Operations Research

Cutting Planes for Multistage Stochastic Integer Programs

Yongpei Guan, Shabbir Ahmed, George L. Nemhauser,

To cite this article:
Yongpei Guan, Shabbir Ahmed, George L. Nemhauser, (2009) Cutting Planes for Multistage Stochastic Integer Programs. Operations Research 57(2):287-298. https://doi.org/10.1287/opre.1080.0535

Operations Research
Vol. 57, No. 2, March–April 2009, pp. 287–298
ISSN 0030-364X | EISSN 1526-5463 | 09 | 5702 | 0287

DOI 10.1287/opre.1080.0535
© 2009 INFORMS

Cutting Planes for Multistage Stochastic Integer Programs

Yongpei Guan
School of Industrial Engineering, University of Oklahoma, Norman, Oklahoma 73019, yguan@ou.edu

Shabbir Ahmed, George L. Nemhauser
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332
{sahmed@isye.gatech.edu, gnemhaus@isye.gatech.edu}

This paper addresses the problem of finding cutting planes for multistage stochastic integer programs. We give a general method for generating cutting planes for multistage stochastic integer programs based on combining inequalities that are valid for the individual scenarios. We apply the method to generate cuts for a stochastic version of a dynamic knapsack problem and for stochastic lot-sizing problems. We give computational results, which show that these new inequalities are very effective in a branch-and-cut algorithm.

Subject classifications: programming, integer: cutting plane/facet; programming: stochastic; inventory/production, lot sizing: stochastic.
Area of review: Optimization.
History: Received September 2006; revisions received March 2007, October 2007; accepted November 2007. Published online in *Articles in Advance* December 9, 2008.

1. Introduction

This paper deals with polyhedral aspects of multistage stochastic integer programs. Our basic idea is to extend known results concerning cutting planes for a deterministic model of the problem to a stochastic model. In other words, suppose that we know valid inequalities that make it possible to solve efficiently the deterministic model by linear programming or branch and cut. In this paper, we show how to use this knowledge to get valid inequalities for a stochastic scenario-tree-based model of the problem so that it too can be solved by a branch-and-cut algorithm. Multiperiod production-planning problems are a typical example where there is considerable knowledge of the convex hull of feasible solutions for various deterministic problems. In Guan et al. (2006), we showed how to apply this idea for uncapacitated lot-sizing problems by generalizing the well-known (l, S) inequalities (Barany et al. 1984) to a stochastic setting. Here we generalize the basic ideas of Guan et al. (2006) so that the results can be applied to general multistage stochastic integer programs involving a scenario tree model of the uncertain parameters. The key idea of our approach is to combine deterministic valid inequalities corresponding to different scenarios to obtain valid inequalities for the whole scenario tree. The general framework is studied in detail in the context of stochastic dynamic knapsack problems and stochastic lot-sizing problems. For these special cases, we provide facet and convex hull defining conditions, and discuss separation procedures. We also present computational results that show that the approach is computationally feasible for stochastic lot-sizing problems.

The remainder of this paper is organized as follows. In the next section, we present notation and terminology used throughout the paper, and also discuss the underlying combination principle in our approach. A general framework for obtaining valid inequalities for scenario-tree-based stochastic integer programs is given in §3. Applications to stochastic dynamic knapsack problems and stochastic lot-sizing problems are presented in §§4 and 5, respectively. Section 6 presents computational results, and §7 gives conclusions and directions for future research.

2. Notation and Preliminaries

2.1. Multistage Stochastic Integer Programs

Consider the deterministic T-period mixed-integer program

$$\begin{aligned} \min\ & \sum_{t=1}^{T}(\alpha_t x_t + \beta_t y_t) \\ \text{s.t.}\ & \sum_{\tau=1}^{t}(G_{t\tau}x_\tau + A_{t\tau}y_\tau) \geqslant b_t, \quad t = 1, \ldots, T, \\ & x_t \in \mathbb{R}^p_+,\ y_t \in \mathbb{Z}^n_+, \quad t = 1, \ldots, T, \end{aligned} \tag{1}$$

where $A_{t\tau}$ and $G_{t\tau}$ are matrices; and α_t, β_t, and b_t are vectors of appropriate dimensions. We assume, without loss of generality, that the decision vectors in each of the time periods $t = 1, \ldots, T$ are of identical dimension.

Now consider the extension of (1) to a stochastic setting. We assume that the problem parameters (α, β, G, A, b) evolve as a discrete-time stochastic process with a finite probability space. This information structure can be interpreted as a scenario tree $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ with T levels (or stages) where a node $i \in \mathcal{V}$ in stage t of the tree gives the state of the system that can be distinguished by information available up to time stage t. The probability associated with the state represented by node i is p_i. The set of nodes on the path from the root node to a node i is denoted by $\mathcal{P}(i)$. The decisions (x_i, y_i) corresponding to a node i are assumed to be made after observing the realizations $(\alpha_i, \beta_i, \{G_{ij}\}_{j\in\mathcal{P}(i)}, \{A_{ij}\}_{j\in\mathcal{P}(i)}, b_i)$, but are nonanticipative with respect to future realizations. The goal is to minimize expected total costs. The multistage stochastic integer programming extension of (1) is then

$$\begin{aligned} \min \quad & \sum_{i\in\mathcal{V}} p_i(\alpha_i x_i + \beta_i y_i) \\ \text{s.t.} \quad & \sum_{j\in\mathcal{P}(i)} (G_{ij}x_j + A_{ij}y_j) \geqslant b_i, \quad i \in \mathcal{V}, \\ & x_i \in \mathbb{R}^p_+,\ y_i \in \mathbb{Z}^n_+, \quad i \in \mathcal{V}. \end{aligned} \tag{2}$$

Any multistage stochastic integer program defined over a scenario tree can be modelled as formulation (2) (cf. Römisch and Schultz 2001, Sen 2005). Specific examples of such problems include stochastic lot-sizing problems (Guan et al. 2006, Lulli and Sen 2004) (see also §5), stochastic capacity-planning models (Ahmed et al. 2003, Singh et al. 2008), and the stochastic unit commitment problem (Nowak and Römisch 2000, Takriti et al. 1996).

We denote the set of feasible solutions of the multistage stochastic integer program (2) by $X_{\mathcal{T}}$, and refer to this set as the *tree set*. In this paper, we develop valid inequalities for the tree set $X_{\mathcal{T}}$ by combining given valid inequalities for *path sets* of the form

$$X_i = \Bigg\{(x_k, y_k)_{k\in\mathcal{P}(i)}\colon \sum_{j\in\mathcal{P}(k)} (G_{kj}x_j + A_{kj}y_j) \geqslant b_k, \\ x_k \in \mathbb{R}^p_+,\ y_k \in \mathbb{Z}^n_+\ \forall\, k \in \mathcal{P}(i)\Bigg\}$$

for $i \in \mathcal{V}$. Note that the path set X_i includes only those constraints of $X_{\mathcal{T}}$ that correspond to the nodes on the path $\mathcal{P}(i)$ from the root node to node i, and hence is a relaxation of the tree set $X_{\mathcal{T}}$. Moreover, the path set X_i is essentially the feasible region of the deterministic multiperiod problem (1) with $t(i)$ periods, where $t(i)$ is the stage number of node i in the scenario tree $\mathcal{T}$. Consequently, known valid inequalities for the deterministic model (1) are valid for the path set X_i and also for the tree set $X_{\mathcal{T}}$. The (deterministic) valid inequalities corresponding to different path sets, called *path inequalities*, can be combined to obtain a new valid inequality, called a *tree inequality*, for the tree set. This idea has been previously explored in Guan et al. (2006), where valid inequalities for deterministic uncapacitated lot sizing were combined to derive valid inequalities for stochastic lot sizing, and in Guan et al. (2007), where valid inequalities for general deterministic two-stage integer programs were combined to obtain inequalities for two-stage stochastic integer programs. The underlying combination scheme in these papers, and in this work, is a simple operation known as *pairing* (Guan et al. 2007), which is described next.

2.2. Pairing

Throughout this paper, we adopt the following convention. Given two vectors a_1 and a_2 of the same dimension, the operations $\min(a_1, a_2)$ and $\max(a_1, a_2)$ are understood to be carried out componentwise. Given a vector a and a scalar b, we define $a + b = a + b\mathbb{1}$ and $\min\{a, b\} = \min\{a, b\mathbb{1}\}$, where $\mathbb{1}$ is a vector of ones of the same dimension as a. Also, because all variables are nonnegative, we say that an inequality $a_1 x \geqslant b_1$ *dominates* another inequality $a_2 x \geqslant b_2$ if $a_1 \leqslant a_2$ and $b_1 \geqslant b_2$.

Theorem 1 Guan et al. (2007). *Suppose that the inequalities $g_1 x + a_1 y \geqslant b_1$ and $g_2 x + a_2 y \geqslant b_2$ with $b_1 \leqslant b_2$ are valid for the set $X \subset \mathbb{R}^p_+ \times \mathbb{Z}^n_+$. Then, the pairing inequality*

$$\varphi x + \phi y \geqslant b_2, \tag{3}$$

where $\varphi = \max\{g_1, g_2\}$ and $\phi = \min\{a_1 + (b_2 - b_1), \max\{a_1, a_2\}\}$, is valid for X.

