Available online at www.sciencedirect.com
ScienceDirect
Discrete Optimization 5 (2008) 270–289
DISCRETE OPTIMIZATION
www.elsevier.com/locate/disopt

Higher-order cover cuts from zero–one knapsack constraints augmented by two-sided bounding inequalities

Hanif D. Sherali[a], Fred Glover[b,*]

[a] Grado Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, United States
[b] Leeds School of Business, University of Colorado, Boulder, CO 80309-0419, United States

Received 4 November 2005; accepted 1 February 2007
Available online 31 October 2007

Abstract

Extending our work on second-order cover cuts [F. Glover, H.D. Sherali, Second-order cover cuts, Mathematical Programming (ISSN: 0025-5610 1436-4646) (2007), doi:10.1007/s10107-007-0098-4. (Online)], we introduce a new class of *higher-order cover cuts* that are derived from the implications of a knapsack constraint in concert with supplementary two-sided inequalities that bound the sums of sets of variables. The new cuts can be appreciably stronger than the second-order cuts, which in turn dominate the classical knapsack cover inequalities. The process of generating these cuts makes it possible to sequentially utilize the second-order cuts by embedding them in systems that define the inequalities from which the higher-order cover cuts are derived. We characterize properties of these cuts, design specialized procedures to generate them, and establish associated dominance relationships. These results are used to devise an algorithm that generates all non-dominated higher-order cover cuts, and, in particular, to formulate and solve suitable separation problems for deriving a higher-order cut that deletes a given fractional solution to an underlying continuous relaxation. We also discuss a lifting procedure for further tightening any generated cut, and establish its polynomial-time operation for unit-coefficient cuts. A numerical example is presented that illustrates these procedures and the relative strength of the generated non-redundant, non-dominated higher-order cuts, all of which turn out to be facet-defining for this example. Some preliminary computational results are also presented to demonstrate the efficacy of these cuts in comparison with lifted minimal cover inequalities for the underlying knapsack polytope.

Keywords: 0–1 integer programming; knapsack cover inequalities; Surrogate constraints; Separation problems; Lifting procedures

1. Introduction

The class of *knapsack cover inequalities* (or *cover cuts*) introduced in [2,17], and Wolsey [32] have enjoyed a well-deserved reputation for being useful to improve the solution of 0–1 integer programming (IP) problems, both in pre-processing and in tightening relaxations (see, e.g., [14,25,27,31]). In a previous paper [12], we introduced a class of *second-order cover cuts* whose members strengthen the classical knapsack cover inequalities by additionally

* Corresponding author.
E-mail addresses: hanifs@vt.edu (H.D. Sherali), fred.glover@colorado.edu (F. Glover).

1572-5286/$ - see front matter
doi:10.1016/j.disopt.2007.02.002

considering an upper bound on the sum of variables. We significantly extend this work in the present paper by proposing a more general class of *higher-order cover cuts* that make it possible to exploit implications of a more extensive set of inequalities.

Our development in this paper is based on a knapsack constraint (1a) in concert with additional two-sided bounding inequalities (1b) and (1c), defined in terms of 0–1 variables x_j, $j \in N \equiv \{1, \ldots, n\}$:

$$\sum_{j \in N} a_j x_j \geq a_0 \tag{1a}$$

$$0 \leq \ell \leq \sum_{j \in N} x_j \leq u \tag{1b}$$

$$0 \leq \ell_i \leq \sum_{j \in N_i} x_j \leq u_i, \quad \forall i \in M \equiv \{1, \ldots, m\}, \tag{1c}$$

where the sets N_i, $i \in M$, constitute a partition of N. The a_j-coefficients in (1a) are real numbers, possibly having mixed signs. Consequently, our results also apply to (1a) in the form $\sum_{j \in N} a_j x_j \leq a_0$.

We also denote

$$X \equiv \{x \text{ binary: (1a)–(1c) hold true}\} \tag{2}$$

and without loss of generality, assume that

$$a_1 \geq a_2 \geq \cdots \geq a_n, \qquad \ell \geq \sum_{i \in M} \ell_i, u_i \leq \min\{|N_i|, u\}, \quad \forall i \in M, u \leq \sum_{i \in M} u_i,$$
$$\text{and} \quad \ell_i < u_i \text{ if } u_i = |N_i|, \forall i \in M, \tag{3}$$

because otherwise, we can accordingly modify the bounds in (1b), (1c), and in the last case, if $\ell_i = u_i = |N_i|$ for any $i \in M$, we can fix $x_j = 1, \forall j \in N_i$.

The second-order cover cuts of our previous work were based on the system consisting of (1a) and (1b), without the inclusion of the inequalities of (1c), and without the consideration of the lower bound ℓ in (1b) (although we showed how the basic cut generation process could handle such a lower bound). As in the case of the second-order cuts, our work here is motivated by the proposals for exploiting systems of nested inequalities in 0–1 linear programming [11]. Our results are also related to the valid inequalities characterized in [20,23,28] for generalized upper-bound constrained knapsack problems, and to the pre-processing strategies analysed and exploited in [26,19] based on nested inequalities employed in conjunction with surrogate constraints in the context of 0–1 multidimensional knapsack problems. Our present paper demonstrates how to exploit and further tighten such nested inequality structures in the process of deriving a stronger class of proposed valid inequalities. Other related areas of application for such inequalities are identified in [18,30]. Furthermore, several extensions of the discussion provided in this paper are evident for more general nested-bounding structures in lieu of the separable restrictions (1c). Hanafi and Glover [19] and Glover and Sherali [13], for example, present some polyhedral analyses for such nested-constrained problems, but we postpone the consideration of such more complex structures in the light of the present work for future research.

Zeng and Richard [34] analyse a special case of X that includes (1a) along with the upper-bounding inequalities in (1c) (i.e., having $\ell = 0, u \equiv |N|$, and $\ell_i = 0, \forall i \in M$). For this structure, they describe a lifting mechanism for related generalized cover inequalities using certain novel super-additive multidimensional and lower approximating lifting functions to the underlying exact lifting function. Some numerical examples are presented to demonstrate that the resulting lifted cover inequalities can dominate those derived for the ordinary 0–1 knapsack polytope.

The constraints in (1c) accommodate two other special noteworthy cases. One arises where $u_i = 1$ for all $i \in M$, which captures the types of constraints found in multiple choice 0–1 problems that abound in practical applications. (See [28] for a characterization of cover cuts in this case, and a polynomial-time lifting of these into facets; also, see [23].) The second case arises where (1c) begins as a single constraint ($m = 1$) over a specified proper subset N_1 of N. To satisfy the condition that the sets N_i, $i \in M$, constitute a partition of N, it suffices to introduce the set $N_2 = N–N_1$ and add the redundant inequality $\sum_{j \in N_2} x_j \leq u_2 \equiv |N_2|$. An interesting use of such a representation occurs in the case where the constraint $\sum_{j \in N_1} x_j \geq \ell_1$, defined over N_1, is one derived as a second-order cover cut. Upon embedding it as indicated in (1c), the second-order cut can then be exploited more fully relative to other knapsack constraints (1a) accompanied by (1b). Such an approach is especially relevant where knapsack constraints arise from surrogate constraints designed to capture more aspects of the problem structure, as for example, by generating weighted combinations of parent constraints by using optimal dual variables from linear programming relaxations. Moreover, our derivations and results in the present paper can be applied to various other important special cases such as those involving multiple choice constraints, and knapsack constraints augmented with inequalities of the type (1b) that are derived by performing standard pre-processing or logical tests on the entire set of problem constraints. Hence, even when the assumed structure (1) is not inherently present in some parent 0–1 programming problem, it can be induced therefrom in order to derive implied valid inequalities of the type proposed herein.

The remainder of this paper is organized as follows. As a preliminary analysis, we begin in Section 2 by delineating certain pre-processing routines to verify whether $X = \varnothing$ or not, and to fix certain variables at 0 or 1 values, as possible. Thereafter, Section 3 describes a general LP-rounding-based procedure, as well as a specialized strongly polynomial stage-wise process to generate a higher-order cover cut. Following this, Section 4 addresses the issue of characterizing non-dominated higher-order cover inequalities, and establishes related dominance results. These properties are used in Section 5 to enumerate all non-dominated higher-order cover inequalities, as well as to solve related separation problems. Section 6 provides an illustrative example to elucidate the basic cut generation and separation routine ideas, and to exhibit the potential of generating significantly tighter inequalities using the structure X defined in Eq. (2), in comparison with non-dominated cover cuts that are implied by the knapsack polytope, or even by the augmented knapsack polytope that leads to the second-order cover inequalities of Glover and Sherali [12]. Section 7 then discusses a sequential lifting procedure for potentially further tightening the generated high-order cover inequalities, and establishes that this lifting can be conducted in polynomial time for the special case of unit-coefficient cuts. Some preliminary computational results are presented in Section 8 to demonstrate the efficacy of the proposed cuts in comparison with lifted minimal covers for the underlying knapsack polytope, and Section 8 provides a summary and some recommendations for future research.

2. Pre-processing routines

We begin by first describing an efficient polynomial-time process to check if $X = \varnothing$ or not. A flow-chart for the associated routine to accomplish this, denoted FEAS(X), is presented in Fig. 1, which returns FEAS(X) = TRUE if $X \neq \varnothing$, and FEAS(X) = FALSE otherwise. The following notation pertains to this procedure.

$$q = \text{Number of (counter for) elements selected from } N, \tag{4a}$$

$$q_i = \text{Number of (counter for) elements selected from } N_i, \forall i \in M. \tag{4b}$$

$$\Sigma = \text{Sum of } a_j\text{-coefficients for the selected indices/elements.} \tag{4c}$$

$$IN(j) = \text{Index } i \in M \text{ for which } j \in N_i, \forall j \in N. \tag{4d}$$

Essentially, based on (3), the method initially selects the first ℓ_i indices from each set N_i having the largest a_j-values. If the total number selected is less than ℓ, the method then continues to sequentially pick the smallest indexed unselected element subject to the upper-bounding restrictions in (1c). If the resultant sum of the selected a_j-coefficients is at least a_0, we have FEAS(X) = TRUE. Otherwise, we continue the sequential selection of admissible elements subject to the upper-bounding restrictions in (1b) and (1c), until we either obtain the sum of selected a_j-coefficients being greater than or equal to a_0, whence FEAS(X) = TRUE, or discover that (1a) is unsatisfiable subject to (1b) and (1c) (hence, FEAS(X) = FALSE). Evidently, given the ordered lists N and $N_i, \forall i \in M$, along with pointers between them (including $IN(j), \forall j \in N$), this process can be implemented in $O(n)$ steps (4d).

Next, we examine how we can augment the foregoing routine to ascertain whether we can *a priori* fix a variable at 0 or 1, and accordingly then, eliminate it from the problem.

