Statistica Neerlandica (2007) Vol. 61, nr. 1, pp. 35–60

Disconnecting graphs by removing vertices: a polyhedral approach

Maarten Oosten*

PROS Revenue Management, 3100 Main Street, Suite #900 Houston, TX 77002, USA

Jeroen H. G. C. Rutten†

ASML, P.O. Box 324, NL-5500 AH Veldhoven, The Netherlands

Frits C. R. Spieksma‡

Department of Operations Research and Business Statistics, Katholieke Universiteit Leuven, Naamsestraat 69, B-3000, Leuven, Belgium

In this paper, we consider the problem of disconnecting a graph by removing as few vertices as possible, such that no component of the disconnected graph has more than a given number of vertices. We give applications of this problem, present a formulation for it, and describe some polyhedral results. Furthermore, we establish ties with other polytopes and show how these relations can be used to obtain facets of our polytope. Finally, we give some computational results.

Keywords and Phrases: clustering, facets, graph partitioning, polyhedral combinatorics.

1 Introduction

In this paper, we consider the following problem. Given is a graph $G=(V,E)$ and an integer $c \geq 1$. Deleting some vertex $i \in V$ from the graph G results in the graph $(V\backslash\{i\}, E\backslash\rho(\{i\}))$, where $\rho(S) := \{\{i,j\} \in E | i \in S, j \in V\}$. The problem is now to delete as few vertices as possible from G, while ensuring that the remaining graph has no components containing more than c vertices. We refer to this problem as DG (which stands for 'disconnecting graphs'). Another way of looking at this problem is the following: find a subgraph $G'=(V\backslash S, E\backslash\rho(S))$ of G with $|S|$ as small as possible such that no component of G' contains more than c vertices. Notice that for $c=1$ this is equivalent to the familiar independent set problem. So DG is a generalization of the independent set problem, and thereby it is NP-hard, and in fact, hard to approximate as well.

*moosten@prosrm.com
†jeroen.rutten@asml.com
‡frits.spieksma@econ.kuleuven.be

Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

Problem DG arises in a variety of settings. For instance, suppose that one is given a linear (integer) program with a corresponding constraint matrix A and some integer $d \geq 1$. Let us call two rows i,j of A *connected* if and only if there exists an h such that $a_{ih} \neq 0$ and $a_{jh} \neq 0$. One is interested in assigning as many rows of A as possible to so-called *blocks* such that no more than d rows are assigned to each block, and a pair of rows assigned to different blocks is not connected. When a constraint matrix can be brought into such a *bordered block diagonal form*, this may help the solution process of the associated linear (or integer) program. This interesting application of DG is described in Borndörfer, Ferreira and Martin (1997, 1998) and Gupta (1997) (see their papers and the references therein for a more elaborate description). It is not difficult to see how one can model an instance of this problem as an instance of DG. Indeed, the graph G is constructed by creating a vertex for each row of A, and by connecting two vertices if and only if the corresponding rows of A are connected; further, let $c := d$. (Notice that different matrices may give rise to the same graph G.) Of course, the problem of decomposing constraint matrices need not be restricted to the area of integer linear programming.

Another application of this problem can be found in the field of *group technology* (see for instance Hyer and Wemmerlöv, 1989 and the references therein). Given is a set of machines $M_1, M_2, \ldots, M_q$, and a set of parts $P_1, P_2, \ldots, P_p$. Furthermore, a $q \times p$ $\{0,1\}$-matrix A is given with $a_{ij} = 1$ if and only if part P_j $(j = 1, \ldots, p)$ has to be processed on machine M_i $(i = 1, \ldots, q)$. Finally, an integer $d \geq 1$ is given. The problem is now to maximize the number of parts produced by the machines such that so-called *production cells* are identified consisting of parts and machines in such a way that each cell contains no more than d parts, each part and each machine occur in at most one cell, and no part has to be processed by machines in different cells. Problems of this type are described in Crama and Oosten (1996) and Hadley (1996). To model an instance of this problem as an instance of DG, we proceed as follows. Construct a graph $G = (V, E)$ by creating a vertex for each column of A, and by connecting two vertices $i,j \in V$ if and only if there exists an $h \in \{1, \ldots, q\}$ with $a_{hi} = a_{hj} = 1$, and set $c := d$.

In a more general context, DG can be seen as a way of identifying (a small set of) vertices that are in some sense *critical*, that is, whose deletion from the graph disconnects the graph into components (with a bounded number of vertices). Problems of this type that are related to DG occur for instance in routing problems in telecommunication networks. In such a network, traffic between pairs of vertices that wish to communicate with each other must be routed through the graph under capacity constraints on the edges. A way of addressing this problem is described in Bartolacci and Wu (1998): the network is partitioned into clusters of bounded size (cf. DG) and a routing algorithm based on these clusters is proposed. Balas and de Souza (2005) investigate a problem where, in a node-weighted graph, a minimum-weight set of vertices needs to be deleted such that two, disconnected, bounded components remain.

Another problem related to DG is the so-called *graph partitioning* problem. Here, one seeks to partition the vertices of some edge-weighted graph, while optimizing a criterion dependent upon the edge weights (see for instance Chopra and Rao 1993, Ferreira *et al.*, 1996).

The goal of this paper is threefold. First, we present an integer programming formulation for DG. This formulation is in the spirit of a formulation given by Crama and Oosten (1996) for a related problem. Second, we investigate the polyhedral structure of the formulation presented here, and its relationship with other well-studied polytopes, like the clique-partitioning polytope and the boolean quadric polytope. In particular, we show that under certain conditions, facets are inherited from other polytopes. Third, we indicate the potential of the formulation given here by solving instances using a branch-and-cut algorithm. The paper is organized as follows. Section 2 gives a formulation of DG, section 3 presents some polyhedral results, section 4 gives computational results, and section 5 presents the conclusions.

2 A formulation for DG

In this section we propose a $\{0,1\}$-programming formulation for DG. Given a graph

$$G=(V,E) \quad \text{with } n=|V| \text{ and } m=\binom{n}{2}$$

and an integer $c \geq 1$ (referred to as the *capacity*), our formulation uses variables $y_i \in \{0,1\}$ and $x_{ij} \in \{0,1\}$ for all $i,j \in V, i \neq j$, where

$$y_i = 1 \quad \text{iff vertex } i \in V \text{ is not deleted from } G$$

and

$$x_{ij} = 1 \quad \text{if vertices } i,j \in V \text{ are assigned to the same component.}$$

Notice that there is a variable x_{ij} present in the formulation even if $\{i,j\} \notin E$. The corresponding $\{0,1\}$-program reads as follows.

$$\begin{aligned}
\text{(DG1) Maximize } & \sum_{i \in V} y_i \\
\text{Subject to } & x_{ij} + x_{ik} - x_{jk} \leq 1 \quad \text{for all distinct } i,j,k \in V \\
& \sum_{j \in V \setminus \{i\}} x_{ij} \leq c-1 \quad \text{for all } i \in V \\
& y_i + y_j - x_{ij} \leq 1 \quad \text{for all } \{i,j\} \in E \\
& x_{ij} \in \{0,1\} \quad \text{for all } i,j \in V, \ i \neq j \\
& y_i \in \{0,1\} \quad \text{for all } i \in V
\end{aligned} \tag{1}$$

The constraints $x_{ij} + x_{ik} - x_{jk} \leq 1$ are called the *triangle* inequalities and they ensure that, for any distinct $i,j,k \in V$, if vertices i and j as well as vertices j and k are

assigned to the same component, then vertices i and k must be assigned to the same component. The constraints

$$\sum_{j \in V \setminus \{i\}} x_{ij} \leq c - 1$$

are called the *capacity* constraints, and they make sure that no more than c vertices can constitute a component. Furthermore, the constraints $y_i + y_j - x_{ij} \leq 1$ are called the *connectivity* constraints, and they imply that if two vertices $i, j \in V$ with $\{i, j\} \in E$ are not deleted from G, then these vertices must be in a same component. Finally, constraints $x_{ij} \in \{0, 1\}$ and $y_i \in \{0, 1\}$ $(i, j \in V, i \neq j)$ are the integrality constraints.

Let us now argue that (DG1) is a valid formulation for DG. The discussion above implies that any feasible solution to DG satisfies these constraints. On the other hand, consider a vector $(x, y) \in \mathbb{R}^{m+n}$ satisfying (1). One can derive a solution to DG as follows: the set of deleted vertices is identified by the y-variables that equal 0. Next, consider the x-variables corresponding to pairs of not deleted vertices. It follows from the connectivity constraints that if such a pair of vertices is connected they are in a same component. Furthermore, since these x-variables satisfy the triangle inequalities and the capacity constraints, they constitute a partition of the associated vertex-set into components of no more than c vertices.

Let us refer to the convex hull of all feasible solutions to (1) as $P(G, c)$. Consider the number of integer variables in (1). We have the following lemma.

