q}$, a flag $f(i, j) \in \{\text{red}, \text{green}, \text{white}\}$ (initialized by white), where $f(i, j) =$ red / green means that we have already detected that $\beta_i = j$ eventually leads to a positive/negative test in Step 20. The modifications that have to be applied to the second part of the algorithm are the following: The selection of the elements in $\Gamma$ in Step 13 must be done in increasing order. Before performing the test in Step 20, we have to check whether $f(i, \beta_i)$ is green. If this is true, then we can proceed with the next $s$ in Step 13 after setting all flags $f(k, \beta_k)$ to green for $s \leq k < i$. Similarly, we need to set all flags $f(k, \beta_k)$ to red for $s \leq k \leq i$ before switching to the next $s$ in Step 22. And finally, we have to set all flags $f(k, \beta_k)$ to green for $s \leq k \leq p$ at the end of the body of the $s$-loop starting in Step 13.

As the running time of this part of the algorithm is proportional to the number of flags changed from white to red or green, the total running time indeed is bounded by $\mathrm{O}(pq)$ (since a flag is never reset). □

## 5. Computational Experiments

We performed computational experiments for the graph partitioning problem mentioned in the Introduction. The code is based on the SCIP framework by Achterberg [1], and we use CPLEX 10.01 as the basic LP solver. The computations were performed on a 3.4 GHz Pentium 4 machine running Linux. All computation times are CPU seconds and are subject to a *time limit of two hours*. Since in this paper we are not interested in the performance of heuristics we initialized all computations with the *optimal primal solution*. We compare different variants of the code by counting *winning* instances. An instance is a winner for variant A compared to variant B, if A finished within the time limit and B did not finish or needed a larger cpu time; if A did not finish, then the instance is a winner for A, if B did not finish and has a larger gap. If the difference between the times and gaps are below 1 sec. and 0.1, respectively, the instance is not counted.

In all variants, we fix the variables $x_{ij}$ with $j > i$ to zero. Furthermore, we heuristically separate clique inequalities $\sum_{i,j \in C} y_{ij} \geq 1$, where $C \subseteq V$ is a clique of size at least $q + 1$. We follow the ideas of Eisenblätter [5]: we

**Table 1:** Results of the branch-and-cut algorithm. All entries are averages over three instances.

| | | basic | | OF | | | OF – first | | |
|---|---|---|---|---|---|---|---|---|---|
| $m$ | $q$ | nsub | cpu | nsub | cpu | #OF | nsub | cpu | #OF |
| 200 | 3 | 905.7 | 4.77 | 719.7 | 4.22 | 5.7 | 858.3 | 2.47 | 5.7 |
| 200 | 6 | 406.3 | 1.06 | 98.3 | 0.31 | 49.0 | 245.3 | 0.41 | 150.3 |
| 200 | 9 | 1.0 | 0.04 | 1.0 | 0.05 | 0.0 | 1.0 | 0.04 | 0.0 |
| 200 | 12 | 1.0 | 0.06 | 1.0 | 0.05 | 0.0 | 1.0 | 0.05 | 0.0 |
| 300 | 3 | 3657.7 | 79.86 | 2849.0 | 63.35 | 7.0 | 2910.3 | 45.18 | 7.0 |
| 300 | 6 | 99040.3 | 460.73 | 10839.7 | 48.96 | 363.0 | 45473.3 | 140.38 | 14149.7 |
| 300 | 9 | 16260.7 | 55.13 | 330.0 | 1.47 | 249.0 | 986.3 | 2.79 | 1394.7 |
| 300 | 12 | 1.0 | 0.13 | 1.0 | 0.14 | 0.0 | 1.0 | 0.14 | 0.0 |
| 400 | 3 | 22400.3 | 1155.93 | 17276.3 | 927.35 | 8.3 | 22474.3 | 921.62 | 8.3 |
| 400 | 6 | 163737.0 | 7200.06 | 334417.0 | 7062.46 | 1547.0 | 305555.0 | 7200.04 | 134886.3 |
| 400 | 9 | 272622.0 | 7200.02 | 52378.3 | 662.64 | 6689.3 | 586002.7 | 5499.17 | 434991.0 |
| 400 | 12 | 616485.0 | 7200.01 | 4345.0 | 41.64 | 2746.3 | 38294.0 | 200.80 | 58536.3 |