Naturally, whenever we fix a variable at 0 or 1, we restructure (1a)–(1c), ensuring (3), and accordingly re-define X in (2). The basic idea here is that for any $j \in N$,

$$\text{if FEAS}(X \cap \{x : x_j = 0\}) = \text{FALSE, then we can fix } x_j = 1 \quad \text{and} \tag{5a}$$

$$\text{if FEAS}(X \cap \{x : x_j = 1\}) = \text{FALSE, then we can fix } x_j = 0. \tag{5b}$$

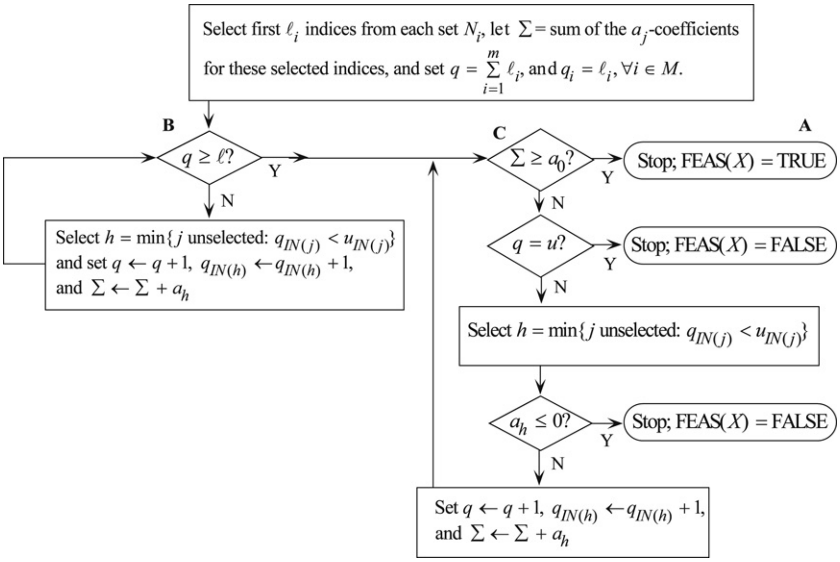


Fig. 1. Flow-chart for routine FEAS(X).

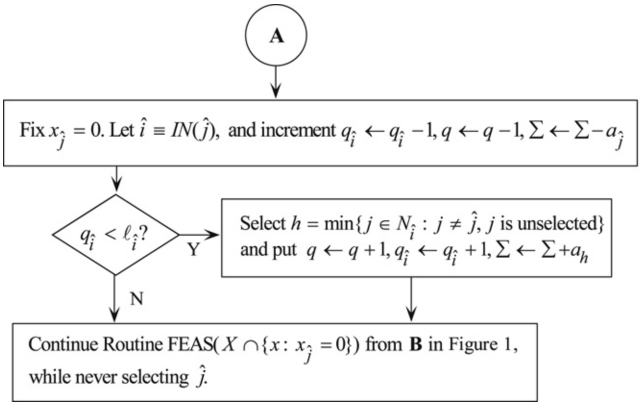


Fig. 2. Pre-processing routine for possibly fixing $x_{\hat{j}} \equiv 1$, for $\hat{j} \in N^+$.

To implement this concept more efficiently than simply running the routine FEAS($\cdot$) from scratch, define the sets

$$N^+ = \{j \in N : j \text{ was selected by FEAS}(X) \text{ at termination}\}, \quad \text{and} \quad N^- \equiv N - N^+. \tag{6}$$

Then, we need to check if x_j can be possibly fixed at 1 only for $j \in N^+$, and likewise, if x_j can be possibly fixed at 0 only for $j \in N^-$. Let A, B, and C be the junctures designated in Fig. 1 for the routine FEAS(X). Then Figs. 2 and 3 link into the structure of this routine for testing whether we can fix a given $x_{\hat{j}} \equiv 1$ for $\hat{j} \in N^+$, or a given $x_{\hat{j}} \equiv 0$ for $\hat{j} \in N^-$, respectively, based on the event that these routines return an indication of FALSE according to (5a) and (5b). The logic for these routines is similar to that for FEAS(X). For example, Fig. 2 commences with the information available at juncture A in Fig. 1, fixes $x_{\hat{j}} = 0$ from its current value of 1, then first checks if this results in $q_{\hat{i}} < \ell_{\hat{i}}$, where $\hat{i} \equiv IN(\hat{j})$. If so, it selects the best (smallest) admissible unselected index from $N_{\hat{i}}$, and then continues the feasibility routine from juncture B as before. The logic in Fig. 3 is similar.

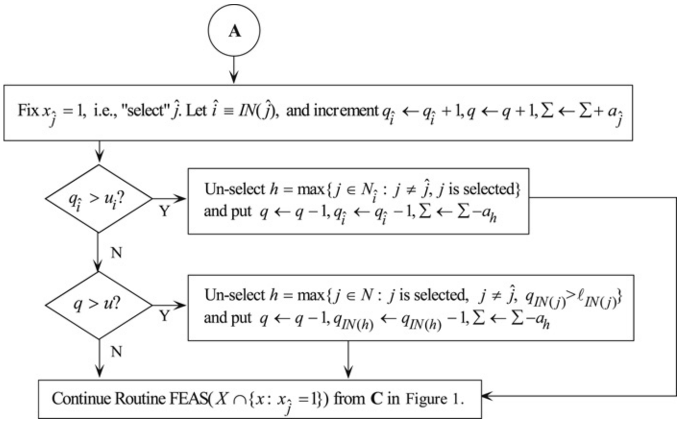


Fig. 3. Pre-processing routine for possibly fixing $x_{\hat{j}} \equiv 0$, for $\hat{j} \in N^-$.

3. Parametric procedure for generating higher-order cover (HOC) inequalities

In this section, we describe a parametric polynomial-time procedure for generating a valid inequality that is implied by X and is of the following form, for any given $J \subseteq N$, $J \neq \varnothing$:

$$\sum_{j \in J} x_j \geq p, \tag{7}$$

where

$$p \equiv \min\left\{\sum_{j \in J} x_j : x \in X\right\}. \tag{8}$$

Note that (7) extends the cover inequalities for knapsack polytopes to the more general set X. In the case of knapsack polytopes (viewed in the complemented variable space), the inequalities of the type (7) are essentially *rank-inequalities* based on an underlying independence system (see [21] for a general discussion of rank-inequalities for independence systems, and also, [3,7,22] for related discussions in the respective contexts of independent sets in graphs, set covering polytopes, and packing designs). However, because of the presence of both lower and upper bounds on sums of subsets of variables in (1b) and (1c), which precludes a direct characterization of feasible solutions to X in terms of an independence system, and also, as motivated in Section 1, since (7) can be derived as further implications of *second-order cover* inequalities of Glover and Sherali [12] that are generated using the pair of parent restrictions $\sum_{j \in N} a_j x_j \geq a_0$ and $\sum_{j \in N} x_j \leq u$ in binary variables, we shall refer to (7) as a *higher-order cover* (***HOC***) *inequality*.

Notationally, given any $J \subseteq N$, define the following for the sake of convenience in reference:

$$NJ \equiv N - J, \qquad J_i \equiv J \cap N_i, \quad \forall i \in M, \quad \text{and} \quad NJ_i \equiv N_i - J = NJ \cap N_i, \quad \forall i \in M. \tag{9}$$

The strongly polynomial-time routine **CUT(*J*)** that we design below for deriving (7) via (8) takes $J \subseteq N$, $J \neq \varnothing$ as an input, and produces p along with an optimal solution x^* to problem (8) as an output.

As an initial step, CUT(J) solves the following problem, whose optimal value is prescribed in closed form by Proposition 1 below.

$$p_0 = \text{Min}\left\{\sum_{j \in J} x_j : \text{(1b) and (1c)}, x \text{ binary}\right\}. \tag{10}$$

Proposition 1. *Consider problem* (10) *and define*

$$\hat{p}_0 = \sum_{i \in M} \max\{0, \ell_i - |NJ_i|\}, \quad \textit{and} \quad Q = \sum_{i \in M} \min\{|NJ_i|, u_i\}. \tag{11}$$

Then

$$p_0 = \begin{cases} \hat{p}_0 & \text{if } \hat{p}_0 + Q \geq \ell \\ \ell - Q, & \text{otherwise}. \end{cases} \tag{12}$$

Proof. Observe that for each $i \in M$, if $|NJ_i| < \ell_i$, then we will necessarily need to select at least $\ell_i - |NJ_i|$ indices from J_i to set the corresponding $x_j = 1$ in any feasible solution to (10). Defining $\hat{p}_0$ as in (11), we therefore have that $p_0 \geq \hat{p}_0$. Now, for each $i \in M$, while satisfying (1c), we can set up to $\min\{|NJ_i|, u_i\}x_j$-variables equal to one using indices $j \in NJ_i$. Hence, defining Q as in (11) and noting (1b) and that $\min\{|NJ_i|, u_i\} = |NJ_i|$ if $|NJ_i| < \ell_i$, if $\hat{p}_0 + Q \geq \ell$, then $p_0 = \hat{p}_0$. Otherwise, if $\hat{p}_0 + Q < \ell$, we will need to set $\ell - Q - \hat{p}_0$ additional x_j-variables at one for $j \in J$, thereby establishing (12). □

Next, consider solving the following problem F(p), parameterized by p, for any value $p \geq p_0$. Note that, by Proposition 1, this problem has an optimum for all $p \geq p_0$.

$$\mathbf{F(p)} : \text{Maximize } \left\{ \sum_{j \in N} a_j x_j : \text{(1b) and (1c)}, \sum_{j \in J} x_j \leq p, x \text{ binary} \right\}. \tag{13}$$

Notationally, henceforth, we denote the optimal value of any optimization problem P by v[P]. Now, observe that the optimal value p for defining the HOC inequality (7) as predicated by (8) is given by the smallest integer $p \geq p_0$ for which $v[\mathrm{F}(p)] \geq a_0$. Hence, in order to solve (8), we can begin with $p = p_0$, and then increment p by one successively until we get $v[\mathrm{F}(p)] \geq a_0$. This is reminiscent of a classical dynamic programming trick to solve (8) by reversing the role of the objective function and the structural knapsack constraint, and resembles Zemel's [33] approach for lifting knapsack cover inequalities in polynomial time (see also [4]). We show below how such a specialized scheme for solving problem (8) can be implemented in strongly polynomial time of complexity $O(n|J|)$. As a point of interest, let us first show that problem (13) can alternatively be solved as a bounded-variable network flow program in an approach of the foregoing type, and that problem (8) can also be solved by rounding up the objective value of its linear programming (LP) relaxation, which implies that inequality (7) is a rank-one Chvatal–Gomory cut (see Nemhauser and Wolsey [24]).

Toward this end, let $\bar{X}$ denote the continuous relaxation of X, let $\bar{F}(p)$ denote problem (13) with the binary restrictions on x replaced by $0 \leq x_j \leq 1, \forall j \in N$, and consider the following two results.