The pairing inequality is a split cut that can be derived as in Cook et al. (1990) or via the mixed-integer rounding procedure (Nemhauser and Wolsey 1988, 1990) and, in the special case where all coefficients are nonnegative, via *mixing* (Günlük and Pochet 2001). The following example illustrates that the pairing inequality may not be obtained by one round of the Chvatal-Gomory (C-G) procedure on the original inequalities, i.e., the C-G rank of the pairing inequality is at least two.

Example 1. Consider the set

$$X := \{(x, y_1, y_2) \in \mathbb{Z}^3_+\colon x + 2y_1 \geqslant 2,\ x + 5y_2 \geqslant 5\}.$$

The pairing inequality for the above system is

$$x + 2y_1 + 3y_2 \geqslant 5. \tag{4}$$

To obtain a C-G inequality for X, let λ_1, λ_2 be the multipliers for the first two inequalities in X, $\rho \geqslant 0$ be the multiplier for $x \geqslant 0$, and $\gamma_1, \gamma_2 \geqslant 0$ be the multipliers for $y_1 \geqslant 0$ and $y_2 \geqslant 0$. To obtain a C-G inequality that dominates (4), these multipliers should satisfy

$$\begin{aligned} \lceil 2\lambda_1 + 5\lambda_2 \rceil \geqslant 5 &\ \Rightarrow\ 4 < 2\lambda_1 + 5\lambda_2, \\ \lceil \lambda_1 + \lambda_2 + \rho \rceil \leqslant 1 &\ \Rightarrow\ \lambda_1 + \lambda_2 \leqslant 1, \\ \lceil 2\lambda_1 + \gamma_1 \rceil \leqslant 2 &\ \Rightarrow\ \lambda_1 \leqslant 1, \\ \lceil 5\lambda_2 + \gamma_2 \rceil \leqslant 3 &\ \Rightarrow\ \lambda_2 \leqslant 3/5. \end{aligned}$$

However, the last three inequalities imply that $2\lambda_1 + 5\lambda_2 \leqslant 3.8 < 4$. Therefore, no C-G multipliers exist that produce an inequality that dominates inequality (4), and the C-G rank of (4) is at least two.

Given a set of valid inequalities, the pairing operation can be carried out repeatedly to generate new valid inequalities. The order in which the inequalities are paired differentiates the inequalities. A natural order is *sequential pairing*. Given K valid inequalities

$$g_i x + a_i y \geqslant b_i, \quad i = 1, \dots, K$$

for a set $X \subset \mathbb{R}^p_+ \times \mathbb{Z}^n_+$, such that $b_1 \leqslant b_2 \leqslant \cdots \leqslant b_K$, the *sequential pairing inequality* is obtained by pairing the inequality for $i = 1$ with the one for $i = 2$, and then pairing the resulting inequality with the one for $i = 3$ and so on in the sequence $i = 1, \dots, K$. Problem structures where sequential pairing dominates any other pairing order have been studied in Guan et al. (2007). One such structure is that of two-stage stochastic integer programs.

3. From Paths to Trees

In this section, we derive a family of valid inequalities for the tree set $X_{\mathcal{T}}$ from a given set of path inequalities. We assume that the coefficients of the path inequalities are nonnegative. This assumption can be enforced by weakening any coefficient α to $\max\{0, \alpha\}$ because all variables are assumed to be nonnegative. We need the following additional notation regarding scenario trees. Each node i of the scenario tree $\mathcal{T}$, except the root node (indexed as $i = 1$), has a unique parent i^-, and is the root of a subtree $\mathcal{T}(i) = (\mathcal{V}(i), \mathcal{E}(i))$, which contains all descendants of node i, including itself. Thus, $\mathcal{T} = \mathcal{T}(1)$ and $\mathcal{V} = \mathcal{V}(1)$. Given a subset of nodes $R \subseteq \mathcal{V}$, let $\mathcal{V}_R = \bigcup_{i \in R} \mathcal{P}(i)$, and $R(j) = R \cap \mathcal{V}(j)$ for each $j \in \mathcal{V}_R$.

3.1. The Tree Inequalities

Given a nonempty subset of nodes $R \subseteq \mathcal{V}$, we consider, for each path set X_i for $i \in R$, the valid path inequality

$$\sum_{j \in \mathcal{P}(i)} (g_{ij} x_j + a_{ij} y_j) \geqslant b_i. \tag{5}$$

Without loss of generality, we assume that the set $R = \{i_1, \dots, i_K\}$ is partially ordered such that $0 =: b_{i_0} \leqslant b_{i_1} \leqslant b_{i_2} \leqslant \cdots \leqslant b_{i_K}$. Corresponding to such a set of path inequalities, we define

$$\begin{aligned} b(R) &= b_{i_K}, \\ \Delta_j(R) &= \sum_{i_k \in R(j)} (b_{i_k} - b_{i_{k-1}}) \quad \forall\, j \in \mathcal{V}_R, \\ \varphi_j(R) &= \max_{i \in R}\{g_{ij}\} \quad \forall\, j \in \mathcal{V}_R, \quad \text{and} \\ \phi_j(R) &= \min\Big\{\max_{i \in R}\{a_{ij}\}, \Delta_j(R)\Big\} \quad \forall\, j \in \mathcal{V}_R. \end{aligned} \tag{6}$$

Theorem 2. *Given a subset $R \subseteq \mathcal{V}$, suppose that the path inequalities of the form* (5) *are valid for the path sets X_i for all $i \in R$. Then, the tree inequality*

$$\sum_{j \in \mathcal{V}_R} \varphi_j(R) x_j + \phi_j(R) y_j \geqslant b(R) \tag{7}$$

is valid for the tree set $X_{\mathcal{T}}$.

Proof. We show, by induction, that the tree inequality (7) corresponding to $R_k = \{i_1, \dots, i_k\}$ is valid for $X_{\mathcal{T}}$ for all $k \in \{1, \dots, K\}$.

For the base case $k = 1$, after coefficient tightening, the path inequality for X_{i_1} is

$$\sum_{j \in \mathcal{P}(i_1)} (g_{i_1 j} x_j + \min\{a_{i_1 j}, b_{i_1}\} y_j) \geqslant b_{i_1}, \tag{8}$$

which is valid for $X_{\mathcal{T}}$. Inequality (8) is precisely the tree inequality (7) with $R = \{i_1\}$. In this case, $\mathcal{V}_R = \mathcal{P}(i_1)$ and $R(j) = \{i_1\}$ for all $j \in \mathcal{V}_R$.

Assume now that the inequality

$$\sum_{j \in \mathcal{V}_{R_k}} (\varphi_j(R_k) x_j + \phi_j(R_k) y_j) \geqslant b_{i_k} \tag{9}$$

is valid for $X_{\mathcal{T}}$ for some $k \in \{1, \dots, K-1\}$. The path inequality for $X_{i_{k+1}}$ is

$$\sum_{j \in \mathcal{P}(i_{k+1})} (g_{i_{k+1} j} x_j + a_{i_{k+1} j} y_j) \geqslant b_{i_{k+1}}, \tag{10}$$

which is also valid for $X_{\mathcal{T}}$.

Next, we pair the valid inequalities (9) and (10) for $X_{\mathcal{T}}$ using Theorem 1. Note that the pairing inequality has a right-hand side equal to $b_{i_{k+1}}$ and includes variables from all the nodes in $\mathcal{V}_{R_{k+1}} = \mathcal{V}_{R_k} \cup \mathcal{P}(i_{k+1})$. We next show that the coefficients in the inequality obtained by pairing (9) and (10) are less than or equal to those of the tree inequality (7) corresponding to R_{k+1}.

We partition $\mathcal{V}_{R_{k+1}}$ into three sets: (i) $\mathcal{P}(i_{k+1}) \backslash \mathcal{V}_{R_k}$, (ii) $\mathcal{V}_{R_k} \backslash \mathcal{P}(i_{k+1})$, and (iii) $\mathcal{V}_{R_k} \cap \mathcal{P}(i_{k+1})$.

(i) For each $j \in \mathcal{P}(i_{k+1}) \backslash \mathcal{V}_{R_k}$, we have

$$\begin{aligned} \varphi &= \max\{0, g_{i_{k+1} j}\} \\ &= \max_{i \in R_{k+1}} \{g_{ij}\} \\ &= \varphi_j(R_{k+1}), \end{aligned}$$

where the second equality follows from the fact that $g_{ij} = 0$ for all $i \in R_k$ for any $j \in \mathcal{P}(i_{k+1}) \backslash \mathcal{V}_{R_k}$. Also,

$$\begin{aligned} \phi &= \min\{\phi_j(R_k) + b_{i_{k+1}} - b_{i_k}, \max\{0, a_{i_{k+1} j}\}\} \\ &= \min\Big\{b_{i_{k+1}} - b_{i_k}, \max_{i \in R_{k+1}} \{a_{ij}\}\Big\} \\ &= \min\Big\{\max_{i \in R_{k+1}} \{a_{ij}\}, \sum_{i_r \in R_{k+1}(j)} (b_{i_r} - b_{i_{r-1}})\Big\} \\ &= \phi_j(R_{k+1}), \end{aligned}$$

where the second equality follows from the fact that $a_{ij} = 0$ for all $i \in R_k$ for any $j \in \mathcal{P}(i_{k+1}) \backslash \mathcal{V}_{R_k}$, and the third equality follows from the fact that $R_{k+1}(j) = \{i_{k+1}\}$ for any $j \in \mathcal{P}(i_{k+1}) \backslash \mathcal{V}_{R_k}$.

