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, revised Dezember 2007)

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 order 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 such 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 symmetry. The method relies on certain polyhedra, called orbitopes, which have been investigated in [11]. It does, however, 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 $i \in [p] := \{1, \dots, p\}$ 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 $\{i, k\} \in E$

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 *Mathematics for key technologies* in Berlin.

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

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

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

The x-variables describe a 0/1-matrix 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 [3, 4, 5], for instance as a relaxation of frequency assignment problems in mobile telecommunication networks. The maximization version is relevant as well [6, 12]. 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 [7, 8, 18, 19]. 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 solving small instances requires 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 (viewed as a 0/1-matrix) 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 interior points of the convex hulls of the individual orbits are responsible for quite weak linear programming (LP) bounds. We will, however, not address this second point in this paper.

In order to remove symmetry, the above model for the graph partitioning problem is often replaced by models containing only edge variables, see, e.g. [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 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 is derived; these polytopes are called *orbitope*. 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. In fact, already the column inequalities are sufficient for this purpose.

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 of avoiding the addition of these inequalities to the LPs is to derive 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 for the whole subtree rooted 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. With orbitopes, however, 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 present a linear time algorithm for *orbitopal fixing*, i.e., for solving the problem to compute simultaneous fixings for orbitopes (Theorem 11).

- We show that, for general 0/1-polytopes, sequential fixing, even with complete and irredundant linear descriptions, is weaker than simultaneous fixing (Theorem 3), We clarify the relationships between different versions of sequential fixing with (shifted) column inequalities, where (despite the situation for general 0/1-polytopes) the strongest one is as strong as orbitopal fixing (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.

Margot [14, 15, 17] 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.

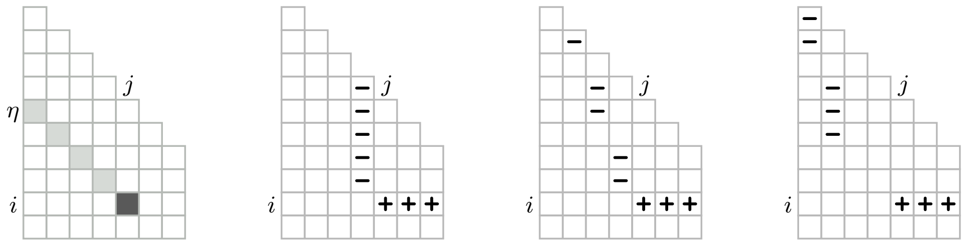
Another approach has recently been proposed by Linderoth et al. [13] (in this volume). They exploit the symmetry arising in each node of a branch-and-bound tree when all fixed variables are removed from the model. Thus one may find additional local symmetries. Nevertheless, for partitioning type symmetries one still may miss some part of the (fixed) global symmetry we are dealing with.

We will elaborate on the relations between orbitopal fixing, isomorphism pruning, and orbital branching in more detail in a journal version of the paper.

2. Orbitopes

Throughout the paper, let p and q be 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 $j \in [q-1]$). Let the symmetric group of size q act on $\{0,1\}^{[p]\times[q]}$ via permuting the columns. 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. As these vertices 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 $q(i) := \min\{i,q\}$ components.

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*.



(a) (b) (c) (d)

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": For $j \in [q]$ and $1 \le \eta \le p - j + 1$, define $\langle \eta, j\rangle := (j + \eta - 1, j)$. Thus (as before) i and j denote the row and the column, respectively, while η is the index of the diagonal (counted from above) containing the respective element; see Figure 1 (a) for an example.

A set $S = \{\langle 1, c_1\rangle, \langle 2, c_2\rangle, \dots, \langle \eta, c_\eta\rangle\} \subset \mathcal{I}_{p,q}$ with $c_1 \le c_2 \le \cdots \le c_\eta$ and $\eta \ge 1$ is called a *shifted column.* For $(i, j) = \langle \eta, j\rangle \in \mathcal{I}_{p,q}$, a shifted column S as above with $c_\eta < j$, and $B = \{(i, j), (i, j + 1), \dots, (i, q(i))\}$, 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 $x \in \mathbb{R}^{\mathcal{I}_{p,q}}$, we denote by $x(A)$ 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* C^d *defined by* (I_0, I_1). All nonempty faces of C^d are of this type.