LEMMA 1. *Let $(x, y) \in \mathbb{R}^m \times \mathbb{R}^n$ be a feasible solution to the Linear Programming relaxation of (DG1) with $y_i \in \{0, 1\}$ for all $i \in V$. Then there exists a feasible solution $(\bar{x}, \bar{y}) \in \{0, 1\}^{m+n}$ to (DG1) with $\bar{y}_i = y_i$ for all $i \in V$.*

PROOF. Notice that, if $i, j, k \in V$ are such that $y_i = y_j = y_k = 1$ and $\{i, j\}, \{i, k\} \in E$, then $x_{ij} = x_{ik} = 1$ due to the connectivity constraints, and hence $x_{jk} = 1$ due to triangle inequalities. Let $G_1 = (V_1, E_1)$ be the subgraph of G induced by $V_1 = \{i \in V | y_i = 1\}$. The observation above implies that $x_{ij} = 1$ if i and j belong to the same component of G_1. Define $(\bar{x}, \bar{y})$ by $\bar{y}_i = y_i$ for all $i \in V$ and $\bar{x}_{ij} = \lfloor x_{ij} \rfloor$ for all $i, j \in V$. Then $(\bar{x}, \bar{y})$ is a feasible solution to (DG1). □

Thus, by Lemma 1 we can relax the constraints $x_{ij} \in \{0, 1\}$ $(i, j \in V, i \neq j)$ in (DG1) to $0 \leq x_{ij} \leq 1$, so that the number of binary variables in (DG1) can be reduced to n.

Observe that formulation (DG1) can be easily extended to formulate more general problems. In a situation where the objective function is more general than the one considered here (for instance by considering weights for the edges of G, see FERREIRA *et al.*, 1996, for a problem of this nature), formulation (DG1) can be easily adapted to incorporate this. In fact, all polyhedral results derived in section 3 for (DG1) remain valid in such a setting.

3 Polyhedral results

In this section, we deal with the facial structure of $P(G,c)$. For an introduction to the field of polyhedral combinatorics we refer to NEMHAUSER and WOLSEY (1988). Apart from focussing on finding the facet-defining inequalities of $P(G,c)$ (section 3.3), we mainly try to exploit the relationship of $P(G,c)$ with other, well-studied, polytopes in order to be able to use existing results for these polytopes. This subject is dealt with in section 3.2. In section 3.1, we investigate some basic properties of $P(G,c)$.

Let us first make the following easy observation concerning the facial structure of $P(G,c)$. The polytope $P(G,1)$ is (isomorphic to) the independent set polytope, and has dimension n. The polytope $P(G,c)$ with $c \geq 2$ is full dimensional, i.e. it has dimension $\frac{1}{2}|V|(|V|+1)$, because it contains the zero vector and all unit vectors. In the sequel, we assume that $c \geq 2$.

Furthermore, for $U \subseteq V$ and $F \subseteq \{\{i,j\} | i,j \in V, i \neq j\}$, we denote by $\chi(U,F) \in \{0,1\}^{n+m}$ the characteristic vector corresponding to U and F, that is $\chi_q(U,F)=1$ if either $q \in U$ or q is associated with a pair of vertices in F, and $\chi_q(U,F)=0$ otherwise.

3.1 Properties of the facial structure of $P(G,c)$

In this subsection we state conditions showing when a facet-defining inequality for $P(G,c)$ for some G and c is facet-defining for other graphs and/or other capacities. In other words, when do facet-defining inequalities for $P(G,c)$ remain facet-defining for other graphs and/or capacities?

THEOREM 1. *Let $G=(V,E)$ be a graph, $c \geq 2$ an integer, and let $(a,b)^T(x,y) \leq a_0$ be a facet-defining inequality for $P(G,c)$.*

(i) If, for any $d \geq c$, the inequality $(a,b)^T(x,y) \leq a_0$ is valid for $P(G,d)$, then $(a,b)^T(x,y) \leq a_0$ is facet-defining for $P(G,d)$.

(ii) If, for any $E' \subseteq E$, the inequality $(a,b)^T(x,y) \leq a_0$ is valid for $P((V,E'),c)$, then $(a,b)^T(x,y) \leq a_0$ is facet-defining for $P((V,E'),c)$.

PROOF. Since $(a,b)^{\mathrm{T}}(x,y) \leq a_0$ is facet-defining for $P(G,c)$, there exists a set M of $\dim(P(G,c))$ affinely independent solutions satisfying $(a,b)^{\mathrm{T}}(x,y)=a_0$. All solutions in M are also contained in $P(G,d)$ and $P(G',c)$, and these two polytopes have the same dimension as $P(G,c)$ (as $c \geq 2$). Hence, the validity of $(a,b)^{\mathrm{T}}(x,y) \leq a_0$ is sufficient. □

Theorem 2 gives sufficient conditions to lift a facet-defining inequality for $P(G',c)$ to a facet-defining inequality for $P(G,d)$ for $d \geq c$, where G' is a vertex-induced subgraph of G.

THEOREM 2. *Let $G=(V,E)$ be a graph, let $c\geq 2$ be an integer, let $G'=(U,E(U))$ be a subgraph of G, and let*

$$\sum_{\substack{i,j\in U\\ i\neq j}} a_{ij}x_{ij}+\sum_{i\in U} b_i y_i \leq a_0 \tag{2}$$

be a facet-defining inequality for $P(G',c)$ such that the following conditions hold.

(i) Inequality (2) is valid for $P(G,d)$ for some integer $d\geq c$.

(ii) There exists an order $(e_1,\ldots,e_p)$ of the elements in $\tilde{E}:=\{(i,j)|i\in U, j\in V\setminus U\}$ such that for each $e_k\in\tilde{E}$ there exists $(x,y)\in P(G,d)$ satisfying (2) at equality with $x_{e_k}=1$ and $x_{e_l}=0$ for all $l=k+1,\ldots,p$.

(iii) For all $i\in V\setminus U$ there exists $(x,y)\in P(G,d)$ with $y_i=1$ satisfying (2) at equality.

Then (2) is facet-defining for $P(G,d)$.

PROOF. As (2) defines a facet of $P(G',c)$ there exists a set M_1 of $\frac{1}{2}|U|(|U|+1)$ affinely independent solutions satisfying this inequality at equality. Let χ_0 be the incidence vector of one of the solutions in M_1. Obviously, the following incidence vectors belong to $P(G,d)$ and satisfy (2) at equality.

$$\chi_0+\chi(\emptyset,\{i,j\}) \quad \forall i,j\in V\setminus U, \quad i\neq j \tag{3}$$

By condition (ii) there exists, for each $e_k\in\tilde{E}$, a point $\chi^k:=(x,y)\in P(G,d)$ with $(a,b)^{\mathrm{T}}(x,y)=a_0$, $x_{e_k}=1$ and $x_{e_l}=0$ for all $l=k+1,\ldots,p$. The set $M_2:=\{\chi^1,\ldots,\chi^p\}$ consists of p affinely independent points. Furthermore, by condition (iii), for all $i\in V\setminus U$ there exists a vector $(x,y)\in P(G,d)$ with $y_i=1$ satisfying (2) at equality. These vectors, together with the ones described by (3) and the sets M_1 and M_2, yield exactly $\frac{1}{2}|V|(|V|+1)$ affinely independent solutions in $P(G,d)$ satisfying (2) at equality. □

To end this subsection, let us give some easy-to-prove properties of $P(G,c)$.

THEOREM 3. *Let $G=(V,E)$ be a graph, let $c\geq 2$ be an integer and let $(a,b)^{\mathrm{T}}(x,y)\leq a_0$ be any facet-defining inequality of $P(G,c)$.*

(i) The inequalities $x_{ij}\geq 0$ $(i,j\in V, i\neq j)$ define (trivial) facets of $P(G,c)$.

(ii) The inequalities $x_{ij}\leq 1$ $(i,j\in V, i\neq j)$ do not define facets of $P(G,c)$.

(iii) The inequalities $y_i\geq 0$ $(i\in V)$ define (trivial) facets of $P(G,c)$.

(iv) The inequalities $y_i\leq 1$ $(i\in V)$ define facets of $P(G,c)$.

(v) $a_0\geq 0$.

(vi) For nontrivial facets: $b_i\geq 0$ for all $i\in V$.

PROOF.

(i) The zero vector and all unit vectors but one satisfy $x_{ij}=0$ $(i,j\in V, i\neq j)$ and are affinely independent.

(ii) The inequality $x_{ij} \leq 1$ is the sum of the triangle inequalities $x_{ij} + x_{ik} - x_{jk} \leq 1$ and $x_{ij} - x_{ik} + x_{jk} \leq 1$, and hence it is not facet-defining.

(iii) The zero vector and all unit vectors but one satisfy $y_i = 0$ $(i \in V)$ and are affinely independent.

(iv) The following vectors satisfy $y_i = 1$ and are linearly independent.

$$\begin{aligned} &\chi(\{i\}, \emptyset) \\ &\chi(\{i\}, \{j,k\}) \quad \text{for all } j,k \in V, j \neq k \\ &\chi(\{i,j\}, \{i,j\}) \quad \text{for all } j \in V, j \neq i. \end{aligned}$$

(v) + (vi) This follows from the fact that the y-variables form an independent system (see page 237 in NEMHAUSER and WOLSEY, (1988)). □

3.2 Ties with other polytopes

In this subsection we study ties between $P(G,c)$ and related polytopes. We prove that under certain conditions facets of the *clique partitioning* polytope, facets of the *capacitated clique partitioning* polytope, facets of the (extended) *maximal weighted clique* polytope, facets of the *boolean quadric* polytope and facets of a polytope denoted by $P(G,B,c)$ each give rise to facets of $P(G,c)$.