(ii) For each $j \in \mathscr{V}_{R_k} \setminus \mathscr{P}(i_{k+1})$, we have

$$\begin{aligned}\varphi &= \max\{\varphi_j(R_k), 0\}\\ &= \max\Big\{\max_{i\in R_k}\{g_{ij}\}, 0\Big\}\\ &= \max_{i\in R_{k+1}}\{g_{ij}\}\\ &= \varphi_j(R_{k+1}).\end{aligned}$$

Also,

$$\begin{aligned}\phi &= \min\{\phi_j(R_k) + b_{i_{k+1}} - b_{i_k}, \max\{\phi_j(R_k), 0\}\}\\ &= \phi_j(R_k)\\ &= \min\Big\{\max_{i\in R_k}\{a_{ij}\}, \sum_{i_r\in R_k(j)} (b_{i_r} - b_{i_{r-1}})\Big\}\\ &= \min\Big\{\max_{i\in R_{k+1}}\{a_{ij}\}, \sum_{i_r\in R_{k+1}(j)} (b_{i_r} - b_{i_{r-1}})\Big\}\\ &= \phi_j(R_{k+1}),\end{aligned}$$

where the second equality follows from the fact that $\phi_j(R_k) \geqslant 0$, and the penultimate equality follows from the fact that $a_{i_{k+1}j} = 0$ and $R_{k+1}(j) = R_k(j)$ for $j \in \mathscr{V}_{R_k} \setminus \mathscr{P}(i_{k+1})$.

(iii) For each $j \in \mathscr{V}_{R_k} \cap \mathscr{P}(i_{k+1})$, we have

$$\begin{aligned}\varphi &= \max\{\varphi_j(R_k), g_{i_{k+1}j}\}\\ &= \max\Big\{\max_{i\in R_k}\{g_{ij}\}, g_{i_{k+1}j}\Big\}\\ &= \max_{i\in R_{k+1}}\{g_{ij}\}\\ &= \varphi_j(R_{k+1}).\end{aligned}$$

Also,

$$\phi = \min\{\phi_j(R_k) + b_{i_{k+1}} - b_{i_k}, \max\{\phi_j(R_k), a_{i_{k+1}j}\}\}, \quad (11)$$

where $\phi_j(R_k) = \min\{\max_{i\in R_k}\{a_{ij}\}, \sum_{i_r\in R_k(j)}(b_{i_r} - b_{i_{r-1}})\}$. Consider the following two cases.

(a) If $\max_{i\in R_k}\{a_{ij}\} \leqslant \sum_{i_r\in R_k(j)}(b_{i_r} - b_{i_{r-1}})$, then $\phi_j(R_k) = \max_{i\in R_k}\{a_{ij}\}$, and from (11),

$$\begin{aligned}\phi &= \min\Big\{\max_{i\in R_{k+1}}\{a_{ij}\}, \max_{i\in R_k}\{a_{ij}\} + b_{i_{k+1}} - b_{i_k}\Big\}\\ &\leqslant \min\Big\{\max_{i\in R_{k+1}}\{a_{ij}\}, \sum_{i_r\in R_k(j)} (b_{i_r} - b_{i_{r-1}}) + b_{i_{k+1}} - b_{i_k}\Big\}\\ &= \min\Big\{\max_{i\in R_{k+1}}\{a_{ij}\}, \sum_{i_r\in R_{k+1}(j)} (b_{i_r} - b_{i_{r-1}})\Big\}\\ &= \phi_j(R_{k+1}).\end{aligned}$$

(b) If $\max_{i\in R_k}\{a_{ij}\} > \sum_{i_r\in R_k(j)}(b_{i_r} - b_{i_{r-1}})$, then $\phi_j(R_k) = \sum_{i_r\in R_k(j)}(b_{i_r} - b_{i_{r-1}})$, and from (11),

$$\begin{aligned}\phi &= \min\Big\{\max\Big\{\sum_{i_r\in R_k(j)} (b_{i_r} - b_{i_{r-1}}), a_{i_{k+1}j}\Big\},\\ &\qquad \sum_{i_r\in R_k(j)} (b_{i_r} - b_{i_{r-1}}) + b_{i_{k+1}} - b_{i_k}\Big\}\\ &\leqslant \min\Big\{\max\Big\{\max_{i\in R_k}\{a_{ij}\}, a_{i_{k+1}j}\Big\}, \sum_{i_r\in R_{k+1}(j)} (b_{i_r} - b_{i_{r-1}})\Big\}\\ &= \min\Big\{\max_{i\in R_{k+1}}\{a_{ij}\}, \sum_{i_r\in R_{k+1}(j)} (b_{i_r} - b_{i_{r-1}})\Big\}\\ &= \phi_j(R_{k+1}).\end{aligned}$$

It then follows that the tree inequality (7) corresponding to R_{k+1} is dominated by an inequality obtained by pairing the valid inequalities (9) and (10), and is therefore valid for $X_{\mathscr{T}}$. □

Example 2. Consider a tree set corresponding to the scenario tree depicted in Figure 1. Assume that the three valid path inequalities corresponding to nodes 2, 3, and 4 are, respectively,

$$2x_1 + 5y_1 + 2x_2 + 5y_2 \geqslant 15,$$
$$3x_1 + 4y_1 + 3x_3 + 4y_3 \geqslant 17, \quad \text{and}$$
$$3x_1 + 6y_1 + 3x_3 + 5y_3 + 9x_4 + 5y_4 \geqslant 18.$$

Then, by Theorem 2, the tree inequalities of the form (7) corresponding to $R = \{2, 3\}$ and $R = \{2, 4\}$,

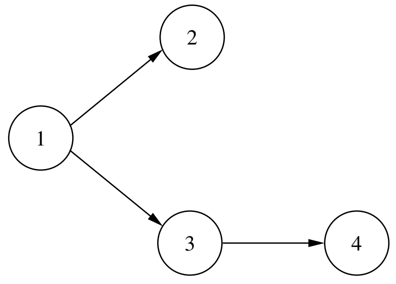
$$3x_1 + 2x_2 + 3x_3 + 5y_1 + 5y_2 + 2y_3 \geqslant 17, \quad \text{and}$$
$$3x_1 + 2x_2 + 3x_3 + 9x_4 + 6y_1 + 5y_2 + 3y_3 + 3y_4 \geqslant 18,$$

are valid.

3.2. Nondominated Tree Inequalities

In general, any $R \subseteq \mathscr{V}$ can produce a tree inequality. However, if the path inequality coefficients g_{ij} and a_{ij} depend only on the variables, i.e., $g_{ij} = g_j$ and $a_{ij} = a_j$ for all i

Figure 1. Scenario tree for Example 2.



and j, then some of the tree inequalities may be dominated by others. In this case, the path inequalities simplify to

$$\sum_{j\in\mathcal{P}(i)} (g_j x_j + a_j y_j) \geqslant b_i \quad \forall\, i\in\mathcal{V}, \tag{12}$$

and we can assume that $b_j \leqslant b_i$ for all $j\in\mathcal{P}(i)$. Also from (6),

$$\varphi_j(R) = g_j \quad \text{and} \quad \phi_j(R) = \min\{a_j, \Delta_j(R)\} \tag{13}$$

for all $j\in\mathcal{V}_R$.

Now, consider a set $R=\{i_1,\ldots,i_K\}$ and suppose that for some node $i_k\in R$, there exists a node $j_k\in\mathcal{P}(i_k)$ such that $j_k\notin R$ and $b_{j_k} > b_{i_{k-1}}$. We let j_k be the closest such node to i_k. We now compare the tree inequalities (7) for the sets R and $R'=\{i_1,\ldots,i_{k-1}, j_k, i_k,\ldots,i_K\}$. Note that $\mathcal{V}_R=\mathcal{V}_{R'}$, $b(R)=b(R')=b_{i_K}$, and $\varphi_j(R)=\varphi_j(R')=g_j$ for all $j\in\mathcal{V}_R$. Now for all $j\in\mathcal{V}_R\backslash\{i_k\}$, we have $\Delta_j(R)=\Delta_j(R')$. However,

$$\Delta_{i_k}(R)-\Delta_{i_k}(R') = (b_{i_k}-b_{i_{k-1}}) - (b_{i_k}-b_{j_k}) = b_{j_k}-b_{i_{k-1}} > 0.$$

Thus, $\phi_j(R)\geqslant\phi_j(R')$, and the tree inequality for R' dominates the one corresponding to R.

To obtain a node set whose corresponding tree inequality is not dominated in the above manner, let $i'_k = \arg\min\{t(j)\colon j\in\mathcal{P}(i_k) \text{ and } b_j > b_{i_{k-1}}\}$ for each $i_k\in R$. If no such i'_k exists, let $i'_k=i_k$. Let $\Omega_R=\bigcup_{i_k\in R}\mathcal{P}(i'_k, i_k)$, where $\mathcal{P}(i'_k,i_k)=\mathcal{P}(i_k)\backslash\mathcal{P}(i'_k)\cup\{i'_k\}$. Then, inductively applying the argument given above over the nodes in $\mathcal{P}(i'_k,i_k)$ proves the following result.