For a polytope $P \subseteq \mathrm{C}^d$ and for a face F of C^d defined by (I_0, I_1), we denote by $\mathrm{Fix}_F(P)$ the smallest face of C^d that contains $P \cap F \cap \{0, 1\}^d$ (i.e.,

$\mathrm{Fix}_F(P)$ is the intersection of all faces of C^d that contain $P \cap F \cap \{0,1\}^d$). If $\mathrm{Fix}_F(P)$ is the nonempty 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 $I_1 \subseteq I_1^\star$, or $\mathrm{Fix}_F(P) = \varnothing$. 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 node can either be pruned, or 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. Unless $\mathrm{Fix}_F(P) = \varnothing$, we call $(I_0^\star, I_1^\star)$ the *fixing of P at (I_0, I_1)*. Similarly, we call $\mathrm{Fix}_F(P)$ the *fixing* of P at F.

Remark 2. *If $P, P' \subseteq \mathrm{C}^d$ are two polytopes with $P \subseteq P'$ and F and F' are two faces of 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 $P_r = \{x \in \mathrm{C}^d \;:\; a_r^T x \leq b_r\}$ is the knapsack relaxation of P for the rth-row $a_r^T x \leq b_r$ of $Ax \leq b$, where $r = 1, \dots, m$. Let F be some face of C^d. The face G of 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 $\mathrm{Fix}_F(Ax \leq b)$ (unless this face is empty) the *sequential fixing of $Ax \leq b$ at (I_0, I_1)*. In the context of sequential fixing we often refer to (the computation of) $\mathrm{Fix}_F(P)$ as *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 full-dimensional 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 fix 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 inequalities are facet defining ($x_4 \geq 0$ and $x_3 \leq 1$ are implied). $\square$

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 $\ell \in [q(i)] \setminus \{j\}$.

In particular, P1 and P2 imply that $I_0 \cap I_1 = \varnothing$. 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 we have $\mathrm{O}^{=}_{p,q} \cap F = \varnothing$. 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 $(I_0^\star, I_1^\star)$, if $\mathrm{Fix}_F(\mathrm{O}^{=}_{p,q})$ is not empty. *Orbitopal fixing* is the problem to compute the simultaneous fixing $(I_0^\star, I_1^\star)$ from (I_0, I_1), or determine that $\mathrm{Fix}_F(\mathrm{O}^{=}_{p,q}) = \varnothing$.

Remark 5. *If $\mathrm{Fix}_F(\mathrm{O}^{=}_{p,q}) \neq \varnothing$, 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. Intersection of Orbitopes with Cube Faces

We start by deriving some structural results on orbitopes that are crucial in our context. 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 $\mathrm{O}^{=}_{p,q}$.

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 α-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 $i \in [p]$. 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 $\alpha_i(I_0) < q(i) \leq i$ 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, $x(B) - x(S_k(I_0)) \le 0$ 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 $\alpha_i(I_0) \ge q(i) - 1$, the claim holds trivially. Otherwise, $B' := B \setminus \{(i, \alpha_i(I_0) + 1)\} \neq \varnothing$. Similarly to the first case, now the shifted column inequality $x(B') - x(S_{i-1}(I_0)) \le 0$ 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 we have $\mu_i(I_0) \le \alpha_i(I_0)$ for all $i \in [p]$, then the point $x^\star = x^\star(I_0) \in \{0,1\}^{\mathcal{I}_{p,q}}$ with $x^\star_{i,\alpha_i(I_0)} = 1$ for all $i \in \Gamma(I_0)$ and $x^\star_{i,\mu_i(I_0)} = 1$ 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 $\alpha_i(I_0) \le \alpha_{i-1}(I_0) + 1$ for all $i \in [p]$ with $i \ge 2$, the point $x^\star$ is contained in $\mathrm{O}^{=}_{p,q}$. It follows from the definitions that $x^\star$ does not have a 1-entry at a position in I_0. Thus, by Remark 4, we have $x^\star \in F$. □

We now characterize the case $\mathrm{O}^{=}_{p,q} \cap F = \varnothing$ (leading to pruning the corresponding node in the branch-and-cut tree) and 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 $\mu_i(I_0) \le \alpha_i(I_0)$ 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 $\mu_i(I_0) = \alpha_i(I_0)$, 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 $i \ge s$ 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 $\mu_i(I_0) \le \alpha_i(I_0)$ 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 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 because $\mathrm{O}^{=}_{p,q} \cap F \neq \varnothing$ (due to Part 1).

