Math. Program., Ser. A 92: 315–333 (2002)

Digital Object Identifier (DOI) 10.1007/s101070100269

Alper Atamtürk* · Deepak Rajan

On splittable and unsplittable flow capacitated network design arc–set polyhedra

Received: August 8, 2000 / Accepted: October 25, 2001
Published online December 6, 2001 – © Springer-Verlag 2001

Abstract. We study the polyhedra of splittable and unsplittable single arc–set relaxations of multicommodity flow capacitated network design problems. We investigate the optimization problems over these sets and the separation and lifting problems of valid inequalities for them. In particular, we give a linear–time separation algorithm for the residual capacity inequalities [19] and show that the separation problem of c–strong inequalities [7] is $\mathcal{NP}$–hard, but can be solved over the subspace of fractional variables only. We introduce two classes of inequalities for the unsplittable flow problems. We present a summary of computational experiments with a branch-and-cut algorithm for multicommodity flow capacitated network design problems to test the effectiveness of the results presented here empirically.

1. Introduction

Given a network, a set of origin–destination vertex pairs (commodities) and demand data for the commodities, the multicommodity capacitated network design problem is to install integer multiples of some capacity unit on the arcs of the network and route the flow of commodities so that the sum of capacity installation and flow routing costs is minimized while meeting the demands for the commodities. Installing or leasing fiberoptic cables on a telecommunication network, determining the capacities of production lines or warehouses in a production–distribution system, determining the number of engines to power a set of trains on a railroad network can all be viewed as installing capacities on the arcs of a network and routing the flow of commodities on the network.

In many applications flow of a commodity is restricted to run through a single path along the network. This is the case, for instance, in telecommunication networks running asynchronous transfer mode (ATM) protocol, production–distribution with single sourcing, and express package delivery, see [4, 12]. These problems are generally formulated using a binary flow variable x_{ka} for each commodity–arc pair (k, a) that takes on a value of 1 if the commodity uses the arc, 0 otherwise and an integer capacity variable y_a. So for each arc of the network there is a capacity constraint of the form

$$\sum_{k \in K} d_k x_{ka} \leq c_{a0} + c_a y_a, \tag{1}$$

where K denotes the set of commodities, d_k the demand of commodity k, c_{a0} existing capacity of the arc and c_a the unit capacity to install. If the flow of commodities is

A. Atamtürk, D. Rajan: Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA 94720-1777, USA, e-mail: {atamturk, deepak}@ieor.berkeley.edu

* Supported, in part, by NSF grants DMI–9908705 and DMI–0070127.

allowed to be split among several paths, then the 0–1 restriction on the flow variables is dropped.

Strong valid inequalities from simple structured relaxations of optimization problems over more complicated sets can be very useful in solving these problems. See for instance [8–10, 20] for successful results with this approach. In this paper, we study the convex hull of solutions to constraints of the form (1). We investigate optimization problems over these polyhedra and the separation and lifting problems of valid inequalities for them. Formally, the sets we consider are defined as

$$\textit{Splittable flow arc set:} \quad \mathcal{F}_S \equiv \{(x, y) \in \mathcal{D}_S : \sum_{i \in N} a_i x_i \le a_0 + y\}$$

$$\textit{Unsplittable flow arc set:} \quad \mathcal{F}_U \equiv \{(x, y) \in \mathcal{D}_U : \sum_{i \in N} a_i x_i \le a_0 + y\},$$

where $N \equiv \{1, 2, \ldots, n\}$ is an index set and

$$\mathcal{D}_S \equiv \{x \in [0, 1]^n, y \in \mathbb{Z}\} \text{ and } \mathcal{D}_U \equiv \{x \in \{0, 1\}^n, y \in \mathbb{Z}\}.$$

One arrives at F_S or F_U by dividing (1) by c_a. The unsplittable flow arc set $\mathcal{F}_U$ is a relaxation of the feasible region of the more familiar 0–1 knapsack problem obtained by introducing a general integer variable y with no bounds. Although the formulation of $\mathcal{F}_U$ is quite similar to the formulation of the knapsack set, as shown in Sect. 3, its structure is significantly different from the latter. The splittable flow arc set $\mathcal{F}_S$ is the relaxation of $\mathcal{F}_U$ obtained by allowing the binary variables to take on any real value between 0 and 1. Finally, we let $\mathcal{F}_L$ denote the relaxation of $\mathcal{F}_S$ obtained by dropping the integrality restriction on y as well.

We assume that the data is rational. Without loss of generality, we assume that $a_i > 0$ for all $i \in N$, since if $a_i < 0$, x_i can be complemented and if $a_i = 0$, x_i can be dropped. We do not impose a sign restriction on the constant term a_0, as for the related separation and lifting problems that will be discussed in the following sections, this term may take on negative or nonnegative values. Throughout the paper, for a vector $v \in \mathbb{R}^n$, we let $v(S) \equiv \sum_{i \in S} v_i$ for $S \subseteq N$.

Related work & contributions of this paper

Polyhedral structure of the 0–1 knapsack set, which is a restriction of $\mathcal{F}_U$, has been studied extensively; see [3, 16, 25, 23, 27, 24, 15]. Another set related to $\mathcal{F}_U$ is the 0–1 knapsack set with single continuous variable, obtained by replacing y with a nonnegative continuous variable, is studied in Marchand and Wolsey [20].

Magnanti et al. [19] study the facial structure of $conv(\mathcal{F}_S)$ when $a_0 = 0$. They define an exponential class of valid inequalities, called the residual capacity inequalities, and show that the residual capacity inequalities and the constraints of $\mathcal{F}_L$ are sufficient to describe $conv(\mathcal{F}_S)$. However, no exact polynomial–time separation algorithm for these inequalities was known until now. In Sect. 2 we present a linear–time algorithm for separating the residual capacity inequalities.

For $\mathcal{F}_U$ Brockmüller et al. [7] introduce the c–strong inequalities and characterize the necessary and sufficient conditions under which the c–strong inequalities are facet–defining. Recently, van Hoesel et al. [17] study $\mathcal{F}_U$ when $a_0 = 0$ as well. In Sect. 3

we prove that the c–strong inequalities constitute all facet–defining inequalities of $conv(\mathcal{F}_U)$ of the form $\sum_{i \in N} \pi_i x_i \leq \pi_o + y$ with integral coefficients. We show that the separation problem of c–strong inequalities is $\mathcal{NP}$–hard and that it is sufficient to solve this separation problem over the subspace of fractional variables only. Furthermore, we introduce two classes of inequalities, both of which include the c–strong inequalities as a special case.

In Sect. 4 we provide a summary of computational studies with a branch–and–cut algorithm for multicommodity flow capacitated network design problems to test the effectiveness of the results presented here empirically.

2. Splittable flow arc set

2.1. Optimization problem

In order to motivate the separation problem of the splittable flow arc set $\mathcal{F}_S$, we start with the related optimization problem. Magnanti et al. [19] state that the optimization of a linear function over F_S can be solved efficiently using an incremental strategy. Here we give a simple algorithm, which is used in Sect. 4 for approximate lifting of valid inequalities for $\mathcal{F}_U$. We consider a maximization problem and without loss of generality assume that the objective coefficient of the capacity variable y is negative and by scaling -1, since otherwise the problem is unbounded. So consider the problem

$$\text{(SFP)} \qquad \zeta = \max\{\sum_{i \in N} c_i x_i - y : \sum_{i \in N} a_i x_i \leq a_0 + y,\ (x, y) \in \mathcal{D}_S\}.$$

We may assume that $c_i > 0$ for all $i \in N$, since otherwise given any optimal solution with $x_i > 0$, there exists an optimal solution that is identical except that $x_i = 0$. Suppose the variables are indexed so that $\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq \cdots \geq \frac{c_n}{a_n}$, ties broken arbitrarily. Let $A_i = \sum_{h=1}^{i} a_h - a_0$ for $i \in N$, $A_0 = -a_0$, and k be the largest index with $\frac{c_k}{a_k} \geq 1$. If $\frac{c_1}{a_1} < 1$, let $k = 0$. Clearly, there exists an optimal solution to the linear programming relaxation of SFP with all positive x_i in the above order and $y = A_k$. Then, by concavity of $\zeta(y)$, there exists an optimal solution (x^*, y^*) to SFP such that $y^* = \lfloor A_k \rfloor$ or $y^* = \lceil A_k \rceil$. Hence the computational burden of finding an optimal solution to SFP is sorting the variables in nonincreasing order of $\frac{c_i}{a_i}$, which can be done in $O(n \log n)$.

Proposition 1. *The optimization problem SFP can be solved in* $O(n \log n)$.

In the light of polynomial equivalence of optimization and separation for a polyhedron [13], the separation problem of $conv(\mathcal{F}_S)$ must also be solvable in polynomial time. In the next section, we show that $conv(\mathcal{F}_S)$ can be separated in linear time.