Theorem 3. *The tree inequality* (7) *corresponding to* Ω_R *dominates the tree inequality corresponding to* R.

In the next section, we will give an example of Theorem 3. Note that Theorem 3 does not need to be applied recursively because from the definition of Ω_R it follows that $\Omega_{\Omega_R}=\Omega_R$. From now on, unless otherwise stated, whenever the path inequality coefficients depend only on the variables, we will assume that the path inequality sets R are such that $R=\Omega_R$.

4. The Stochastic Dynamic Knapsack Problem

The deterministic dynamic knapsack set

$$X_{\text{DK}} = \left\{(x,y)\in\mathbb{R}_+\times\{0,1\}^T\colon x+\sum_{\tau=1}^{t} a_\tau y_\tau \geqslant b_t \quad t=1,\ldots,T\right\},$$

where $a_t\in\mathbb{R}_+$ and $b_t\in\mathbb{R}_+$, has been studied in Loparic et al. (2003). Assuming that the parameters a_t and b_t are stochastic and evolve according to the scenario tree $\mathcal{T}=(\mathcal{V},\mathcal{E})$, and using the notation already described, the *stochastic dynamic knapsack set* is

$$X_{\text{SDK}} = \left\{(x,y)\in\mathbb{R}_+\times\{0,1\}^{|\mathcal{V}|}\colon x+\sum_{j\in\mathcal{P}(i)} a_j y_j \geqslant b_i,\, i\in\mathcal{V}\right\}, \tag{14}$$

where $a_i\in\mathbb{R}_+$ and $b_i\in\mathbb{R}_+$ for all $i\in\mathcal{V}$. Without loss of generality, we assume that $b_j\leqslant b_i$ if $j\in\mathcal{P}(i)$.

The stochastic dynamic knapsack set X_{SDK} is a simple special case of the tree set $X_{\mathcal{T}}$, involving a single binary variable and a single constraint corresponding to each node of the scenario tree, and an additional continuous variable x corresponding to the root node. By applying Theorem 2 to X_{SDK}, we obtain the valid tree inequalities

$$x+\sum_{j\in\mathcal{V}_R}\phi_j(R)y_j \geqslant b(R) \quad \forall\, R\subseteq\mathcal{V}, \tag{15}$$

where $\phi_j(R)=\min\{a_j,\Delta_j(R)\}$ for all $j\in\mathcal{V}_R$.

Note that the inequalities defining X_{SDK} satisfy the requirements of §3.2, and hence a tree inequality (15) for a set R is dominated by the one for Ω_R. The following example illustrates this.

Example 3. Consider an instance of X_{SDK} where the scenario tree has five nodes as shown in Figure 2. The instance parameters are

$$a_1=40,\quad a_2=15,\quad a_3=20,\quad a_4=20,\quad a_5=40 \quad\text{and}$$
$$b_1=5,\quad b_2=15,\quad b_3=17,\quad b_4=20,\quad b_5=40.$$

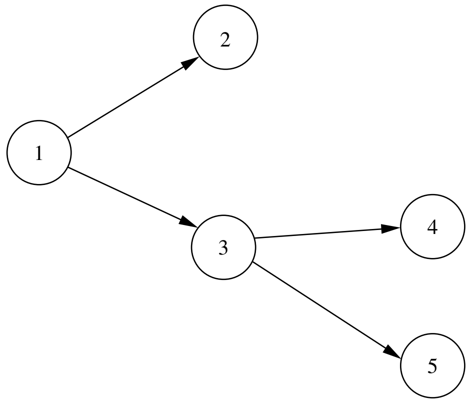
Note that because $b_3>b_2$, any tree inequality (15) obtained from a set R that contains node 2 and either node 4 or 5, but not node 3, will be dominated by the tree inequality obtained from $R\cup\{3\}$. For example, the tree inequality

$$x+20y_1+10y_2+5y_3+5y_4\geqslant 20 \tag{16}$$

corresponding to $R=\{1,2,4\}$ is dominated by the inequality

$$x+20y_1+10y_2+5y_3+3y_4\geqslant 20 \tag{17}$$

Figure 2. Scenario tree for Example 3.



corresponding to $\Omega_R = \{1, 2, 3, 4\}$. Moreover, the following set of six affinely independent points,

$$\begin{bmatrix} x \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 20 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 5 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 15 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 17 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

are feasible to X_{SDK} and satisfy (17) at equality. Hence, (17) defines a facet of the convex hull of X_{SDK}.

Example 3 illustrates that the tree inequalities (15) can be facet defining. A set of sufficient conditions on the problem parameters under which the tree inequalities (15) are guaranteed to be facet defining is given in Guan (2005). A particularly interesting condition is when the coefficients a_j are large relative to the right-hand sides b_i, which is studied next.

4.1. Special Case: Large Coefficients

In this section, we consider instances of X_{SDK} with large coefficients, in particular, $a_j \geqslant \max\{b_k, k \in \mathcal{V}(j)\}$ for all $j \in \mathcal{V}$, so that X_{SDK} simplifies to

$$X'_{\text{SDK}} = \left\{(x, y) \in \mathbb{R}_+ \times \{0,1\}^{|\mathcal{V}|}\colon x + M \sum_{j \in \mathcal{P}(i)} y_j \geqslant b_i, i \in \mathcal{V}\right\},$$

where $M > \max_{j \in \mathcal{V}}\{b_j\}$. We show that, in this case, the tree inequalities (15) suffice to describe the convex hull of X'_{SDK}. We use the following result, which is a simpler version of Theorem 7 in Miller and Wolsey (2003). Note that conv(X) denotes the convex hull of a set X.

Theorem 4 (Miller and Wolsey 2003). *Let*

$$Z = \{(w, z) \in \mathbb{R}_+ \times \mathbb{Z}^n_+\colon w + z_i \geqslant f_i \; i = 1, \ldots, n\},$$

with $0 \leqslant f_i < 1$ *for all* i. *Then,*

$$conv(Z) = \left\{(w, z) \in \mathbb{R}_+ \times \mathbb{R}^n_+\colon w + \sum_{i_k \in R} (f_{i_k} - f_{i_{k-1}}) z_{i_k} \geqslant f_{i_K} \right.$$
$$\left. \forall R = \{i_1, \ldots, i_K\} \subseteq \{1, \ldots, n\}\right\},$$

where $R = \{i_1, \ldots, i_K\}$ *is such that* $0 =: f_{i_0} \leqslant f_{i_1} \leqslant \cdots \leqslant f_{i_K}$. *Moreover, the set*

$$conv(Z) \cap \{z\colon Bz \leqslant d\}$$

is integral if B is the transpose of a network flow matrix and d is integral.

Theorem 5. *The tree inequalities* (15) *for all* $R \subseteq \mathcal{V}$, *together with* $x \geqslant 0$ *and* $0 \leqslant y_j \leqslant 1$ *for each* $j \in \mathcal{V}$, *describe the convex hull of* X'_{SDK}.

Proof. Consider the following system

$$Z' = \{(w, z) \in \mathbb{R}_+ \times \mathbb{Z}^{|\mathcal{V}|}_+\colon w + z_i \geqslant f_i,$$
$$0 \leqslant z_i - z_{i^-} \leqslant 1 \; \forall i \in \mathcal{V}\}$$

with $z_{1^-} := 0$ and $f_i := b_i/M$ for all $i \in \mathcal{V}$. Observe that there is a one-to-one correspondence between the solutions of the continuous relaxation of X'_{SDK} and those of the continuous relaxation of Z' through the transformation $w = x/M$ and $z_i = \sum_{j \in \mathcal{P}(i)} y_j$ for all $i \in \mathcal{V}$. Note that the transformation from z_i to y_i is given by $y_i = z_i - z_{i^-}$ for all $i \in \mathcal{V}$. Because the constraint matrix of the system $0 \leqslant z_i - z_{i^-} \leqslant 1$ for all $i \in \mathcal{V}$ is the transpose of a Network flow matrix and the right-hand sides are integral, it follows from Theorem 4 that conv(Z') is given by

$$\left\{(w, z) \in \mathbb{R}_+ \times \mathbb{R}^{|\mathcal{V}|}_+\colon w + \sum_{i_k \in R} (f_{i_k} - f_{i_{k-1}}) z_{i_k} \geqslant f_{i_K} \right.$$
$$\left. \forall R = \{i_1, \ldots, i_K\} \subseteq \mathcal{V},\; 0 \leqslant z_i - z_{i^-} \leqslant 1 \; \forall i \in \mathcal{V}\right\},$$

where $R = \{i_1, \ldots, i_K\}$ is such that $0 =: f_{i_0} \leqslant f_{i_1} \leqslant \cdots \leqslant f_{i_K}$. Transforming back to the (x, y)-space, we obtain that conv(X_{SDK}) is given by

$$\left\{(x, y) \in \mathbb{R}_+ \times \mathbb{R}^{|\mathcal{V}|}_+\colon x + \sum_{j \in \mathcal{V}_R} \Delta_j(R) y_j \geqslant b(R) \right.$$
$$\left. \forall R \subseteq \mathcal{V},\; 0 \leqslant y_i \leqslant 1 \; \forall i \in \mathcal{V}\right\}. \quad \square$$

Theorem 5 generalizes the convex hull results for the deterministic case, i.e., $|R| = 1$, in Barany et al. (1984), and the case where there are only two periods, i.e., $t(j) \leqslant 2$ for each $j \in \mathcal{V}$, in Günlük and Pochet (2001).

