On the Mixed 0-1 Knapsack Set with a GUB Constraint

Wei-Kun Chen[1] and Yu-Hong Dai[2]

[1] School of Mathematics and Statistics/Beijing Key Laboratory on MCAACI, Beijing Institute of Technology, Beijing 100081, China, chenweikun@bit.edu.cn
[2] LSEC, ICMSEC, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100090, China, dyh@lsec.cc.ac.cn

Abstract. In this paper, we investigate the polyhedral structure of a mixed 0-1 set which consists of a mixed knapsack constraint and a generalized upper bound (GUB) constraint with n binary variables and m bounded continuous variables. This set appears as a common substructure of the network design problems with discrete capacity installation costs. For this set, we first introduce a family of valid inequalities, called coefficient strengthening (CS) inequalities, derived by relaxing some continuous variables and strengthening the coefficients of some binary variables in the mixed knapsack constraint. Then we give a necessary and sufficient condition to guarantee the CS inequality to be facet-defining and prove that together with the initial constraints, the CS inequalities are sufficient to describe the convex hull of this set. Furthermore, we develop an exact polynomial-time separation algorithm for the CS inequalities. Finally, we perform numerical experiments on using the CS inequalities as cutting planes for solving the network design problems. Numerical results demonstrate the effectiveness of the CS inequalities and the proposed exact separation algorithm.

Keywords: Coefficient strengthening · Cutting plane · GUB constraint· Mixed 0-1 knapsack polytope · Separation algorithm

1 Introduction

Consider an extension of the mixed 0-1 knapsack set in which the 0-1 variables are subjected to a generalized upper bound constraint (GUB):

$$X = \left\{ (x, y) \in \{0,1\}^n \times \mathbb{R}_+^m \, : \, \sum_{j=1}^{m} y_j \leq b + \sum_{i=1}^{n} a_i x_i, \;\; \sum_{i=1}^{n} x_i \leq 1, \; y \leq u \right\},$$

where $a_i > 0$ for $i \in N := \{1, \dots, n\}$, $u_j > 0$ for all $j \in M := \{1, \dots, m\}$, and $b \geq 0$ are rational. This set appears as a common substructure of the network design problems with discrete capacity installation costs [4,6]. Specifically, in such problems, variable y_j denotes the amount of commodity j with demand

u_j flowing along an edge in the network. The total flow on this edge is upper bounded by capacity $b + \sum_{i=1}^n a_i x_i$, i.e,

$$\sum_{j=1}^m y_j \leq b + \sum_{i=1}^n a_i x_i. \tag{1}$$

Here b denotes the preinstalled capacity, a_i stands for the capacity of module i, and x_i symbolizes that whether or not module i is installed on this edge. Installing different modules on an edge will incur different costs. We are interested in the model that at most one module may be installed on every edge, that is,

$$\sum_{i=1}^n x_i \leq 1. \tag{2}$$

Notice that this requirement, however, can be relaxed. Indeed, the model that allows to install combinations of different modules on every edge can also be transformed into the above model; see [8,18].

When the GUB constraint (2) is relaxed, X reduces to the mixed 0-1 knapsack set whose polyhedral structure has been studied in the literature. In particular, Richard et al. [16,17] described facets for the mixed 0-1 knapsack polytope using lifting techniques. Narisetty et al. [13] developed the mixed lifted cover and pack inequalities and gave conditions under which they are facet-defining. Marchand and Wolsey [11] studied a closely related set where there exists only a single unbounded continuous variable. We remark that with the GUB constraint (2), the polyhedral structure of $\text{conv}(X)$ is intrinsically different from that of the mixed 0-1 knapsack polytope. Specifically, the optimization of a linear function over set X can be done in polynomial time as by enumeration of the 0-1 solutions for variables x, we can reduce the problem to $n+1$ linear programming (LP) problems. This implies the possibility to derive a polyhedral description and to develop an efficient separation algorithm for polytope $\text{conv}(X)$; see [15]. In sharp contrast, for the mixed 0-1 knapsack polytope, there is little hope of doing this as it is NP-hard to optimize a linear function over it.

Another closely related set is the splittable flow arc set $Y = \{(x, y) \in \mathbb{Z}_+ \times \mathbb{R}^n_+ : \sum_{j=1}^m y_j \leq b + cx,\ y \leq u\}$ where $c > 0$. Indeed, let $n = \max\{0, \lceil (u(M) - b)/c \rceil\}$ and $a_i = i*c$ for $i = 1, \ldots, n$. Then set X can be seen as a binary extended formulation of set Y; see [5]. There exist several studies on polyhedron $\text{conv}(Y)$. Magnanti et al. [10] proposed the residual capacity inequalities and showed that when $b = 0$, polyhedron $\text{conv}(Y)$ can be described by these inequalities and the initial constraints (see [1] for the generalized results of polyhedron $\text{conv}(Y)$ with an arbitrary b). Atamtürk and Rajan [2] developed a linear-time separation algorithm for the residual capacity inequalities. Unfortunately, with arbitrary values of parameters a_i with $a_i > 0$, $i = 1, \ldots, n$, the above results cannot directly be applied to polytope $\text{conv}(X)$.