2.2. Separation problem

For $S \subseteq N$ let $\eta = \lceil a(S) - a_0 \rceil$ and $r = a(S) - a_0 - \lfloor a(S) - a_0 \rfloor$. Magnanti et al. [19] show that for any $S \subseteq N$ the *residual capacity inequality*

$$\sum_{i \in S} a_i (1 - x_i) \geq r(\eta - y) \tag{2}$$

is valid for $\mathcal{F}_S$ when $a_0 = 0$. Inequality (2) is valid for $\mathcal{F}_S$ when $r = 0$ or $y \geq \eta$ since $x_i \leq 1$ for all $i \in N$. To see that it is also valid otherwise, observe that $\sum_{i \in S} a_i(1-x_i) \geq a(S)-a_0-y \geq \eta-(1-r)-y = (1-r)(\eta-1)+r\eta-y \geq (1-r)y+r\eta-y = r(\eta-y)$. The residual capacity inequality can also be viewed as a mixed–integer rounding inequality from a suitable relaxation of $\mathcal{F}_S$ [21].

Magnanti et al. [19] prove that the residual capacity inequalities together with the constraints of $\mathcal{F}_L$ are sufficient to describe $conv(\mathcal{F}_S)$ when $a_0 = 0$. This result extends to the case when $a_0 \neq 0$ as well. Since the constraints of $\mathcal{F}_L$ can be checked for violation simply by testing in linear time, the separation problem of $conv(\mathcal{F}_S)$ reduces to, given a point $(\bar{x}, \bar{y}) \in \mathcal{F}_L$, either finding a residual capacity inequality violated by $(\bar{x}, \bar{y})$, or concluding that $(\bar{x}, \bar{y}) \in conv(\mathcal{F}_S)$. Without loss of generality, we may assume that $\bar{y} \notin \mathbb{Z}$, because no $(\bar{x}, \bar{y}) \in \mathcal{F}_L$ with $\bar{y} \in \mathbb{Z}$ violates a residual capacity inequality as residual capacity inequalities are valid for $\mathcal{F}_S$. So we look for $S \subseteq N$ such that $\sum_{i \in S} a_i(1 - \bar{x}_i) < r(\eta - \bar{y})$. We are interested in only S with $r > 0$, since no residual capacity inequality with $r = 0$ is violated by $(\bar{x}, \bar{y}) \in \mathcal{F}_L$ as $\bar{x}_i \leq 1$ for all $i \in N$. Then, the separation problem can be formulated as

$$\text{(SP1)} \qquad \begin{aligned} \varsigma = \min \; & \sum_{i \in N} a_i(1 - \bar{x}_i)z_i - r(\eta - \bar{y}) \\ \text{s.t.:} \; & \sum_{i \in N} a_i z_i = a_0 + (\eta - 1) + r \\ & 0 < r < 1, \quad \eta \in \mathbb{Z}, \quad z_i \in \{0, 1\} \;\; i \in N, \end{aligned}$$

where $z \in \{0, 1\}^n$ is the characteristic vector of S. If $\varsigma < 0$, then the residual capacity inequality corresponding to an optimal (z, η, r) is violated by $(\bar{x}, \bar{y})$, otherwise, no residual capacity inequality is violated by $(\bar{x}, \bar{y})$. SP1 is a nonlinear mixed–integer optimization problem, which is hard to solve in general.

Lemma 1. *A point* $(\bar{x}, \bar{y}) \in \mathcal{F}_L$ *does not violate any residual capacity inequality* (2) *with* $\eta \leq \bar{y}$ *or* $\eta \geq \bar{y} + 1$.

Proof. Since residual capacity inequality (2) is the mixed–integer rounding inequality for the relaxation $\sum_{i \in S} a_i(1-x_i)+y \geq a(S)-a_0$, it is dominated by $\sum_{i \in S} a_i(1-x_i) \geq 0$ and $\sum_{i \in S} a_i(1 - x_i) + y \geq a(S) - a_0$ unless $\eta - 1 < y < \eta$.

□

From Lemma 1 any residual capacity inequality violated by $(\bar{x}, \bar{y})$ has $\eta = \lceil \bar{y} \rceil$. Since $\bar{y} \notin \mathbb{Z}$, we have $\lfloor \bar{y} \rfloor = \eta - 1$. After fixing η to $\lceil \bar{y} \rceil$, the separation problem can be formulated as the following linear mixed 0–1 optimization problem

$$\text{(SP2)} \qquad \begin{aligned} \varsigma = \min \; & \sum_{i \in N} a_i(1 - \bar{x}_i)z_i - r(\lceil \bar{y} \rceil - \bar{y}) \\ \text{s.t.:} \; & \sum_{i \in N} a_i z_i = a_0 + \lfloor \bar{y} \rfloor + r \\ & 0 < r < 1, \quad z_i \in \{0, 1\} \;\; i \in N. \end{aligned}$$

Eliminating the bounded continuous variable r, we rewrite the separation problem as a 0–1 problem with two strict inequalities

$$\text{(SP3)}\qquad \begin{aligned} \varsigma = \min \ & \sum_{i \in N} a_i(1 - \bar{x}_i - \lceil \bar{y} \rceil + \bar{y}) z_i + (\lceil \bar{y} \rceil - \bar{y})(a_0 + \lfloor \bar{y} \rfloor) \\ \text{s.t.: } & a_0 + \lfloor \bar{y} \rfloor < \sum_{i \in N} a_i z_i < a_0 + \lceil \bar{y} \rceil \\ & z_i \in \{0, 1\} \;\; i \in N. \end{aligned}$$

Next we show that in order to find a violated residual capacity inequality, it is sufficient to consider only variables with a negative coefficient in the objective function of SP3. Let $T \equiv \{i \in N : 1 - \bar{x}_i < \lceil \bar{y} \rceil - \bar{y}\}$.

Lemma 2. *If there exists a residual capacity inequality* (2) *violated by a point* $(\bar{x}, \bar{y}) \in \mathcal{F}_L$*, then there exists one given by* $S \subseteq T$.

Proof. Suppose the residual capacity inequality given by $C \subseteq N$ is violated by $(\bar{x}, \bar{y})$. Then, $a(C \cap T) + a(C \setminus T) = a_0 + \lfloor \bar{y} \rfloor + r$. Consider the following two cases.

(1) $a(C \setminus T) < r$. In this case $a_0 + \lfloor \bar{y} \rfloor < a(C \cap T) < a_0 + \lceil \bar{y} \rceil$ and $C \cap T$ has an objective value in SP3 that is no more than that of C. So let $S = C \cap T$.

(2) $a(C \setminus T) \geq r$. In this case $a(C \cap T) \leq a_0 + \lfloor \bar{y} \rfloor$. Therefore, the objective value for C in SP3 is $\sum_{i \in C} a_i(1 - \bar{x}_i - \lceil \bar{y} \rceil + \bar{y}) + (\lceil \bar{y} \rceil - \bar{y})(a_0 + \lfloor \bar{y} \rfloor) \geq \sum_{i \in C} a_i(1 - \bar{x}_i - \lceil \bar{y} \rceil + \bar{y}) + (\lceil \bar{y} \rceil - \bar{y}) a(C \cap T) = \sum_{i \in C \cap T} a_i(1 - \bar{x}_i) + \sum_{i \in C \setminus T} a_i(1 - \bar{x}_i - \lceil \bar{y} \rceil + \bar{y}) \geq 0$. However, this contradicts the assumption that the residual capacity given by C is violated.

□

Lemma 3. *If* $a(T) \leq a_0 + \lfloor \bar{y} \rfloor$ *or* $a(T) \geq a_0 + \lceil \bar{y} \rceil$*, then there exists no residual capacity inequality violated by* $(\bar{x}, \bar{y}) \in \mathcal{F}_L$.

Proof. Suppose $a(T) \leq a_0 + \lfloor \bar{y} \rfloor$. Then, there exists no $S \subseteq T$ that satisfies the constraints of SP3 and therefore, from Lemma 2, there exists no residual capacity inequality that is violated by $(\bar{x}, \bar{y})$. Now suppose $a(T) \geq a_0 + \lceil \bar{y} \rceil$ and, for contradiction, suppose there exists a set S that gives a violated inequality. From Lemma 2, we may assume that $S \subseteq T$. The objective value for S in SP3 is $\sum_{i \in S} a_i(1 - \bar{x}_i - \lceil \bar{y} \rceil + \bar{y}) + (\lceil \bar{y} \rceil - \bar{y})(a_0 + \lfloor \bar{y} \rfloor) \geq \sum_{i \in T} a_i(1 - \bar{x}_i - \lceil \bar{y} \rceil + \bar{y}) + (\lceil \bar{y} \rceil - \bar{y})(a_0 + \lfloor \bar{y} \rfloor) \geq (\lceil \bar{y} \rceil + a_0)(1 - \lceil \bar{y} \rceil + \bar{y}) - (a_0 + \bar{y}) + (\lceil \bar{y} \rceil - \bar{y})(a_0 + \lfloor \bar{y} \rfloor) = 0$. This contradicts the assumption that the inequality is violated by $(\bar{x}, \bar{y})$.