Let us first describe these polytopes. Given a complete graph $K_n = (V_n, E_n)$ on n vertices with edge weights $w_{ij} \in \mathbb{R}$ for all $\{i,j\} \in E_n$, the clique partitioning problem (CPP) is described by

$$\begin{aligned} &\text{Maximize} \sum_{\{i,j\} \in E_n} w_{ij} x_{ij} \\ \text{(CPP)} \quad &\text{subject to } x_{ij} + x_{ik} - x_{jk} \leq 1 \quad \text{for all distinct } i,j,k \in V_n \\ &x_{ij} \in \{0,1\} \quad \text{for all } \{i,j\} \in E_n \end{aligned} \tag{4}$$

The facial structure of this polytope, called here P_n^{cpp}, is extensively studied in GRÖTSCHEL and WAKABAYASHI (1990), OOSTEN, RUTTEN and SPIEKSMA (2001), SØRENSEN (1995), and RUTTEN (1998).

In case one imposes a bound on the number of vertices in a clique (say c), we arrive at the capacitated clique partitioning problem: add to (CPP) the inequalities

$$\text{(CapCPP)} \quad \sum_{j \in V \setminus \{i\}} x_{ij} \leq c - 1 \quad \text{for all } i \in V \tag{5}$$

The resulting polytope, called $P_{n,c}^{\text{capcpp}}$ is studied in FAIGLE, SCHRADER and SULETZKI (1987) (this polytope is called the c-restricted partition polytope in DIJKHUIZEN and FAIGLE (1993). Given a graph $G = (V,E)$ on n vertices, one could also add to (CapCPP) the variables $y_i \in \{0,1\}$ for all $i \in V$ and the appropriate connectivity constraints. This gives us the polytope $P(G,c)$.

When one is interested in finding a single maximal weighted clique, one needs to add the inequalities:

$$\text{(Clique)} \quad x_{ij} + x_{jk} + x_{kl} - x_{ik} - x_{jl} \leq 1 \quad \text{for all distinct } i,j,k,l \in V \tag{6}$$

The resulting polytope, called $P_{n,c}^{\text{clique}}$ is studied by Dijkhuizen and Faigle (1993). Park, Lee and Park (1996) also study the problem of finding a single maximal weighted clique; however, they introduce an extended formulation which looks as follows:

$$\begin{aligned} &\text{Maximize} \sum_{\{i,j\}\in E_n} w_{ij}x_{ij} \\ \text{(EXCLIQ)}\quad &\text{subject to } y_i + y_j - x_{ij} \le 1 \quad \text{for all } \{i,j\}\in E_n \\ &\sum_{i\in V_n} y_i \le c \\ &x_{ij} - y_i \le 0, x_{ij} - y_j \le 0 \quad \text{for all } \{i,j\}\in E_n \\ &x_{ij}\in\{0,1\} \quad \text{for all } \{i,j\}\in E_n \\ &y_i \in \{0,1\} \quad \text{for all } i\in V_n \end{aligned} \tag{7}$$

This polytope, called $P_{n,c}^{\text{excliq}}$ here, is also investigated by Macambira and de Souza (2000) and Sørensen (2004). Observe that, when omitting constraints

$$\sum_{i\in V_n} y_i \le c$$

from the above formulation, the Boolean quadric polytope (P_n^{quad}) arises which has been studied by Padberg (1989).

Finally, consider now a formulation proposed by Borndörfer *et al.* (1997) in the context of decomposing constraint matrices, and by Kumar, Kusiak, and Vanelli (1986) in a cell-formation context. This formulation has as an additional input parameter an integer $B\in\mathbb{N}$, which is an upper bound on the number of components to be formed. The formulation uses variables $z_{ib}\in\{0,1\}$ for all $i\in V$ and $b\in\{1,\ldots,B\}$, where

$$z_{ib} = \begin{cases} 1 & \text{if vertex } i\in V \text{ is assigned to component } b \\ 0 & \text{otherwise} \end{cases}$$

The corresponding $\{0,1\}$-program reads as follows.

$$\begin{aligned} \text{(MD)}\quad &\text{Maximize} \sum_{i\in V}\sum_{b=1}^{B} z_{ib} \\ &\text{Subject to} \sum_{b=1}^{B} z_{ib} \le 1 \quad \text{for all } i\in V \\ &\sum_{i\in V} z_{ib} \le c \quad \text{for all } b\in\{1,\ldots,B\} \\ &z_{ib} + z_{jb'} \le 1 \quad \text{for all } \{i,j\}\in E, b,b'\in\{1,\ldots,B\},\ b\neq b' \\ &z_{ib}\in\{0,1\} \quad \text{for all } i\in V, b\in\{1,\ldots,B\} \end{aligned} \tag{8}$$

We denote the convex hull of all feasible solutions to (8) by $P(G,B,c)$. Given a vertex $z \in P(G,B,c)$ it is straightforward to construct a vertex $(x,y) \in \mathbb{R}^{m+n}$ of $P(G,c)$. Set

$$y_i = \sum_{b=1}^{B} z_{ib}$$

and $x_{ij} = 1$ if there exists $b \in \{1,\ldots,B\}$ such that $z_{ib} = z_{jb} = 1$, and $x_{ij} = 0$ otherwise. Notice that different vertices of $P(G,B,c)$ which correspond to permutations of the components all give rise to the same vertex of $P(G,c)$. This observation implies that valid inequalities $a^{\mathrm{T}} z \leq a_0$ for $P(G,B,c)$ with $a_{ib} = a_{ib'}$ for each $i \in V$ and all $b, b' \in \{1,\ldots,B\}$ can be turned into valid inequalities for $P(G,c)$ by substituting

$$y_i = \sum_{b=1}^{B} z_{ib}.$$

Let us refer to inequalities for $P(G,B,c)$ with $a_{ib} = a_{ib'}$ for all $i \in V$ and all $b, b' \in \{1,\ldots,B\}$ as *block- invariant* inequalities (see Borndörfer *et al.*, 1998). Thus a block-invariant inequality can be written as

$$\sum_{i \in V} a_i \sum_{b=1}^{B} z_{ib} \leq a_0. \tag{9}$$

Let us now summarize the relationships between the various polytopes introduced.

Lemma 2 (Inheritance of Valid Inequalities).

(i) An inequality valid for P_n^{cpp} is valid for $P_{n,c}^{capcpp}$ for all $n, c \in \mathbb{N}$.
(ii) An inequality valid for $P_{n,c}^{\mathrm{capcpp}}$ is valid for $P_{n,c}^{\mathrm{clique}}$ for all $n, c \in \mathbb{N}$ as well as valid for $P(G,c)$.
(iii) An inequality valid for $P_{n,c}^{\mathrm{clique}}$ is valid for $P(G,c)$ if G is a clique, for all $c \in \mathbb{N}$.
(iv) A block-invariant inequality valid for $P(G,B,c)$ is valid for $P(G,c)$ when substituting

$$y_i = \sum_{b=1}^{B} z_{ib} \quad \text{for } i = 1,\ldots,n.$$

(v) An inequality valid for $P(G,c)$ is valid for $P_{n,c}^{\mathrm{excliq}}$ for all $n, c \in \mathbb{N}$.

Proof. Straightforward. □

A graphical representation of Lemma 2 is given in Figure 1.

Of course, a more powerful result would concern the facetness of inequalities instead of their validity. Given a facet-defining inequality $a^{\mathrm{T}} x \leq a_0$ of P_n^{cpp} (P_n^{quad}) there

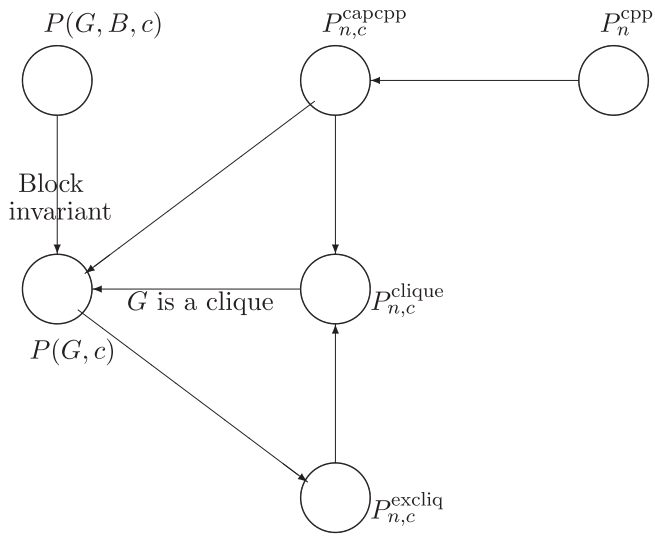

Fig. 1. Inheritance of valid inequalities.

exists a set M_a of $\dim(P_n^{\text{cpp}})$ ($\dim(P_n^{\text{quad}})$) affinely independent solutions satisfying this inequality at equality. Let us denote by $k_{M_a}^{\text{cpp}}$ ($k_{M_a}^{\text{quad}}$) the size of the largest clique (in terms of number of vertices) present in a solution in M_a, and let

$$k_a^{\text{cpp}} := \min\{k_{M_a} : M_a \text{ affinely independent}, \; |M_a| = \dim(P_n^{\text{cpp}}), a^{\mathrm{T}}x = a_0 \forall x \in M_a\}$$

$$(k_a^{\text{quad}} := \min\{k_{M_a}^{\text{quad}} : M_a \text{ affinely independent}, \; |M_a| = \dim(P_n^{\text{quad}}), a^{\mathrm{T}}x = a_0 \forall x \in M_a\}).$$

Lemma 3 (Inheritance of Facets).