To the best of our knowledge, a polyhedral study on polytope $\text{conv}(X)$ is missing in the literature. The motivation of this paper is to fill the research gap. In particular, we propose a family of valid inequalities, called coefficient strengthening (CS) inequalities, derived by first relaxing some continuous variables y_j and then strengthening the coefficients of some binary variables x_i in the mixed knapsack constraint (1). We give a necessary and sufficient condition to guarantee the CS inequality to be facet-defining for polytope $\mathrm{conv}(X)$. In addition, together with the initial constraints, the CS inequalities are shown to be sufficient to describe polytope $\mathrm{conv}(X)$. Furthermore, we develop an exact polynomial-time separation algorithm for the CS inequalities. We also perform numerical experiments on using the CS inequalities as cutting planes for solving the network design problems. Numerical results demonstrate the effectiveness of the CS inequalities and the proposed exact separation algorithm.

We would like to emphasize that the proposed CS inequality can be seen as the well-known mixed integer rounding (MIR) inequality [12,14]. However, our exact separation algorithm for the CS inequalities is much more effective in finding cuts and hence can improve the solution efficiency of the branch-and-cut framework, as compared with the existing heuristic algorithm in [12].

The remainder of the paper is organized as follows. Section 2 describes the CS inequality and the polyhedral description of $\mathrm{conv}(X)$. Section 3 presents a separation algorithm for the CS inequalities. Finally, section 4 provides the computational results.

Assumptions and notations. We assume $0 < a_1 \leq \cdots \leq a_n$ since otherwise we can reorder the variables x_i, $i = 1, \ldots, n$. Denote $w(S) = \sum_{i \in S} w_i$ for a vector $w \in \mathbb{R}^m$ and a subset $S \subseteq M$. Without loss of generality, we assume $a_n \leq u(M) - b$ since otherwise, we can equivalently strengthen a_n as $u(M) - b$. For notations convenience, we denote $a_0 = 0$ and $a_{n+1} = +\infty$. Throughout this paper, we denote X_{L} as the linear relaxation of set X.

2 Polyhedral description

In this section, we first present the CS inequalities for set X and give a necessary and sufficient condition for them to be facet-defining for polytope $\mathrm{conv}(X)$. Then we show that together with the initial constraints, the CS inequalities are sufficient to describe polytope $\mathrm{conv}(X)$.

Let $S \subseteq M$ such that $b_S := u(S) - b > 0$. First, relaxing variables y_j to zero for all $j \in M \backslash S$ in inequality (1), we obtain inequality

$$y(S) \leq b + \sum_{i=1}^{n} a_i x_i. \tag{3}$$

Then, using the fact that x_i, $i = 1, \ldots, n$, are binary variables, we can strengthen inequality (3) as

$$y(S) \leq b + \sum_{i=1}^{r} a_i x_i + b_S \sum_{i=r+1}^{n} x_i, \tag{4}$$

where $r \in N \cup \{0\}$ satisfying $a_r \leq b_S < a_{r+1}$ (note that we assume $a_0 = 0$ and $a_{n+1} = +\infty$). Clearly, inequality (4) is valid for X. We call (4) as the CS inequality. When it is associated with a specific subset S, we also call it as S-CS inequality. Notice that if $b_S \geq a_n$, inequality (4) is dominated by or equivalent to inequality (1) (as it is assumed that $a_n \leq u(M) - b$). Hence, we are only interested in the CS inequality (4) with $b_S < a_n$, or equivalently, $r + 1 \leq n$.

Remark 1. The CS inequality (with $0 < b_S < a_n$) can be seen as an MIR inequality [12,14]. Indeed, inequality (3) can be written as

$$\sum_{i=1}^{n} \frac{a_i}{a_n} x_i + s \geq \frac{b_S}{a_n}, \tag{5}$$

where $s = \frac{1}{a_n} \sum_{j \in S} (u_j - y_j)$. The MIR inequality based on inequality (5) is

$$\sum_{i=1}^{n} \min \left\{ \frac{b_S}{a_n}, \frac{a_i}{a_n} \right\} x_i + s \geq \frac{b_S}{a_n}. \tag{6}$$

As $a_1 \leq \cdots \leq a_n$, $\sum_{i=1}^{n} \min \left\{ \frac{b_S}{a_n}, \frac{a_i}{a_n} \right\} x_i = \sum_{i=1}^{r} \frac{a_i}{a_n} x_i + \sum_{i=r+1}^{n} \frac{b_S}{a_n} x_i$. Then, substituting $s = \frac{1}{a_n} \sum_{j \in S} (u_j - y_j)$ into inequality (6) and multiplying (6) by a_n, we obtain the CS inequality (4) (for more details of the MIR inequality, see [12,14]).

The following proposition give a necessary and sufficient condition for the CS inequality (4) to be facet-defining for polytope $\operatorname{conv}(X)$. The proof can be found in [3].

Proposition 1. *Let $S \subseteq M$ such that $0 < b_S < a_n$. Then the CS inequality (4) defines a facet of polytope $\operatorname{conv}(X)$ if and only if at least one of the following three conditions holds:* (i) $a_1 < b_S$; (ii) $|S| = 1$; (iii) $b > 0$.

We now establish the main result of this section.