Proposition 2. *Problem $\bar{F}(p)$ is a bounded-variable network flow program.*

Proof. Consider the transformations:

$$s_2 = \sum_{j \in N} x_j - \ell, \quad \text{and} \quad z_i = \sum_{j \in N_i} x_j - \ell_i, \quad \forall i \in M.$$

Then, the constraints of problem $\bar{F}(p)$ can be equivalently restated as follows, where s_1 is the slack in $\sum_{j \in J} x_j \leq p$:

$$-\sum_{j \in J} x_j - s_1 = -p$$

$$s_1 - \sum_{j \in NJ} x_j + s_2 = p - \ell$$

$$\sum_{j \in N_i} x_j - z_i = \ell_i, \quad \forall i \in M$$

$$s_1 \geq 0, \qquad 0 \leq s_2 \leq u - \ell, \qquad 0 \leq z_i \leq u_i - \ell_i, \quad \forall i \in M, \qquad 0 \leq x_j \leq 1, \quad \forall j \in N.$$

This transformed constraint set now displays a (totally unimodular) bounded-variable network flow programming structure (see Bazaraa et al. [5], for example). □

Proposition 3. *The optimal value of problem (8) is given by rounding up its LP relaxation value.*

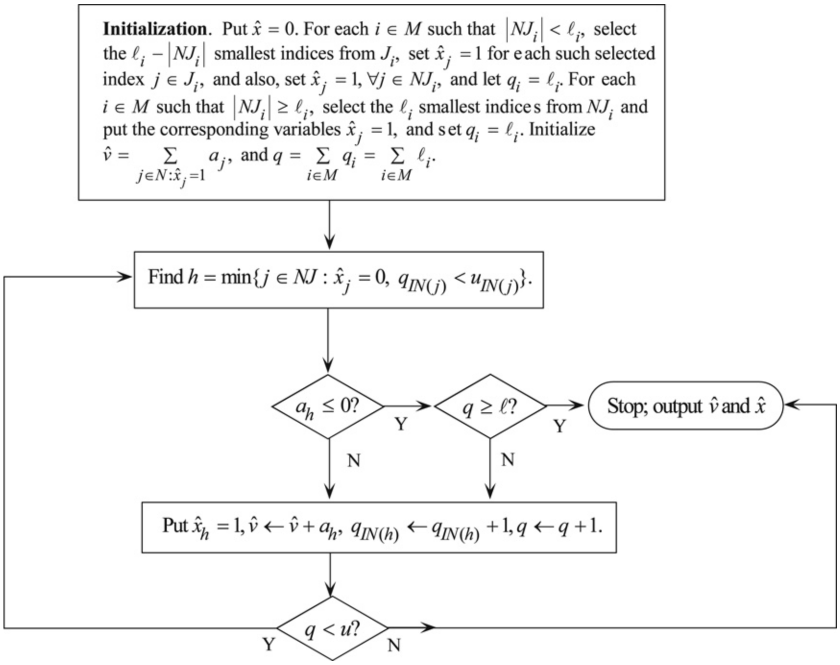


Fig. 4. Solution of problem $\text{F}(p_0)$, for the case of $p_0 = \hat{p}_0$.

Proof. As noted above, if p is given by (8), then $v[F(p)] \geq a_0$ and $v[F(p-1)] < a_0$. By Proposition 2, therefore, we have that $v[\bar{F}(p-1)] < a_0$. This means that $p \geq \min\{\sum_{j \in J} x_j : x \in \bar{X}\} > (p-1)$, where the latter inequality holds because otherwise, if there exists an $\hat{x} \in \bar{X}$ such that $\sum_{j \in J} \hat{x}_j \leq (p-1)$, then $\hat{x}$ would be feasible to $\bar{F}(p-1)$ with $v[\bar{F}(p-1)] \geq a_0$, a contradiction. Hence, we get $\lceil \min\{\sum_{j \in J} x_j : x \in \bar{X}\} \rceil = p$. □

Remark 1. Notwithstanding Proposition 3, which does have significance in practical implementations although the complexity of this approach is in general $O(n^3 L)$, where L is the number of binary bits required to store the data (see Bazaraa et al. [5], for example), we continue to describe a specialized parametric $O(n|J|)$ algorithm for solving problem (8). This procedure commences by solving problem (13) for $p = p_0$, and then utilizes an efficient scheme to iteratively update the solution to $\text{F}(p+1)$ from an optimal solution to $\text{F}(p)$, for $p \geq p_0$, while $v[\text{F}(p)]$ remains less than a_0. In the light of Proposition 2, such a process could alternatively be implemented as a sensitivity analysis-based update to the optimal solution for the network flow program $\bar{F}(p)$ as p is iteratively increased (see [1,11] for related algorithmic approaches). However, the specialized routine described below accomplishes this task in a more efficient direct manner with a lower complexity order, and is hence of interest in its own right. Nevertheless, it is worth noting here that certain more general overlapping or multilayer nested structured-bounding constraints have been identified by Glover [11] to be solvable as network flow programs. When the set X possesses such a more general structure, the aforementioned sequential network flow programming approach can then be utilized for generating similar cover inequalities. □

Accordingly, first consider the solution to problem $\text{F}(p_0)$. As an initial case, suppose that $p_0 = \hat{p}_0$ as defined in Proposition 1. In this case, for each $i \in M$ such that $|NJ_i| < \ell_i$, we must pick some $\ell_i - |NJ_i|$ indices from J_i to set the corresponding $x_j = 1$ in solving $\text{F}(p_0)$. Naturally, we select the smallest $\ell_i - |NJ_i|$ indices from each such set J_i. Since this accounts for all the permissible $\hat{p}_0$ indices that can be selected from J, the remaining selected indices for setting the corresponding x_j-variables to one at optimality must come from NJ, including all $j \in NJ_i$ if $|NJ_i| < \ell_i, i \in M$, and at least the ℓ_i smallest indices from NJ_i if $|NJ_i| \geq \ell_i, i \in M$, plus additional indices from NJ that are selected in increasing order (to maximize $\sum_{j \in N} a_j x_j$) while respecting the bounds in (1b) and (1c). This process is presented in Fig. 4.

Next, consider the case where $\hat{p}_0 + Q < \ell$ in Proposition 1, so that we then have $p_0 = \ell - Q$. Note that since we will be selecting p_0 indices from J in this case, and noting that $\ell = p_0 + Q$, we must select at least Q indices

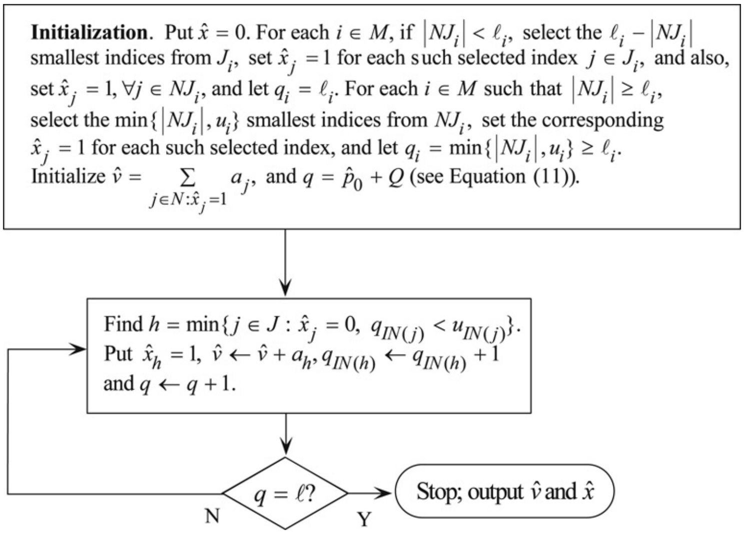


Fig. 5. Selection of problem $F(p_0)$ for the case of $p_0 = \ell - Q$.

from NJ (where *selecting an index* means setting the corresponding variable $x_j = 1$ at optimality in $F(p)$). However, by (11), $Q = \sum_{i \in M} \min\{|NJ_i|, u_i\}$. Hence, for each $i \in M$, we select the smallest $\min\{|NJ_i|, u_i\}$ indices from NJ_i, and also, the smallest $\ell_i - |NJ_i|$ indices from J_i for those $i \in M$ for which $|NJ_i| < \ell_i$ as above. This accounts for a total of $\hat{p}_0 + Q$ selected indices. Since $\hat{p}_0 + Q < \ell$, and $p_0 = \ell - Q$, we must now select the best possible $\ell - \hat{p}_0 - Q$ additional indices from J, subject to the bounds in (1b) and (1c). This process is described in the flow-chart of Fig. 5.

Having solved $F(p_0)$, if we have that $v[F(p_0)] \geq a_0$, then $p = p_0$ in problem (8). Otherwise, beginning with $p = p_0$, at any stage in the proposed sequential process, given an optimum $\hat{x}$ to $F(p)$ such that $v[F(p)] < a_0$, we update this solution to an optimum for $F(p + 1)$ as prescribed by Proposition 4 below. To present this result, consider the following notation, where $\hat{x}$ solves problem $F(p)$ for a given $p \geq p_0$:

$$q_i = \sum_{j \in N_i} \hat{x}_j, \quad \forall i \in M, \quad \text{and} \quad q = \sum_{j \in N} \hat{x}_j = \sum_{i \in M} q_i. \tag{14a}$$

$$j(\max) = \max\{j \in NJ : \hat{x}_j = 1\} \tag{14b}$$

$$j(i) = \begin{cases} \min\{j \in J_i : \hat{x}_j = 0\} & \text{if } \sum_{k \in J_i} \hat{x}_k < u_i \\ 0, & \text{otherwise}, \forall i \in M, \end{cases} \tag{14c}$$

$$j[i] = \begin{cases} \max\{j \in NJ_i : \hat{x}_j = 1\} & \text{if } q_i = u_i \\ j(\max) & \text{if } q_i < u_i, \text{and either } q = u \text{ or } a_{j(\max)} \leq 0 \\ n + 1, & \text{otherwise (where } a_{n+1} \equiv 0), \forall i \in M, \end{cases} \tag{14d}$$

$$\alpha(i) = \begin{cases} a_{j(i)} - a_{j[i]} & \text{if } j(i) \neq 0 \\ 0, & \text{otherwise}, \forall i \in M \end{cases} \tag{14e}$$

and

$$i^* = \arg\max\{\alpha(i) : i \in M\}. \tag{14f}$$

Proposition 4. *Suppose that $\hat{x}$ solves problem $F(p)$ for some given $p \geq p_0$ such that $v[F(p)] < a_0$, and consider the definitions (14a)–(14f). Then, given that $X \neq \emptyset$, we must have $\alpha(i^*) > 0$, and moreover, an optimal solution to problem $F(p + 1)$ is obtained by revising $\hat{x}$ by setting $\hat{x}_{j(i^*)} = 1$, and $\hat{x}_{j[i^*]} = 0$ (if $j[i^*] \neq n + 1$), with an optimal objective value $v[F(p + 1)] = v[F(p)] + \alpha(i^*)$.*