(i) *An inequality $a^{\mathrm{T}}x \le a_0$ facet-defining for P_n^{cpp} is also facet-defining for $P_{n,c}^{\text{capcpp}}$ for all $c \ge k_a^{\text{cpp}}$.*

(ii) *An inequality $a^{\mathrm{T}}x \le a_0$ facet-defining for $P_{n,c}^{\text{capcpp}}$ is facet-defining for $P(G, c)$ for all graphs $G = (V, E)$ with $|V| = n$.*

(iii) *An inequality $a^{\mathrm{T}}x \le a_0$ facet-defining for $P_{n,c}^{\text{excliq}}$ which is valid for $P(G, c)$ is facet-defining for $P(G, c)$ for all graphs $G = (V, E)$ with $|V| = n$.*

(iv) *An inequality $a^{\mathrm{T}}x \le a_0$ facet-defining for P_n^{quad} is also facet-defining for $P(G, c)$ for all $c \ge k_a^{\text{quad}}$ and for all graphs $G = (V, E)$ with $|V| = n$.*

(v) *A block-invariant inequality*

$$\sum_{i \in V} b_i \sum_{b=1}^{B} z_{ib} \le b_0 \tag{10}$$

facet-defining for $P(G,B,c)$ *is facet-defining for* $P(G,c)$ *when substituting*

$$y_i = \sum_{b=1}^{B} z_{ib} \quad \textit{for } i = 1,\ldots,n.$$

PROOF.

(i) Let M_a be a maximal set of affinely independent solutions such that $k^{\text{cpp}}_{M_a} = k^{\text{cpp}}_a$. Then $M_a \subseteq P^{\text{capcpp}}_{n,c}$ for $c \geq k_a$, and from $\dim(P^{\text{capcpp}}_{n,c}) = \dim(P^{\text{cpp}}_n)$ it follows that $a^{\text{T}}x \leq a_0$ defines a facet of $P^{\text{capcpp}}_{n,c}$.

(ii) Let M be a maximal set of affinely independent solutions in $P^{\text{capcpp}}_{n,c}$, and let χ_0 be the incidence vector of one of the solutions in M. The set M together with the set $\{\chi_0 + \chi(\{i\},\emptyset) | i \in V\}$ is still affinely independent and has cardinality $\dim(P(G,c))$.

(iii) Notice that any feasible solution of $P^{\text{excliq}}_{n,c}$ is also feasible in $P(G,c)$; thus, since $|V| = n$ and $\dim(P^{\text{excliq}}_{n,c}) = \dim(P(G,c))$ the result follows.

(iv) Let M_a be a maximal set of affinely independent solutions such that $k^{\text{quad}}_{M_a} = k^{\text{quad}}_a$. Then $M_a \subseteq P(G,c)$ for $c \geq k_a$, and from $\dim(P^{\text{quad}}_n) = \dim(P(G,c))$ it follows that $a^{\text{T}}x \leq a_0$ defines a facet of $P(G,c)$.

(v) Consider inequalities (10) and substitute y_i for $\sum_b z_{ib}$ for all i. Let

$$F := \{(x,y) \in P(G,c) | (0,b)^{\text{T}}(x,y) = b_0\}$$

and suppose that F is not a facet of $P(G,c)$. Then there exist $(p,q) \in \mathbb{R}^{m+n}$ and $p_0 \in \mathbb{R}$ such

$$\sum_{\substack{i,j \in V \\ i \neq j}} p_{ij}x_{ij} + \sum_{i \in V} q_i y_i = p_0 \tag{11}$$

for all $(x,y) \in F$. Consider now, for all $(x,y) \in F$, all pairs of distinct vertices $i,j \in V$. Two situations can occur.

(1) There exists $(x,y) \in F$ with $y_i + y_j \leq 1$. Then $p_{ij} = 0$, because the solution with $x_{ij} = 0$ as well as the solution with $x_{ij} = 1$ are in F.

(2) All $(x,y) \in F$ have $y_i = y_j = 1$. This implies that the face induced by (10) is contained in the hyperplane

$$\sum_{b=1}^{B} (z_{ib} + z_{jb}) = 2,$$

a contradiction, since (10) defines a facet of $P(G,B,c)$. Hence $p_{ij} = 0$ for all $i,j \in V, i \neq j$. Thus (11) boils down to

$$\sum_{i \in V} q_i y_i = p_0$$

which is satisfied by all $(x,y) \in F$, and hence

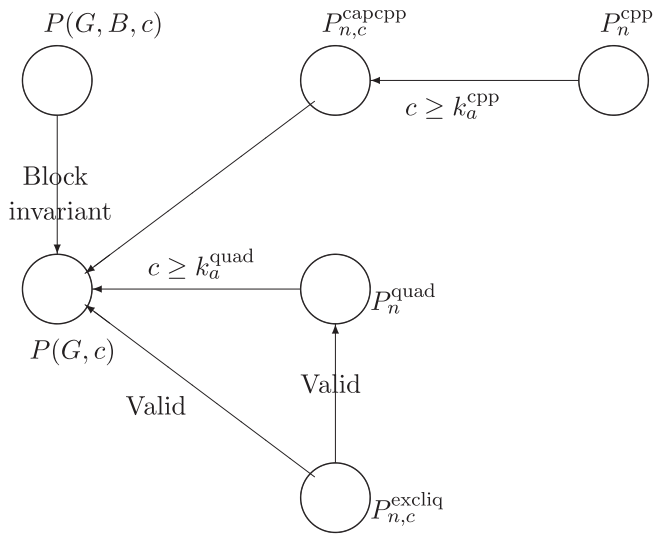

Fig. 2. Inheritance of facet-defining inequalities.

$$\sum_{i \in V} q_i \sum_{b=1}^{B} z_{ib} = p_0$$

holds for all $z \in P(G,B,c)$ satisfying (10) at equality. Thus, $(p,q,p_0) = \beta \cdot (a,b,a_0)$ for some $\beta > 0$, and we conclude that (v) holds. □

A graphical representation of Lemma 3 is given in Figure 2.

Lemma 3 gives us quite some information concerning the facetness of inequalities for $P(G,c)$. Using this lemma and the results described in Faigle (1987), Grötschel and Wakabayashi (1990), Oosten (1996), Borndörfer *et al.* (1998), Park *et al.* (1996), and Bandelt *et al.* (1999), one can conclude for a number of classes of inequalities that they are facet-defining for $P(G,c)$ for some $c \geq 2$. Here, we mention the following examples listed as corollaries.

Corollary 1. *Consider the circuit inequalities defined by* Faigle *et al. (1987). Let* $C = \{v_1, \ldots, v_{c+1}\} \subseteq V$. *The corresponding circuit inequality is given by*

$$x_{v_1,v_2} + x_{v_2,v_3} + \cdots + x_{v_{c+1},v_1} \leq c - 1$$

These inequalities define facets of $P(G,c)$.

Proof. This follows from Lemma 3 and the fact proven in Faigle *et al.* (1987) that these inequalities define facets of $P_{n,c}^{\text{capcpp}}$. □

COROLLARY 2. *Another example is the class of the two-partition inequalities introduced by* GRÖTSCHEL *and* WAKABAYASHI *(1990) (see also* BANDELT *et al., 1999). Let* $S, T \subseteq V$ *be two disjoint subsets of vertices. The corresponding two-partition inequality is given by*

$$\sum_{i \in S} \sum_{j \in T} x_{ij} - \sum_{\substack{i,j \in S \\ i \neq j}} x_{ij} - \sum_{\substack{i,j \in T \\ i \neq j}} x_{ij} \leq \min\{|S|, |T|\} \tag{12}$$

These inequalities define facets of $P(G, c)$ *for all* $c \geq 4$ *if and only if* $|S| \neq |T|$.

PROOF. This follows from Lemma 3, the proof in GRÖTSCHEL and WAKABAYASHI (1990) that these inequalities define facets of P_n^{cpp} and the observation that $k_a \leq 4$ in this proof. □

Notice that Corollary 2 implies that the triangle inequalities [which occur in the LP-relaxation of (DG1)] are facet-defining for $P(G, c)$.

COROLLARY 3. *Let us now consider the clique inequalities introduced by* PADBERG *(1989) (see also* PARK *et al., 1996;* MACAMBIRA *and* DE SOUZA*, 1997). Let* $G = (V, E)$ *be a graph, let* c, p *be two integers with* $c \geq p + 1$*, and let* $U \subseteq V$ *be such that* $(U, E(U))$ *is a clique in* G*. Then the inequality*

$$p \cdot \sum_{i \in U} y_i - \sum_{\{i,j\} \in E(U)} x_{ij} \leq \binom{p+1}{2} \tag{13}$$

defines a facet if and only if $|U| \geq p + 2$.