□

Separation Algorithm

Since the residual capacity inequalities, together with the inequalities of $\mathcal{F}_L$ describe $conv(\mathcal{F}_S)$, we have the following simple procedure for separating $(\bar{x}, \bar{y}) \in \mathcal{F}_L \setminus \mathcal{F}_S$ from $conv(\mathcal{F}_S)$: Let $T \equiv \{i \in N : \bar{x}_i > \bar{y} - \lfloor \bar{y} \rfloor\}$. If $a_0 + \lfloor \bar{y} \rfloor < a(T) < a_0 + \lceil \bar{y} \rceil$ and $\sum_{i \in T} a_i(1 - \bar{x}_i - \lceil \bar{y} \rceil + \bar{y}) + (\lceil \bar{y} \rceil - \bar{y})(a_0 + \lfloor \bar{y} \rfloor) < 0$, then the inequality $\sum_{i \in T} a_i(1 - x_i) \geq r(\eta - y)$ is violated by $(\bar{x}, \bar{y})$. Otherwise, there exists no residual capacity inequality violated by $(\bar{x}, \bar{y})$. Clearly, this procedure can be performed in linear time.

Theorem 1. *The separation problem for the residual capacity inequalities* (2) *can be solved in* $O(n)$.

3. Unsplittable flow arc set

3.1. Optimization problem

Even though our ultimate goal is to find strong valid inequalities for the unsplittable flow arc set $\mathcal{F}_U$, it is helpful to study the maximization of a linear function over $\mathcal{F}_U$ first. As for $\mathcal{F}_S$ we may assume that the objective coefficient of the capacity variable y is -1 and state the problem as

$$\text{(UFP)} \qquad \xi = \max\{\sum_{i \in N} c_i x_i - y : \sum_{i \in N} a_i x_i \leq a_0 + y,\ (x, y) \in \mathcal{D}_U\}.$$

UFP is a relaxation of the more familiar 0–1 knapsack problem. Although the formulation of UFP is quite similar to the formulation of the knapsack problem, its structure is significantly different from the latter. Below we present properties of optimal solutions of UFP that will be useful when studying $conv(\mathcal{F}_U)$ in Sect. 3.2.

Proposition 2. *UFP has an optimal solution* (x^*, y^*) *such that*

$$x_i^* = \begin{cases} 1 & \textit{if } c_i \geq \lceil a_i \rceil \\ 0 & \textit{if } c_i \leq \lfloor a_i \rfloor \end{cases} \quad \textit{for } i \in N.$$

Proof. For $S \subseteq N$ let $\xi(S)$ be the maximum value of the objective of UFP when $x_k = 1$ for all $k \in S$, and $x_k = 0$ otherwise, i.e., $\xi(S) = c(S) - \lceil a(S) - a_0 \rceil$. Suppose $i \notin S$. Then $\xi(S) - \xi(S \cup i) = \lceil a(S) + a_i - a_0 \rceil - \lceil a(S) - a_0 \rceil - c_i$. Since for any $a, b \in \mathbb{R}$, we have $\lceil a \rceil + \lfloor b \rfloor \leq \lceil a + b \rceil \leq \lceil a \rceil + \lceil b \rceil$, it follows that

$$\xi(S) - \xi(S \cup i) \begin{cases} \leq 0 & \text{if } c_i \geq \lceil a_i \rceil \\ \geq 0 & \text{if } c_i \leq \lfloor a_i \rfloor. \end{cases}$$

□

Due to Proposition 2, all binary variables except the ones with $\lfloor a_i \rfloor < c_i < \lceil a_i \rceil$ can be eliminated from UFP since optimal values for them can be determined a priori.

Corollary 1. *UFP can be solved in* $O(n)$ *if either* $c_i \in \mathbb{Z}$ *or* $a_i \in \mathbb{Z}$ *for all* $i \in N$.

Theorem 2. *UFP is* $\mathcal{NP}$*–hard for any fixed value of* a_0.

Proof. The proof is by reduction from PARTITION [11]. Given a set N and weights a_i $i \in N$ with $a(N) = 2$, PARTITION is the question whether there exists $S \subset N$ such that $a(S) = 1$. Let a_0 be fixed to $\bar{a}_0$ and let $\bar{f}_0 = \bar{a}_0 - \lfloor \bar{a}_0 \rfloor$. In order to answer PARTITION, we construct the following instance of UFP with $a_0 = \bar{a}_0$

$$\begin{aligned} \varsigma = \max \ & \sum_{i \in N} a_i x_i + 3\alpha x_o - \alpha y \\ \text{s.t.:} \ & \sum_{i \in N} a_i x_i + (\epsilon + \bar{f}_0(1 + \epsilon))x_o \leq (1 + \epsilon)(\bar{a}_0 + y) \\ & x_i \in \{0, 1\}\ i \in N,\ x_o \in \{0, 1\},\ y \in \mathbb{Z}, \end{aligned}$$

where $\alpha > 1$ and $1 > \epsilon > 0$. After dividing the objective by α and the constraint by $1+\epsilon$, since $\lceil \frac{\epsilon}{1+\epsilon} + \bar{f}_0 \rceil < 3$, from the proof of Proposition 2 we see that $x_o = 1$ and hence $y \geq \lceil -\bar{a}_0 \rceil + 1$ in every optimal solution to UFP. Also since $a(N) + \epsilon + \bar{f}_0(1+\epsilon) < (1+\epsilon)(\bar{f}_0 + 2)$, we have $y \leq \lceil -\bar{a}_0 \rceil + 2$ in any optimal solution. The objective value $\varsigma(y)$ as a function of y satisfies the following: $\varsigma(\lceil -\bar{a}_0 \rceil + 1) \leq 1 + 2\alpha + \alpha \lfloor \bar{a}_0 \rfloor$ and $\varsigma(\lceil -\bar{a}_0 \rceil + 2) \leq 2 + \alpha + \alpha \lfloor \bar{a}_0 \rfloor$. Let S^* be the index set of binary variables at value 1 in an optimal solution. Since $\alpha > 1$, $\varsigma = 1 + 2\alpha + \alpha \lfloor \bar{a}_0 \rfloor$ if and only $a(S^*) = 1$. Hence, the PARTITION problem has an affirmative answer if and only if the optimal value of UFP equals $1 + 2\alpha + \alpha \lfloor \bar{a}_0 \rfloor$.

□

Remark 1. Theorem 2 states that UFP remains $\mathcal{NP}$–hard for any fixed value of a_0. In Sect. 3.2 we will see that a_0 may take on negative values in the separation problem of c–strong inequalities.

Now we give a canonical form of UFP. Without loss of generality, we assume that $\lfloor a_i \rfloor < c_i < \lceil a_i \rceil$ for all $i \in N$ for UFP, since all other variables can be eliminated from the problem by Proposition 2. Then we can further simplify the problem into one where the data consists of only the fractional parts of a_i and c_i. Let $f_0 = a_0 - \lfloor a_0 \rfloor$ and for $i \in N$ let $f_i = a_i - \lfloor a_i \rfloor$ and $r_i = c_i - \lfloor c_i \rfloor$ and define

$$(\text{UFP}_f) \qquad \xi_f = \max\{\sum_{i \in N} r_i x_i - y : \sum_{i \in N} f_i x_i \leq f_0 + y,\ (x, y) \in \mathcal{D}_U\}.$$

Proposition 3. *UFP$_f$ is equivalent to UFP in the sense that $S \subseteq N$ maximizes UFP if and only if it maximizes UFP$_f$ and $\xi_f = \xi - \lfloor a_0 \rfloor$.*

Proof. For $S \subseteq N$ let $\xi(S) = c(S) - \lceil a(S) - a_0 \rceil$. Then

$$\begin{aligned} \xi(S) &= \sum_{i \in S} \lfloor c_i \rfloor + r(S) - \sum_{i \in S} \lfloor a_i \rfloor + \lfloor a_0 \rfloor - \lceil f(S) - f_0 \rceil \\ &= r(S) + \lfloor a_0 \rfloor - \lceil f(S) - f_0 \rceil = \xi_f(S) + \lfloor a_0 \rfloor. \end{aligned}$$

□

Optimization algorithm

Next we give a pseudo polynomial algorithm for solving UFP$_f$, which is used to show strongly polynomial–time lifting valid inequalities in Theorem 5. Let λ be a common multiple of the denominators of the rational numbers f_i $i = 0, 1, \ldots, n$. By multiplying f_i with λ, the constraint of UFP$_f$ can be written with integral coefficients only. Observe that since $f_i < 1$, y can take on a value at most $\lceil f(N) - f_0 \rceil \leq n$ in an optimal solution to UFP$_f$. For $\nu \in \{0, 1, \ldots, \lceil f(N) - f_0 \rceil\}$, consider an optimal solution x^* to the 0–1 knapsack problem

$$(\text{KP1}(\nu)) \quad \varsigma(\nu) = \max\{\sum_{i \in N} r_i x_i : \sum_{i \in N} \lambda f_i x_i \leq \lambda(f_0 + \nu),\ x \in \{0, 1\}^n\}.$$

Since $r_i > 0$ and $f_i < 1$ for all $i \in N$, we have $\lambda f_0 + \lambda(\nu - 1) < \sum_{i \in N} \lambda f_i x_i^* \leq \lambda f_0 + \lambda \nu$. Hence, $\xi_f = \max_{\nu \in \{0,1,\ldots,\lceil f(N) - f_0 \rceil\}} \{\varsigma(\nu) - \nu\}$. Notice that all of these related, at most

$n+1$, knapsack problems can be solved in a total of $O(\lambda n^2)$ by dynamic programming, since when solving KP1(n), we already complete the computations required for solving KP1(ν) for all $\nu \in \{0, 1, \dots, n-1\}$.