Example 3 (continued). If we modify the coefficients to $a_1 = a_2 = a_3 = a_4 = a_5 = 40$, then inequalities (15) corresponding to $R = \{1\}$, $\{1, 2\}$, $\{1, 3\}$, $\{1, 3, 4\}$, $\{1, 3, 5\}$, $\{1, 2, 3\}$, $\{1, 2, 3, 4\}$, $\{1, 2, 3, 5\}$, $\{1, 3, 4, 5\}$, and $\{1, 2, 3, 4, 5\}$, together with $x \geqslant 0$ and $0 \leqslant y_1, \ldots, y_5 \leqslant 1$, describe the convex hull of all feasible solutions.

4.2. Separation

For X'_{SDK}, separation of the tree inequalities (15) can be carried out by solving shortest-path problems on a directed graph G with nodes $\mathcal{V} \cup \{0\}$ (node 0 is a dummy node with $b_0 = 0$) and arcs (i, i') for all i, i' with $b_{i'} \geqslant b_i$. Given a point (x^*, y^*), the separation problem of determining whether there exists a violated tree inequality can be reduced to finding a shortest path from node 0 to node r for each $r \in \mathcal{V}$, where the length of arc (i, i') is given by $\sum_{j \in \mathcal{P}(i')} (b_{i'} - b_i) y^*_j$. This is true because a path $P = (0, i_1, i_2, \ldots, i_K)$, where $i_K = r$ in G corresponds to a valid tree inequality of the form (15) with $R = \{i_1, i_2, \ldots, i_K\}$ because the length of the path P is $\sum_{k=1}^{K} \sum_{j \in \mathcal{P}(i_k)} (b_{i_k} - b_{i_{k-1}}) y^*_j = \sum_{j \in \mathcal{V}_R} (\sum_{i_k \in R(j)} (b_{i_k} - b_{i_{k-1}})) y^*_j = \sum_{j \in \mathcal{V}_R} \Delta_j(R) y^*_j$, which plus x^* is equal to the left-hand side of the inequality (15). Therefore, there is a violated inequality with right-hand side b_{i_K} if and only if the length of a shortest path from node 0 to node $r = i_K$ is less than $b_{i_K} - x^*$. Using Dijkstra's algorithm, the separation problem can be solved in $O(|\mathcal{V}|^2)$ time and we can find as many as $|\mathcal{V}|$ violated inequalities from the shortest paths from node 0 to node k for each $k \in \mathcal{V}$.

Theorem 6. *If $a_j \geqslant \max\{b_k, k \in \mathcal{V}(j)\}$ for all $j \in \mathcal{V}$, there exists a polynomial-time separation algorithm for the tree inequalities* (15).

For X_{SDK}, i.e., when the condition that $a_j \geqslant \max\{b_k, k \in \mathcal{V}(j)\}$ does not hold, the above algorithm can be used as a separation heuristic by first finding a tree inequality assuming large coefficients—i.e., finding the set R and coefficients $\Delta_j(R)$ as above—and then tightening the coefficient $\Delta_j(R)$ of each y_j variable to $\min\{a_j, \Delta_j(R)\}$.

4.3. Dominance of Sequential Pairing

Recall that the tree inequality (15) for a set R is obtained by sequential pairing of the original inequalities of X_{SDK} corresponding to $i \in R$ according to the partial order on R. In this subsection, we argue that this pairing order is in some sense optimal.

Lemma 1. *Given $R \subseteq \mathcal{V}$, suppose that*

$$x + \sum_{j \in \mathcal{V}_R} \theta_j(R) y_j \geqslant b(R)$$

is a valid inequality obtained by an arbitrary sequence of pairings on the inequalities of X'_{SDK} corresponding to $i \in R$. Then, the valid inequality corresponding to R obtained by the same sequence of pairing operations for X_{SDK} is

$$x + \sum_{j \in \mathcal{V}_R} \min\{a_j, \theta_j(R)\} y_j \geqslant b(R).$$

Proof. We show that the conclusion holds for each pairing in the arbitrary sequence. Note that the first pairing involves two inequalities of the form (14), so the result follows from Theorem 2. For any subsequent operation, suppose that we are pairing the two inequalities

$$x + \sum_{j \in \mathcal{V}_{R_1}} \min\{a_j, \theta_j(R_1)\} y_j \geqslant b(R_1)$$

and

$$x + \sum_{j \in \mathcal{V}_{R_2}} \min\{a_j, \theta_j(R_2)\} y_j \geqslant b(R_2)$$

with $b(R_1) \leqslant b(R_2)$. Then, the coefficient of each variable y_j for the new inequality corresponding to $R = R_1 \cup R_2$ will be as follows. (Recall that $\theta_j(R) = \min\{\theta_j(R_1) + b(R_2) - b(R_1), \max\{\theta_j(R_1), \theta_j(R_2)\}\}$.)

(1) If $a_j \geqslant \theta_j(R_1)$ and $a_j \geqslant \theta_j(R_2)$, then the coefficient of y_j is $\theta_j(R) \leqslant a_j$ because $\theta_j(R) \leqslant \max\{\theta_j(R_1), \theta_j(R_2)\}$.

(2) If $a_j \geqslant \theta_j(R_1)$ and $a_j \leqslant \theta_j(R_2)$, then the coefficient of y_j is $\min\{\theta_j(R_1) + b(R_2) - b(R_1), a_j\} = \min\{\theta_j(R_1) + b(R_2) - b(R_1), \theta_j(R_2), a_j\} = \min\{\theta_j(R), a_j\}$, where the first equality follows from $a_j \leqslant \theta_j(R_2)$ and the second equality follows from the definition of $\theta_j(R)$.

(3) If $a_j \leqslant \theta_j(R_1)$ and $a_j \geqslant \theta_j(R_2)$, then the coefficient of y_j is $a_j \leqslant \theta_j(R)$ because $\theta_j(R) = \theta_j(R_1)$, which is based on the fact that $\theta_j(R_1) \geqslant \theta_j(R_2)$.

(4) If $a_j \leqslant \theta_j(R_1)$ and $a_j \leqslant \theta_j(R_2)$, then the coefficient of y_j is $a_j \leqslant \theta_j(R)$ because $\theta_j(R) \geqslant \min\{\theta_j(R_1), \theta_j(R_2)\}$ by (3).

Therefore, the coefficient of y_j is $\min\{a_j, \theta_j(R)\}$ for all cases and the conclusion holds. □

Theorem 7. *A valid inequality generated by an arbitrary sequence of pairing operations on a subset of the original inequalities of X_{SDK} is dominated by a convex combination of the tree inequalities* (15) *for all $R \subseteq \mathcal{V}$.*

Proof. For X'_{SDK} the claim is true because, by Theorem 5, the tree inequalities suffice to describe $\text{conv}(X'_{\text{SDK}})$. Thus, a valid inequality

$$x + \sum_{j \in \mathcal{V}_R} \theta_j(R) y_j \geqslant b(R)$$

obtained by an arbitrary sequence of pairing operations of the original inequalities for a subset $R \subseteq \mathcal{V}$, is dominated by a convex combination of tree inequalities (15),

$$x + \sum_{j \in \mathcal{V}_R^k} \Delta_j(R^k) y_j \geqslant b(R^k), \quad k = 1, \ldots, K,$$

corresponding to subsets of nodes $R^1, \ldots, R^K$. That is, there exists a set of weights $\lambda_1, \ldots, \lambda_K$ with $\lambda_k \geqslant 0$ and $\sum_{k=1}^K \lambda_k = 1$, such that for all j,

$$\theta_j(R) \geqslant \sum_{k=1}^K \lambda_k \Delta_j(R^k) \quad \text{and} \quad b(R) \leqslant \sum_{k=1}^K \lambda_k b(R^k). \tag{18}$$

Now consider X_{SDK}. According to Lemma 1, a valid inequality obtained by an arbitrary sequence of pairing operations on the original constraints of X_{SDK} corresponding to R is of the form

$$x + \sum_{j \in \mathcal{V}_R} \min\{a_j, \theta_j(R)\} y_j \geqslant b(R).$$

Similarly, a tree inequality corresponding to $R^k \subseteq \mathcal{V}$ is of the form

$$x + \sum_{j \in \mathcal{V}_R^k} \min\{a_j, \Delta_j(R^k)\} y_j \geqslant b(R^k).$$