PROOF. As PARK *et al.* (1996) show that these inequalities are facet-defining for $P_{n,c}^{\text{excliq}}$, it follows from Lemma 3 that the validity of these inequalities for $P(G, c)$ ensures that they are facet-defining. Validity follows by observing that for a fixed p the left-hand side of (13) is maximal if either p or $p + 1$ vertices from U are assigned to a component, and in that case the left-hand side and the right-hand side of (13) are equal. □

Observe that for $p = 1$ and $|E(U)| = 1$ in (13) the connectivity constraints arise. Notice that other facet-defining inequalities of $P_{n,c}^{\text{excliq}}$ may violate the validity condition that is required to ensure facetness for $P(G, c)$. This happens for instance with the so-called *cut* inequalities described in PADBERG (1989) that are facet-defining for $P_{n,c}^{\text{excliq}}$ (see PARK *et al.*, 1996; MACAMBIRA and DE SOUZA, 1997).

BORNDÖRFER *et al.* (1998) describe several classes of valid and facet-defining block-invariant inequalities for $P(G, B, c)$, each of which is, by Lemmas 2 and 3, valid and facet-defining for $P(G, c)$. Here we list only those classes which are used in our branch-and-cut algorithm (see section 4).

Let $W \subseteq V$ with $|W| = c + k$ be such that the induced subgraph $(W, E(W))$ of G is k-vertex-connected. Then the inequality

$$\sum_{i \in W} \sum_{b=1}^{B} z_{ib} \leq c \tag{14}$$

is valid for $P(G, B, c)$. It is facet-defining for $P(G, B, c)$ if and only if for each $i \notin W$ there exists some vertex cut N in $(W \cup \{i\}, E(W \cup \{i\}))$ of size k with $i \notin N$ (see Borndörfer *et al.*, 1998). Hence, we have Corollary 4.

Corollary 4. *Let $G = (V, E)$ be a graph, $c \geq 2$ an integer, and let $W \subseteq V$ with $|W| = c + k$ be such that $(W, E(W))$ is k-vertex-connected. Then the generalized cover inequality*

$$\sum_{i \in W} y_i \leq c \tag{15}$$

is valid for $P(G, c)$. It defines a facet if for all $i \notin W$ there exists some vertex-cut N in $(W \cup \{i\}, E(W \cup \{i\}))$ of size k with $i \notin N$.

Proof. This follows from Lemma 3 and the proof in Borndörfer *et al.* (1998) that these inequalities define facets of $P(G, B, c)$. □

In a similar fashion, we can deduce the following two corollaries.

Corollary 5. *Let $G = (V, E)$ be a graph, $c \geq 2$ an integer, and let $W \subseteq V$ be such that $(W, E(W))$ is a clique of G. Then the capclique inequality*

$$\sum_{i \in W} y_i \leq c \tag{16}$$

is valid for $P(G, c)$. It defines a facet if $|W| \geq c + 1$ and for each vertex $i \notin W$ there exists $U \subseteq W$ with $|U| = c$, such that i is not adjacent to any vertex in U.

Proof. This follows from Lemma 3 and the proof in Borndörfer *et al.* (1998) that these inequalities define facets of $P(G, B, c)$. □

Notice that the condition in Corollary 5 implies maximality of the clique W.

Corollary 6. *Let $G = (V, E)$ be a graph, $c \geq 2$ an integer, and let $i \in V$ be such that $|N(i)| \geq c$. Then the star inequality*

$$(|N(i)| - c + 1) \cdot y_i + \sum_{j \in N(i)} y_j \leq |N(i)| \tag{17}$$

is valid for $P(G, c)$.

Proof. Trivial. □

3.3 More facets of $P(G,c)$

In this section, we show that all facet-defining inequalities with right-hand side 1 are contained in the previous section and we describe some more classes of facet-defining inequalities.

THEOREM 4. *Let $G=(V,E)$ be a graph and let $c\geq 4$ be an integer. All facet-defining inequalities of $P(G,c)$ with integral coefficients and right-hand side 1 are given by the inequalities $y_i\leq 1$ for $i\in V$, the two-partition inequalities [see (12)] with $|S|=1$ and inequalities (13) with $p=1$.*

PROOF. We prove the theorem by showing that any inequality $(a,b)^{\mathrm{T}}(x,y)\leq 1$ valid for $P(G,c)$ is implied by one of the inequalities mentioned above. Obviously, no b_i ($i\in V$) can have value 2 or more. Let $D=\{i\in V|b_i=1\}$. For each pair $i,j\in D$, we have $\{i,j\}\in E$ (otherwise there exists a solution that violates the inequality). So D corresponds to the vertices of a clique in G. We distinguish three cases.

CASE 1. $|D|\geq 2$. It is evident that all coefficients a_{ij} with $i\notin D$ or $j\notin D$ (or both) can have value 0 at most. Consider now any edge $\{i,j\}\in E$ with $i,j\in D$. Its coefficient can be at most -1, because otherwise we take $y_i=y_j=x_{ij}=1$ and all other variables 0, which would violate the inequality. Thus in this case, a valid inequality is implied by (13) with $p=1$.

CASE 2. $|D|=1$. Consider any pair $i,j\in V$. The value of its corresponding coefficient a_{ij} can be at most 0. Hence, a valid inequality of this form is implied by the upper bound constraints $y_i\leq 1$.

CASE 3. $D=\emptyset$. Obviously, no a_{ij} with $i,j\in V$ can have a value 2 or more. Moreover, any two edges with coefficient 1 must have a vertex in common, say vertex s. Let $T=\{i\in V|a_{si}=1\}$. Now, the coefficient of a_{ij} ($i,j\in T$) can be at most -1 (otherwise consider $x_{si}=x_{sj}=x_{ij}=1$ and all other variables 0). It follows that a valid inequality of this form is implied by the two-partition inequalities with $|S|=1$. □

Let us finally provide three classes of facet-defining inequalities related to specific structures in the graph G.

THEOREM 5. *Let $G=(V,E)$ be a graph, let $c\geq 3$ be an integer, and let $i,j,k\in V$ be such that $\{i,j\},\{i,k\}\in E$ and $\{j,k\}\notin E$. Then the inequality*

$$y_i+y_j+y_k-x_{jk}\leq 2 \tag{18}$$

is valid and facet-defining for $P(G,c)$.

PROOF. Obviously, the inequality is valid. The following six solutions are linearly independent and satisfy (18) at equality.

$$\begin{array}{ll} \chi(\{i,j\},\{i,j\}) & \chi(\{j,k\},\{i,j\}) \\ \chi(\{i,k\},\{i,k\}) & \chi(\{j,k\},\{i,k\}) \\ \chi(\{j,k\},\emptyset) & \chi(\{i,j,k\},E(\{\{i,j,k\}\})) \end{array}$$

Consider all pairs of distinct vertices $l,m \in V\setminus\{i,j,k\}$. The following incidence vectors satisfy (18) at equality, and each incidence vector has a nonzero entry at a position where all the previous incidence vectors had a zero entry.

$$\begin{array}{l} \chi(\{j,k\},\{i,l\}\}) \\ \chi(\{i,k\},\{\{i,k\},\{j,l\}\}) \\ \chi(\{k,l\},\{\{i,j\},\{k,l\}\}) \\ \chi(\{i,j,k\},E(\{j,k,l\})) \\ \chi(\{j,k\},\{\{l,m\}\}) \end{array}$$

Together this yields $\dim(P(G,c))$ linearly independent solutions satisfying (18) at equality. □

We will refer to these inequalities as $K_{1,2}$-inequalities.

THEOREM 6. *Let $G=(V,E)$ be a graph, let $c\geq 3$ an integer, and let $U:=\{i,j,k,l\}\subseteq V$ be such that $(U,E(U))$ is a clique in G. Then the arrow inequality (see Figure 3)*

$$y(\{i,j,l\})+x_{ik}-x(E(\{j,k,l\}))\leq 2 \tag{19}$$

is valid and facet-defining for $P(G,c)$.

PROOF. It is easy to check that (19) is facet-defining for $P(K_4,3)$ and valid for $P(G,c)$. In order to prove that the arrow constraints define facets of $P(G,c)$ whenever K_4 is a subgraph of G we use Theorem 2. For each $m\in V\setminus U$ set

$$\begin{array}{l} \chi^1_m := \chi(\{i,l\},\{\{i,l\},\{j,m\}\}) \\ \chi^2_m := \chi(\{i,l\},\{\{i,l\},\{k,m\}\}) \\ \chi^3_m := \chi(\{i,j\},\{\{i,j\},\{l,m\}\}) \\ \chi^4_m := \chi(\{i,j\},E(\{i,j,m\})) \end{array}$$

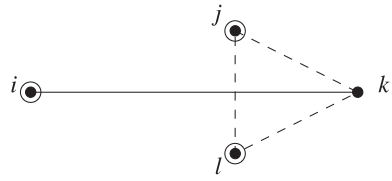

Fig. 3. The support of an arrow inequality.

and define $M_2 := \{\chi^1_m, \ldots, \chi^4_m | m \in V \setminus U\}$. Furthermore, define

$$M_3 := \{\chi(\{i,j,m\}, E(\{i,j,m\})) | m \in V \setminus U\}$$

The set M_2 shows that condition (ii) of Theorem 2 is fulfilled, and M_3 shows that condition (iii) holds. Hence, the arrow constraints define facets of $P(G,c)$. □

THEOREM 7. *Let $G=(V,E)$ be a graph, let $c \geq 3$ be an integer, and let $C := \{v_1, v_2, v_3, v_4\}$ be a subset of V that induces a cycle in G. Then the four-cycle inequality*

$$y(C) - x_{v_1, v_3} - x_{v_2, v_4} \leq 2 \tag{20}$$

is valid and facet-defining for $P(G,c)$.