Alternatively, let μ be a common multiple of the denominators of the fractional numbers r_i $i = 1, 2, \dots, n$. Instead of solving KP1(ν), we may solve the dual knapsack problem

$$\text{(KP2}(t)) \quad \omega(t) = \min\{\sum_{i\in N} f_i x_i : \sum_{i \in N} \mu r_i x_i \geq t,\ x \in \{0,1\}^n\}$$

so that $\varsigma(\nu) = \omega(t)$ if $\omega(t) \leq \lambda(f_0 + \nu) < \omega(t+1)$. Since KP2($t$) is infeasible for $t \geq \mu n > \mu r(N)$, $\varsigma(\nu)$ for all $\nu \in \{0, 1, \dots, n\}$ can be computed in $O(\mu n^2)$.

Theorem 3. *UFP can be solved in* $O(\min\{\lambda, \mu\} n^2)$.

3.2. Valid inequalities

In this section we discuss three classes of valid inequalities for $\mathcal{F}_U$. The first class is the c–strong inequalities introduced by Brockmüller et al. [7]. The next two classes are new and both of them subsume the c–strong inequalities. Before describing specific valid inequalities, we present some general properties of $conv(\mathcal{F}_U)$ that will be useful in the analysis of those inequalities. First of all, it is easy to check that the convex hull of $\mathcal{F}_U$, $conv(\mathcal{F}_U)$, is full–dimensional and inequalities $x_i \geq 0$ and $x_i \leq 1$ for all $i \in N$ are facet–defining for $conv(\mathcal{F}_U)$. We call these inequalities as the *trivial* valid inequalities of $conv(\mathcal{F}_U)$. The next proposition provides bounds on the coefficients of non–trivial facet–defining valid inequalities of $conv(\mathcal{F}_U)$.

Proposition 4. [2,17] *Every non–trivial facet–defining inequality* $\sum_{i\in N} \pi_i x_i \leq \pi_o + y$ *of* $conv(\mathcal{F}_U)$ *has* $\lfloor a_i \rfloor \leq \pi_i \leq \lceil a_i \rceil$ *for all* $i \in N$ *and* $\pi_o \geq \lfloor a_0 \rfloor$. $\pi_o = \max\{\sum_{i\in N} \pi_i x_i - y : (x, y) \in \mathcal{F}_U\}$.

As before let $f_i = a_i - \lfloor a_i \rfloor$ for $i = 0, 1, \dots, n$ and define

$$\mathcal{F}_{Uf} \equiv \{(x, y) \in \mathcal{D}_U : \sum_{i\in N} f_i x_i \leq f_0 + y\}.$$

Proposition 5. *An inequality* $\sum_{i\in N} \pi_i x_i \leq \pi_o + y$ *with* $\lfloor a_i \rfloor \leq \pi_i \leq \lceil a_i \rceil$ *is valid for* $\mathcal{F}_U$ *if and only if* $\sum_{i\in N}(\pi_i - \lfloor \pi_i \rfloor) x_i \leq \pi_o - \lfloor a_0 \rfloor + y$ *is valid for* $\mathcal{F}_{Uf}$.

Proof. Follows from Proposition 3.

□

Remark 2. From Proposition 4 when looking for strong valid inequalities for $\mathcal{F}_U$, we can restrict our attention to inequalities $\sum_{i\in N} \pi_i x_i \leq \pi_o + y$ with $\lfloor a_i \rfloor \leq \pi_i \leq \lceil a_i \rceil$ for all $i \in N$. But then, from Proposition 5, instead of working with $\mathcal{F}_U$, we can work with $\mathcal{F}_{Uf}$ defined using the fractional parts of the data.

Nonrestrictive assumption

Due to Propositions 4 and 5, without loss of generality, we assume that $0 < a_i < 1$ for all $i \in N$ and $0 \le a_0 < 1$; so $\mathcal{F}_U = \mathcal{F}_{Uf}$ for the rest of the paper. Consequently, from Proposition 4, $0 \le \pi_i \le 1$ for all $i \in N$ for all non–trivial facet–defining inequalities of $conv(\mathcal{F}_U)$.

3.2.1. *c–strong inequalities*

For $S \subseteq N$ let $c_S = |S| - \lceil a(S) - a_0 \rceil$. S is said to be *maximal c–strong* if $c_{S\setminus\{i\}} = c_S$ for all $i \in S$ and $c_{S\cup\{i\}} = c_S + 1$ for all $i \in N \setminus S$. Brockmüller et al. [7] show that for any $S \subseteq N$ the *c–strong* inequality

$$\sum_{i\in S} x_i \le c_S + y \tag{3}$$

is valid for $\mathcal{F}_U$ when $a_0 = 0$. A c–strong inequality is facet–defining for $conv(\mathcal{F}_U)$ if and only if S is maximal c–strong.

Theorem 4. *The maximal c–strong inequalities* (3) *constitute all facet–defining inequalities of* $conv(\mathcal{F}_U)$ *of the form* $\sum_{i\in N} \pi_i x_i \le \pi_o + y$ *with integral coefficients.*

Proof. Let $\sum_{i\in N} \pi_i x_i \le \pi_o + y$ be a facet–defining inequality of $conv(\mathcal{F}_U)$ with integral coefficients. From Proposition 4, it follows that $\pi_i \in \{0, 1\}$ for $i \in N$. Let $S = \{i \in N : \pi_i = 1\}$. From Proposition 2, we have that $\pi_0 = c_S$.

□

In the next sections, we show that $conv(\mathcal{F}_U)$ does have facet–defining inequalities $\sum_{i\in N} \pi_i x_i \le \pi_o + y$ with fractional coefficients.

Separation

Given a point $(\bar{x}, \bar{y})$, there exists a c–strong inequality violated by $(\bar{x}, \bar{y})$ if and only if $\exists S \subseteq N$ such that $\sum_{i\in S} \bar{x}_i - c_S > \bar{y}$. Then, a c–strong inequality is violated if and only if $\max_{S\subseteq N}\{\sum_{i\in S} \bar{x}_i - \lfloor a_0 + \sum_{i\in S}(1 - a_i)\rfloor\} = \max\{\sum_{i\in N} \bar{x}_i z_i - w : \sum_{i\in N}(1 - a_i)z_i + a_0 + 1/\lambda \le w, (z, w) \in \mathcal{D}_U\} + 1 > \bar{y}$, where λ is the least common multiple of the denominators of the rational numbers $(1 - a_i)$ and a_0. From Theorem 2 the last maximization problem with the constant term $-a_0 - 1/\lambda$ is $\mathcal{NP}$–hard.

Although the separation problem of c–strong inequalities is $\mathcal{NP}$–hard, from Proposition 2, it has an optimal solution (z^*, w^*) such that $z_i^* = 1$ if $\bar{x}_i = 1$ and $z_i^* = 0$ if $\bar{x}_i = 0$. Therefore, we can fix such variables to their optimal values and solve the separation problem over $i \in N$ such that $0 < \bar{x}_i < 1$, which in practice can be done very efficiently even by enumeration, as most variables take on values either 0 or 1 in the LP relaxations of network design problems.

3.2.2. *k–split c–strong inequalities*

In this section we describe new valid inequalities for $\mathcal{F}_U$ that are motivated by Proposition 2. An inequality $\sum_{i\in N} \pi_i x_i \le \pi_o + y$ is valid for $\mathcal{F}_U$ if and only if

$\max\{\sum_{i\in N}\pi_i x_i - y : (x,y)\in\mathcal{F}_U\} \le \pi_o$. As shown in Sect. 3.1 solving this maximization problem is $\mathcal{NP}$–hard. However, if the maximum of $\sum_{i\in N}\pi_i x_i - y$ over a suitable relaxation of $\mathcal{F}_U$ is no more than π_o, then we can deduce that $\sum_{i\in N}\pi_i x_i \le \pi_o + y$ is valid for this relaxation and hence for $\mathcal{F}_U$. The relaxation of $\mathcal{F}_U$ that we consider for this purpose is obtained by splitting the integer capacity variable. In a *k–split relaxation*, the capacity variable y is allowed to take values that are integer multiples of $1/k$, where k is a positive integer. Let $\mathcal{F}_U^k \equiv \{\sum_{i\in N} a_i x_i \le a_0 + z/k,\ (x,z)\in\mathcal{D}_U\}$. Thus we define an infinite set of relaxations for $\mathcal{F}_U$ with $conv(\mathcal{F}_U^1) = conv(\mathcal{F}_U)$ and $\lim_{k\to\infty} conv(\mathcal{F}_U^k) = \mathcal{F}_L$. The last equation follows from the fact that $x\in\{0,1\}^n$ for all extreme points (x,y) of $\mathcal{F}_L$. For a given instance of $\mathcal{F}_U$, since there exists a finite k for which $kc_i \le \lfloor ka_i\rfloor$ if $c_i \le a_i$ and $kc_i \ge \lceil ka_i\rceil$ if $c_i \ge a_i$ for all $i\in N$, from Proposition 2, there exists a k–split relaxation of $\mathcal{F}_U$, for which the optimization problem is trivial to solve and hence the validity of a given inequality $\sum_{i\in N}\pi_i x_i \le \pi_o + y$ can be checked easily for this relaxation. Alternatively, for each k we can define an inequality that can be easily verified to be valid for the corresponding k–split relaxation.