Theorem 1. *Together with the initial constraints (1), (2), $0 \leq x_i \leq 1$, $i \in N$, and $0 \leq y_j \leq u_j$, $j \in M$, the CS inequalities (4) are sufficient to describe polytope $\operatorname{conv}(X)$.*

Proof. We use the technique of [9]. Given an arbitrary objective function $(c, d) \neq (0, 0)$, let $\mathcal{O}(c, d)$ denote the set of optimal solutions of problem

$$\min \left\{ c^\top x + d^\top y \,:\, (x, y) \in X \right\}. \tag{7}$$

We shall prove the statement by showing that there exists an inequality listed in the theorem such that it holds at equality for all $(x, y) \in \mathcal{O}(c, d)$.

If $d_j > 0$ for some $j \in M$, the statement is true since $y_j = 0$ for all $(x, y) \in \mathcal{O}(c, d)$. Thus we assume $d_j \leq 0$ for all $j \in M$. Let $S = \{ j \,:\, d_j < 0,\ j \in M \}$. Consider the following two cases.

(i) $u(S) - b \leq 0$. If $|S| \geq 1$, then for every point $(x, y) \in \mathcal{O}(c, d)$, $y_q = u_q$ must be true for $q \in S$, and thus the statement holds true. Otherwise, it follows

$d_j = 0$ for all $j \in M$. As $(c,d) \neq (0,0)$, there must exist some $p \in N$ such that $c_p \neq 0$. If $c_p > 0$, we have $x_p = 0$ for each $(x,y) \in \mathcal{O}(c,d)$. If $c_p < 0$, since the point $(\bar{x},\bar{y})$ defined by $\bar{x}_p = 1$, $\bar{x}_i = 0$ for $i \in N\backslash\{p\}$, and $\bar{y}_j = 0$ for $j \in M$ is feasible and gives a negative objective value in problem (7), $\sum_{i=1}^n x_i = 1$ must be satisfied at each point $(x,y) \in \mathcal{O}(c,d)$.

(ii) $u(S) - b > 0$. This implies that $S \neq \varnothing$ as $b \geq 0$. We complete the proof by showing that the S-CS inequality (4) holds at equality for all $(x,y) \in \mathcal{O}(c,d)$. To do this, we proceed by contradiction. Suppose that there exists a point $(\hat{x},\hat{y}) \in \mathcal{O}(c,d)$ such that

$$\hat{y}(S) < b + \sum_{i=1}^{r} a_i \hat{x}_i + b_S \sum_{i=r+1}^{n} \hat{x}_i. \tag{8}$$

Together with $\sum_{i=1}^n \hat{x}_i \leq 1$ (as $(\hat{x},\hat{y}) \in X$), we can derive $\hat{y}(S) < b + b_S = b + u(S) - b = u(S)$, which implies that $\hat{y}_q < u_q$ must hold for some $q \in S$. Then, for each $j \in M\backslash S$, since $d_j = 0$, it follows $\hat{y}_j = 0$. As a result, we have

$$\sum_{i=1}^{n} a_i \hat{x}_i - \hat{y}(M) = \sum_{i=1}^{n} a_i \hat{x}_i - \hat{y}(S) \geq \sum_{i=1}^{r} a_i \hat{x}_i + b_S \sum_{i=r+1}^{n} \hat{x}_i - \hat{y}(S) > b,$$

where the last inequality follows from (8). However, this indicates that increasing $\hat{y}_q$ by a small value $\epsilon > 0$ gives another feasible solution of problem (7) whose objective value is smaller than that of point $(\hat{x},\hat{y})$. As a result, it contradicts with the fact that $(\hat{x},\hat{y})$ is optimal for problem (7). $\square$

3 Separation problem

In this section, we consider the separation problem of polytope $\mathrm{conv}(X)$. Since the constraints in set X_{L} can be checked for violations in linear time, the separation problem of polytope $\mathrm{conv}(X)$ can be reduced to, given a point $(\bar{x},\bar{y}) \in X_{\mathrm{L}}$, either find a violated inequality (4) or prove that no such one exists. We note that due to the potentially exponential number of selections of subset $S \subseteq M$ in the CS inequality (4), it is unrealistic to solve the separation problem by enumeration. In the following, we show that by considering a linear number of subsets of M, a most violated CS inequality (4) can be found.

First, to find a CS inequality (4) violated by point $(\bar{x},\bar{y}) \in X_{\mathrm{L}}$, we may assume $\bar{x} \notin \{0,1\}^n$ since otherwise $(\bar{x},\bar{y}) \in \mathrm{conv}(X)$, and hence no violated one exists. In addition, the CS inequality (4) with $b_S \leq 0$ or $b_S \geq a_n$ cannot be violated by point $(\bar{x},\bar{y})$ (as $(\bar{x},\bar{y}) \in X_{\mathrm{L}}$), and hence we can also limit to consider the CS inequality (4) with $0 < b_S < a_n$. We next further divide the separation problem of the CS inequalities into n subproblems where each subproblem attempts to find a most violated CS inequality (4) with $a_{k-1} \leq b_S < a_k$ where $k \in N$. In particular, to find a most violated CS inequality (4) with $a_{k-1} \leq b_S < a_k$, we can solve problem