PROOF. Straightforward when using Theorem 2. □

Notice that although the facets of Theorems 5, 6, and 7 have support graphs with at most four nodes, straightforward lifting of these facets (e.g. using Theorem 2) can produce inequalities with larger support graphs.

4 Computational results

4.1 Branch-and-cut algorithm

We implemented a branch-and-cut algorithm for DG based on (DG1) that uses the results described in Section 3. The structure of the branch-and-cut algorithm is depicted in Figure 4. The linear programs were solved with GLPK 4.8, the GNU Linear Programming Kit (see GLPK). The linear programming solver was embedded in a branch-and-cut algorithm, which was written in C. All computations are performed on a AMD Athlon 2600+ processor running Linux.

Our algorithm does not differ much from standard branch-and-cut algorithms, except that at several points we could benefit from specific knowledge of the problem instances we wanted to solve. The objective for all instances, consisting of a graph $G=(V,E)$ and an integer $c \geq 1$, is to maximize the number of vertices of G that can be assigned to some component such that no component has more than c vertices. If $i,j \in V, i \neq j$ are such that $N(i) \subseteq N(j)$ and there exists an optimal vertex assignment in which vertex j is assigned to some component, then there also exists an optimal vertex assignment in which vertex i is assigned to some component. The same holds if $\{i,j\} \in E$ and $N(i) \setminus \{j\} \subseteq N(j) \setminus \{i\}$. This implies that, for $i,j \in V, i \neq j$ with $N(i) \setminus \{j\} \subseteq N(j) \setminus \{i\}$, we can add the inequality $y_i \geq y_j$ to our linear program, although it is not valid for $P(G,c)$, but at least one optimal integral solution is not cut off by this inequality. If $N(i) \setminus \{j\} = N(j) \setminus \{i\}$ we add only one of the inequalities $y_i \leq y_j$ and $y_i \geq y_j$, because otherwise all optimal solutions could be cut off, namely

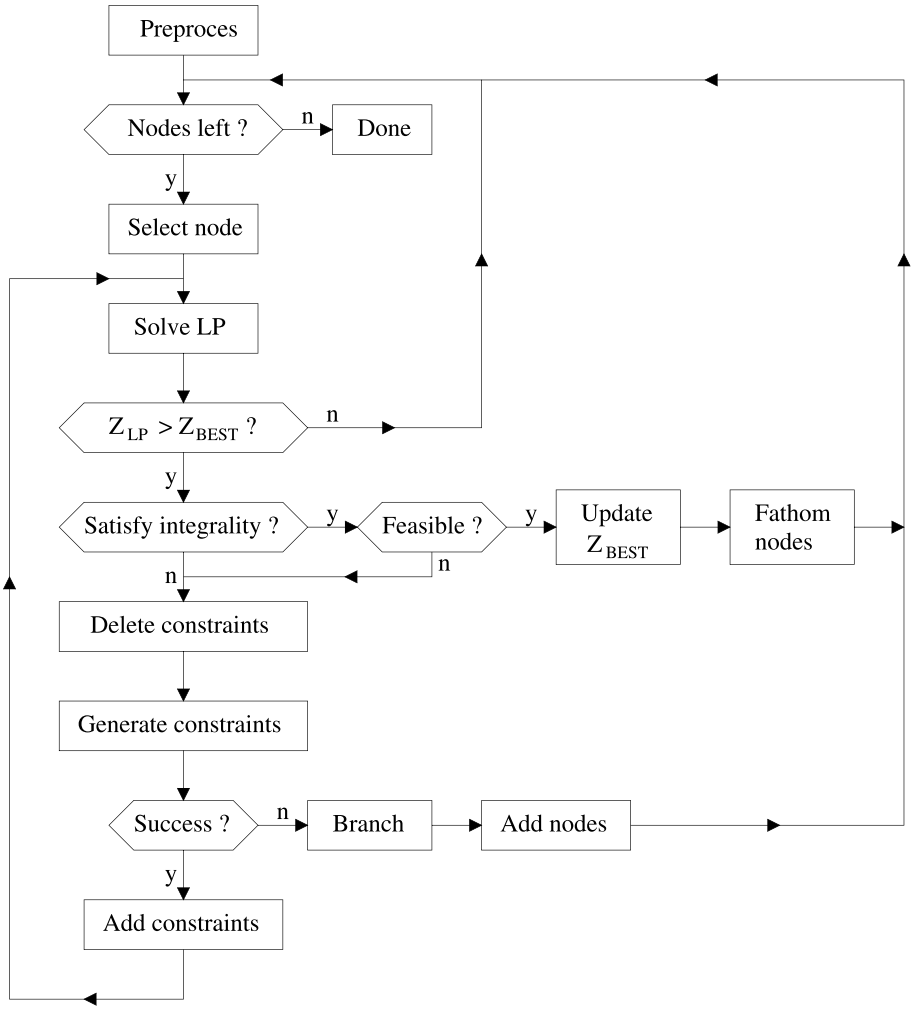

Fig. 4. Structure of the branch-and-cut algorithm.

if it is optimal to assign exactly one of the vertices i and j. Constraints of this type are added to the linear program in the preprocessing phase of our algorithm, and they are never deleted later on. Our initial linear program consists of the capacity constraints and the connectivity constraints. The 'cutting' part of our branch-and-cut algorithm uses the following classes of (facet-defining) inequalities.

1. The $K_{1,2}$-inequalities: for all distinct $i,j,k \in V$ with $\{i,j\},\{j,k\} \in E$ the inequality $y_i + y_j + y_k - x_{ik} \leq 2$ must be satisfied. Searching violated $K_{1,2}$-inequalities is done by enumeration. The number of $K_{1,2}$-inequalities is $O(n^3)$ and enumeration can be done fast.
2. The cover inequalities (15) with $|W| = c + k$. We use the following heuristic to separate cover inequalities with $k = 1$ and $k = 2$ (compare Borndörfer *et al.*,

1998). Searching $W \subseteq V$ such that $(W, E(W))$ is two-vertex connected is done as follows.

For each edge $\{i,j\} \in E$
Begin
  Set $W := \{i,j\}$ and $N := N(i) \cap N(j)$
  While $(N \neq \emptyset)$ and $(|W| < c+2)$ do
  Begin
    Choose $l \in N$
    $W := W \cup \{l\}$
    $N := \{v \in V \setminus W : |E(\{v\}, W)| \geq 2\}$
  End
  If $|W| = c+2 \rightarrow$ Check cover inequality
End

In the worst case, this takes $O(n^3c^2)$ time.

3. The star inequalities (17). The number of star inequalities is linear in n, and therefore separation for this class is simply done by enumeration.
4. The four-cycle inequalities (20). Searching violated inequalities in this class is done by enumeration, which takes $O(n^4)$ in the worst case.
5. The clique inequalities (13) and the capclique inequalities (16). Finding violated inequalities in these classes requires the search for a (maximal) clique in G (see TSUKIYAMA *et al.*, 1977). Maximal cliques in G can be constructed via a recursive algorithm as follows. We assume that there is an order $(v_1, \ldots, v_n)$ of the vertices in V. Call the subroutine 'Clique' with $U = \emptyset$ and $N = V$, where Clique has the following structure.

Clique(U, N)
Begin
  $k := \min\{l | (v_l \in N) \wedge (l > \max\{m | v_m \in U\})\}$
  While $(k \leq n)$ do
  Begin
    $U := U \cup \{v_k\}$
    $N' := \{v \in N | \{v, v_k\} \in E\}$
    if $(N' = \emptyset) \rightarrow U$ is maximal clique
    else Clique(U, N')
    $U := U \setminus \{v_k\}$
    $k := \min\{l | (v_l \in N) \wedge (l > k)\}$
  End
End

Notice that Clique is an exact algorithm for finding all maximal cliques in a graph and hence has exponential running time in the worst case. For the instances we considered it is very fast. Whenever a (maximal) clique-inducing subset $U \subseteq V$ has been constructed, the corresponding inequalities (13) and (16) are checked. Violated inequalities are added to the current linear program.

6. The arrow constraints (19). Searching violated arrow inequalities is done by enumeration, which takes $O(n^4)$ in the worst case.
7. The triangle inequalities $x_{ij} + x_{ik} - x_{jk} \leq 1$ for all distinct $i, j, k \in V$. Separation is done by enumeration.

Each class of cutting planes is searched only if the number of violated inequalities found in all the previous classes is less than a given number. The class of triangle inequalities is searched only if the current solutions satisfy all integrality conditions and no other violated inequalities are found.

The branching part of our algorithm uses a depth-first search. We branch on the variable y_i ($i \in V$) for which $\min\{y_i, 1 - y_i\}$ is maximal. If $y_i < 0.5$ we first examine the branch corresponding to $y_i = 0$, if $y_i \geq 0.5$ we start with the case where $y_i = 1$.