Proposition 6. *Any inequality $\sum_{i\in N}\pi_i x_i \le \pi_o + ky$ with k positive integer and $\pi_i \le \lfloor ka_i\rfloor$ or $\pi_i \ge \lceil ka_i\rceil$ for all $i\in N$ is valid for $\mathcal{F}_U$ for $\pi_o = \pi(S) - \lceil ka(S) - ka_0\rceil$, where $S = \{i\in N : \pi_i \ge \lceil ka_i\rceil\}$.*

Proof. The proof is an immediate consequence of Proposition 2.

$$\begin{aligned}\pi_o &= \pi(S) - \lceil ka(S) - ka_0\rceil \\ &= \max\{\sum_{i\in N}\pi_i x_i - z : \sum_{i\in N} ka_i x_i \le ka_0 + z,\ (x,z)\in\mathcal{D}_U\} \\ &\ge \max\{\sum_{i\in N}\pi_i x_i - ky : \sum_{i\in N} a_i x_i \le a_0 + y,\ (x,y)\in\mathcal{D}_U\}\ \ (ky = z).\end{aligned}$$

□

Then for any positive integer k and any $S\subseteq N$, we can define strong valid inequalities by letting $\pi_i\ i\in N$ equal either $\lfloor ka_i\rfloor$ or $\lceil ka_i\rceil$. Let $c_S^k = \sum_{i\in S}\lceil ka_i\rceil - \lceil ka(S) - ka_0\rceil$ and define the *k–split c–strong inequality* as

$$\sum_{i\in S}\lceil ka_i\rceil x_i + \sum_{i\in N\setminus S}\lfloor ka_i\rfloor x_i \le c_S^k + ky. \tag{4}$$

Observe that k–split c–strong inequality (4) is a c–strong inequality for $\mathcal{F}_U^k$. Therefore, since $\mathcal{F}_U^k$ is a relaxation of $\mathcal{F}_U$, a necessary condition for inequality (4) to be facet–defining for $\mathcal{F}_U$ is that S is maximal c–strong in the k–split relaxation $\mathcal{F}_U^k$. Recall that $a_i < 1$ without loss of generality. Below we give a sufficient condition for the k–split c–strong inequality to be facet–defining for $\mathcal{F}_U$. As the example in Sect. 3.2.4 illustrates, k–split c–strong inequalities may be facet–defining more generally.

Proposition 7. *The k–split c–strong inequality* (4) *is facet–defining for $conv(\mathcal{F}_U)$ if* (*i*) *S is maximal c–strong in the k–split relaxation,* (*ii*) *$f_S > (k-1)/k$ and $a_0 \ge 0$,* (*iii*) *$a_i > f_S$ for all $i\in S$, $a_i < 1 - f_S$ for all $i\in N\setminus S$, where $f_S = a(S) - a_0 - \lfloor a(S) - a_0\rfloor$.*

Proof. Let (C, z) denote a point $(x, z) \in \mathbb{B}^n \times \mathbb{Z}$ where $x_i = 1$ for all $i \in C$ and $x_i = 0$ otherwise. Consider the following $n + 1$ affinely independent points of $\mathcal{F}_U^k$:

$$(S, \lceil k(a(S) - a_0)\rceil) \text{ if } c_S^k \neq 0, (\emptyset, 0) \text{ if } c_S^k = 0;$$
$$(S \setminus \{i\}, \lceil k(a(S) - a_i - a_0)\rceil) \text{ for } i \in S;$$
$$(S \cup \{i\}, \lceil k(a(S) + a_i - a_0)\rceil) \text{ for } i \in N \setminus S.$$

Since S is maximal c–strong in k–split relaxation, we have $\lceil k(a(S) - a_0)\rceil = \lceil k(a(S) - a_i - a_0)\rceil + \lceil ka_i\rceil$ for all $i \in S$ and $\lceil k(a(S) - a_0)\rceil = \lceil k(a(S) + a_i - a_0)\rceil + \lfloor ka_i \rfloor$ for all $i \in N \setminus S$. Therefore, after replacing z with $y = z/k$, these points satisfy the k–split c–strong inequality (4) at equality. To complete the proof, it is enough to show that z/k is integer for the points above. For any $w \in \mathbb{R}$, let $f_w = w - \lfloor w \rfloor$. If $kf_w > k - 1$, then $\lceil kf_w \rceil = k$ since $f_w < 1$, so $\lceil kw \rceil = k\lfloor w \rfloor + \lceil kf_w \rceil = k\lceil w \rceil$. Thus, by part (*ii*) of the proposition, for the first point $\lceil k(a(S) - a_0)\rceil/k = \lceil a(S) - a_0 \rceil$. Similarly, by part (*iii*), $\lceil k(a(S) - a_i - a_0)\rceil/k = \lfloor a(S) - a_0 \rfloor$ for $i \in S$ and $\lceil k(a(S) + a_i - a_0)\rceil/k = \lceil a(S) - a_0 \rceil$ for $i \in N \setminus S$ follow.

□

3.2.3. *Lifted knapsack cover inequalities*

Let N_0 and N_1 be two disjoint subsets of N and ν be a nonnegative integer. Consider the 0–1 knapsack set $\mathcal{F}_U(\nu, N_0, N_1)$ obtained by projecting the capacity variable y to ν, all binary variables indexed with N_0 to 0 and all binary variables indexed with N_1 to 1, i.e., $\mathcal{F}_U(\nu, N_0, N_1) \equiv \{(x, y) \in \mathcal{F}_U : y = \nu,\ x_i = 0 \text{ for all } i \in N_0 \text{ and } x_i = 1 \text{ for all } i \in N_1\}$. For this projection $C \equiv N \setminus (N_0 \cup N_1)$ is called a *cover* if $r = a(C) + a(N_1) - a_0 - \nu > 0$. C is said to be a *minimal cover* if $a_i \geq r$ for all $i \in C$.

The knapsack cover inequality $\sum_{i \in C} x_i \leq |C| - 1$ is well–known to be facet–defining for $conv(\mathcal{F}_U(\nu, N_0, N_1))$ if and only if C is a minimal cover [22]. By lifting the knapsack cover inequalities of minimal covers with the projected variables, one can obtain facet–defining inequalities of $conv(\mathcal{F}_U)$. One practical way of lifting inequalities is sequential lifting, in which projected variables are introduced to an inequality one at a time in some sequence. Van Hoesel et al. [17] have independently lifted knapsack cover inequalities to get strong valid inequalities for $\mathcal{F}_U$ as well. Here we show that given a minimal cover, a lifted cover inequality can be constructed in $O(n^3)$ if the capacity variable y is lifted first. We further show that inequalities obtained in this manner subsume all c–strong inequalities.

Now we describe the lifting procedure. We introduce the capacity variable y to the cover inequality first. Let $\mathcal{F}_U(N_0, N_1) \equiv \{(x, y) \in \mathcal{F}_U : x_i = 0 \text{ for all } i \in N_0 \text{ and } x_i = 1 \text{ for all } i \in N_1\}$ and C be a cover. Inequality $\sum_{i \in C} x_i + \alpha(\nu - y) \leq |C| - 1$ is valid for $\mathcal{F}_U(N_0, N_1)$ if and only if

$$\alpha \leq \overline{\alpha} \equiv \min \left\{ \frac{|C| - 1 - \sum_{i \in C} x_i}{\nu - y} : y < \nu,\ (x, y) \in \mathcal{F}_U(N_0, N_1) \right\}$$

and

$$\alpha \geq \underline{\alpha} \equiv \max \left\{ \frac{\sum_{i \in C} x_i - |C| + 1}{y - \nu} : y > \nu,\ (x, y) \in \mathcal{F}_U(N_0, N_1) \right\}.$$

If C is a minimal cover and α equals either $\underline{\alpha}$ or $\overline{\alpha}$, then $\sum_{i \in C} x_i + \alpha(\nu - y) \le |C| - 1$ is facet-defining for $conv(\mathcal{F}_U(N_0, N_1))$, which follows from Wolsey [26]. The existence of a valid lifting coefficient α follows from the next proposition.