$$\begin{aligned} v_k = \min_{z,\bar{b}} \quad & b + \sum_{i=1}^{k-1} a_i \bar{x}_i + \bar{b} \sum_{i=k}^{n} \bar{x}_i - \sum_{j \in M} \bar{y}_j z_j \\ \text{s.t.} \quad & a_{k-1} \leq \bar{b} = \sum_{j \in M} u_j z_j - b < a_k, \ z_j \in \{0,1\}, \ j \in M, \end{aligned} \tag{P$_k$}$$

where $z \in \{0,1\}^m$ is the characteristic vector of subset $S \subseteq M$. If $v_k < 0$, then the CS inequality (4) corresponding to an optimal solution $(z, \bar{b})$ of problem (P$_k$) is violated by point $(\bar{x}, \bar{y})$; otherwise, no violated one with $a_{k-1} \leq b_S < a_k$ exists. The separation problem of the CS inequalities can then be solved by enumerating v_k, $k = 1, \ldots, n$. To be more specific, let $v = \min_{k=1,\ldots,n}\{v_k\}$. If $v < 0$, we can find a most violated inequality (4); otherwise, we conclude that no violated one exists. Problems (P$_k$), $k = 1, \ldots, n$, are mixed integer programming (MIP) problems, which are hard to solve in general. In the following, by investigating the relationship of different problems (P$_k$), $k = 1, \ldots, n$, we are able to provide a strongly polynomial-time separation algorithm for solving the separation problem of the CS inequalities (4).

We first substitute $\bar{b} = \sum_{j \in M} u_j z_j - b$ into the objective function of problem (P$_k$) and obtain an equivalent problem:

$$\begin{aligned} v_k = \min_{z} \quad & \sum_{i=1}^{k-1} a_i \bar{x}_i + b \left(1 - \sum_{i=k}^{n} \bar{x}_i \right) + \sum_{j \in M} \left(u_j \sum_{i=k}^{n} \bar{x}_i - \bar{y}_j \right) z_j \\ \text{s.t.} \quad & a_{k-1} \leq \sum_{j \in M} u_j z_j - b < a_k, \ z_j \in \{0,1\}, \ j \in M. \end{aligned} \tag{P$'_k$}$$

Let T_k denote the index set of negative objective coefficients of variables z_j, $j \in M$, in problem (P$'_k$), i.e.,

$$T_k = \left\{ j \,:\, u_j \sum_{i=k}^{n} \bar{x}_i - \bar{y}_j < 0, \ j \in M \right\}. \tag{9}$$

In addition, for $k \in N$, let $g_k(z)$ denote the objective value of problem (P$'_k$) at point $z \in \{0,1\}^m$, i.e.,

$$g_k(z) = \sum_{i=1}^{k-1} a_i \bar{x}_i + b \left(1 - \sum_{i=k}^{n} \bar{x}_i \right) + \sum_{j \in M} \left(u_j \sum_{i=k}^{n} \bar{x}_i - \bar{y}_j \right) z_j. \tag{10}$$

We have the following observation.

Observation 1 *Let T_k be defined as in (9) and $\hat{z} \in \{0,1\}^m$ be the associated characteristic vector. Then $v_k \geq g_k(\hat{x})$. Furthermore, (i) if $g_k(\hat{z}) \geq 0$, no CS inequality (4) with $a_{k-1} \leq b_S < a_k$ violated by point $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$ exists; (ii) if $a_{k-1} \leq b_{T_k} < a_k$ and $g(\hat{z}) < 0$, among the CS inequalities (4) with $a_{k-1} \leq b_S < a_k$, the T_k-CS inequality is the most violated one for point $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$.*

By Observation 1, we only need to consider problem (P'_k) with $b_{T_k} < a_{k-1}$ or $b_{T_k} \geq a_k$.

Lemma 1. *Let T_k be defined as in (9) and $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$. If $b_{T_k} \leq 0$ or $b_{T_k} \geq a_n$, no CS inequality (4) with $a_{k-1} \leq b_S < a_k$ can be violated by point $(\bar{x}, \bar{y})$.*

Proof. Let $\hat{z} \in \{0,1\}^m$ be the characteristic vector of set T_k. By Observation 1 (i), it is enough to show that if $b_{T_k} \leq 0$ or $b_{T_k} \geq a_n$, $g_k(\hat{z}) \geq 0$ must be hold where

$$g_k(\hat{z}) = \sum_{i=1}^{k-1} a_i \bar{x}_i + b\left(1 - \sum_{i=k}^{n} \bar{x}_i\right) + \sum_{j \in T_k} \left(u_j \sum_{i=k}^{n} \bar{x}_i - \bar{y}_j\right).$$