For Part 2c, let $s \in \Gamma(I_0)$ with $\mu_s(I_0) < \alpha_s(I_0)$ and define the new set $I_0' := I_0 \cup \{(s, \alpha_s(I_0))\}$. 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 $i \ge s$ with $\mu_i(I_0) > \alpha_i(I_0')$. Part 1 (applied to I_0' instead of I_0) implies that $\mathrm{O}^{=}_{p,q} \cap F$ does not contain a

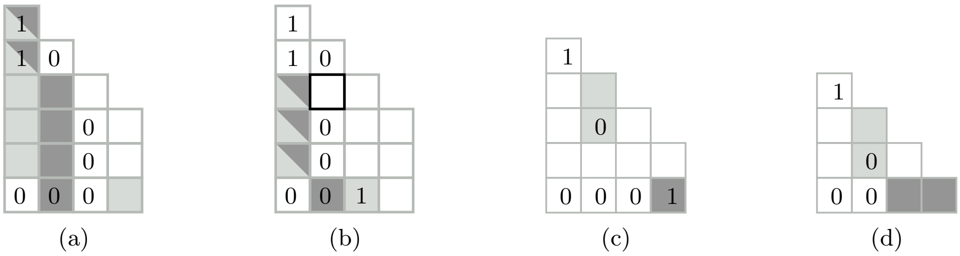


Figure 2: (a): Example for Prop. 9 (1). Light-gray entries indicate the entries $(i, \mu_i(I_0))$ and dark-gray entries indicate entries $(i, \alpha_i(I_0))$. (b): Example of fixing an entry to 1 for Prop. 9 (2c). As before light-gray entries indicate entries $(i, \mu_i(I_0))$. Dark-gray entries indicate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

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 $\mathrm{O}_{p,a}^{=} \cap F \neq \varnothing$). $\square$

4.2. Sequential Fixing for Orbitopes

Let us, for some fixed $p \geq q \geq 2$, denote by $\mathcal{S}_{\mathrm{SCI}}$ the system of the nonnegativity inequalities, the row-sum equations (each one written as two inequalities, in order to be formally correct) and all shifted column inequalities. Thus, according to Theorem 1, $\mathrm{O}_{p,q}^{=}$ is the set of all $x \in \mathbb{R}^{\mathcal{I}_{p,q}}$ that satisfy $\mathcal{S}_{\mathrm{SCI}}$. Let $\mathcal{S}_{\mathrm{CI}}$ be the subsystem of $\mathcal{S}_{\mathrm{SCI}}$ containing only the column inequalities (and all nonnegativity inequalities and row-sum equations).

At first sight, it is not clear whether sequential fixing with the exponentially large system $\mathcal{S}_{\mathrm{SCI}}$ 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 ϱ is the number of variables that are fixed by the procedure.

In fact, one can achieve more: One can compute sequential fixings with respect to the affine hull of the orbitope. In order to explain this, consider a polytope $P = \{x \in \mathrm{C}^d : Ax \leq b\}$, and let $S \subseteq \mathbb{R}^d$ be some affine subspace containing P. As before, we denote the knapsack relaxations of P obtained from $Ax \leq b$ by $P_1, \ldots, P_m$. Let us define $\mathrm{Fix}_F^S(P_r)$ as the smallest cube-face that contains $P_r \cap S \cap \{0,1\}^d \cap F$. Similarly to the definition of $\mathrm{Fix}_F(Ax \leq b)$, denote by $\mathrm{Fix}_F^S(Ax \leq b)$ the face of C^d that is obtained by setting $G := F$ and then iteratively replacing G by $\mathrm{Fix}_G^S(P_r)$ as long as there is some $r \in [m]$ with $\mathrm{Fix}_G^S(P_r) \subsetneq G$. We call $\mathrm{Fix}_F^S(Ax \leq b)$ the *sequential fixing of $Ax \leq b$ at F relative to S*. Obviously, we have $\mathrm{Fix}_F(P) \subseteq \mathrm{Fix}_F^S(Ax \leq b) \subseteq \mathrm{Fix}_F(Ax \leq b)$. In contrast to sequential fixing, sequential fixing relative to affine subspaces *in general* is NP-hard (as it can be used to decide whether a linear equation has a 0/1-solution).