Proposition 8. [2,17] *For any cover C, $\underline{\alpha} \le 1 \le \overline{\alpha}$ holds.*

Note that $\underline{\alpha} = 1/(\lceil a(C) + a(N_1) - a_0 \rceil - \nu) = 1/\lceil r \rceil$, which is computed in linear time for any cover. For a minimal cover C, since $0 < r \le a_i < 1$, we have $\underline{\alpha} = 1$. The upper bound $\overline{\alpha}$ can be computed efficiently as well: Suppose $a_1 \le a_2 \le \cdots \le a_{|C|}$. Let $A_0 = a(N_1) - a_0$ and $A_i = A_{i-1} + a_i$ for $i = 1, 2, \dots, |C|$. Since the coefficients in the cover inequality are the same, we have

$$\overline{\alpha} = \min_{i \in \{0,1,\dots,|C|-2\}} \left\{ \frac{|C| - 1 - i}{\nu - \lceil A_i \rceil} : \lceil A_i \rceil < \nu \right\}.$$

Therefore $\overline{\alpha}$ is computed by selecting the minimum of at most $|C| - 1$ terms after sorting a_i $i \in C$ in nondecreasing order, which can be done in $O(n \log n)$.

Next we introduce the projected binary variables to inequality $\sum_{i \in C} x_i + \alpha(\nu - y) \le |C| - 1$ one at a time in some arbitrary sequence. As shown in the example in Sect. 3.2.4, different sequences may lead to different lifted inequalities. Let $L_0 \subseteq N_0$ and $L_1 \subseteq N_1$ be the index sets of variables that are already used in lifting and the current lifted inequality be

$$\sum_{i \in C} x_i + \sum_{i \in L_0} \alpha_i x_i + \sum_{i \in L_1} \alpha_i (1 - x_i) + \alpha(\nu - y) \le |C| - 1. \quad (5)$$

Then the lifting coefficient of a variable x_k, $k \in (N_0 \setminus L_0) \cup (N_1 \setminus L_1)$ is computed by solving the lifting problem

$$\begin{aligned} \alpha_k = |C| - 1 - \max \sum_{i \in C} x_i + \sum_{i \in L_0} \alpha_i x_i + \sum_{i \in L_1} \alpha_i (1 - x_i) + \alpha(\nu - y) \\ \text{(BLP)} \qquad \text{s.t.:} \sum_{i \in C \cup L_0 \cup L_1} a_i x_i \le a_0 - a(N_1 \setminus L_1) \mp a_k + y \\ x_i \in \{0, 1\}\ i \in C \cup L_0 \cup L_1, \quad y \in \mathbb{Z}. \end{aligned}$$

In the rhs of the constraint of BLP, we have $-a_k$ if $k \in N_0$ and $+a_k$ if $k \in N_1$.

Proposition 9. *The maximal c–strong inequalities are equivalent to the lifted minimal cover inequalities with $\alpha = \underline{\alpha}$.*

Proof. Let S be maximal c–strong. Then the corresponding c–strong inequality $\sum_{i \in S} x_i \le c_S + y$ is facet–defining for $conv(\mathcal{F}_U)$. S is a minimal cover with $\nu = \lfloor a(S) - a_0 \rfloor$, $N_0 = N \setminus S$ and $N_1 = \emptyset$. Consider the cover inequality lifted with the capacity variable using $\alpha = \underline{\alpha} = 1$, $\sum_{i \in S} x_i \le |S| - 1 - \nu + y$. Since $a_i < 1$ and since S is maximal c–strong, we have $\lfloor a(S) - a_0 \rfloor < a(S) - a_0$, it follows that $|S| - 1 - \nu = c_S$. Since the c–strong inequality is facet–defining, the lifting coefficients of all of the projected binary variables must be 0. Thus, the maximal c–strong inequality is indeed a lifted minimal cover inequality. The other direction follows from Theorem 4 since $\underline{\alpha} = 1$ and hence the coefficients of the lifted cover inequality are integer.

□

Remark 3. The proof of Proposition 9 also shows that projecting binary variables to 1 does not lead to new inequalities when $\underline{\alpha}$ is used as the lifting coefficient for the capacity variable, because by Proposition 9 and Theorem 4 all facet–defining inequalities $\sum_{i \in N} \pi_i x_i \leq \pi_o + y$ of $conv(\mathcal{F}_U)$ with integer coefficients can be obtained by lifting minimal cover inequalities using $\alpha = \underline{\alpha}$ and $N_1 = \emptyset$, and letting $N_1 \neq \emptyset$ does not lead to fractional lifting coefficients. So when $\alpha = \underline{\alpha}$, the lifted minimal cover inequalities take the simple form $\sum_{i \in C} x_i \leq |C| - 1 - \nu + y = c_C + y$.

□

Lemma 4. *If C is a minimal cover and* $\alpha = \overline{\alpha}$*, then the lifting coefficients of inequality* (5) *satisfy* $\alpha_i \leq |C| - 1$ *for all* $i \in N_0$ *and* $-\alpha_i \leq |C| - 1$ *for all* $i \in N_1$.

Proof. From the definition of $\overline{\alpha}$, observe that $\overline{\alpha} \leq |C| - 1$ with $y = \nu - 1$. Since C is a minimal cover, the lifted inequality is facet–defining for $conv(\mathcal{F}_U)$. Then from Proposition 4, since $0 < a_i < 1$, we have $\alpha_i \leq \overline{\alpha}$ for $i \in N_0$ and $-\alpha_i \leq \overline{\alpha}$ for $i \in N_1$.

□

Theorem 5. *For a minimal cover, a lifted cover inequality with* $\alpha = \overline{\alpha}$ *can be constructed in* $O(n^3)$.

Proof. We have already argued that $\overline{\alpha}$ can be computed in $O(n \log n)$. Since the coefficients of the objective function of the lifting problem BLP is bounded, when computing α_i $i \in N_0 \cup N_1$, it is more efficient to solve BLP with the dual knapsack formulation KP2 in Sect. 3.1. After putting BLP in the form of UFP$_f$, since $\overline{\alpha}$ is a common multiple of the coefficients $\alpha_i / \overline{\alpha}$ and $\overline{\alpha} \leq |C| - 1 < n$, from Theorem 3 the lifting problem for a single binary variable can be solved in $O(n^3)$ by dynamic programming. However, similar to the lifting of 0–1 knapsack cover inequalities [27], the lifting coefficients of all projected variables can also be computed in $O(n^3)$ by dynamic programming, since the set of variables in the knapsack problems solved for lifting are nested.

□

Remark 4. For the 0–1 knapsack set Zemel [27] gives an $O(n^2)$ algorithm to compute the lifting coefficients of a minimal cover inequality when $N_1 = \emptyset$. However, no polynomial–time algorithm is known for constructing a lifted cover inequality for the 0–1 knapsack set if some of the variables are projected to 1, i.e., $N_1 \neq \emptyset$. For the 0–1 knapsack set Gu et al. [14] give an example where the lifting coefficients are bounded from below by an exponential function of n. In the case of the unsplittable flow arc set $\mathcal{F}_U$, we are able to bound the coefficients of the lifted cover inequality from above by n in Lemma 4 by lifting the integer capacity variable y first.

□

3.2.4. Example

Let $\mathcal{F}_U = \{(x, y) \in \{0, 1\}^5 \times \mathbb{Z} : \frac{1}{3}x_1 + \frac{1}{3}x_2 + \frac{1}{3}x_3 + \frac{1}{2}x_4 + \frac{2}{3}x_5 \leq y\}$. Below we list the lifted cover inequalities of $\mathcal{F}_U$ that are not c–strong inequalities.

v	(C, N_0, N_1)	Inequalities
1	$(\{2,3,4\}, \{1,5\}, \emptyset)$	$x_2 + x_3 + x_4 + x_5 \leq 2y$
1	$(\{1,4,5\}, \{2,3\}, \emptyset)$	$x_1 + x_2 + x_4 + x_5 \leq 2y$ and $x_1 + x_3 + x_4 + x_5 \leq 2y$
2	$(\{1,2,3,4\}, \emptyset, \{5\})$	$x_1 + x_2 + x_3 + x_4 + 2x_5 \leq 2y + 1$
2	$(\{1,2,3,5\}, \emptyset, \{4\})$	$x_1 + x_2 + x_3 + 2x_4 + x_5 \leq 2y + 1$

These are all of the facet–defining inequalities that can be obtained by the lifting procedure described here for minimal cover inequalities. We remark that the last two inequalities cannot be obtained by lifting any cover inequality (including non–minimal covers) unless $N_1 \neq \emptyset$. Thus, this example illustrates that, unlike the case when $\alpha = \underline{\alpha}$, with $\alpha = \overline{\alpha}$ projecting binary variables to 1 does lead to inequalities that cannot be obtained otherwise by sequential lifting of cover inequalities. Our second remark is that all of the inequalities above except the last one are also 2–split c–strong inequalities. The 3–split c-strong inequality $x_1+x_2+x_3+2x_4+2x_5 \leq 3y$ with $S = \{1, 2, 3, 4, 5\}$, which is facet–defining for $conv(\mathcal{F}_U)$, however, cannot be obtained by the lifting procedure described here for any cover. Hence, this example shows that the k–split c–strong inequalities and the lifted cover inequalities are indeed different classes of inequalities.