Indeed, if $b_{T_k} = u(T_k) - b \leq 0$, or equivalently, $b \geq u(T_k)$, we have

$$\begin{aligned} g_k(\hat{z}) \geq & \sum_{i=1}^{k-1} a_i \bar{x}_i + u(T_k)\left(1 - \sum_{i=k}^{n} \bar{x}_i\right) + \sum_{j \in T_k} \left(u_j \sum_{i=k}^{n} \bar{x}_i - \bar{y}_j\right) \\ = & \sum_{i=1}^{k-1} a_i \bar{x}_i + \sum_{j \in T_k} (u_j - y_j) \geq 0, \end{aligned}$$

where the first inequality follows from $\sum_{i=k}^{n} \bar{x}_i \leq \sum_{i=1}^{n} \bar{x}_i \leq 1$ and the second one follows from $\bar{x}_i \geq 0$ and $\bar{y}_j \leq u_j$ for all $i \in N$ and $j \in M$, respectively (as $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$). If $b_{T_k} = u(T_k) - b \geq a_n$, or equivalently, $-b \geq a_n - u(T_k)$, we have

$$\begin{aligned} g_k(\hat{z}) = & \sum_{i=1}^{k-1} a_i \bar{x}_i + b - b\sum_{i=k}^{n} \bar{x}_i + \sum_{j \in T_k} \left(u_j \sum_{i=k}^{n} \bar{x}_i - \bar{y}_j\right) \\ \geq & \sum_{i=1}^{k-1} a_i \bar{x}_i + b + [a_n - u(T_k)]\sum_{i=k}^{n} \bar{x}_i + \sum_{j \in T_k} \left(u_j \sum_{i=k}^{n} \bar{x}_i - \bar{y}_j\right) \\ = & \sum_{i=1}^{k-1} a_i \bar{x}_i + a_n \sum_{i=k}^{n} \bar{x}_i + b - \sum_{j \in T_k} \bar{y}_j \geq \sum_{i=1}^{n} a_i \bar{x}_i + b - \sum_{j \in M} \bar{y}_j \geq 0, \end{aligned}$$

where the second inequality follows from $a_n \geq a_i$ and $\bar{x}_i \geq 0$ for $i = k, \ldots, n$ and $\bar{y}_j \geq 0$ for $j \in M \backslash T_k$ and the last one follows from $\sum_{j \in M} \bar{y}_j \leq b + \sum_{i=1}^{n} a_i \bar{x}_i$ (as $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$). $\square$

Together with Observation 1 and Lemma 1, we are left with the case that $a_{\tau-1} < b_{T_k} \leq a_\tau$ for some τ with $\tau \neq k$ and $1 \leq \tau \leq n$. The following lemma shows that in this case, if there exists an S-CS inequality with $a_{k-1} < b_S \leq a_k$ violated by point $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$ by ϵ, then there must also exist an S'-CS inequality with $a_{\tau-1} < b_{S'} \leq a_\tau$ violated by point $(\bar{x}, \bar{y})$ by $\epsilon_1 \geq \epsilon$.

Lemma 2. *Let T_k be defined as in (9) and $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$. If $a_{\tau-1} < b_{T_k} \leq a_\tau$ for some τ with $\tau \neq k$ and $1 \leq \tau \leq n$, then $v_\tau \leq v_k$.*

Proof. Let $\hat{z} \in \{0,1\}^m$ be the characteristic vector of set T_k. The difference of the objective values of problems (P'_k) and (P'_τ) at point $\hat{z}$ is

$$\begin{aligned} g_k(\hat{z}) - g_\tau(\hat{z}) = \quad & \sum_{i=1}^{k-1} a_i \bar{x}_i + b\left(1 - \sum_{i=k}^{n} \bar{x}_i\right) + \sum_{j \in T_k} \left(u_j \sum_{i=k}^{n} \bar{x}_i - \bar{y}_j\right) - \\ & \left[\sum_{i=1}^{\tau-1} a_i \bar{x}_i + b\left(1 - \sum_{i=\tau}^{n} \bar{x}_i\right) + \sum_{j \in T_k} \left(u_j \sum_{i=\tau}^{n} \bar{x}_i - \bar{y}_j\right)\right]. \end{aligned}$$

If $\tau < k$, since $a_i \geq a_\tau \geq b_{T_k} = u(T_k) - b$ and $\bar{x}_i \geq 0$ for $i = \tau, \ldots, k-1$, we have

$$g_k(\hat{z}) - g_\tau(\hat{z}) = \sum_{i=\tau}^{k-1} a_i \bar{x}_i + b \sum_{i=\tau}^{k-1} \bar{x}_i - u(T_k) \sum_{i=\tau}^{k-1} \bar{x}_i = \sum_{i=\tau}^{k-1} \left[a_i + b - u(T_k)\right] \bar{x}_i \geq 0.$$

Otherwise, it follows $\tau > k$ as $\tau \neq k$. Since $a_i \leq a_{\tau-1} < b_{T_k} = u(T_k) - b$ and $\bar{x}_i \geq 0$ for $i = k, \ldots, \tau - 1$, we have

$$g_k(\hat{z}) - g_\tau(\hat{z}) = -\sum_{i=k}^{\tau-1} a_i \bar{x}_i - b \sum_{i=k}^{\tau-1} \bar{x}_i + u(T_k) \sum_{i=k}^{\tau-1} \bar{x}_i = \sum_{i=k}^{\tau-1} [-a_i - b + u(T_k)] \bar{x}_i \geq 0.$$