Because $b(R) \leqslant \sum_{k=1}^K \lambda_k b(R^k)$ from (18), we only need to verify that for each $j \in \mathcal{V}$,

$$\min\left\{a_j, \theta_j(R)\right\} \geqslant \sum_{k=1}^K \lambda_k \min\{a_j, \Delta_j(R^k)\right\},$$

with $\lambda_k \geqslant 0$ and $\sum_{k=1}^{K} \lambda_k = 1$. Indeed, if $a_j \geqslant \theta_j(R)$, then we have

$$\min\{a_j, \theta_j(R)\} = \theta_j(R) \geqslant \sum_{k=1}^{K} \lambda_k \Delta_j(R^k)$$
$$\geqslant \sum_{k=1}^{K} \lambda_k \min\{a_j, \Delta_j(R^k)\},$$

where the first inequality follows from (18). On the other hand, if $a_j \leqslant \theta_j(R)$, then

$$\min\{a_j, \theta_j(R)\} = a_j = \sum_{k=1}^{K} \lambda_k a_j \geqslant \sum_{k=1}^{K} \lambda_k \min\{a_j, \Delta_j(R^k)\}. \quad \square$$

5. Stochastic Lot Sizing

A multistage stochastic integer programming formulation of the single-item stochastic lot-sizing problem defined over a scenario tree $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ is (cf. Guan et al. 2006)

$$\begin{aligned} \min \quad & \sum_{i\in\mathcal{V}} p_i(\alpha_i s_i + \beta_i x_i + \gamma_i y_i) + \alpha_{1^-} s_{1^-} \\ \text{s.t.} \quad & s_{i^-} + x_i = d_i + s_i, \quad i \in \mathcal{V}, \\ & 0 \leqslant x_i \leqslant a_i y_i, \quad i \in \mathcal{V}, \\ & s_{1^-} \geqslant 0,\ s_i \geqslant 0,\ y_i \in \{0, 1\}, \quad i \in \mathcal{V}, \end{aligned}$$

where s, x, and y denote the inventory, production, and setup variables, and the parameters d, α, β, γ, and a denote demands, holding costs, production costs, setup costs, and production capacities, respectively. Eliminating the inventory variables s_i for $i \in \mathcal{V}$ and using s to denote the initial inventory variable s_{1^-}, the feasible region of the stochastic lot-sizing problem is

$$X_{\text{SLP}} = \Big\{(s, x, y) \in \mathbb{R}_+ \times \mathbb{R}_+^{|\mathcal{V}|} \times \{0, 1\}^{|\mathcal{V}|}: \\ s + \sum_{j\in\mathcal{P}(i)} x_j \geqslant d_{1i},\ x_i \leqslant a_i y_i,\ i \in \mathcal{V}\Big\}, \quad (19)$$

where $d_{1i} = \sum_{j\in\mathcal{P}(i)} d_j$ is the cumulative demand up to node i. Replacing x_i with $a_i y_i$, we have the relaxation of X_{SLP}:

$$X_{\text{RSLP}} = \Big\{(s, y) \in \mathbb{R}_+ \times \{0, 1\}^{|\mathcal{V}|}: \\ s + \sum_{j\in\mathcal{P}(i)} a_j y_j \geqslant b_i,\ i \in \mathcal{V}\Big\}, \quad (20)$$

where $b_i = d_{1i}$. Note that X_{RSLP} is precisely the stochastic dynamic knapsack set X_{SDK}. Hence, the valid inequalities developed in §4 are also valid for X_{RSLP}, and therefore for X_{SLP}. The following lemma allows us to include the x_j variables in these valid inequalities.

Lemma 2. *If $s + \sum_{j\in\mathcal{V}_R} \pi_j y_j \geqslant \pi_0$ is a valid inequality for X_{SLP} for some $R \subseteq \mathcal{V}$, and $S_R \subseteq \mathcal{V}_R$, then*

$$s + \sum_{j\in S_R} x_j + \sum_{j\in \bar{S}_R} \pi_j y_j \geqslant \pi_0, \quad (21)$$

where $\bar{S}_R = \mathcal{V}_R \backslash S_R$ is a valid inequality for X_{SLP}.

Proof. Consider a point $(s^*, x^*, y^*) \in X_{\text{SLP}}$. Now construct a point $(\hat{s}, \hat{x}, \hat{y})$ such that $\hat{x}_j = x_j^*$ and $\hat{y}_j = y_j^*$ for each $j \in \mathcal{V}_R \backslash S_R$, $\hat{x}_j = \hat{y}_j = 0$ for each $j \in S_R$, and $\hat{s} = s^* + \sum_{j\in S_R} x_j^*$. Then, for each $i \in \mathcal{V}$,

$$\hat{s} + \sum_{j\in\mathcal{P}(i)} \hat{x}_i = s^* + \sum_{j\in S_R} x_i^* + \sum_{j\in\mathcal{P}(i)\backslash S_R} x_i^* \geqslant s^* + \sum_{j\in\mathcal{P}(i)} x_i^* \geqslant d_{1i}.$$

Thus, $(\hat{s}, \hat{x}, \hat{y}) \in X_{\text{SLP}}$. Then,

$$\pi_0 \leqslant \hat{s} + \sum_{j\in\mathcal{V}_R} \pi_j \hat{y}_j = s^* + \sum_{j\in S_R} x_j^* + \sum_{j\in\mathcal{V}_R\backslash S_R} \pi_j y_j^*.$$

Therefore, inequality (21) is valid for X_{SLP}. $\square$

Theorem 8. *Given a subset $R \subseteq \mathcal{V}$ and a subset $S_R \subseteq \mathcal{V}_R$, the inequality*

$$s + \sum_{j\in S_R} x_j + \sum_{j\in \bar{S}_R} \phi_j(R) y_j \geqslant b(R) \quad (22)$$

is valid for X_{SLP}, where $\bar{S}_R = \mathcal{V}_R \backslash S_R$ and $\phi_j(R) = \min\{a_j, \Delta_j(R)\}$.

Proof. The result follows immediately by applying Lemma 2 to inequality (15) for the stochastic dynamic knapsack relaxation X_{RSLP}. $\square$

For constant production capacities, i.e., $a_j = a$ for all $j \in \mathcal{V}$, valid inequalities derived from a mixing set relaxation of X_{SLP} are presented in Di Summa and Wolsey (2008).

5.1. The Uncapacitated Case

If $a_j \geqslant \max\{b_k, k \in \mathcal{V}(j)\}$ for all $j \in \mathcal{V}$, then the production variables are uncapacitated and (22) simplifies to

$$s + \sum_{j\in S_R} x_j + \sum_{j\in \bar{S}_R} \Delta_j(R) y_j \geqslant b(R). \quad (23)$$

Inequalities of the form (23) were obtained by Guan et al. (2006) for certain restricted subsets R. Specifically, Guan et al. (2006) consider subsets $\mathcal{Q} = \{i_1, \ldots, i_K\} \subseteq \mathcal{V}$ with $d_{1i_1} \leqslant \cdots \leqslant d_{1i_K}$ such that, for any $j \in \mathcal{V}$, if $\{i_m, i_n\} \subseteq \mathcal{Q}(j) := \mathcal{Q} \cap \mathcal{V}(j)$, then $\{i_{m+1}, i_{m+2}, \ldots, i_{n-1}\} \subset \mathcal{Q}(j)$. Valid path inequalities corresponding to the nodes in $\mathcal{Q}$ are then combined to obtain a valid inequality, called a $(\mathcal{Q}, S_{\mathcal{Q}})$ inequality, for X_{SLP}. It can be shown that every such $(\mathcal{Q}, S_{\mathcal{Q}})$ inequality is an inequality of the form (23) with $R = \Omega_{\mathcal{Q}}$ (Guan et al. 2005). Clearly, not every tree inequality (23) is a $(\mathcal{Q}, S_{\mathcal{Q}})$ inequality, and moreover, such tree inequalities may be required in the description of the convex hull of X_{SLP}. The following example illustrates this.

Example 4. Consider a stochastic uncapacitated lot-sizing problem for the scenario tree structure shown in Figure 1. Let $d_1 = 10$, $d_2 = 15$, $d_3 = 5$, and $d_4 = 20$. The tree inequality corresponding to $R = \{1,3,2,4\}$ with $S_R = \{1\}$ is

$$s + x_1 + 10y_2 + 15y_3 + 10y_4 \geqslant 35.$$

This inequality is facet defining. However, it is not a $(\mathcal{Q}, S_{\mathcal{Q}})$ inequality because the set $\{1,3,2,4\}$ does not satisfy the necessary requirements on $\mathcal{Q}$ (note that $b_1 < b_3 < b_2 < b_4$ and $\mathcal{Q}(3) = \{3,4\}$ but $2 \notin \mathcal{Q}(3)$).

5.2. Separation

Consider the uncapacitated case first. Separation of the tree inequalities (22) corresponds to finding a subset of nodes R and a partition of $\mathcal{V}_R$ into S_R and $\bar{S}_R$. Unfortunately, this does not appear to be easy, so we use a heuristic approach. Recall that if we fix $S_R = \varnothing$, then the lot-sizing tree inequalities (22) are the dynamic knapsack tree inequalities (15), and hence can be separated exactly in polynomial time by a shortest-path scheme as in §4.2. Once we have identified the most violated dynamic knapsack tree inequality (15), i.e., a subset of nodes R, we can then set $S_R = \{j \in \mathcal{V}_R : x_j^* < \phi_j(R) y_j^*\}$, where (x^*, y^*) is the current fractional solution, to find a tree inequality (22). This heuristic can be further enhanced by setting S_R as above for each of the dynamic knapsack tree inequalities identified in the course of the separation algorithm of §4.2 and obtaining a resulting lot-sizing tree inequality, and then choosing the most violated lot-sizing tree inequality from among these.