4. Computational results

In order to test the effectiveness of the results in the preceding sections on network design arc sets empirically, we developed a branch–and–cut algorithm using CPLEX callable library (version 6.5.1) for solving multicommodity flow capacitated network design problems formulated as

$$\begin{aligned}
\min \quad & \sum_{a\in A, k\in K} g_{ka} x_{ka} + \sum_{a \in A} h_a y_a \\
\text{s.t.:} \quad & \sum_{a\in\delta^+(v)} x_{ka} - \sum_{a\in\delta^-(v)} x_{ka} = b_{kv}, \qquad \forall v \in V,\ \forall k \in K, \\
& \sum_{k\in K} d_k x_{ka} \leq c_a\, y_a, \qquad \forall a \in A, \\
& x_{ka} \in \{0,1\},\ \forall a \in A,\ \forall k \in K;\ \ y_a \in \mathbb{Z}_+, \forall a \in A,
\end{aligned}$$

where V is the set of vertices, A is the set of arcs, and K is the set of commodities. $\delta^+(v)$ and $\delta^-(v)$ are the outbound and inbound arcs of vertex v, respectively, and b_{kv} is 1 if v is the supply vertex of commodity k, -1 if it is the demand vertex of k, and 0 otherwise. Variable x_{ka} is the fraction of commodity k's demand sent through arc a, whereas y_a is the number of capacity units installed on arc a.

The branch–and–cut algorithm generates cutting planes from the splittable and unsplittable arc sets. All of the computations are done on a Sun Ultra 5 workstation with 1 hour time limit using a best–bound node selection strategy in the branch–and–bound search tree.

Our data set is based on the unsplittable multicommodity flow problem instances used in [4]. In these instances capacity is fixed and demand for commodities ranges between 5 and 60. In order to make them capacitated network design problems, we introduced capacity variables with unit capacities 4, 25, 60, and 120 and unit installation costs 50, 250, 450, and 720, respectively. Here we report on our computations with 5 problems that CPLEX had the most difficulty in solving. These problem instances are available at `http://ieor.berkeley.edu/~atamturk/data` and their basic characteristics are summarized in Table 1. Cut–set inequalities [18,6,1] are known to improve the LP relaxations of network design problems significantly; however, their separation problem is $\mathcal{NP}$–hard [5]. Therefore, before solving the problems we added only the cut–set inequalities defined for one and two–node subsets of the network to the formulations and used these formulations as the basis for our comparisons.

Table 1. Problem instances

Problem	Commodities	Nodes	Arcs	Variables		Constraints
				Flow	Capacity	
1	70	29	61	8540	61	2181
2	58	18	29	3364	29	1120
3	93	27	37	7178	37	2612
4	87	24	42	7308	42	2196
5	81	27	36	5832	36	2284

4.1. Experiments with splittable flow problems

The first experiment is done to test the effectiveness of the exact separation algorithm for the residual capacity inequalities described in Sect. 2.2. Since these inequalities are developed for splittable flow network design problems, we relaxed the binary flow variables of our data set to continuous variables. Table 2 summarizes the computations with the branch–and–cut algorithm using residual capacity cuts. In this table, for each problem (`prob`) and capacity (`cap`) combination, we report the number of residual capacity cuts added (`cuts`), percentage improvement in the integrality gap obtained by the cuts at the root node of the search tree (`gap impr`), the number of nodes evaluated in the search tree (`b&b nodes`), elapsed CPU time in seconds (`time`) or percentage gap between the best upper bound and the best lower bound at termination if the time limit is reached (`endgap`). The gap improvement is calculated as $100 \times \frac{\texttt{zroot-zprep}}{\texttt{zub-zprep}}$, where `zprep` is the objective value of the LP relaxation after preprocessing, `zroot` the value of the LP relaxation at the root node just before branching, and `zub` the value of the best integer solution known for an instance. In order to reflect the improvement obtained by the cut–set inequalities as well, we used the `zprep` values of formulations before adding the cut–set inequalities. In all tables columns with heading (1) show the performance of the branch–and–bound algorithm for the base formulation that includes the cut–set inequalities.

In any branch–and–cut algorithm the frequency of applying a separation routine in the search tree has an important effect on the computations. In this experiment the separation routine for the residual capacity inequalities is run only at the root node. This choice is based on our observation that most of the effective cuts are found at the root node of the search tree. Observe that a large number of cuts are added compared with the

Table 2. Computational results with splittable flow arc sets

cap	prob	cuts	gap impr (1)	gap impr (2)	b&b nodes (1)	b&b nodes (2)	time/(endgap) (1)	time/(endgap) (2)
4	1	20	35.7	44.3	27630	22310	(0.1)	(0.1)
	2	17	25.8	43.2	6040	2918	177.4	85.9
	3	12	48.9	62.3	15736	17720	437.5	435.3
	4	24	81.8	81.8	29683	23282	(0.0)	(0.0)
	5	17	47.3	80.7	1052	1172	38.5	41.4
25	1	80	37.7	56.2	6858	2929	(0.8)	(0.8)
	2	68	44.9	61.3	5840	2967	266.7	277.4
	3	31	63.1	75.2	7177	5476	206.4	157.5
	4	57	56.9	59.3	9554	7441	(0.7)	(0.2)
	5	29	32.3	68.6	9037	8384	285.4	260.4
60	1	262	17.0	55.1	5403	967	(3.9)	(2.7)
	2	206	19.2	58.5	1788	483	133.6	161.5
	3	114	59.8	72.4	7753	5893	245.6	224.5
	4	257	22.4	55.3	8250	1832	(1.6)	(1.4)
	5	77	48.7	79.5	2166	1047	78.8	49.5
120	1	625	23.2	52.1	3650	363	(17.9)	(13.8)
	2	500	26.4	58.1	39465	9662	2118	3411
	3	340	66.3	83.7	797	626	39.2	38.4
	4	629	12.9	47.6	6665	624	(10.8)	(7.4)
	5	209	55.4	75.9	1056	1616	52.4	78.4

(1) base formulation., (2) residual cap. ineqs.

number of arcs in the problems and the number of cuts added increases with the capacity. The addition of the cuts reduces the integrality gap at the root node significantly and for almost all problems decreases the total number of nodes evaluated. For the problems that could not be solved within the time limit, `endgap` is smaller for all of the problems when the residual capacity cuts are added. Since the residual capacity inequalities describe the convex hull of the splittable arc sets $\mathcal{F}_S$ and we use an exact algorithm to separate them, the integrality gap improvement shown under heading (2) of Table 2 is the best that can be achieved by using cutting planes from individual arc sets for these instances.

4.2. Experiments with unsplittable flow problems

The next set of experiments are on the multicommodity unsplittable flow network design problem. First, we test the impact of the inequalities described in Sect. 3 in reducing the integrality gap at the root node of the search tree. Under headings (2), (3) and (4) of Table 3, we report the number of cuts added and the integrality gap improvement for c–strong inequalities, k–split c–strong inequalities, and lifted cover inequalities, respectively. As in Table 2, we use `zprep` values of formulations before adding the cut–set inequalities to calculate `impr` so that the improvement can be compared with the one obtained by cut–set inequalities, shown under heading (1) of Table 3.

Given a fractional point $(\bar{x}, \bar{y})$, in order to find violated c–strong inequalities, we use the fact there exists an optimal solution S^* to the separation problem such that $i \in S^*$ if $\bar{x}_i = 1$ and $i \in N \setminus S^*$ if $\bar{x}_i = 0$. Therefore after fixing the variables with integral LP values, for each value of c we choose the elements of S in nondecreasing order of $\bar{x}_i a_i$ in a greedy fashion for the fractional variables. We observed that in the separation problem usually more than 90% of the variables are fixed by the optimality criteria of Proposition 2. A greedy heuristic is used to separate the knapsack cover inequalities [15] after letting $\nu = \lceil \bar{y} \rceil$. When lifting cover inequalities, we let the lifting coefficient of the capacity variable $\alpha = \overline{\alpha}$, since c–strong inequalities correspond to lifted cover inequalities with $\alpha = \underline{\alpha}$. In order to find the lifting coefficients for the projected binary variables, rather than solving the lifting problems exactly, we solve their splittable relaxation as described in Sect. 2.1. To generate k–split c–strong cuts, we use the separation routine for c–strong inequalities, after simply multiplying the coefficients of the arc–set inequality by k. A preliminary test showed that the quality of k–split c–strong cuts degrade for high values of k. Therefore, in these computations, we set the maximum value of k to 4. Comparing headings (3) and (4) with (2) we observe that although a good number of lifted cover and k–split c–strong cuts are generated, further improvement of the integrality gap is limited. The root improvement is slightly better with k–split c–strong cuts for most of the instances. Recall that a lifted cover inequality is generated in two steps. First a violated knapsack cover inequality is found and then it is lifted with the projected variables. Even though a heuristic, the one step separation routine for the k–split c–strong inequalities may find more cuts than the knapsack cover separation that does not take into account the lifting coefficients.