Theorem 10.

(1) *There are cube-faces F^1, F^2, F^3 with the following properties:*
   (a) $\mathrm{Fix}_{F^1}(\mathcal{S}_{SCI}) \subsetneq \mathrm{Fix}_{F^1}(\mathcal{S}_{CI})$
   (b) $\mathrm{Fix}_{F^2}^{\mathrm{aff}(\mathrm{O}^=_{p,q})}(\mathcal{S}_{CI}) \subsetneq \mathrm{Fix}_{F^2}(\mathcal{S}_{SCI})$
   (c) $\mathrm{Fix}_{F^3}^{\mathrm{aff}(\mathrm{O}^=_{p,q})}(\mathcal{S}_{SCI}) \subsetneq \mathrm{Fix}_{F^3}^{\mathrm{aff}(\mathrm{O}^=_{p,q})}(\mathcal{S}_{CI})$

(2) *For all cube-faces F, we have* $\mathrm{Fix}_{F}^{\mathrm{aff}(\mathrm{O}^=_{p,q})}(\mathcal{S}_{SCI}) = \mathrm{Fix}_F(\mathrm{O}^=_{p,q})$.

Proof. For Part 1(a), we chose $p = 5$, $q = 4$, and define the cube-face F_1 via $I_0^1 = \{(3,2),(5,1),(5,2),(5,3)\}$ and $I_1^1 = \{(1,1),(5,4)\}$. The shifted column inequality with shifted column $\{(2,2),(3,2)\}$ and bar $\{(5,4)\}$ allows to fix x_{22} to one (see Fig. 2 (c)), while no column inequality (and no nonnegativity constraint and no row-sum equation) allows to fix any variable.

For Part 1(b), let $p = 4$, $q = 4$, and define F^2 via $I_0^2 = \{(3,2),(4,1),(4,2)\}$ and $I_1^2 = \{(1,1)\}$. Exploiting $x_{43} + x_{44} = 1$ for all $x \in \mathrm{aff}(\mathrm{O}^=_{p,q}) \cap F^2$, we can use the column inequality with column $\{(2,2),(3,2)\}$ and bar $\{(4,3),(4,4)\}$ to fix x_{22} to one (see Fig. 2 (d)), while no fixing is possible with $\mathcal{S}_{\mathrm{SCI}}$ only.

For Part 1(c), we use $F^3 = F^1$. The proof of Part 2 is omitted here. □

The different versions of sequential fixing for partitioning orbitopes are dominated by each other in the following sequence: $\mathcal{S}_{\mathrm{CI}} \to \{\mathcal{S}_{\mathrm{SCI}}, \text{affine } \mathcal{S}_{\mathrm{CI}}\} \to$ affine $\mathcal{S}_{\mathrm{SCI}}$, which finally is as strong as orbitopal fixing. For each of the arrows there exists an instance for which dominance is strict. The examples in the proof of Theorem 10 also show that there is no general relation between $\mathcal{S}_{\mathrm{SCI}}$ and affine $\mathcal{S}_{\mathrm{CI}}$.

In particular, we could compute orbitopal fixings by the polynomial time algorithm for sequential fixing relative to $\mathrm{aff}(\mathrm{O}^=_{p,q})$. It turns out, however, that this is not the preferable choice. In fact, we will describe below a linear time algorithm for solving the orbitopal fixing problem directly.

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 β_i for $\alpha_i(I_0 \cup \{(s, \alpha_s(I_0))\})$.

Theorem 11. *A slight modification of Algorithm 1 solves the orbitopal fixing problem in time* $\mathrm{O}(pq)$.