In both cases, we have $g_\tau(\hat{z}) \leq g_k(\hat{z})$. Notice that as $a_{\tau-1} < b_{T_k} \leq a_\tau$, point $\hat{z}$ is a feasible solution of problem (P'_τ), implying that $v_\tau \leq g_\tau(\hat{z})$. This, together with $g_k(\hat{z}) \leq v_k$ in Observation 1, shows that $v_\tau \leq v_k$. □

By Observation 1 and Lemmas 1-2, to find a most violated CS inequality by point $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$, it suffices to test the n ones associated with subsets $T_1, \ldots, T_n$. More specifically, we first initialize $v := 0$. Then, for $k = 1, \ldots, n$, we test whether

$$a_{k-1} < b_{T_k} \leq a_k \text{ and } v_k = b + \sum_{i=1}^{k-1} a_i \bar{x}_i + b_{T_k} \sum_{i=k}^{n} \bar{x}_i - \bar{y}(T_k) < v \tag{11}$$

hold or not and if yes, update $v := v_k$. In the end, if $v < 0$, a most violated CS inequality (4) is found; otherwise, no violated one exists. Apparently, this gives an $\mathcal{O}(n(m+n))$ separation algorithm. The following important observation, however, enables us to design a much more efficient separation algorithm when n is large (see Algorithm 1).

Observation 2 *Let $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$. Rewriting T_k (defined in (9)) as*

$$\left\{ j \, : \, \sum_{i=k}^{n} \bar{x}_i < \frac{\bar{y}_j}{u_j}, \ j \in M \right\}, \ k = 1, \ldots, n, \tag{12}$$

we have $T_1 \subseteq \cdots \subseteq T_n$.

In Algorithm 1, we reorder the variables y_j, $j = 1, \ldots, m$, such that $\frac{\bar{y}_1}{u_1} \leq \cdots \leq \frac{\bar{y}_m}{u_m}$ in step 1 in the complexity of $\mathcal{O}(m \log m)$. By Observation 2 and this

Algorithm 1: A separation algorithm for the CS inequalities (4)

Input: The set X and the point $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$;
1 Reorder the variables y_j, $j = 1, \ldots, m$, such that $\frac{\bar{y}_1}{u_1} \leq \cdots \leq \frac{\bar{y}_m}{u_m}$;
2 Initialize $v := 0$, $k_0 = -1$, $\mathrm{SAX}_0 = 0$, $\mathrm{SAX}_1 = a_1 \bar{x}_1$, $\mathrm{SX}_1 = \sum_{i=1}^{n} \bar{x}_i$, $\mathrm{YT}_1 = \bar{y}(T_1)$, $\mathrm{UT}_1 = u(T_1)$;
3 **for** $k = 2, \ldots, n$ **do**
4   Compute $\mathrm{YT}_k = \mathrm{YT}_{k-1} + \bar{y}(T_k \backslash T_{k-1})$ and $\mathrm{UT}_k = \mathrm{UT}_k + u(T_k \backslash T_{k-1})$;
5   Compute $\mathrm{SAX}_k = \mathrm{SAX}_{k-1} + a_k \bar{x}_k$ and $\mathrm{SX}_k = \mathrm{SX}_{k-1} - \bar{x}_{k-1}$;
6 **for** $k = 1, \ldots n$ **do**
7   **if** $a_{k-1} < b_{T_k} = \mathrm{UT}_k - b \leq a_k$ **then**
8   Compute $v_k = b + \mathrm{SAX}_{k-1} + b_{T_k} * \mathrm{SX}_k - \mathrm{YT}_k$;
9   Update $v := v_k$ and $k_0 := k$ if $v_k < v$;
10 If $v < 0$, the CS inequality (4) with subset T_{k_0} is violated by point $(\bar{x}, \bar{y})$; otherwise no violated one exists.

ordering, the computation of $\mathrm{YT}_k = \bar{y}(T_k)$ and $\mathrm{UT}_k = u(T_k)$ for $k = 1, \ldots, n$ in steps 3-4 can be done in $O(n)$. Clearly, the computation of $\mathrm{SAX}_k = \sum_{i=1}^{k} a_i \bar{x}_i$ and $\mathrm{SX}_k = \sum_{i=k}^{n} \bar{x}_i$ for $k = 1, \ldots, n$ can also be done in $O(n)$. Finally, with the arrays YT, UT, SAX, and SX, we can test whether conditions (11) hold or not for all $k = 1, \ldots, n$ in the complexity of $\mathcal{O}(n)$; see steps 6-10. Overall, the complexity of Algorithm 1 is $\mathcal{O}(n + m \log m)$. Together with the previously mentioned $\mathcal{O}(n(n + m))$ separation algorithm, we have the following.

Theorem 2. *The separation problem for the CS inequalities* (4) *can be solved in* $\mathcal{O}(\min\{n(m + n), n + m \log m\})$.