For the capacitated case, as in §4.2, we first use the above scheme to find tree inequalities assuming $a_j \geqslant \max\{b_k, k \in \mathcal{V}(j)\}$ and then tighten the coefficient of each y_j variable by taking the minimum value of a_j and the coefficient obtained by the shortest-path algorithm.

6. Computational Experiments

In this section, we present computational results with a branch-and-cut algorithm to demonstrate the effectiveness of the inequalities generated by our pairing scheme on randomly generated instances of single-item uncapacitated and capacitated stochastic lot-sizing problems. All computations have been carried out on a Linux workstation with dual 2.4 GHz Intel Xeon processors and 2 GB RAM.

6.1. Instance Generation

Instances were generated based on different structures of the underlying scenario trees, different ratios of the production cost to the inventory holding cost, and different ratios of the setup cost to the inventory holding cost. We assumed that the underlying scenario tree is balanced with T stages and K branches per stage. For the uncapacitated instances, we used stage-branch combinations $(T,K) = (9,2)$, $(10,2)$, $(6,3)$, and $(7,3)$; production to holding cost ratios $\beta/h = 2$ and 4; and setup to holding cost ratios $\gamma/h = 200$ and 400.

Corresponding to each of the 16 combinations of K, T, γ/h, and β/h, three random instances were generated. In these instances, corresponding to each node i of the tree, the holding cost h_i is a random number uniformly distributed in the interval $[0, 10]$; the production cost β_i is uniformly distributed in the interval $[0.8(\alpha/h)\bar{h}, 1.2(\alpha/h)\bar{h}]$, where $\bar{h}$ is the average holding cost; the setup cost γ_i is uniformly distributed in the interval $[0.8(\beta/h)\bar{h}, 1.2(\beta/h)\bar{h}]$; and demand d_i is uniformly distributed in the interval $[0, 100]$. Finally, all K children of a node occur with equal probability $1/K$.

For the capacitated instances, we used $(T,K) = (9,2)$ and $(6,3)$. Two sizes of production capacities a_i were used, a large capacity that is uniformly distributed in the interval $[40T, 60T]$ and a small capacity that is uniformly distributed in the interval $[20T, 40T]$. All other parameters were generated in the same way as in the uncapacitated case.

6.2. Results

We used CPLEX 8.1 in the default mode as a control and compared its performance to our customized algorithm, which augments default CPLEX by repeatedly solving the linear programming relaxation and adding the most violated cut found by the separation heuristics until no more cuts can be found, at each node of the branch-and-cut tree. To get a better understanding of the value of our cuts, we also evaluated how much they improved the LP at the root node.

Computational results for the stochastic uncapacitated case are shown in Tables 1 and 2. Table 1 gives the effectiveness of the tree inequalities in tightening the LP relaxation gap at the root node. The LP relaxation gap of the original formulation without adding any of our cuts is shown in the column labelled "LP gap %." It is calculated with respect to the best feasible solution found by our branch-and-cut algorithm. The column labelled "Path" corresponds to the results from adding all violated path inequalities (i.e., $|R| = 1$); the column labelled "$(\mathcal{Q}, S_{\mathcal{Q}})$" corresponds to the results after adding violated $(\mathcal{Q}, S_{\mathcal{Q}})$ inequalities as done in Guan et al. (2006); the column labelled "Tree" corresponds to the results after adding violated tree inequalities (22) in a separate experiment by the heuristic separation algorithm.

For each combination of K, T, γ/h, and β/h, there are two rows corresponding to the columns labelled "Path," "$(\mathcal{Q}, S_{\mathcal{Q}})$," and "Tree." The first row gives the LP relaxation gap after adding inequalities, and the second row gives the number of inequalities added. Note that all reported numbers are averages over three instances. Significant tightening of the LP relaxation is achieved with the tree inequalities. In most cases, the LP relaxation gap is reduced from over 20% to less than 1%. Furthermore, in most cases,

Table 1. Results of the root node for the uncapacitated case.

K	T	γ/h	β/h	LP gap (%)	Path	$(\mathscr{Q}, S_{\mathscr{Q}})$	Tree	$\lvert R\rvert$
2	9	200	2	16.74	2.37	0.77	0.11	(2, 8, 28)
					433	12,440	916	
2	9	200	4	13.76	2.20	0.92	0.30	(2, 8, 28)
					417	13,106	784	
2	9	400	2	20.67	3.54	1.04	0.13	(2, 9, 31)
					318	13,094	1,027	
2	9	400	4	18.22	3.10	1.37	0.33	(2, 8, 24)
					359	12,760	734	
2	10	200	2	15.56	2.73	1.26	1.01	(2, 7, 23)
					849	15,399	620	
2	10	200	4	20.12	3.97	1.50	0.92	(2, 7, 23)
					631	15,809	1,165	
2	10	400	2	12.63	2.30	0.93	0.33	(2, 9, 30)
					846	15,254	1,112	
2	10	400	4	18.90	4.66	1.96	0.76	(2, 9, 30)
					729	15,656	1,579	
3	6	200	2	19.19	3.92	0.96	0.32	(2, 12, 34)
					197	11,059	849	
3	6	200	4	16.13	3.57	0.73	0.18	(2, 12, 33)
					183	10,877	922	
3	6	400	2	25.04	5.11	1.67	0.16	(2, 14, 36)
					172	11,265	1,127	
3	6	400	4	22.05	4.65	1.59	0.32	(2, 13, 40)
					168	11,405	1,158	
3	7	200	2	22.01	4.17	1.96	1.10	(2, 9, 31)
					739	15,020	1,386	
3	7	200	4	17.64	3.12	1.51	1.35	(2, 10, 30)
					696	14,507	991	
3	7	400	2	30.80	8.92	3.82	2.04	(2, 12, 35)
					634	14,779	1,656	
3	7	400	4	24.48	4.24	2.31	0.98	(2, 12, 38)
					638	14,810	2,056	

we observe significant improvement by adding tree inequalities to the path inequalities. The tree inequalities also give better performance than the $(\mathscr{Q}, S_{\mathscr{Q}})$ inequalities, and many fewer tree inequalities are needed to get this improved performance.

The final column labelled "$\lvert R\rvert$" records the minimum, average, and maximum number of elements in R corresponding to each tree inequality, which gives an indication of how much of the scenario tree is used by each inequality. We observed that $\lvert R\rvert$ ranges from 2 to 40 with an average around 10, and that the average $\lvert R\rvert$ for the cases with $K=2$ is less than those with $K=3$.

Table 2 presents our branch-and-cut results. We compared the number of cuts added by default CPLEX and by our branch-and-cut scheme, respectively, the relative optimality gap upon termination, the number of nodes explored (apart from the root node), and the total CPU time. For the two rows corresponding to each combination of K, T, γ/h, and β/h in the table, the first one gives the performance of default CPLEX and the second one gives the performance of our branch-and-cut scheme. The reported data

Table 2. Results of the branch-and-cut algorithm for the uncapacitated case.

K	T	γ/h	β/h	No. of cuts	Optimality gap	No. of nodes	CPU secs.
2	9	200	2	563	0.59 [3]	1,657,049	$***$
				3,823	0	248	149.4
2	9	200	4	551	0.47 [3]	1,640,825	$***$
				8,425	0	189	894.3
2	9	400	2	596	0.99 [3]	1,570,548	$***$
				14,642	0.02 [1]	264	956.5
2	9	400	4	521	0.92 [3]	1,616,461	$***$
				16,420	0.08 [1]	190	437
2	10	200	2	780	1.78 [3]	943,455	$***$
				18,567	0.17 [2]	655	3,264
2	10	200	4	1,026	0.95 [3]	835,008	$***$
				21,241	0.05 [2]	133	3,521
2	10	400	2	885	2.1 [3]	891,822	$***$
				17,450	0.42 [3]	946	$***$
2	10	400	4	858	2.02 [3]	924,457	$***$
				27,642	1.31 [3]	85	$***$
3	6	200	2	723	0.61 [3]	1,996,296	$***$
				9,046	0.08 [1]	76	87
3	6	200	4	801	0.24 [3]	1,988,059	$***$
				5,545	0	156	512.5
3	6	400	2	566	0.81 [3]	2,608,384	$***$
				7,535	0	291	1,045
3	6	400	4	546	0.65 [3]	3,005,068	$***$
				9,812	0.17 [1]	130	195.1
3	7	200	2	1,129	2.29 [3]	790,023	$***$
				29,009	0.69 [3]	24	$***$
3	7	200	4	1,014	1.77 [3]	828,985	$***$
				37,766	0.98 [3]	45	$***$
3	7	400	2	945	3.62 [3]	1,000,364	$***$
				25,187	1.24 [3]	0	$***$
3	7	400	4	1,069	2.55 [3]	1,123,622	$***$
				26,690	0.82 [3]	0	$***$

is averaged over three instances. In the column labelled "Optimality gap," the numbers in square brackets indicate the number of instances not solved to default CPLEX optimality tolerance within the allotted time limit of one hour. The default CPLEX MIP solver added several types of cuts, including cover cuts, flow cuts, Gomory fractional cuts, and mixed-integer rounding cuts. Our branch-and-cut algorithm added up to 500 tree inequalities as cuts at each node after the CPLEX default cuts have been added. For the total CPU time, as shown in the column labelled "CPU secs." we report the average CPU time for instances that are solved to default CPLEX optimality tolerance within the allotted time limit of one hour. Label "$***$" represents the case that no instance is solved to default CPLEX optimality tolerance within the allotted time.