Proof. The correctness of the algorithm follows from 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 β-values by β_i ($i \geq s$) and by β'_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 $\beta_k = \beta'_k$

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** $i = 2, \dots, p$ **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 "Orbitopal fixing is empty"
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** $i = s+1, \dots, p$ **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.

for all $k \geq i$. Thus, as soon as a pair (i, β_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 Γ 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 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 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). $\square$

5. Computational Experiments

We performed computational experiments for the graph partitioning problem mentioned in the introduction. The code is based on the SCIP 0.90 framework by Achterberg [1], and we use CPLEX 10.01 as the basic LP solver. The computations were performed on a 3.2 GHz Pentium 4 machine with 2 GB of main memory and 2 MB cache running Linux. All computation times are CPU seconds and are subject to a *time limit of four hours.* Since in this paper we are not interested in the performance of heuristics, we

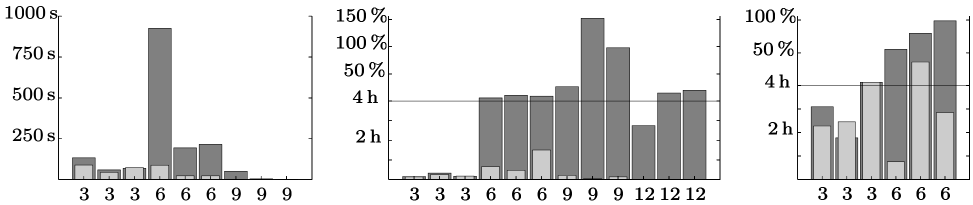


Figure 3: Computation times/gaps for the basic version (dark gray) and the version with orbitopal fixing (light gray). From left to right: instances with $n = 30$, $m = 300$, instances for $n = 30$, $m = 400$, instances for $n = 50$, $m = 560$. The number of partitions q is indicated on the x-axis. Values above 4 hours indicate the gap in percent.

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 in case that B did also not finish, leaving, however, a larger gap than A. If the difference between the times or 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 general clique inequalities $\sum_{i,j \in C} y_{ij} \geq b$, where $b = \frac{1}{2}t(t-1)(q-r) + \frac{1}{2}t(t+1)r$ and $C \subseteq V$ is a clique of size $tq + r > q$ with integers $t \geq 1$, $0 \leq r < q$ (see [3]). The separation heuristic for a fractional point $y^\star$ follows ideas of Eisenblätter [5]. We generate the graph $G' = (V, E')$ with $\{i, k\} \in E'$ if and only if $\{i, k\} \in E$ and $y^\star_{ik} < b(b+1)/2$, 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 branch-and-bound node limit of 10 000) and check whether the corresponding inequality is violated.

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

We generated random instances with n vertices and m edges of the following types. For $n = 30$ we used $m = 200$ (*sparse*), 300 (*medium*), and 400 (*dense*). Additionally, for $n = 50$ we choose $m = 560$ in search for the limits of our approach. For each type we generated three instances by picking edges uniformly at random (without recourse) until the specified number of edges is reached. The edge weights are drawn independently uniformly at random from the integers $\{1, \ldots, 1000\}$. 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 variant (*basic*) without symmetry breaking (except for the zero-fixing of the upper right x-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: 26, basic winners: 3), see also Figure 3.

Table 1: Results of the branch-and-cut algorithm. All entries are rounded averages over three instances. CPU times are given in seconds.

			basic		Iso Pruning		OF		
n	m	q	nsub	cpu	nsub	cpu	nsub	cpu	#OF
30	200	3	1 082	6	821	4	697	5	6
30	200	6	358	1	122	0	57	0	25
30	200	9	1	0	1	0	1	0	0
30	200	12	1	0	1	0	1	0	0
30	300	3	3 470	87	2 729	64	2 796	69	7
30	300	6	89 919	445	63 739	168	8 934	45	353
30	300	9	8 278	19	5 463	5	131	0	73
30	300	12	1	0	1	0	1	0	0
30	400	3	11 317	755	17 433	800	9 864	660	8
30	400	6	458 996	14 400	1 072 649	11 220	159 298	3 142	1 207
30	400	9	2 470 503	14 400	1 048 256	2 549	70 844	450	7 305
30	400	12	3 668 716	12 895	37 642	53	2 098	12	1 269
50	560	3	309 435	10 631	290 603	14 400	288 558	10 471	10
50	560	6	1 787 989	14 400	3 647 369	14 400	1 066 249	9 116	4 127
50	560	9	92	0	2 978	5	10	0	10
50	560	12	1	0	1	0	1	0	0

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 $m = 300$ and 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. Of course, symmetry breaking methods become more important when q gets larger.