To end this section, we would like to highlight the advantage of our exact separation algorithm for the CS inequalities over the heuristic algorithm in [12]. Recall that in Remark 1, the CS inequality (4) is shown to be the MIR inequality. As for the separation of the MIR inequalities, the author in [12] suggested to heuristically choose subset S as

$$\{j \,:\, \bar{y}_j > u_j - \bar{y}_j, \; j \in M\} = \{j \,:\, \bar{y}_j / u_j > 0.5, \; j \in M\}$$

and test whether or not the corresponding inequality is violated by point $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$. To the best of our knowledge, this heuristic algorithm is still employed in state-of-the-art MIP solvers; see, for example, [19]. We note that this heuristic algorithm uses a "0.5" strategy, i.e., it prefers to leave variables y_j, with $\bar{y}_j / u_j > 0.5$, in the CS inequality (or MIR inequality). Our exact separation algorithm for the CS inequality also prefers to leave variables y_j, with a large $\bar{y}_j / u_j$, in the CS inequality; see T_k in (12). However, in contrast to the "0.5" heuristic strategy that does not consider values $\bar{x}_i$, $i \in N$, our separation algorithm finds a most violated CS by enumerating subsets $T_1, \ldots, T_n$ and taking fully values $\bar{x}_i$, $i \in N$, into consideration. Moreover, it guarantees to find a violated inequality by point $(\bar{x}, \bar{y}) \in X_{\mathrm{L}}$ (if such one exists). In the next section, we will present computational results to illustrate this advantage.

4 Numerical results

In this section, we present numerical results to illustrate the effectiveness of using the CS inequalities (4) as cutting planes and the proposed exact separation algorithm in the branch-and-cut solver SCIP [7] for solving the single-source network design problem formulated as:

$$\begin{aligned} \min_{x,y} \quad & \sum_{e\in E} l_e \sum_{i\in N} c_{ei}x_{ei} & \\ \text{s.t.} \quad & \sum_{e\in\delta^+(v)} y_{ej} - \sum_{e\in\delta^-(v)} y_{ej} = \begin{cases} -u_j & \text{if } j = v \\ u_j & \text{if } j = r \\ 0 & \text{otherwise} \end{cases} & \forall v \in V \\ & (x_{e\cdot}, y_{e\cdot}) \in X^e & \forall e \in E \end{aligned} \tag{13}$$

where V and E are the sets of nodes and edges respectively, $r \in V$ is the root node routing demand u_j to node $j \in M \subseteq V$, $\delta^+(v)$ and $\delta^-(v)$ are the outgoing and incoming edges of node v, respectively, l_e is the length of edge e, and X^e is defined by

$$X^e = \Big\{ (x_{e\cdot}, y_{e\cdot}) \in \{0,1\}^{|N|} \times \mathbb{R}_+^{|M|} \,:\, \sum_{j\in M} y_{ej} \le \sum_{i\in N} a_{ei}x_{ei},\ \sum_{i\in N} x_{ei} \le 1, \\ y_{ej} \le u_j,\ j \in M \Big\}.$$

Variable y_{ej} describes the amount of flow, from node r to node j, through edge e, and variable x_{ei} stands for whether module i, with capacity a_{ei} and cost c_{ei}, is installed on edge e.

Our data set is generated using the random procedure in [8,18]. The number of nodes $|V|$ is chosen in $\{20, 30, 40\}$. There are two ways of choosing the root node r. One selects r to be the central node and the other one selects r randomly. There are also two ways of generating the demands $u = (u_1, \dots, u_{|M|})$, which randomly select u_j, $j \in M$, in $[0, 30]$ and $[0, 60]$, respectively. We refer to [18] for a procedure of generating capacities and costs of the modules and a detailed description of the random procedure. For each combination of triple $(|V|, r, u)$, we generate 100 instances. Thus, in total, we have 1200 instances. All numerical experiments were conducted on a cluster of Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz computers, with 180 GB RAM, running Linux (in 64 bit mode). We set a time limit of 3600 seconds for SCIP. Unless otherwise stated, we use the default setting of SCIP.

We first compare the performance of adding the CS inequalities (4) into SCIP (**CS**) with the default setting of SCIP (**Default**). Table 1 summarizes the computational results of the two settings where we remove those instances that cannot be solved by both **CS** and **Default** within the time limit. We report the number of instances solved to optimality (Solved), the average running time (Time) which includes the time spent in separating the cuts, the average number of explored nodes (Nodes), the average percentage gap improvement (Gap)

Table 1: Performance comparison of the setting that uses CS cuts with the default setting.

$\lvert V\rvert$	Total	CS				Default			
		Solved	Time	Nodes	Gap	Solved	Time	Nodes	Gap
20	397	397	12.48	318	92.85	397	14.08	382	92.39
30	325	317	130.29	4563	87.33	312	161.11	6867	87.16
40	141	135	512.01	16316	86.55	117	685.77	29360	86.08

defined by $100 \cdot (z_{\text{ROOT}} - z_{\text{LP}})/(z_{\text{MIP}} - z_{\text{LP}})$, where z_{LP} is the objective value of the initial LP relaxation, z_{ROOT} is the objective value of the LP relaxation after adding cuts, and z_{MIP} is the objective value of the optimal solution.