Proof. First of all, note that the constraint $\sum_{j \in J} x_j \leq p + 1$ must be active at optimality in $F(p + 1)$, because otherwise, the same solution $\hat{x}$ solves $F(p'), \forall p' \geq p$, implying that $X = \emptyset$, a contradiction to our standing assumption of $X \neq \emptyset$. Furthermore, by the nature of problem F(·), an optimal solution is composed by appropriately selecting indices in increasing order from each set J_i and NJ_i, $i \in M$, for setting the corresponding variable to one. In particular, as far as the set J is concerned in composing an optimal solution to F($p+1$), we set $\hat{x}_j = 1$ for the same indices $j \in J$ as in the optimum found for F(p), plus then, set an additional variable $\hat{x}_j = 1$ for some $j \in J$. The candidate index for this additional variable from each set J_i, $i \in M$, is identified by $j(i)$ via (14c), where no such candidate is possible ($j(i) \equiv 0$) in case $\sum_{k \in J_i} \hat{x}_k$ already equals u_i. Now, if we set $\hat{x}_{j(i)} = 1$ for any particular $i \in M$, then we might need to set some $\hat{x}_r = 0$ for $r \in NJ$. This index r is identified by (3) as $j[i]$. If $q_i = u_i$, then $j[i] = \max\{j \in NJ_i : \hat{x}_j = 1\}$. Otherwise, if $q_i < u_i$ but $q = u$, then $j[i] = j(\max)$. Also, if $q_i < u_i$ and $q < u$, but $a_{j(\max)} \leq 0$, then we set $\hat{x}_{j(\max)} = 0$; hence, again, $j[i] = j(\max)$ in this case. Else, no $\hat{x}_j$-variable for $j \in NJ$ needs to be switched to zero from one, whence we let $j[i] \equiv n+1$, where $a_{n+1} \equiv 0$. Eq. (14e) then computes the gain in objective value by making $\hat{x}_{j(i)} = 1$ and $\hat{x}_{j[i]} = 0$ (if $j[i] \neq n+1$), whenever $j(i) \neq 0$. Note that the best gain given by $\alpha(i^*)$ as defined by (14f) must be positive, because otherwise, $\hat{x}$ itself would solve F($p+1$), and as argued above, this would mean that $X = \emptyset$, a contradiction. Therefore, revising $\hat{x}$ by setting $\hat{x}_{j(i^*)} = 1$ and $\hat{x}_{j[i^*]} = 0$ (if $j[i^*] \neq n+1$) yields an optimal solution to F($p+1$). □

Based on Propositions 1 and 4, we adopt the following routine, CUT(J), to determine an HOC inequality (7), given any $J \subseteq N$, $J \neq \emptyset$.

Routine CUT(J), for $J \subseteq N, J \neq \emptyset$.

Initialization. Determine p_0 via Eq. (2) of Proposition 1. Set $p = p_0$ and solve F(p_0) using the procedure of Fig. 4 or Fig. 5, depending on whether $p_0 = \hat{p}_0$ or $p_0 = \ell - Q$, respectively. Let $\hat{x}$ be the solution obtained with $\hat{v} \equiv v[\mathrm{F}(p_0)]$.

Step 1. If $\hat{v} \geq a_0$, proceed to Step 2. Else, set $p \leftarrow p+1$ and use Proposition 4 to revise the current $\hat{x}$ to an optimal solution to problem F($p+1$), with $\hat{v}$ being the updated objective function value. Repeat Step 1.

Step 2. Set $x^* = \hat{x}$. Then x^* solves problem (8) with the current p being the optimal objective value, and the associated HOC inequality is given by (7). Prescribe $\{p$ and $x^*\}$ as an output to CUT(J).

Proposition 5. *Routine CUT(J) is of complexity $O(n|J|)$.*

Proof. Given the ordered lists J and NJ, finding p_0 by Proposition 1, solving F(p_0) by the procedure of Fig. 4 or 5, and updating the solution to F(·)$O(|J|)$ times are of respective complexity orders: $O(n)$, $O(n)$, and $O(n|J|)$. □

4. Characterization of non-dominated higher-order cover inequalities

Consider a pair of HOC inequalities for some non-empty subsets J and J' of N:

$$\sum_{j \in J} x_j \geq p \tag{15a}$$

$$\sum_{j \in J'} x_j \geq p'. \tag{15b}$$

We say that (15a) *dominates* (15b) over the unit hypercube if either

(i) $J \subseteq J'$ and $p \geq p'$(with at least one relation strict), or (16a)

(ii) $J = J' \cup \{j\}$ for some $j \in N - J'$and$p = p' + 1$. (16b)

Note that if (15a) dominates (15b), then (15a) implies (15b) over the unit hypercube, i.e.,

$$\min\left\{\sum_{j \in J'} x_j : \sum_{j \in J} x_j \geq p, 0 \leq x_j \leq 1, \forall j \in N\right\} \geq p'. \tag{17}$$

Moreover, we say that a given HOC inequality (15a) is *non-dominated* if $p \geq 1$ (i.e., it is non-trivial), and if there does not exist another valid HOC inequality that dominates it (over the unit hypercube). We are interested in two related aspects: (a) solving a *separation problem* using HOC inequalities, i.e., generating a non-dominated HOC inequality that deletes some given underlying LP relaxation solution $\bar{x}$ to a parent problem, and (b) prescribing the entire set of non-dominated HOC inequalities. Our focus in this section is to provide a principal characterization of non-dominated inequalities, which will enable us in the following section to design a sequential algorithm for generating the set of non-dominated HOC inequalities that automatically *fathoms* (skips over) several dominated members of this class of valid inequalities. This will also prompt one approach to solve the aforementioned separation problem.

Toward this end, consider the following result.

Proposition 6. *Consider any $J \subseteq N$, $J \neq \emptyset$ such that the routine CUT(J) produces the HOC inequality (15a), yielding the output $\{p$ and $x^*\}$. Then, (15a) dominates all HOC inequalities that are based on sets J' of the type:*

$$J' = J \cup \Delta J, \quad \text{where } \Delta J \subseteq \{j \in NJ : x_j^* = 0\}, \Delta J \neq \emptyset. \tag{18}$$

Proof. Consider any J' of the type indicated by (18). By (8), we have,

$$p' = \min \left\{ \sum_{j \in J'} x_j : x \in X \right\}. \tag{19}$$

But since x^* solves for $p = \min\{\sum_{j \in J} x_j : x \in X\}$, we get from (18) that x^* is feasible to problem (19) with an objective value of p. Hence, $p' \leq p$, and since $J \subset J'$, we get by (16a) that (15a) dominates (15b). □

Corollary 6.1. *Under the condition of Proposition 6, suppose that the set $\{j \in NJ : x_j^* = 1\}$ is a singleton, given by $\{j^*\}$. Then, any possibly non-dominated HOC inequality that is based on a set J' of the type $J' = J \cup \Delta J$, for $\Delta J \subseteq NJ$, $\Delta J \neq \emptyset$, must necessarily include the index j^*.*

Proof. Follows directly from Proposition 6. □

5. Generating the set of non-dominated HOC inequalities and solving separation problems

To begin with, we first propose in Section 5.1 an algorithm for implicitly enumerating all possible sets $J \subseteq N$ in a suitable sequential order with the intent of identifying the set of all non-dominated HOC inequalities. Next, in Section 5.2, we shall address the issue of generating only a particular member of this class of non-dominated HOC inequalities in order to delete a given relaxation solution, i.e., demonstrate how to solve a suitable associated separation problem.

5.1. Generating all non-dominated HOC inequalities

Consider a binary enumeration tree that is based on the dichotomy that any index $j \in N$ either belongs to J or not. Following the recipe prescribed by Glover [10] (see also [9]), we manage an implicit enumeration process in a depth-first fashion on the aforementioned binary tree via a *partial solution* list *PS*. At any stage in this enumeration process, the list *PS* contains a subset of indices of N, where each included index j appears as $+j$ or $-j$, indicating respectively that j is confined to belong to J or to NJ. Moreover, this index $\pm j$ is *underlined* $(\pm \underline{j})$ if the complement branch corresponding to the node pertaining to the subset of *PS* up to this underlined element has already been *fathomed* (eliminated from further consideration). Following the schema of Glover [10] and Geoffrion [9], whenever we *fathom* *PS*, we identify the right-most non-underlined element in *PS* (if no such element exists, we set $PS = \emptyset$ and terminate), complement and underline it, and delete all elements to its right.

Now, at any step in this process, consider a partial solution list $PS \neq \emptyset$. Let the set J *induced by PS* be defined as

$$J = \{j : +j \text{ or } +\underline{j} \text{ belongs to } PS\}. \tag{20}$$

Let $\widehat{\textbf{CUT}}(\boldsymbol{PS})$ denote the routine CUT(J) as applied to the set J induced by *PS*. Note that for any list *PS*, if we identify the right-most positive element and drop all the (negative) elements appearing to the right of it to obtain a list *PS*′ (where $PS' = \emptyset$ if *PS* has only negative elements), then both *PS* and *PS*′ induce the same set J. To avoid the duplication of executing $\widehat{\text{CUT}}(PS)$ given that $\widehat{\text{CUT}}(PS')$ has already been previously run, we ensure that the final element in any *PS* for which $\widehat{\text{CUT}}(PS)$ is invoked is always positive (underlined or not). This also ensures that the routine $\widehat{\text{CUT}}(PS)$ is called with the corresponding induced set $J \neq \emptyset$.

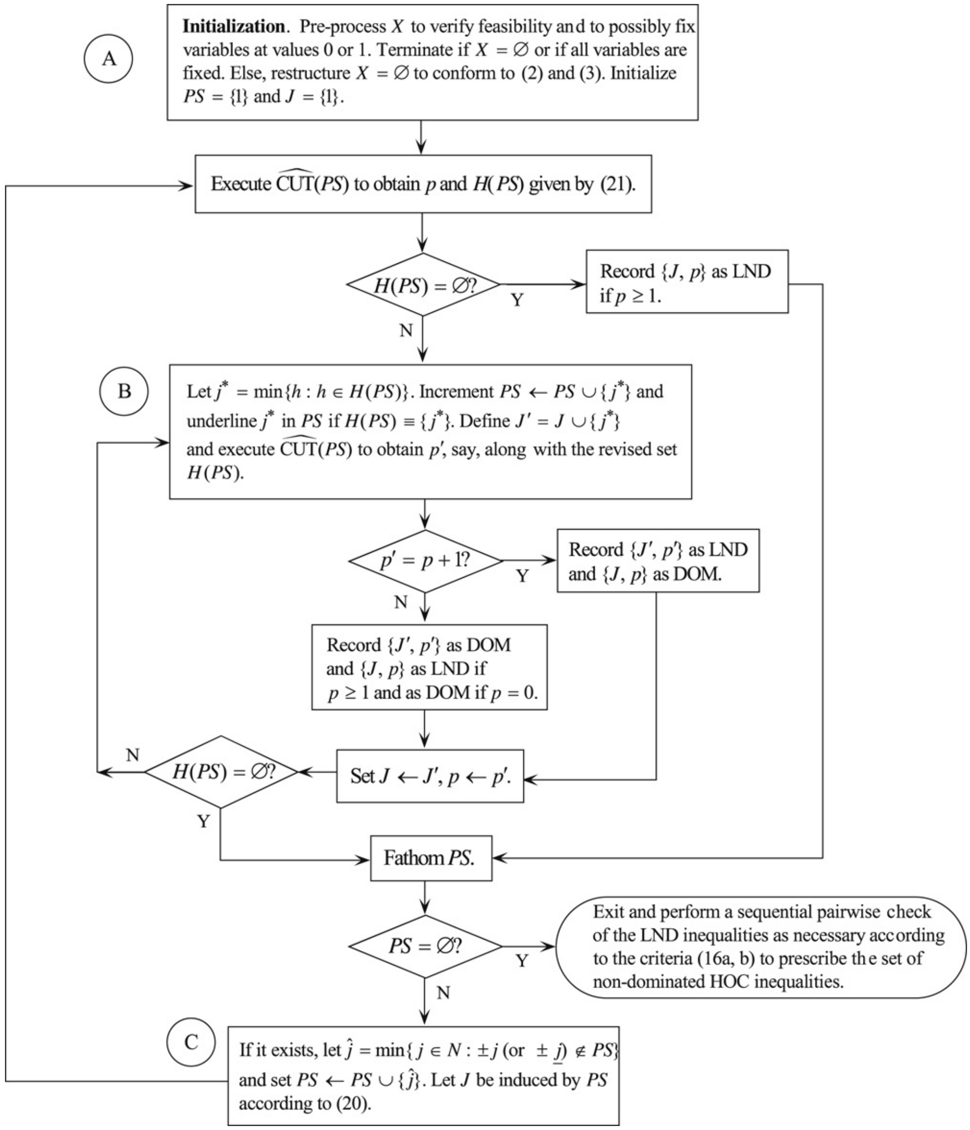


Fig. 6. Flow-chart for generating the set of non-dominated HOC inequalities.