Table 3. Improvement of integrality gap at the root node

cap	prob	(1) impr	(2) cuts	(2) impr	(3) cuts	(3) impr	(4) cuts	(4) impr
4	1	4.3	161	53.8	358	55.2	140	54.1
	2	6.4	74	69.2	153	72.6	63	70.7
	3	37.7	70	79.6	137	83.6	70	56.6
	4	9.8	157	42.4	324	44.5	143	43.0
	5	38.4	52	82.9	77	88.4	44	84.5
25	1	0.00	270	32.2	633	34.6	225	32.7
	2	22.8	119	57.4	385	64.6	106	62.1
	3	54.2	84	62.0	218	63.6	79	62.0
	4	0.00	204	12.8	531	14.4	165	13.2
	5	17.2	126	54.1	218	56.3	115	54.1
60	1	4.7	317	25.7	854	28.4	297	26.9
	2	14.0	185	44.5	462	48.0	153	47.7
	3	53.7	153	61.2	355	61.6	127	61.1
	4	0.6	267	17.5	683	19.7	203	19.0
	5	38.0	201	68.5	431	69.3	196	68.6
120	1	9.4	486	30.7	1291	31.2	438	30.7
	2	28.0	379	52.8	809	51.6	301	52.7
	3	62.0	420	71.1	732	71.6	316	71.4
	4	2.4	531	39.3	1094	39.7	437	38.9
	5	35.5	539	63.0	746	63.3	498	63.0

(1) base formulation, (2) c–strong ineqs.,
(3) k–split c–strong ineqs., (4) lifted cover ineqs.

In the next table we compare the overall performance of the branch–and–cut algorithm for c–strong cuts only, for k–split c–strong cuts, with $1 \leq k \leq 4$, and finally for all cuts, including the lifted covers. In Table 4 we report the number of nodes evaluated (`nodes`), and elapsed CPU time in seconds (`time`) or percentage gap between the best known upper bound and the best lower bound at termination if the time limit is reached (endgap). In the separation routine for k–split c–strong inequalities, for each arc we increase the value of k only if no cut is found with the current value. Separation routines are run in the first 50 nodes, which correspond to nodes that are high in the search tree, because we use a best–bound node selection strategy. Comparing columns with headings (1) and (2) we see that generating c–strong cuts reduces the number of nodes and the overall CPU time significantly. Generating k–split c–strong inequalities and lifted cover inequalities in addition to c–strong inequalities generally has a positive effect. However, the additional improvement is not as significant as adding c–strong inequalities on the base formulation.

Table 4. Computational results with unsplittable flow arc sets

		b&b nodes				time / (endgap)			
cap	prob	(1)	(2)	(3)	(4)	(1)	(2)	(3)	(4)
4	1	18536	4269	2521	2282	(2.4)	(1.0)	(1.0)	(0.9)
	2	61101	757	458	494	(1.4)	133.8	93.4	117.1
	3	78659	40359	18723	16898	(0.3)	1701	819.2	728.0
	4	17494	5273	7572	5585	(1.6)	(1.0)	(1.2)	(1.1)
	5	99686	3345	936	1318	(0.1)	160.0	56.8	72.1
25	1	11283	2084	1351	1268	(12.7)	(9.5)	(9.3)	(9.3)
	2	43535	6791	2362	5396	(8.6)	2838	1767	(0.0)
	3	71023	2399	4877	1896	2687	133.4	268.9	125.1
	4	14798	2469	1191	1392	(8.8)	(7.9)	(7.8)	(7.9)
	5	95088	32464	28679	21012	(0.1)	1832	(0.1)	1878
60	1	9949	4154	2561	1817	(26.8)	(25.5)	(25.0)	(24.8)
	2	12008	5050	7902	6246	842.9	1397	1835	1890
	3	9400	9062	6632	6050	330.1	403.4	293.6	288.6
	4	12050	6043	2952	2653	(11.2)	(11.3)	(11.1)	(10.7)
	5	7210	2533	4208	1935	256.8	134.3	202.4	106.4
120	1	4869	968	932	720	(27.2)	(24.6)	(24.7)	(24.9)
	2	19082	12431	13208	16321	1147	2878	2371	(0.1)
	3	835	680	606	570	42.8	60.0	53.0	67.3
	4	6261	1128	1817	1332	(14.7)	(10.0)	(8.8)	(9.5)
	5	22827	2511	2888	2820	933.2	149.0	189.9	151.9

(1) base formulation, (2) c–strong ineqs., (3) k–split c–strong ineqs., (4) all ineqs.

Based on these experimental results, our conclusion is that inequalities from the arc sets of capacitated network design problems do a reasonably good job in strengthening the LP relaxations and improving the performance of LP based search algorithms. However, inequalities that capture additional structures of the network design problems seem to be necessary for solving them more efficiently.

For the splittable flow problems, the exact separation algorithm for residual capacity inequalities empirically provides the maximum integrality gap improvements based on inequalities from arc sets. It would be interesting to know the value of the maximum possible improvement that can be obtained by using inequalities from arc sets for the unsplittable flow problems as well. One possible way of finding this bound is to solve LP relaxations of Dantzig–Wolfe reformulations of the unsplittable multicommodity flow network design problem by relaxing the demand constraints. One can solve this LP relaxation by generating columns over the pricing subproblems consisting of individual arc sets with the dynamic programming algorithm given in Sect. 3.1.

References

1. Atamtürk, A.: On capacitated network design cut–set polyhedra. To appear in Math. Program.
2. Atamtürk, A., Rajan, D. (2000): On splittable and unsplittable flow capacitated network design arc–set polyhedra. BCOL Research Report 00.02, July 2000
3. Balas, E. (1975): Facets of the knapsack polytope. Math. Program. **8**, 146–164
4. Barnhart, C., Hane, C.A., Vance, P.H. (2000): Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. Operations Research **48**, 318–326
5. Bienstock, D. (2001): Personal communication
6. Bienstock, D., Günlük, O. (1996): Capacitated network design – Polyhedral structure and computation. INFORMS Journal on Computing **8**, 243–259
7. Brockmüller, B., Günlük, O., Wolsey, L.A. (1996): Designing private line networks – Polyhedral analysis and computation. CORE Discussion Paper 9647, Université Catholique de Louvain
8. Ceria, S., Cordier, C., Marchand, H., Wolsey, L.A. (1998): Cutting planes for integer programs with general integer variables. Math. Program. **81**, 201–214
9. Cordier, C., Marchand, H., Laundy, R., Wolsey, L.A. (1999): *bc-opt*: a branch-and-cut code for mixed integer programs. Math. Program. **86**, 335–354
10. Crowder, H., Johnson, E.L., Padberg, M.W. (1983): Solving large-scale zero-one linear programming problems. Operations Research **31**, 803–834
11. Garey, M.R., Johnson, D.S. (1979): Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York
12. Gavish, B., Altinkemer, K. (1990): Backbone network design tools with economic tradeoffs. ORSA Journal on Computing **2**, 58–76
13. Grötschel, M., Lovász, L., Schrijver, A. (1981): The ellipsoid method and its consequences in combinatorial optimization. Combinatorica **1**, 169–197
14. Gu, Z., Nemhauser, G.L., Savelsbergh, M.W.P. (1994): Lifted knapsack covers inequalities for 0–1 integer programs: Fast algorithms. Manuscript, Georgia Institute of Technology, Atlanta
15. Gu, Z., Nemhauser, G.L., Savelsbergh, M.W.P. (1998): Lifted cover inequalities for 0–1 integer programs: Computation. INFORMS Journal on Computing **10**, 427–437
16. Hammer, P.L., Johnson, E.L., Peled, U.N. (1975): Facets of regular 0–1 polytopes. Math. Program. **8**, 179–206
17. van Hoesel, S.P.M., Koster, A.M.C.A., van de Leensel, R.L.M.J., Savelsbergh, M.W.P. (2000): Polyhedral results for the edge capacity polytope. Technical Report SC 00-22, Konrad-Zuse-Zentrum für Informationstechnik Berlin
18. Magnanti, T.L., Mirchandani, P. (1993): Shortest paths, single origin-destination network design, and associated polyhedra. Networks **23**, 103–121
19. Magnanti, T.L., Mirchandani, P., Vachani, R. (1993): The convex hull of two core capacitated network design problems. Math. Program. **60**, 233–250
20. Marchand, H., Wolsey, L.A. (1999): The 0–1 knapsack problem with a single continuous variable. Math. Program. **85**, 15–33
21. Marchand, H., Wolsey, L.A. (2001): Aggregation and mixed integer rounding to solve MIPs. Operations Research **49**, 363–371
22. Nemhauser, G.L., Wolsey, L.A. (1988): Integer and Combinatorial Optimization. John Wiley and Sons, New York
23. Padberg, M.W. (1979): Covering, packing and knapsack problems. Annals of Discrete Mathematics **4**, 265–287
24. Weismantel, R. (1997): On the 0/1 knapsack polytope. Math. Program. **77**, 49–68
25. Wolsey, L.A. (1975): Faces for linear inequality in 0–1 variables. Math. Program. **8**, 165–178
26. Wolsey, L.A. (1976): Facets and strong valid inequalities for integer programs. Operations Research **24**, 367–372
27. Zemel, E. (1989): Easily computable facets of the knapsack polytope. Mathematics of Operations Research **14**, 760–764