Recall that we described in section 1 the problem of bringing a constraint matrix into bordered block diagonal form as an instance of problem DG. In fact, BORNDÖRFER *et al.* (1998) took a subset of the constraint matrices in the MIPLIB and NETLIB problem libraries as their test set using formulation (MD) (available at: ftp://ftp.zib.de/pub/Packages/mp-testdata/madlib/index.html). We use the same test set here, and to keep the number of variables in the LPs within reasonable bounds, we restricted ourselves to those instances with at most 100 vertices. For the MIPLIB instances we used $c = \lceil 1.05 \cdot |V|/2 \rceil$ as the value for the capacity, and for the NETLIB instances we used $c = \lceil \frac{1}{4}|V| \rceil$ (see BORNDÖRFER *et al.*, 1998). Notice that BORNDÖRFER *et al.* (1998) solve formulation (MD) with $B = 2$ for all instances arising from the MIPLIB problem library, and with $B = 4$ for all instances from the NETLIB problem library. That is, they require a partition of the assigned vertices into at most two and four components, respectively. In our case, we allow for an arbitrary number of components, thus we solve a different problem. Obviously, it follows that the value of an optimal solution we find may differ from the one found by BORNDÖRFER *et al.* (1998); in fact, the optimal value found there is a lower bound for our problem.

Each instance was allotted at most 1 hour of computing time. The performance of our branch-and-cut algorithm on these instances is given in Tables 1 and 2. Both tables have the same structure. The first column lists the name of each problem, and the next three columns give the size of the corresponding graph (number of vertices, and number of edges) and the capacity. The next three columns list the computing time (in minutes and seconds), the number of nodes in the branch-and-bound tree, and the number of iterations needed (an iteration consists of one round of adding violated inequalities and reoptimizing the linear program). The last three columns give the used lower bound (as described before, we use the value found in BORNDÖRFER *et al.* (1998) as a lower bound for our problem instances), the best solution found (which is a lower bound for the optimum in case the problem is not solved within 1 hour), and the best known upper bound (which is of course equal to the best solution if the problem is solved to optimality within the allowed computing time).

The results in Tables 1 and 2 show that a branch-and-cut approach based upon formulation (DG1) is a viable approach to solving instances of DG. From the 28

Table 1. Computational results of the MIPLIB instances.

Name	$\|V\|$	$\|E\|$	c	Time	Nodes	Iter.	Lb.	Obj.	Ub.
mod008	6	15	4	00:00	1	1	4	4	4
stein9	13	66	7	00:00	5	11	7	7	7
p0040	13	30	7	00:00	1	3	10	10	10
p0033	15	40	1	00:00	1	2	12	12	12
gt1	15	61	8	00:01	7	14	10	10	10
flugpl	16	28	9	00:00	5	9	14	15	15
bm23	20	190	11	00:00	1	2	11	11	11
enigma	21	118	12	00:08	15	44	12	12	12
air01	23	137	13	00:00	5	8	20	21	21
rgn	24	75	13	00:02	91	224	19	19	19
pipex	25	153	14	00:12	7	28	16	16	16
lseu	28	136	15	00:06	9	41	21	21	21
gt2	28	173	15	00:24	7	35	17	17	17
sentoy	30	435	16	00:00	1	3	16	16	16
stein15	36	350	19	60:00	3193	14517	19	19	24
misc02	43	454	23	03:52	36	149	28	29	29
sample2	45	97	24	06:20	469	963	41	41	41
air02	50	1126	27	04:55	21	91	28	29	29
misc01	54	929	29	60:00	57	460	30	31	40
mod013	62	144	33	00:11	17	35	56	56	56
mod014	74	127	39	00:02	3	8	72	72	72
lp4l	85	1644	45	60:00	1	147	50	50	67
bell5	87	226	46	05:15	19	128	83	83	83
p0291	92	521	49	14:18	11	116	85	85	85
misc03	96	2894	51	60:00	1	85	52	52	76
l152lav	97	1866	51	60:00	1	96	61	61	78
khb05250	100	1323	53	60:00	2	95	75	75	83
harp2	100	1225	53	45:51	5	94	83	83	83

Table 2. Computational results of the NETLIB instances.

Name	$\|V\|$	$\|E\|$	c	Time	Nodes	Iter.	Lb.	Obj.	Ub.
seba	2	0	1	00:00	1	1	2	2	2
afiro	20	20	5	00:00	1	2	18	18	18
fit1d	24	228	6	00:07	15	49	8	8	8
fit2d	25	279	7	01:41	25	79	7	7	7
sc50b	28	110	7	00:02	31	91	17	17	17
sc50a	29	95	8	00:01	13	29	21	21	21
kb2	39	330	10	00:14	41	81	25	25	25
vtpbase	51	354	13	02:06	41	166	37	37	37
bore3d	52	615	13	08:55	59	221	29	29	29
adlittle	53	239	14	00:16	11	39	43	43	43
blend	54	548	14	12:50	55	249	34	34	34
recipe	55	129	14	00:00	1	3	54	55	55
scagr7	58	661	15	05:18	41	110	37	37	37
sc105	59	356	15	09:30	205	619	43	43	43
stocfor1	62	272	16	01:02	19	59	52	52	52
scsd1	77	202	20	60:00	49	223	69	69	70
beaconfd	90	1199	23	60:00	2	136	64	64	68
share2b	93	619	24	60:00	121	445	84	84	85

MIPLIB instances we can solve 22 within 1 hour of computing; for the NETLIB instances we can solve 15 of 18. The number of nodes in the branching tree is very modest for all but one of the problem instances, as is the number of iterations.

Another interesting question is whether allowing an arbitrary number of components improves the solution compared to having an upper bound of 2 (Table 1) or 4

Table 3. Generated cutting planes for the MIPLIB instances.

Name	K_{12}	Cover	Star	Cycle	Clique	Arrow	Triangle	Total
mod008	0	0	0	0	0	0	0	0
stein9	129	1	13	463	4	0	0	610
p0040	165	0	3	0	0	0	0	168
p0033	89	0	0	0	0	0	0	89
gt1	359	0	6	336	0	0	0	701
flugpl	36	0	0	0	0	0	325	351
bm23	0	0	20	0	9	0	0	29
enigma	2052	1	1	4149	0	0	0	6203
air01	120	101	0	99	0	0	0	320
rgn	984	0	0	160	3	101	0	1248
pipex	2124	23	15	3265	0	0	0	5427
lseu	1178	10	6	1630	0	12	0	2836
gt2	1747	0	14	4135	0	0	0	5896
sentoy	0	0	30	0	0	0	0	30
stein15	392,387	0	1	683,066	802	7829	0	1,084,085
misc02	6752	0	3	13,909	437	442	418	21,961
sample2	2670	0	0	681	0	0	13,200	16,551
air02	1794	0	71	13,381	0	4355	177	19,778
misc01	20,818	0	42	77,771	688	1503	1101	101,923
mod013	1643	0	0	641	0	0	0	2284
mod014	576	0	0	9	0	0	800	1385
lp4l	29,646	0	2	103,600	209	0	0	133,455
bell5	1068	407	0	146	3	5	7600	9229
p0291	11,736	31	0	8487	0	190	0	20,444
misc03	33,239	0	32	0	0	0	0	33,271
l152lav	28,632	0	2	8000	0	0	0	36,634
khb05250	17,841	0	0	12,400	0	0	0	30,241
harp2	31,646	0	16	2656	0	0	0	34,318

Table 4. Generated cutting planes for the NETLIB instances.

Name	K_{12}	Cover	Star	Cycle	Clique	Arrow	Triangle	Total
seba	0	0	0	0	0	0	0	0
afiro	27	0	0	0	0	0	0	27
fit1d	409	0	43	3072	242	606	0	4372
fit2d	840	0	66	6879	5500	1048	0	14,333
sc50b	1065	36	12	289	40	20	0	1462
sc50a	536	3	1	334	5	0	0	879
kb2	2222	3	30	1970	36	140	0	4401
vtpbase	6124	80	35	8835	37	96	0	15,207
bore3d	9623	0	49	19,954	0	1601	0	31,227
adlittle	2012	0	2	1453	0	104	0	3571
blend	13,922	1	18	24,571	16	916	0	39,444
recipe	14	0	0	0	0	0	14	28
scagr7	2474	0	39	7886	0	2932	0	13,331
sc105	18,943	15	1	22,186	7	659	400	42,211
stocfor1	2548	9	0	1634	0	198	0	4389
scsd1	3418	209	0	698	9	5	16,400	20,739
beaconfd	8532	0	26	41,380	0	779	0	50,717
share2b	7591	1558	8	7999	10	225	12,400	29,791

(Table 2). Indeed, comparing columns 'Lb.' and 'Obj.' in these tables indicates when allowing more than 2 (4) components is beneficial. We observe that in at least six of the 46 cases allowing an arbitrary number of blocks yields a better solution (flugpl, air01, misc02, air02, misc01, and recipe).

Furthermore, we have kept track of the numbers of the different types of cutting planes that we generated for each instance (see Tables 3 and 4). It is interesting to see that all classes of inequalities are used, and that, apart from the $K_{1,2}$ inequalities, the cycle inequalities are the ones most often found. This might suggest that lifting these inequalities is a promising possibility to find more effective cutting planes.