Furthermore, suppose that given a list PS having an induced set $J \neq \varnothing$, we execute $\widehat{\text{CUT}}(PS)$ and obtain the associated HOC inequality (15a) with output $\{p$ and $x^*\}$. Define

$$H(PS) = \{j \in NJ \equiv N - J : x_j^* = 1 \text{and} -j \,(\text{or} -\underline{j}) \notin PS\}. \tag{21}$$

By Proposition 6, if $H(PS) = \varnothing$, then any further augmentation of PS would only produce induced sets J' for which the corresponding HOC inequality (15b) would be dominated by (15a). Hence, we can fathom PS. Otherwise, we *increment* PS by the operation $PS \leftarrow PS \cup \{j^*\}$, where $j^* = \min\{h : h \in H(PS)\}$, and where, by Corollary 6.1, we also underline j^* (i.e., $PS \leftarrow PS \cup \{\underline{j^*}\}$) in case $H(PS) \equiv \{j^*\}$ is a singleton set.

Fig. 6 presents a flow-chart for generating the set of non-dominated HOC inequalities, where each HOC inequality is recorded as $\{J, p\}$. (Redundant inequalities that are implied by the other constraints can be detected and eliminated via solving a linear program if necessary (see Bazaraa et al. [5], for example).) Note that at the point of generation and subsequent incrementing of the current list PS, certain HOC inequalities are identified as being dominated (labeled **DOM**) while others are labeled as "*locally non-dominated*," or **LND**, being candidates for the final set of non-dominated inequalities. This latter set is obtained through a sequential pairwise examination of the LND inequalities, which deletes any that is found to be dominated by another according to the dominance criteria (16a) and (16b).

5.2. Solving separation problems

Suppose that we are given a fractional solution $\bar{x} \equiv (\bar{x}_j, j \in N)$ as (part of) an optimum to some continuous relaxation to an underlying parent 0–1 program. We propose below two methods to possibly generate a non-dominated HOC inequality (7) to delete $\bar{x}$. The following section illustrates these procedures on a numerical example.

5.2.1. Method 1

In this approach, we first determine an HOC inequality, which is at least LND, in order to possibly delete $\bar{x}$ by employing the procedure given in the flow-chart of Fig. 6 with the following modifications. Noting that we desire to determine an inequality (7) that has a negative (hopefully, minimum) value of $\sum_{j \in J} \bar{x}_j - p$, we increment the partial solution list *PS* at the respective junctures denoted A, B, and C in Fig. 6 by selecting the corresponding indices to increment *PS* as follows:

$$\text{A}: PS = J = \text{arglexmin}\,\{\bar{x}_j, j: j \in N\}$$
$$\text{B}: j^* = \text{arglexmin}\,\{\bar{x}_h, h: h \in H(PS)\}$$
$$\text{C}: \hat{j} = \text{arglexmin}\,\{\bar{x}_j, j: \quad j \in N, \text{and } \pm j (\text{or } \pm \underline{j}) \notin PS\}.$$

The procedure is terminated at the first instance when a violated (LND) HOC inequality is detected, if at all. Alternatively, since the procedure of Fig. 6 prioritizes the consideration of indices to increment *PS* of the order of non-increasing a_j-values, which tends to focus on generating stronger HOC inequalities earlier, we could simply run the original method of Fig. 6 until a first violated (LND) HOC inequality is detected, if at all. (In practice, an upper bound can be imposed on the total number of LND inequalities generated in order to control the solution effort.)

In either case, having obtained an LND inequality $\sum_{j \in J} x_j \geq p$ that deletes $\bar{x}$, we can further ascertain if there exists a tighter dominating inequality as follows. Let x^* be the optimum obtained for problem (8) corresponding to this given HOC inequality. Following the proof of Proposition 6, for each $k \in J$ such that $x_k^* = 0$ in turn, we examine if minimum $\{\sum_{j \in J - \{k\}} x_j : x \in X\} = p$, and if so, then we have found another dominating valid HOC inequality $\sum_{j \in J - \{k\}} x_j \geq p$ and we repeat. Otherwise, based on Proposition 6, we next consider each $k \in NJ$ such that $x_k^* = 1$ in turn, and check if min $\{\sum_{j \in J \cup \{k\}} x_j : x \in X\} = p + 1$, and if so, then again we have found another dominating valid HOC inequality $\sum_{j \in J \cup \{k\}} x_j \geq p + 1$ (see (16b)). Else, we output the resulting non-dominated HOC cut at hand. If the original LND inequality is indeed non-dominated, this process will determine it to be such in polynomial time with complexity $O(n^3)$; else, it will progressively generate tighter cuts, each with the same complexity. Section 6 illustrates both these alternative schemes for implementing Method 1.

5.2.2. Method 2

In this approach, we directly formulate a separation problem in order to generate a non-dominated violated HOC inequality. Note that we would like to determine an index set J and a right-hand-side value p for (7) in order to minimize $\sum_{j \in J} \bar{x}_j - p$ in an attempt to drive it negative. Toward this end, in the spirit of Crowder et al. [8], define binary variables y_j, $j \in N$, such that $y_j = 1$ if j is selected to lie in J, and equals zero otherwise. Then, we would like to find a $J \subseteq N$ and an integer $p \geq 1$ such that $\sum_{j \in J} \bar{x}_j - p$ is minimized, while for the selected J and p, we have that $v[F(p-1)] \leq a_0 - 1$, as defined in (13). This would then imply that the resulting HOC inequality is valid and that $p \geq 1$ is as large as possible. Hence, we consider the *separation problem*:

$$\textbf{SEP}: \text{Minimize} \left\{ \sum_{j \in N} \bar{x}_j y_j - p \colon g(y, p-1) \leq a_0 - 1,\ p \geq 1,\ p \text{ integer},\ y \text{ binary} \right\}, \tag{22}$$

where for a *given* binary y and integer $p \geq 1$, we have

$$g(y, p-1) \equiv \max \left\{ \sum_{j \in N} a_j x_j : \text{(1b) and (1c)}, \sum_{j \in N} x_j y_j \leq p - 1, x \text{ binary} \right\}. \tag{23}$$

Noting Proposition 2, $g(y, p-1)$ can be equivalently derived via its LP relaxation as follows, where we have designated specified dual variables against each constraint set.

$$g(y, p-1) = \text{Maximum} \sum_{j\in N} a_j x_j$$

$$\text{subject to} \sum_{j\in N} x_j \le u \quad (\lambda_1)$$

$$-\sum_{j\in N} x_j \le -\ell \quad (\lambda_2)$$

$$\sum_{j\in N_i} x_j \le u_i, \quad \forall i \in M \quad (\mu_{1i}, \forall i \in M)$$

$$-\sum_{j\in N_i} x_j \le -\ell_i, \quad \forall i \in M \quad (\mu_{2i}, \forall i \in M)$$

$$\sum_{j\in N} x_j y_j \le p-1 \quad (\gamma)$$

$$x_j \le 1, \quad \forall j \in N \quad (\theta_j, \forall j \in N)$$

$$x_j \ge 0, \quad \forall j \in N.$$

Writing the dual to the above program, we have,

$$g(y, p-1) = \text{Minimum} \left\{ u\lambda_1 - \ell\lambda_2 + \sum_{i\in M} u_i\mu_{1i} - \sum_{i\in M} \ell_i\mu_{2i} + (p-1)\gamma + \sum_{j\in N} \theta_j \right\} \tag{24a}$$

subject to

$$\lambda_1 - \lambda_2 + \mu_{1i} - \mu_{2i} + \gamma y_j + \theta_j \ge a_j, \quad \forall j \in N_i, \forall i \in M \tag{24b}$$

$$(\lambda_1, \lambda_2, \mu_{1i}, \forall i \in M, \mu_{2i}, \forall i \in M, \gamma, \theta_j, \forall j \in N) \ge 0. \tag{24c}$$

Now, the structural constraint in (22) holds true if and only if there exists a feasible solution to (24b) and (24c) for which the objective value in (24a) is less than or equal to $a_0 - 1$. This leads to the following equivalent reformulation of problem SEP.

$$\textbf{SEP}: \text{Minimize} \sum_{j\in N} \bar{x}_j y_j - p \tag{25a}$$

$$\text{subject to } u\lambda_1 - \ell\lambda_2 + \sum_{i\in M} u_i\mu_{1i} - \sum_{i\in M} \ell_i\mu_{2i} + (p-1)\gamma + \sum_{j\in N} \theta_j \le a_0 - 1 \tag{25b}$$

$$\lambda_1 - \lambda_2 + \mu_{1i} - \mu_{2i} + \gamma y_j + \theta_j \ge a_j, \quad \forall j \in N_i, \forall i \in M \tag{25c}$$

$$(\lambda_1, \lambda_2, \mu_{1i}, \forall i \in M, \mu_{2i}, \forall i \in M, \gamma, \theta_j, \forall j \in N) \ge 0,\ p \ge 1,$$
$$p \text{ integer, } y \text{ binary.} \tag{25d}$$

Observe that SEP is a nonlinear 0–1 mixed-integer program (MIP) because of the terms $(p-1)\gamma$ and γy_j in constraints (25b) and (25c), respectively. Whereas this problem can be solved to global optimality using the procedure described in Sherali and Tuncbilek [29], it possesses a special structure that we can exploit, in that for a fixed γ, it is a linear 0–1 MIP. Note that for $\gamma = 0$, since $g(y, p-1)$ via (24) is given by $P \equiv \max\{\sum_{j\in N} a_j x_j : \text{(1b) and (1c)}, 0 \le x_j \le 1, \forall j \in N\} \ge a_0$, this would yield infeasibility in (25), because (25) then essentially seeks a dual feasible solution to the foregoing problem P having an objective value less than or equal to $a_0 - 1$. Hence, we will have $\gamma > 0$ at optimality in (25). Observe also that for the special case of generating a separating inequality having $p = 1$, (25) can be solved as a single MIP by fixing γ at a sufficiently large value by virtue of the terms involving γ in (25b) and (25c). In general, denoting the optimal value of SEP for a fixed $\gamma \ge 0$ as $f(\gamma)$, problem SEP essentially seeks to solve $\inf\{f(\gamma): \gamma \ge 0\}$. This is a univariate "line search" problem for which we can parametrically search for an optimum γ value. However, since f is generally discontinuous in our context, and noting that it suffices to find a

$\bar{\gamma}$ such that $f(\bar{\gamma}) < 0$ to obtain a cut to delete $\bar{x}$, we can approximate this search by performing a single iteration of the quadratic-fit line search method described in Bazaraa et al. [6]. (A good trial value for γ in the light of (25c) might be $\gamma = \max\{a_j : j \in N\}$.) If the resultant best γ-value found in this process yields a negative objective value in problem SEP given by Eq. (25), we will have detected a violated HOC inequality. Moreover, to encourage the generation of a non-dominated inequality, we can replace zero values of $\bar{x}_j$ in (25a) by some small tolerance $\varepsilon > 0$. In any case, the resultant HOC inequality could be tightened to a non-dominated cut by invoking the procedure described in Section 5.2.1.