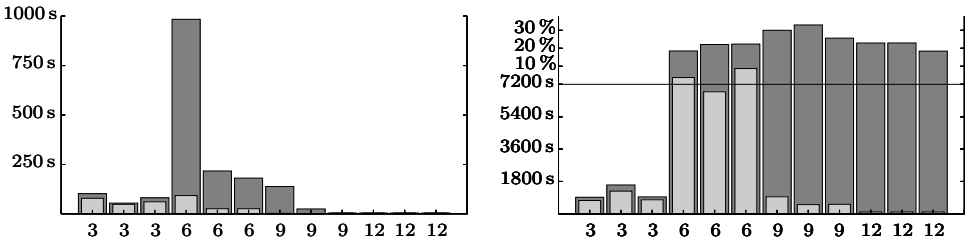



**Figure 3:** Computation times/gaps for the basic version (dark gray) and the version with orbitopal fixings (light gray). Left for all instances with $m = 300$ and right for $m = 400$. The number of partitions $q$ is indicated on the $x$-axis. Values above 7200.0 indicate the gap in percent.

generate the graph $G' = (V, E')$ with $\{i, k\} \in E'$ if and only if $\{i, k\} \in E$ and $y^\star_{ik} < q(q+1)/2$, where $y^\star$ is the $y$-part of an LP-solution. We search for maximum cliques in $G'$ with the branch-and-bound method implemented in SCIP (with a node limit of 10 000).

Our default branching rule combines *first index* and *reliability branching*. We branch on the first fractional variable in the variable order used for defining orbitopes, but we skip columns in which a 1 has appeared before. If no fractional variable could be found, we perform reliability branching as described by Achterberg, Koch, and Martin [2].

We generated 9 random instances with 30 vertices each. The graphs have $m = 200$ (sparse), 300 (medium), and 400 (dense) edges, where edges are picked uniformly at random (without recourse) until the specified number of edges is reached. For each size we generated three instances. For each instance we computed results for $q = 3$, 6, 9, and 12.

In a first experiment we tested the speedup that can be obtained by performing orbitopal fixing. For this we compare the basic variant (basic) without symmetry breaking (except the zero-fixing of the upper right variables) and the version in which we use orbitopal fixing (OF), see Table 1 for the results. Columns "nsub" give the number of nodes in the branch-and-bound tree. The results show that orbitopal fixing is clearly superior (OF winners: 22, basic winners: 0), see also Figure 3.

Table 1 shows that the sparse instances are extremely easy, the instances with $m = 300$ are quite easy, while the dense instances are hard. One effect

is that often for small $m$ and large $q$ the optimal solution is 0 and hence no work has to be done. For $m = 300, 400$, the hardest instances arise when $q = 6$. It seems that for $q = 3$ the small number of variables helps, while for $q = 12$ the small objective function values help. On the other side, symmetry breaking methods get more effective when $q$ gets larger.

In a second experiment we investigated the symmetry breaking built into CPLEX. We suspect that it breaks symmetry within the tree, but no detailed information was available. We ran CPLEX 10.01 on the IP formulation stated in Sect. 1. In one variant, we fixed variables $x_{ij}$ with $j > i$ to zero, but turned symmetry breaking off. In a second variant, we turned symmetry breaking on and did not fix variables to zero (otherwise CPLEX seems not to recognize the symmetry). These two variants performed about equally good (turned-on winners: 13, turned-off winners: 12). The variant with no symmetry breaking and no fixing of variables performed extremely badly. The results obtained by the OF-variant above are superior to the best CPLEX results (CPLEX could not solve 10 instances within the time limit, while OF could not solve 2). Probably this is at least partially due to the separation of clique inequalities and the special branching rule in our code.