From Table 1, we observe that **CS** performs better than **Default**, especially for problems with a larger size. In particular, when $|V| = 40$, (i) **CS** is about $1.34\times$ (685.77/512.01) faster than **Default** and can solve 18 (135-117) more instances to optimality; (ii) the number of nodes decreases by a factor of 1.80 (29360/16316). In addition, we observe that the gap improvement of **CS** is only slightly better than that of **Default**. This can be explained by the reason that the CS cuts can be seen as the MIR cuts (see Remark 1) and **Default** also generates the MIR cuts. Notice that the above results also imply that the proposed exact separation algorithm for the CS inequalities contributes this improvement, i.e., it enables to find more CS/MIR cuts, as compared with the heuristic algorithm of SCIP. To further verify this, we perform another test where we do not generate other cuts in SCIP and compare the performance of using the exact separation algorithm (**EXACT**) and the "0.5" heuristic algorithm (**HEUR**) to find CS cuts. Table 2 reports the number cuts (found at the root node) and the gap improvement of using **EXACT** and **HEUR**. Apparently, using the exact separation algorithm, we can find more cuts than the "0.5" heuristic algorithm. As a result, we can also observe a clear gap improvement in this case.

Table 2: Performance comparison of using the exact separation algorithm and the "0.5" heuristic algorithm to find CS cuts.

$\lvert V\rvert$	EXACT		HEUR	
	Cuts	Gap	Cuts	Gap
20	66	71.70	42	69.22
30	122	70.05	75	67.62
40	185	66.55	109	64.30

In summary, our computational results show that (i) the CS cuts can strengthen the LP relaxation and improve the solution efficiency of solving the single-source network design problem; (ii) the proposed exact separation algorithm is much more effective in finding cuts, as compared with the "0.5" heuristic algorithm.

References

1. Atamtürk, A., Günlük, O.: Network design arc set with variable upper bounds. Networks **50**(1), 17–28 (2007)
2. Atamtürk, A., Rajan, D.: On splittable and unsplittable flow capacitated network design arc-set polyhedra. Mathematical Programming **92**(2), 315–333 (2002)
3. Chen, W.K., Dai, Y.H.: Online supplement for paper: on the mixed 0-1 knapsack set with a GUB constraint. Tech. rep., Beijing Institute of Technology (2020), http://math.bit.edu.cn/docs/20201119170158081652.pdf
4. Dahl, G., Stoer, M.: A cutting plane algorithm for multicommodity survivable network design problems. INFORMS Journal on Computing **10**(1), 1–11 (1998)
5. Dash, S., Günlük, O., Hildebrand, R.: Binary extended formulations of polyhedral mixed-integer sets. Mathematical Programming **170**(1), 207–236 (2018)
6. Gabrel, V., Knippel, A., Minoux, M.: Exact solution of multicommodity network optimization problems with general step cost functions. Operations Research Letters **25**(1), 15–23 (1999)
7. Gleixner, A., Maher, S.J., Fischer, T., Gally, T., Gamrath, G., Gottwald, R.L., Hendel, G., Koch, T., Lübbecke, M.E., Miltenberger, M., Müller, B., Pfetsch, M.E., Puchert, C., Rehfeldt, D., Schenker, S., Schwarz, R., Serrano, F., Shinano, Y., Weninger, D., Witt, J.T., Witzig, J.: The SCIP optimization suite 6.0. ZIB-Report (2018)
8. Ljubić, I., Putz, P., Salazar-González, J.J.: Exact approaches to the single-source network loading problem. Networks **59**(1), 89–106 (2012)
9. Lovász, L.: Graph theory and integer programming. In: Hammer, P.L., Johnson, E.L., Korte, B.H. (eds.) Discrete Optimization I, pp. 141–158. Elsevier, New York (1979)
10. Magnanti, T.L., Mirchandani, P., Vachani, R.: The convex hull of two core capacitated network design problems. Mathematical Programming **60**(1-3), 233–250 (1993)
11. Marchand, H., Wolsey, L.A.: The 0-1 knapsack problem with a single continuous variable. Mathematical Programming **85**(1), 15–33 (1999)
12. Marchand, H., Wolsey, L.A.: Aggregation and mixed integer rounding to solve MIPs. Operations Research **49**(3), 363–371 (2001)
13. Narisetty, A.K., Richard, J.P.P., Nemhauser, G.L.: Lifted tableaux inequalities for 01 mixed-integer programs: a computational study. INFORMS Journal on Computing **23**(3), 416–424 (2010)
14. Nemhauser, G.L., Wolsey, L.A.: A recursive procedure to generate all cuts for 0-1 mixed integer programs. Mathematical Programming **46**(1-3), 379–390 (1990)
15. Pochet, Y., Wolsey, L.A.: Production Planning by Mixed Integer Programming. Springer, New York (2006)
16. Richard, J.P., de Farias Jr, I.R., Nemhauser, G.L.: Lifted inequalities for 0-1 mixed integer programming: Basic theory and algorithms. Mathematical Programming **98**(1), 89–113 (2003)
17. Richard, J.P., de Farias Jr, I.R., Nemhauser, G.L.: Lifted inequalities for 0-1 mixed integer programming: Superlinear lifting. Mathematical Programming **98**(1), 115–143 (2003)
18. Salman, F.S., Ravi, R., Hooker, J.N.: Solving the capacitated local access network design problem. INFORMS Journal on Computing **20**(2), 243–254 (2008)
19. Wolter, K.: Implementation of cutting plane separators for mixed integer programs. Diploma thesis, Dissertation for Diploma Degree. Berlin: Technische Universität Berlin (2006)