As an alternative implementation, denoting $\bar{f}(\gamma)$ as the optimal value of the continuous relaxation to (25), which is an LP for a fixed γ, we could further conserve effort by solving (perhaps approximately as above via one or two iterations of the quadratic-fit line search method) the problem to minimize $\{\bar{f}(\gamma) : 0 \le \gamma \le \bar{\gamma}\}$ for some sufficiently large upper bound $\bar{\gamma}$. The resultant γ-value could then be used in (25) to derive the final solution. We illustrate both these alternative approaches in the following section.

To conclude this section, it is interesting to see how SEP generalizes the separation problem solved by Crowder et al. [8] to generate a minimal cover for deleting $\bar{x}$ for the case of a knapsack inequality (1a) in the absence of (1b) and (1c). In this case, problem SEP given by Eq. (25) reduces to (noting that $p = 1$ in this context):

$$\text{Minimize} \left\{ \sum_{j \in N} \bar{x}_j y_j - 1 : \sum_{j \in N} \theta_j \le a_0 - 1, \gamma y_j + \theta_j \ge a_j, \forall j \in N, (\gamma, \theta_j, \forall j \in N) \ge 0, y \text{ binary} \right\}. \tag{26}$$

It is easy to see that an optimal value for γ in (26) is given by $\gamma = \max\{a_j : \forall j \in N\}$, whence $\theta_j \equiv a_j(1 - y_j), \forall j \in N$, at optimality, so that (26) is equivalent to solving:

$$\text{Minimize} \left\{ \sum_{j \in N} \bar{x}_j y_j - 1 : \sum_{j \in N} a_j(1 - y_j) \le a_0 - 1, y \text{ binary} \right\}, \tag{27}$$

which is precisely the separation problem formulated by Crowder et al. [8] in this simple special case.

6. Illustrative example

Consider the following constraints of type (1a)–(1c) in binary variables x_j, $j = 1, \ldots, 10$:

$$13x_1 + 12x_2 + 9x_3 + 7x_4 + 5x_5 + 4x_6 + 3x_7 + 2x_8 + 2x_9 + 2x_{10} \ge 25 \tag{28a}$$

$$1 \le \sum_{j=1}^{10} x_j \le 3, \qquad 0 \le \sum_{j=1}^{5} x_j \le 3, \qquad 1 \le \sum_{j=6}^{10} x_j \le 3. \tag{28b}$$

It is readily verified that the corresponding set $X \ne \emptyset$, (3) holds true, and no variable can be fixed at 1. Moreover, the routine of Fig. 3 fixes $x_4 = x_5 = 0$. For example, consider x_4. If $x_4 = 1$, then even with the best consequent choices of $x_1 = x_6 = 1$ subject to (28b), we get the sum on the left-hand side of (28a) to be $24 < 25$. Hence, $x_4 = 0$. Similarly, $x_5 = 0$. Eliminating x_4 and x_5 from (28a) and (28b) and running the algorithm of Fig. 6, we generate the following non-dominated cuts based on the corresponding then-current lists PS as identified below. Note that (29e) reproduces the second-last inequality in (28b), while (29b) asserts that the second inequality in (28b) should hold as an equality, in addition to having $x_4 = x_5 = 0$. Also, observe that while (29b) is non-dominated, it is implied by (29a) and (29e), as seen by summing these inequalities. In particular, the set of non-redundant, non-dominated HOC inequalities are given by (29a), (29c) and (29d) for this example.

$$PS = \{1, 2, 3\} : x_1 + x_2 + x_3 \ge 2 \tag{29a}$$

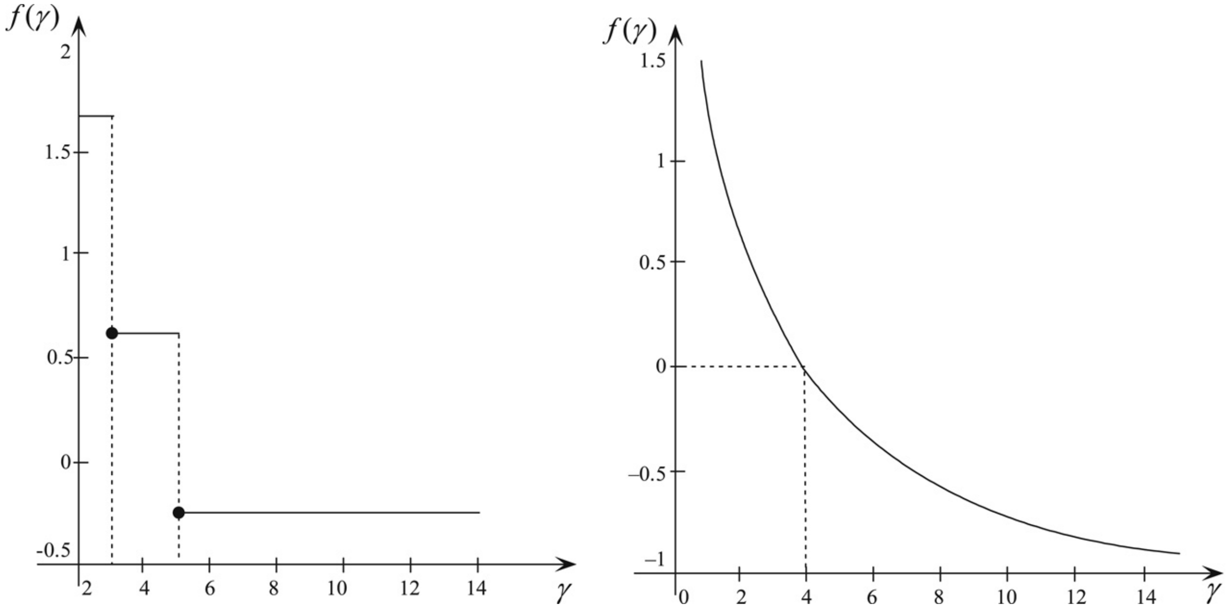
$$PS = \{1, 2, 3, \underline{6}, \underline{7}, \underline{8}, \underline{9}, \underline{10}\} : x_1 + x_2 + x_3 + x_6 + x_7 + x_8 + x_9 + x_{10} \ge 3 \tag{29b}$$

$$PS = \{1, 2, -\underline{3}, 6, \underline{7}\} : x_1 + x_2 + x_6 + x_7 \ge 2 \tag{29c}$$

$$PS = \{1, -\underline{2}, -\underline{3}, 6\} : x_1 + x_6 \ge 1 \tag{29d}$$

$$PS = \{-\underline{1}, -\underline{2}, -\underline{3}, 6, \underline{7}, \underline{8}, \underline{9}, \underline{10}\} : x_6 + x_7 + x_8 + x_9 + x_{10} \ge 1. \tag{29e}$$

It is interesting to compare these inequalities with the ordinary non-dominated knapsack inequalities obtained from (28a) itself, as well as with the second-order cover cuts of Glover and Sherali [12] that are based on (28a) and



(a) Case of y binary-restricted.

(b) Case of y relaxed to be continuous.

Fig. 7. Parametric plots of the objective value of problem SEP versus γ.

$\sum_{j=1}^{10} x_j \leq 3$ from (28b). For example, a non-dominated knapsack cover inequality is given by $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \geq 2$. This is strictly dominated by the non-dominated second-order cover inequalities $x_1 + x_2 + x_3 + x_5 \geq 2$ and $x_1 + x_2 + x_3 + x_4 \geq 2$, which are generated using the procedure developed in Glover and Sherali [12]. Observe that the HOC cut (29a), which is given by $x_1 + x_2 + x_3 \geq 2$, and is based on considering the additional information in (28b), strictly dominates all these inequalities.

Finally, we illustrate the separation routines discussed in Section 5.2 to delete a given solution $\bar{x}$ to some continuous relaxation, where we take $\bar{x}$ for this example to be given by

$$\bar{x}_1 = 1, \qquad \bar{x}_2 = 2/3, \qquad \bar{x}_6 = 1, \quad \text{and} \quad \bar{x}_j = 0 \text{ otherwise.}$$

Implementing Method 1 of Section 5.2.1 with the original procedure of Fig. 6, the first LND cut generated is $x_1 + x_2 + x_3 \geq 2$, which happens to be the non-dominated HOC inequality (29a). On the other hand, using the modified flow-chart procedure described in Section 5.2.1, we obtained $x_2 + x_3 + x_8 \geq 1$ as the first violated LND cut, with a corresponding optimal solution to (8) given by x^* having $x_1^* = x_2^* = x_6^* = 1$, and $x_j^* = 0$, otherwise. Checking its non-dominance, we find that min $\{x_2 + x_3 \colon x \in X\} = 1$, which leads to the dominating HOC inequality $x_2 + x_3 \geq 1$. Continuing with the latter as discussed in Section 5.2.1 (and with the same solution x^*), we find that with $k = 1 \in NJ$, $\min\{x_1 + x_2 + x_3 \colon x \in X\} = 2$, which produces the same non-dominated HOC cut as above, given by $x_1 + x_2 + x_3 \geq 2$.

To illustrate Method 2 in Section 5.2.2 based on the parametric separation problem SEP given by (25), starting with $\gamma = \max\{a_j : j \in N\} = 13$ as recommended, we obtained an optimal solution having $p = 1$, $y_2 = y_3 = 1$, and $y_j = 0$ otherwise, with an objective value of –1/3. A parametric plot for the underlying function $f(\gamma)$ is depicted in Fig. 7(a). This solution yields the cut $x_2 + x_3 \geq 1$, which can be tightened to the non-dominated HOC inequality $x_1 + x_2 + x_3 \geq 2$ as shown above, following the procedure of Section 5.2.1. Actually, it turns out that for $\gamma = 5$, we obtain an alternative optimal solution having $p = 2$, $y_1 = y_2 = y_3 = 1$, and $y_j = 0$ otherwise, thereby directly yielding the non-dominated inequality $x_1 + x_2 + x_3 \geq 2$.