The third experiment concerns branching rules. We used orbitopal fixing and compared our default branching rule with first index branching. First index branching with orbitopal fixing is close to Margot's approach (mentioned in the Introduction) with the row-by-row variable order. It turns out that the version using our branching rule beats the first index rule (default winners: 14, first index: 8 – the total avg. running times are 777.59 and 1236.45 sec., respectively), see Table 1. Indeed, it seems that our branching rule avoids much of the symmetry, whereas the first index variant needs many more orbitopal fixings to do the same (see column "#OF"). Apparently, the additional freedom to choose an arbitrary branching rule improves the performance.

Finally, we turned off orbitopal fixing and separated shifted column inequalities in every node of the tree. The results are that the OF-version is better than this variant (OF winners: 12, SCI winners: 3), but the results are relatively close together (OF average time: 777.59, SCI average time: 792.82). Although in theory, the SCI-version does not produce more nodes in the tree, the LPs get harder and the process slows down a bit.

## References


[1] T. Achterberg, *SCIP – A framework to integrate constraint and mixed integer programming*, Report 04–19, ZIB, 2004. `http://www.zib.de/Publications/abstracts/ZR-04-19/`.

[2] T. Achterberg, T. Koch, and A. Martin, *Branching rules revisited*, Oper. Res. Lett., 33 (2005), pp. 42–54.

[3] S. Chopra and M. Rao, *The partition problem.*, Math. Programming, 59 (1993), pp. 87–115.

[4] ———, *Facets of the k-partition polytope.*, Discrete Appl. Math., 61 (1995), pp. 27–48.

[5] A. Eisenblätter, *Frequency Assignment in GSM Networks: Models, Heuristics, and Lower Bounds*, PhD thesis, TU Berlin, 2001.

[6] J. Falkner, F. Rendl, and H. Wolkowicz, *A computational study of graph partitioning.*, Math. Programming, 66 (1994), pp. 211–239.

[7] C. Ferreira, A. Martin, C. de Souza, R. Weismantel, and L. Wolsey, *Formulations and valid inequalities of the node capacitated graph partitioning problem.*, Math. Programming, 74 (1996), pp. 247–266.

[8] ———, *The node capacitated graph partitioning problem: A computational study.*, Math. Programming, 81 (1998), pp. 229–256.

[9] M. Grötschel and Y. Wakabayashi, *A cutting plane algorithm for a clustering problem.*, Math. Programming, 45 (1989), pp. 59–96.

[10] ———, *Facets of the clique partitioning polytope.*, Math. Programming, 47 (1990), pp. 367–387.

[11] V. Kaibel and M. E. Pfetsch, *Packing and partitioning orbitopes.* To appear in: Math. Programming, Ser. A.

[12] G. Kochenberger, F. Glover, B. Alidaee, and H. Wang, *Clustering of microarray data via clique partitioning.*, J. Comb. Optim., 10 (2005), pp. 77–92.

[13] F. Margot, *Pruning by isomorphism in branch-and-cut*, Math. Programming, 94 (2002), pp. 71–90.

[14] F. Margot, *Exploiting orbits in symmetric ILP.*, Math. Programming, 98 (2003), pp. 3–21.

[15] A. Mehrotra and M. A. Trick, *Cliques and clustering: A combinatorial approach.*, Oper. Res. Lett., 22 (1998), pp. 1–12.

[16] M. M. Sørensen, *Polyhedral computations for the simple graph partitioning problem*, tech. rep., AArhus School of Business, 2005.

Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany
*E-mail address*: `[kaibel,peinhardt,pfetsch]@zib.de`