4.2 The influence of varying capacity

Another issue we investigated is the influence of the capacity on the number of components in an optimal solution and the hardness of the problem. To do this, we fixed four values of c, namely 3, 5, 10 and 15 and ran, for each value of c, each of the fourteen smallest MIPLIB instances. The results can be found in Table 5. Notice that the value of the lower bound mentioned in these tables is found using the value of an optimal solution to an instance with a smaller value of c. To be precise, with $c_1 = 3, c_2 = 5, c_3 = 10$ and $c_4 = 15$, let $\text{opt}(c_i)$ equal the optimal value of an instance with capacity c_i and let Lb_i denote the lower bound used in an instance with capacity c_i $(i = 1, 2, 3, 4)$. We have: $\text{Lb}_i = \min\{\max\{c_i, \text{opt}(c_{i-1})\}, n\}$ for $i = 1, 2, 3, 4$, with $\text{opt}(c_0) = 0$. From the results in Table 5 it seems difficult to make a general statement regarding a relationship between the capacity and the hardness of the problem. Note that the instances mod008, bm23, and sentoy correspond to complete graphs, which makes it easy to find the optimal solution independent of the capacity. However, when looking at instances enigma, rgn, lseu, and gt2 one might conclude that they tend to get more difficult with increasing capacity.

5 Conclusions and summary

In this paper, we propose formulation (DG1) for the problem of disconnecting graphs by removing vertices (problem DG). We study the facial structure of the polytope associated with (DG1). This is mainly done by establishing ties with other polytopes, notably the (capacitated) clique-partitioning polytope and the boolean quadric polytope. It turns out that a number of facet-defining inequalities for this polytope remain facet-defining for the polytope associated with (DG1). We characterize all facet-defining inequalities with right-hand side 1.

Furthermore, we have done some computational experiments by implementing a branch-and-cut algorithm using the polyhedral results derived. Although it is tempting to compare the results of BORNDÖRFER *et al.* (1997, 1998) for (MD) with the results presented here, one should be careful in doing so, because in BORNDÖRFER *et al.* (1997, 1998) a different problem is solved because of restricting the number of components to be found.

From our computational results we conclude that (DG1) is a useful model, at least for modest problem instances. Using a relatively simple code, we could solve most

Table 5. Computational results for varying capacity.

Name	c	Lb.	Obj.	Nodes	Iter.
mod008	3	3	3	1	2
$\|V\|=6$	5	5	5	1	1
$\|E\|=15$	10	6	6	1	1
	15	6	6	1	1
stein9	3	3	3	19	39
$\|V\|=13$	5	5	5	17	32
$\|E\|=66$	10	10	10	1	3
	15	13	13	1	1
p0040	3	3	10	1	5
$\|V\|=13$	5	10	10	1	3
$\|E\|=30$	10	10	10	3	6
	15	13	13	1	1
p0033	3	3	9	12	33
$\|V\|=15$	5	9	10	8	16
$\|E\|=40$	10	10	13	3	8
	15	15	15	1	1
gt1	3	3	9	2	9
$\|V\|=15$	5	9	9	3	7
$\|E\|=61$	10	10	11	7	12
	15	15	15	1	1
flugpl	3	3	11	8	14
$\|V\|=16$	5	11	14	1	4
$\|E\|=28$	10	14	15	5	10
	15	15	15	1	3
bm23	3	3	3	1	2
$\|V\|=20$	5	5	5	1	2
$\|E\|=190$	10	10	10	1	2
	15	15	15	1	2
enigma	3	3	10	7	29
$\|V\|=21$	5	10	10	9	34
$\|E\|=118$	10	10	11	23	57
	15	15	15	7	20
air01	3	3	6	6	19
$\|V\|=23$	5	6	10	7	19
$\|E\|=137$	10	10	18	6	19
	15	18	22	1	5
rgn	3	3	12	9	20
$\|V\|=24$	5	12	14	180	435
$\|E\|=75$	10	14	19	10	35
	15	19	19	205	495
pipex	3	3	16	1	8
$\|V\|=25$	5	16	16	1	4
$\|E\|=153$	10	16	16	5	14
	15	16	16	17	50
lseu	3	3	18	1	14
$\|V\|=28$	5	18	18	5	18
$\|E\|=136$	10	18	20	35	121
	15	20	21	22	79
gt2	3	3	17	1	9
$\|V\|=28$	5	5	17	1	4
$\|E\|=173$	10	17	17	3	19
	15	17	17	7	35
sentoy	3	3	3	1	2
$\|V\|=30$	5	5	5	1	2
$\|E\|=435$	10	10	10	1	2
	15	15	15	1	2

instances with up to 100 vertices by branch-and-cut without evaluating more than 500 nodes or 1000 iterations (only one instance required more nodes and iterations, and it was not solved within 1 hour). More knowledge of the polyhedral structure of $P(G, c)$ will probably make it possible to solve larger problem instances.

References

Balas, E. and C. C. de Souza (2005), The vertex separator problem: a polyhedral investigation, *Mathematical Programming* **103**, 583–608.

Bandelt, H.-J., M. Oosten, J. H. G. C. Rutten and F. C. R. Spieksma (1999), Lifting theorems and facet characterization for a class of clique partitioning inequalities, *Operations Research Letters* **24**, 235–243.

Bartolacci, M. R. and S. D. Wu (1998), A virtual clustering approach for routing problems in telecommunication networks, *INFORMS Journal on Computing* **10**, 12–24.

Borndörfer, R., C. E. Ferreira and A. Martin (1997), *Matrix decomposition by branch-and-cut*, Preprint SC-97-14, Konrad-Zuse-Zentrum für Informationstechnik, Berlin.

Borndörfer, R., C. E. Ferreira and A. Martin (1998), Decomposing matrices into blocks, *SIAM Journal on Optimization* **9**, 236–269.

Chopra, S. and M. R. Rao (1993), The partition problem, *Mathematical Programming* **59**, 87–115.

Crama, Y. and M. Oosten (1996), Models for machine-part grouping in cellular manufacturing, *International Journal of Production Research* **34**, 1693–1713.

Dijkhuizen, G. and U. Faigle (1993), A cutting-plane approach to the edge-weighted maximal clique problem, *European Journal of Operational Research* **69**, 121–130.

Faigle, U., R. Schrader and R. Suletzki (1987), A cutting-plane algorithm for optimal graph partitioning, *Methods of Operations Research* **57**, 109–116.

Ferreira, C. E., A. Martin, C. C. de Souza, R. Weismantel and L. A. Wolsey (1996), Formulations and valid inequalities for the node capacitated graph partitioning problem, *Mathematical Programming* **74**, 247–266.

GNU Linear Programming Kit (GLPK), Open source package for solving mixed integer programs, http://www.gnu.org/software/glpk/glpk.html.

Grötschel, M. and Y. Wakabayashi (1990), Facets of the clique partitioning polytope, *Mathematical Programming* **47**, 367–387.

Gupta, A. (1997), Fast and effective algorithms for graph partitioning and sparse-matrix ordering, *IBM Journal of Research and Development* **41**, 171–184.

Hadley, S. W. (1996), Finding part-machine families using graph partitioning techniques, *International Journal of Production Research* **34**, 1821–1839.

Hyer, N. L. and U. Wemmerlöv (1989), Group technology in the US manufacturing industry: a survey of current practices, *International Journal of Production Research* **29**, 287–304.

Kumar, K. R., A. Kusiak and A. Vanelli (1986), Grouping parts and components in flexible manufacturing systems, *European Journal of Operational Research* **24**, 387–397.

Macambira, E. M. and C. C. de Souza (2000), The edge-weighted clique problem: valid inequalities, facets and polyhedral computations, *European Journal of Operational Research* **123**, 346–371.

Nemhauser, G. L. and L. A. Wolsey (1988), *Integer and combinatorial optimization*, Wiley, New York.

Oosten, M., J. H. G. C. Rutten and F. C. R. Spieksma (2001), The clique partitioning problem: facets and patching facets, *Networks* **38**, 209–226.

Oosten, M. (1996), *A polyhedral approach to grouping problems*, PhD thesis, Maastricht University, The Netherlands.

Padberg, M. (1989), The Boolean quadric polytope: some characteristics, facets and relatives, *Mathematical Programming* **45**, 139–172.

PARK, K., K. LEE and S. PARK (1996), An extended formulation approach for the edge-weighted maximal clique problem, *European Journal of Operational Research* **95**, 671–682.
RUTTEN, J. H. G. C. (1998), *Polyhedral clustering*, PhD thesis, Maastricht University, The Netherlands.
SØRENSEN, M. M. (1995), *A polyhedral approach to graph partitioning*, PhD thesis, Aarhus School of Business, Denmark.
SØRENSEN, M. M. (2004), New facets and a branch-and-cut algorithm for the weighted clique problem, *European Journal of Operational Research* **154**, 57–70.
TSUKIYAMA, S., M. IDE, H. ARIYOSHI and I. SHIRAKAWA (1977), A new algorithm for generating all the maximal independent sets, *SIAM Journal on Computing* **6**, 505–517.

Received: September 2005. Received: September 2006.