Finally, consider the case where problem SEP is solved with the y-variables relaxed to be continuous. A plot of the corresponding objective function value $\bar{f}(\gamma)$, $\gamma \geq 0$, as defined in Section 5.2.2 is displayed in Fig. 7(b). Here, the objective value turns negative beyond $\gamma = 4$ (where for $\gamma = 4$, the optimal solution obtained is given by $p = 2$, $y_1 = y_2 = 1$, $y_3 = y_6 = 0.25$, and $y_j = 0$ otherwise). In particular, commencing with $\gamma = \max\{a_j : j \in N\} = 13$ as recommended and solving this relaxation of problem SEP, we would obtain a negative objective value, so that continuing to then solve this problem with the y-variables restricted to be binary-valued would ultimately yield the cut $x_1 + x_2 + x_3 \geq 2$ as above.

7. Lifting HOC inequalities

In this section, we briefly address the issue of (sequentially) lifting a non-dominated HOC inequality, typically one that has been generated to delete some continuous relaxation solution $\bar{x}$, in order to further tighten this cut. At any stage in this process, beginning with an HOC inequality

$$\sum_{j \in J} x_j \geq p, \quad \text{where } p = \min\left\{\sum_{j \in J} x_j : x \in X\right\}, \text{ solved by } x^* \in X, \tag{30}$$

suppose that we have a current valid inequality

$$\sum_{j \in J \cup L} \pi_j x_j \geq \pi_0, \tag{31}$$

where $L \subset NJ \equiv N - J$. Following Gu et al. [15,16], for example, we now consider two cases of down- and up-lifting, respectively.

Case (i): Down-lifting based on $k \in NJ - L$ such that $x_k^* = 1$.

Consider the following augmentation of (31):

$$\sum_{j \in J \cup L} \pi_j x_j - \pi_k(1 - x_k) \geq \pi_0. \tag{32}$$

Clearly, given the validity of (31) for X, the inequality (32) is valid when $x_k = 1$. In order to make it also valid when $x_k = 0$ while rendering the resulting cut as tight as possible, we derive

$$\pi_k = \min\left\{\sum_{j \in J \cup L} \pi_j x_j : x \in X, x_k = 0\right\} - \pi_0. \tag{33}$$

Case (ii): Up-lifting based on $k \in NJ - L$ such that $x_k^* = 0$.

Following a parallel argument to Case (i), we consider the following augmentation to (31):

$$\sum_{j \in J \cup L} \pi_j x_j - \pi_k x_k \geq \pi_0, \tag{34}$$

which is clearly valid when $x_k = 0$, and is validated when $x_k = 1$ by selecting

$$\pi_k = \min\left\{\sum_{j \in J \cup L} \pi_j x_j : x \in X, x_k = 1\right\} - \pi_0. \tag{35}$$

Hence, revising (31) according to (32) or (34) as the case might be, and replacing $L \leftarrow L \cup \{k\}$, we repeat this process until $L = NJ$ and then terminate with the resulting lifted inequality. The following result establishes that $\pi_j \geq 0, \forall j \in N$, for the foregoing standard procedure, and that, in particular, the same solution x^* given by (30) continues to minimize the left-hand side of (31) over $x \in X$ with objective value π_0 at each stage, so that the choice of whether to down-lift or up-lift coefficients based on this minimizing solution remains invariant through this process. A proof is included for the sake of completeness in providing this additional latter insight.

Proposition 7. *Inductively, at each step of the foregoing lifting process, we have*

$$\pi_0 = \min\left\{\sum_{j \in J \cup L} \pi_j x_j \ : \ x \in X\right\}, \text{ which is solved by } x^*, \tag{36}$$

and that π_j is a nonnegative integer, $\forall j \in N$.

Proof. By induction, given that this is true to begin with for the case $L = \emptyset$ by (30), assume that (36) is true for (31) and consider the subsequent step of applying Case (i) (the argument for applying Case (ii) is similar). Examine the problem

$$\min\left\{\sum_{j\in J\cup L}\pi_j x_j + \pi_k x_k : x \in X\right\}. \tag{37}$$

When we fix $x_k = 1$ in (37), then by (36) and the induction hypothesis, an optimal solution is given by x^* and yields an objective value of $\pi_0 + \pi_k$. But also, when we fix $x_k = 0$ in (37), then by (33), the optimal objective value in (37) is given by $\pi_0 + \pi_k$. Hence, the optimal objective value in (37) is given by $\pi_0 + \pi_k$, which is the revised right-hand-side value in (32), and noting that $x_k^* = 1$, this optimal value is achieved for $x^* \in X$. This establishes (36) for the next step. Moreover, by the induction hypothesis (36), we have that $\pi_k \geq 0$ in (33). □

The next result identifies a special case, namely, the class of unit-coefficient lifted HOC cuts, for which the foregoing lifting process can be conducted in polynomial time.

Proposition 8. *Consider the class of HOC inequalities for which the process of sequential lifting produces valid inequalities of the type* $\sum_{j\in N}\pi_j x_j \geq \pi_0$, *where* $0 \leq \pi_j \leq 1, \forall j \in N$. *Then the corresponding lifting process can be accomplished in polynomial time with complexity* $O(n^3)$.

Proof. At any stage in the lifting process, the bottleneck effort is in solving problems (33) and (35). Consider problem (33) (the case of problem (35) is similar). In the spirit of Zemel [33], this problem can be solved by examining the parametric problem:

$$\text{Maximize}\left\{\sum_{j\in N} a_j x_j : \sum_{j\in J\cup L}\pi_j x_j \leq q, \text{(1b) and (1c)}, x_k = 0, x \text{ binary}\right\}, \tag{38}$$

where we seek the smallest value of q (say, q^*) for which the optimal objective function value in (38) is at least a_0. Noting by Proposition 2 that problem (38) is a network flow program, and following the scheme used to solve the similar problem (13) in Section 3, we can compute q^* in polynomial time of order $O(n^2)$, whence, $\pi_k = q^* - \pi_0$ in (33). Hence, the overall lifting process can be conducted in $O(n^3)$ time. □

When the lifting coefficients take on more general (integral) values, whereas we can polynomially bound the cut coefficients as in Zemel [33], because of the additional constraints (1b) and (1c) in (38), an efficient dynamic programming algorithm for parametrically finding q^* as in Proposition 8 becomes elusive. In this case, we would need to contend with solving (33) and (35) as 0–1 programs, although as implemented in [8], we could perform a weaker lifting process by solving the LP relaxation to each of these problems and rounding up the resulting objective value (setting this to zero by Proposition 7 in case the value obtained is negative) in order to determine the cut coefficients.

For the example of Section 6, it turns out that no further tightening of any of the non-redundant, non-dominated HOC inequalities (29a), (29c) and (29d) is possible by using the foregoing sequential lifting process. However, for the sake of illustration, consider the valid HOC inequality $x_1 + x_6 + x_7 \geq 1$, which is actually dominated by (29d). The problem to minimize $\{x_1 + x_6 + x_7 : x \in X\}$ is solved by x^* having $x_2^* = x_3^* = x_6^* = 1$, and $x_j^* = 0$ otherwise, with a unit-objective value. Selecting the index $k = 2$ for down-lifting, for example, we compute using (33) that

$$\pi_2 = \min\{x_1 + x_6 + x_7 : x \in X, x_2 = 0\} - 1 = 1,$$

as attained by the solution $x_1 = x_3 = x_6 = 1$, and $x_j = 0$ otherwise. This yields the lifted inequality (32) given by $x_1 + x_2 + x_6 + x_7 \geq 2$, which is (29c). As mentioned above, any further lifting attempt produces zero cut coefficients.

An interesting question to raise at this point is whether the non-redundant, non-dominated HOC inequalities (29a), (29c) and (29d) that we were unable to lift above happen to be facet-defining for $X_c \equiv \text{conv}[X]$, the convex hull of X. This is indeed the case, as demonstrated below. First of all, note that by virtue of (29b) and that we must necessarily have $x_4 = x_5 = 0$, we get that $\sum_{j=1}^{10} x_j = 3$ must hold true for any $x \in X$. Furthermore, this together with (29a) and (29e) implies that both the latter inequalities must hold as equalities for any $x \in X$. Hence, in effect, the set X is given by

$$13x_1 + 12x_2 + 9x_3 + 7x_4 + 5x_5 + 4x_6 + 3x_7 + 2x_8 + 2x_9 + 2x_{10} \geq 25 \tag{39a}$$

$$x_1 + x_2 + x_3 = 2, \quad x_4 = x_5 = 0, \quad x_6 + \cdots + x_{10} = 1, \quad x \text{ binary}. \tag{39b}$$

Because of the linearly independent equalities in (39b), we have that the dimension of X_c (denoted $\dim(X_c)$) satisfies $\dim(X_c) \leq 6$. Indeed, in this case, we have $\dim(X_c) = 6$, as evidenced by the following seven points $x^i \in X$, $i = 1, \ldots, 7$, which are readily verified to be affinely independent:

$$x_1^i = x_2^i = 1, \quad x_{6+(i-1)}^i = 1, \quad \text{with } x_j^i = 0 \text{ otherwise, for } i = 1, \ldots, 5$$

$$x_1^6 = x_3^6 = x_6^6 = 1, \quad \text{with } x_j^6 = 0, \text{ otherwise,} \quad \text{and} \quad x_2^7 = x_3^7 = x_6^7 = 1, \quad \text{with } x_j^7 = 0, \text{ otherwise.}$$

Now, let us examine the HOC inequalities (29a), (29c) and (29d), and demonstrate in each case that the inequality is facet-defining for X_c by identifying six affinely independent points $x^i \in X$, $i = 1, \ldots, 6$, for which this cut is active. Note that by virtue of (39b), the inequality (29a) defines an improper facet of X_c. Next, consider (29c). The required set of six points, which are readily verified to be affinely independent, are given by

$$x_1^i = x_2^i = x_{8+(i-1)}^i = 1, \quad x_j^i = 0, \quad \text{otherwise, for } i = 1, 2, 3$$

$$x_1^i = x_3^i = x_{6+(i-4)}^i = 1, \quad x_j^i = 0, \quad \text{otherwise, for } i = 4, 5, \text{ and}$$

$$x_2^6 = x_3^6 = x_6^6 = 1, \quad x_j^6 = 0, \quad \text{otherwise.}$$

Finally, for the HOC inequality (29d), the required set of six affinely independent points can be verified to be given by

$$x_1^i = x_2^i = x_{7+(i-1)}^i = 1, \quad \text{with } x_j^i = 0, \text{ otherwise, for } i = 1, \ldots, 4$$

$$x_2^5 = x_3^5 = x_6^5 = 1, \quad x_j^5 = 0, \quad \text{otherwise, and}$$

$$x_1^6 = x_3^6 = x_7^6 = 1, \quad \text{with } x_j^6 = 0, \text{ otherwise.}$$

We mention here in closing that, in general, Johnson and Padberg [20], Nemhauser and Vance [23], Sherali and Lee [28], and Glover and Sherali [12] have characterized and identified particular classes of facet-defining inequalities for certain special cases of X. We recommend exploring additional cases of this type in the present more general context for future research.