In a second experiment we investigated the symmetry breaking capabilities 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 clearly 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.

In another experiment, we turned off orbitopal fixing and separated shifted column inequalities in every node of the tree. The results show that the OF-version is slightly better than this variant (OF winners: 13, SCI winners: 10), but the results are quite close (OF average time: 1563.3, SCI average time: 1596.7). Although by Part 2 of Theorem 10, orbitopal fixing is not stronger than fixing with SCIs (with the same branching decisions), the LPs get harder and the process slows down a bit.

Finally, we compared orbitopal fixing to the isomorphism pruning approach of Margot. We implemented the *ranked branching rule* (see [16]) adapted to the special symmetry we exploit, which simplifies Margot's algorithm significantly. It can be seen from Table 1 that isomorphism pruning is inferior to both orbitopal fixing (OF winners: 25, isomorphism pruning winners: 3) and shifted column inequalities (26:2), but is still a big improvement over the basic variant (23:7).

6. Concluding Remarks

The main contribution of this paper is a linear time algorithm for the orbitopal fixing problem, which provides an efficient way to deal with partitioning type symmetries in integer programming models. The result can easily be extended to "packing orbitopes" (where, instead of $x(\mathrm{row}_i) = 1$, we require $x(\mathrm{row}_i) \leq 1$). Our proof of correctness of the procedure uses the linear description of $\mathrm{O}^{=}_{p,q}$ given in [11]. However, we only need the validity of the shifted column inequalities in our arguments. In fact, one can devise a similar procedure for the case where the partitioning constraints $x(\mathrm{row}_i) = 1$ are replaced by covering constraints $x(\mathrm{row}_i) \geq 1$, though, for the corresponding "covering orbitopes" no complete linear descriptions are known at this time. A more detailed treatment of this will be contained in a journal version of the paper, which will also include comparisons to the isomorphism pruning method [14, 15, 17] and to orbital branching [13].

References

[1] T. Achterberg, *SCIP – A framework to integrate constraint and mixed integer programming*, Report 04-19, Zuse Institute Berlin, 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. Program., 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. Program., 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. Program., 74 (1996), pp. 247–266.

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

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

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

[11] V. Kaibel and M. E. Pfetsch, *Packing and partitioning orbitopes*, Math. Program., (2007). In press.

[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] J. Linderoth, J. Ostrowski, F. Rossi, and S. Smriglio, *Orbital branching*, in Proceedings of IPCO XII, M. Fischetti and D. Williamson, eds., vol. 4513 of LNCS, Springer-Verlag, 2007, pp. 106–120.

[14] F. Margot, *Pruning by isomorphism in branch-and-cut*, Math. Program., 94 (2002), pp. 71–90.
[15] ———, *Exploiting orbits in symmetric ILP*, Math. Program., 98 (2003), pp. 3–21.
[16] ———, *Small covering designs by branch-and-cut*, Math. Program., 94 (2003), pp. 207–220.
[17] ———, *Symmetric ILP: Coloring and small integers*, Discrete Opt., 4 (2007), pp. 40–62.
[18] A. Mehrotra and M. A. Trick, *Cliques and clustering: A combinatorial approach*, Oper. Res. Lett., 22 (1998), pp. 1–12.
[19] M. M. Sørensen, *Polyhedral computations for the simple graph partitioning problem*, working paper L-2005-02, Århus School of Business, 2005.

(V. Kaibel and M. Peinhardt) Otto-von-Guericke Universität Magdeburg, Fakultät für Mathematik, Universitätsplatz 2, 39106 Magdeburg, Germany
E-mail address: `{kaibel,peinhard}@ovgu.de`

(M. E. Pfetsch) Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany
E-mail address: `pfetsch@zib.de`