8. Some preliminary computational results

To provide some computational evidence for the proposed class of HOC cuts and the related cut strengthening and lifting procedures, we performed the following experiment. We randomly generated 15 test instances of different sizes as indicated in Table 1 of the type: Minimize $\{c^T x : x \in X\}$ where X is defined by (1). The c_j-and the a_j-coefficients were generated uniformly on the interval [2, 15] for Instances 1–10, and on [2, 33] for Instances 11–15, and we let $\ell_i \in \{0, 1\}$ and $u_i = 3$, $\forall i \in M$, $\ell = 1$, $u = \lceil 1.5m \rceil$, and $a_0 = \lfloor 0.2a_{0\,\min} + 0.8a_{0\,\max} \rfloor$, where $a_{0\,\min} \equiv \min\{\sum_{j=1}^n a_j x_j \,:\, \text{(1b) and (1c)}, x \text{ binary}\}$ and $a_{0\,\max} \equiv \max\{\sum_{j=1}^n a_j x_j \,:\, \text{(1b) and (1c)}, x \text{ binary}\}$. In the results reported in Table 1, GAP I refers to the usual LP-IP percentage gap for the generated test instances; GAP II refers to the final LP-IP percentage gap after generating up to 5 non-dominated inequalities using Method 1 of Section 5.2.1 that were subsequently lifted as discussed in Section 7; GAP III refers to the final LP-IP percentage gap after generating up to 5 non-dominated cuts using Method 2 as described in Section 5.2.2, which involves invoking the separation routine (25) with $\gamma \equiv \max_{j \in N}\{a_j\}$, tightening the resulting cut to ensure that it is non-dominated as in Section 5.2.1, and then lifting it as possible as discussed in Section 7, and finally, for comparison purposes, GAP IV refers to the final LP-IP percentage gap after sequentially generating up to 5 minimal cover inequalities for the underlying knapsack polytope using the separation problem (27), which were subsequently lifted (exactly) in each case into facets for this polytope (see Balas and Zemel [4] and Crowder et al. [8]). The results indicate that both the proposed Methods 1 and 2 generated strong cuts that helped close the LP-IP gap in most instances (more so for Method 1) in contrast with using lifted minimal cover facet-inducing inequalities of the knapsack polytope that ignore the additional structure defined by (1b) and (1c). We recommend further tests using more general IP instances for future research.

Table 1
Optimality gap comparison

Problem	Size (n, m)	LP-IP gap (%)			
		GAP I (Initial gap)	GAP II (Method 1)	GAP III (Method 2)	GAP IV (Lifted minimal covers)
1	(10, 2)	30.00	0	0	7.27
2		20.83	0	0	13.63
3		29.62	0	0	19.04
4	(15, 3)	7.14	0	0	3.80
5		8.97	0	0	5.12
6		17.50	1.39	0	12.50
7	(20, 4)	7.14	3.57	0	3.57
8		8.69	6.52	6.52	6.52
9		3.70	0	0	0
10	(25, 5)	9.09	4.45	6.81	9.09
11		14.51	3.13	3.13	13.11
12		5.41	0	0	1.67
13	(30, 5)	6.12	0	0	5.18
14		11.08	0	0	7.89
15		9.37	6.25	0	1.70

9. Conclusions

We have presented in this paper a new class of higher-order cover (HOC) inequalities for 0–1 programs that are implied by a knapsack constraint in concert with lower- and upper-bounding restrictions on the sum of all variables appearing in this constraint, as well as on partitioned subsets of these variables. We have shown how we can perform pre-processing to check feasibility, possibly fix variables at 0 or 1 values, and to generate members of this class of HOC inequalities, all in (strongly) polynomial time. An algorithm was also proposed based on certain established dominance results to generate the entire class of non-dominated HOC inequalities. Procedures for solving underlying separation problems and for sequentially lifting generated HOC inequalities to possibly further tighten them, along with polynomial-time implementations for certain special cases, were also developed. The ability to derive stronger valid inequalities than those obtained via non-dominated knapsack cover or second-order cover inequalities was demonstrated using an illustrative example. For this particular numerical example, all the generated non-redundant, non-dominated HOC inequalities were shown to be facet-defining. Some preliminary computational results were also presented to exhibit the efficacy of exploiting the augmented knapsack structure in generating lifted HOC inequalities.

The focus of the present paper has been to introduce the class of higher-order cover cuts and to establish some fundamental results and procedures pertaining to the generation of non-dominated inequalities from this class. There are several theoretical and practical research issues that arise based on this work, which we propose to explore in follow-on studies. Among these are an investigation of conditions under which HOC inequalities would be facet-defining for conv (X), the extension of the analysis of this paper to consider more general nested-bounding constraints as in [11,19,13], and the computational implementation and testing of using lifted, non-dominated HOC cuts in the solution of 0–1 programs.

Acknowledgements

This research has been supported in part by the *National Science Foundation* under Grant Number DMI-0552676. The authors also thank the guest editor, Professor Egon Balas, and three anonymous referees for their constructive comments and suggestions that have significantly improved the contents of this paper. This paper is dedicated to the memory of George B. Dantzig.

References

[1] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice-Hall, Upper Saddle River, NJ, 1993.
[2] E. Balas, Facets of the knapsack polytope, Mathematical Programming 8 (1975) 146–164.
[3] E. Balas, E. Zemel, Critical cutsets of graphs and canonical facets of set-packing polytopes, Mathematics of Operations Research 2 (1977) 15–19.

[4] E. Balas, E. Zemel, Facets of the knapsack polytope from minimal covers, SIAM Journal on Applied Mathematics 34 (1978) 119–148.
[5] M.S. Bazaraa, J.J. Jarvis, H.D. Sherali, Linear Programming and Network Flows, third ed., John Wiley & Sons, Inc., New York, NY, 2005.
[6] M.S. Bazaraa, H.D. Sherali, C.M. Shetty, Nonlinear Programming: Theory and Algorithms, third ed., John Wiley & Sons, Inc., New York, NY, 2006.
[7] G. Cornuéjols, A. Sassano, On the 0–1 facets of the set covering polytope, Mathematical Programming 43 (1989) 45–55.
[8] H.P. Crowder, E.L. Johnson, M.W. Padberg, Solving large-scale zero–one linear programming problems, Operations Research 31 (1983) 803–834.
[9] A.M. Geoffrion, An improved implicit enumeration approach for integer programming, Operations Research 17 (1969) 437–454.
[10] F. Glover, A multiphase-dual algorithm for the zero–one integer programming problem, Operations Research 13 (1965) 879–919.
[11] F. Glover, Flows in arborescences, Management Science 17 (9) (1971) 568–586.
[12] F. Glover, H.D. Sherali, Second order cover cuts, Mathematical Programming (ISSN: 0025-5610 1436-4646) (2007), doi:10.1007/s10107-007-0098-4. (Online).
[13] F. Glover, H.D. Sherali, Some classes of valid inequalities and convex hull characterizations for dynamic fixed-charge problems under nested constraints, Annals of Operations Research, State-of-the-Art in Integer Programming 140 (1) (2005) 215–234. M. Guignard-Spielberg, K. Spielberg (Eds.).
[14] F. Glover, H.D. Sherali, Y. Lee, Generating cuts from surrogate constraint analysis for zero–one and multiple choice programming, Computational Optimization and Applications 8 (2) (1997) 152–172.
[15] Z. Gu, G.L. Nemhauser, M.W.P. Savelsbergh, Cover inequalities for 0–1 linear programs: Computation, INFORMS Journal on Computing 10 (1998) 427–437.
[16] Z. Gu, G.L. Nemhauser, M.W.P. Savelsbergh, Sequence independent lifting in mixed integer programming, Journal of Combinatorial Optimization 4 (2000) 109–129.
[17] P.L. Hammer, E.L. Johnson, U.N. Peled, Facets of regular 0–1 polytopes, Mathematical Programming 8 (1975) 179–206.
[18] S. Hanafi, Contribution à la résolution de problèmes duaux de grande taille en optimisation combinatoire, Ph.D. Thesis, University of Valenciennes, France, 1993.
[19] S. Hanafi, F. Glover, Exploiting nested inequalities and surrogate constraints, Research Report, University of Valenciennes, France, and University of Colorado, Boulder, CO, 2005.
[20] E.L. Johnson, M.W. Padberg, A note on the knapsack problem with special ordered sets, Operations Research Letters 1 (1981) 18–22.
[21] M. Laurent, A generalization of antiwebs to independence systems and their canonical facets, Mathematical Programming 45 (1989) 97–108.
[22] L. Moura, Rank inequalities and separation algorithms for packing designs and sparse triple systems, Theoretical Computer Science 297 (1–3) (2003) 367–384.
[23] G.L. Nemhauser, P. Vance, Lifted cover facets of the 0–1 knapsack polytope with GUB constraints, Operations Research Letters 16 (1994) 255–263.
[24] G.L. Nemhauser, L.A. Wolsey, Integer and Combinatorial Optimization, John Wiley & Sons, Inc., New York, NY, 1999.
[25] G.L. Nemhauser, M.W.P. Savelsbergh, G.S. Sigismondi, MINTO, a mixed INTeger optimizer, Operations Research Letters 15 (1994) 47–58.
[26] M.A. Osorio, F. Glover, P. Hammer, Cutting and surrogate constraint analysis for improved multidimensional knapsack solutions, Annals of Operations Research 117 (2002) 71–93.
[27] M.W.P. Savelsbergh, Preprocessing and probing for mixed integer programming problems, ORSA Journal on Computing 6 (1994) 445–454.
[28] H.D. Sherali, Y. Lee, Sequential and simultaneous liftings of minimal cover inequalities for generalized upper bound constrained knapsack polytopes, SIAM Journal on Discrete Mathematics 8 (1) (1995) 133–153.
[29] H.D. Sherali, C.H. Tuncbilek, A global optimization algorithm for polynomial programming problems using a reformulation–linearization technique, Journal of Global Optimization 2 (1992) 101–112.
[30] K. Spielberg, M. Guignard, A sequential (quasi) hot start method for BB (0, 1) mixed integer programming, in: Mathematical Programming Symposium, Atlanta, GA, 2000.
[31] M. Vasquez, Y. Vimont, Improved results on the 0–1 multidimensional knapsack problem, European Journal of Operational Research 165 (2005) 70–81.
[32] L.A. Wolsey, Faces for a linear inequality in 0-1 variables, Mathematical Programming 8 (1975) 165–178.
[33] E. Zemel, Easily computable facets of the knapsack polytope, Mathematics of Operations Research 14 (1989) 760–764.
[34] B. Zeng, J.-P.P. Richard, Sequence independent lifting for 0–1 knapsack problems with disjoint cardinality constraints, School of Industrial Engineering, Purdue University, West Lafayette, IN, 2006, Manuscript.