Our branch-and-cut algorithm performs much better than default CPLEX. Our algorithm solves to optimality half of the instances for $K=2$ and 10 out of 24 instances for $K=3$, whereas the default CPLEX cannot solve any of the instances to optimality. For those instances unsolved

Table 3. Results of the root node for the capacitated case.

Capacity	K	T	γ/h	β/h	LP gap (%)	Path	Tree	$\|R\|$
$U[40T, 60T]$	2	9	200	2	14.57	2.64	0.17	(2, 6, 19)
						439	657	
	2	9	200	4	11.18	2.16	0.28	(2, 6, 20)
						404	598	
	2	9	400	2	17.54	3.28	0.24	(2, 8, 24)
						332	721	
	2	9	400	4	14.73	3.18	0.25	(2, 9, 28)
						342	1,013	
	3	6	200	2	13.84	5.16	1.62	(2, 8, 27)
						202	897	
	3	6	200	4	10.91	4.04	1.33	(2, 8, 21)
						208	785	
	3	6	400	2	16.06	8.00	2.80	(2, 10, 29)
						178	894	
	3	6	400	4	14.18	7.34	2.50	(2, 12, 32)
						182	1,039	
$U[20T, 40T]$	2	9	200	2	12.82	3.16	0.49	(2, 6, 21)
						403	696	
	2	9	200	4	9.56	2.65	0.43	(2, 6, 20)
						383	701	
	2	9	400	2	15.07	4.56	1.23	(2, 8, 20)
						316	893	
	2	9	400	4	12.33	4.06	1.03	(2, 8, 24)
						339	944	
	3	6	200	2	12.36	4.35	0.48	(2, 8, 24)
						189	773	
	3	6	200	4	8.64	3.85	0.43	(2, 8, 22)
						185	877	
	3	6	400	2	14.08	6.35	1.19	(2, 9, 25)
						162	918	
	3	6	400	4	11.28	5.79	1.12	(2, 10, 28)
						175	1,004	

by both algorithms, our algorithm yielded much smaller optimality gaps. Moreover, our cuts dramatically reduced the number of nodes in the branch-and-bound tree and, although we added many more cuts, the running times were smaller as well. Furthermore, by limiting the number of cuts added as a function of tree depth, we were able to decrease the running times a bit more than those shown in Table 2.

Tables 3 and 4 present results for the capacitated case. We also tested three instances for each combination. Table 3 shows the optimality gap reduction after adding path inequalities and the substantially bigger reductions after adding tree inequalities at the root node. For the branch-and-cut algorithm, as shown in Table 4, default CPLEX cannot solve any of the large-capacity instances to optimality, whereas our algorithm solves 16 out of 24 instances to optimality, including all two-branch instances. For those unsolved instances, our algorithm obtains smaller optimality gaps, and all final gaps are smaller than 0.5%. For the small-capacity case, 21 out of the 24 instances

Table 4. Results of the branch-and-cut algorithm for the capacitated case.

Capacity	K	T	γ/h	β/h	No. of cuts	Optimality gap	No. of nodes	CPU secs.
$U[40T, 60T]$	2	9	200	2	590	0.55 [3]	1,680,099	∗∗∗
					2,695	0	194	73.3
	2	9	200	4	569	0.36 [3]	1,682,063	∗∗∗
					4,567	0	215	121.2
	2	9	400	2	538	1.11 [3]	1,727,792	∗∗∗
					6,498	0	208	244.4
	2	9	400	4	551	0.85 [3]	1,801,839	∗∗∗
					10,789	0	257	821.9
	3	6	200	2	487	0.37 [3]	2,370,519	∗∗∗
					5,300	0	222	204.4
	3	6	200	4	530	0.18 [3]	2,167,495	∗∗∗
					5,057	0	173	215.6
	3	6	400	2	561	1.1 [3]	2,164,056	∗∗∗
					24,018	0.45 [3]	245	∗∗∗
	3	6	400	4	586	0.94 [3]	2,053,463	∗∗∗
					24,398	0.47 [3]	284	∗∗∗
$U[20T, 40T]$	2	9	200	2	561	0.45 [3]	1,762,030	∗∗∗
					3,396	0	248	138.6
	2	9	200	4	589	0.30 [3]	1,732,004	∗∗∗
					4,328	0	266	269.1
	2	9	400	2	653	0.82 [3]	1,667,112	∗∗∗
					6,915	0	414	626.3
	2	9	400	4	573	0.63 [3]	1,806,365	∗∗∗
					8,940	0	497	835.6
	3	6	200	2	630	0.23 [2]	1,494,500	2,444.6
					10,948	0.15 [1]	211	1,269.3
	3	6	200	4	520	0.21 [2]	1,800,160	2,404.0
					5,096	0	209	222.7
	3	6	400	2	571	0.26 [2]	1,504,807	2,428.7
					14,879	0.15 [1]	230	1,448
	3	6	400	4	483	0.24 [2]	1,835,754	2,411.3
					11,406	0.06 [1]	384	1,323.3

are solved to optimality by our algorithm, whereas default CPLEX can only solve 4 out of the 24 instances to optimality. Our final optimality gaps are within 0.15%.

7. Conclusions and Future Research

We have presented a general method for generating valid inequalities for multistage stochastic integer programs based on combining inequalities that are valid for the individual scenarios. We have applied the method to a stochastic version of a dynamic knapsack problem and to stochastic lot-sizing problems. Our computational results show that these new inequalities are very effective in a branch-and-cut algorithm and give much better results than default CPLEX. Because multistage stochastic integer programs are very difficult to solve, and arise in many domains, including network reliability, routing, capacity planning, and scheduling, we are now investigating the application of our method to different structural models. Decomposition methods involving Lagrangian relaxation (Carøe and Schultz 1999, Nowak and Römisch 2000) and column generation (Lulli and Sen

2004, Sen 2005, Singh et al. 2008) have been very effective in solving various classes of multistage stochastic integer programs. Integration of the proposed cut generation scheme within such decomposition frameworks is an important unresolved issue.

Acknowledgments

This research was supported in part by the National Science Foundation under grants DMI-0121495, DMI-0522485, CMMI-0700868, and IIP-0725843. The authors are indebted to two anonymous referees for helpful comments.

References

Ahmed, S., A. J. King, G. Parija. 2003. A multi-stage stochastic integer programming approach for capacity expansion under uncertainty. *J. Global Optim.* **26** 3–24.

Barany, I., T. van Roy, L. A. Wolsey. 1984. Uncapacitated lot-sizing: The convex hull of solutions. *Math. Programming Stud.* **22** 32–43.

Carøe, C. C., R. Schultz. 1999. Dual decomposition in stochastic integer programming. *Oper. Res. Lett.* **24** 37–45.

Cook, W., R. Kannan, A. J. Schrijver. 1990. Chvátal closures for mixed integer programming problems. *Math. Programming* **47** 155–174.

Di Summa, M., L. A. Wolsey. 2008. Lot-sizing on a tree. *Oper. Res. Lett.* **36**(1) 7–13.

Guan, Y. 2005. Pairing inequalities and stochastic lot-sizing problems: A study in integer programming. Ph.D. thesis, Georgia Institute of Technology, Atlanta.

Guan, Y., S. Ahmed, G. L. Nemhauser. 2007. Sequential pairing of mixed integer inequalities. *Discrete Optim.* **4** 21–39.

Guan, Y., S. Ahmed, G. L. Nemhauser, A. J. Miller. 2006. A branch-and-cut algorithm for the stochastic uncapacitated lot-sizing problem. *Math. Programming* **105** 55–84.

Günlük, O., Y. Pochet. 2001. Mixing MIR inequalities for mixed integer programs. *Math. Programming* **90** 429–457.

Loparic, M., H. Marchand, L. A. Wolsey. 2003. Dynamic knapsack sets and capacitated lot-sizing. *Math. Programming* **95** 53–69.

Lulli, G., S. Sen. 2004. A branch and price algorithm for multi-stage stochastic integer programming with application to stochastic batch-sizing problems. *Management Sci.* **50** 786–796.

Miller, A. J., L. A. Wolsey. 2003. Tight formulations for some simple mixed integer programs and convex objective integer programs. *Math. Programming* **78** 73–88.

Nemhauser, G. L., L. A. Wolsey. 1988. *Integer and Combinatorial Optimization*. Wiley, New York.

Nemhauser, G. L., L. A. Wolsey. 1990. A recursive procedure for generating all cuts for 0–1 mixed integer programs. *Math. Programming* **46** 379–390.

Nowak, M. P., W. Römisch. 2000. Stochastic Lagrangian relaxation applied to power scheduling in a hydro-thermal system under uncertainty. *Ann. Oper. Res.* **100** 251–272.

Römisch, W., R. Schultz. 2001. Multistage stochastic integer programs: An introduction. M. Grötschel, S. O. Krumke, J. Rambau, eds. *Online Optimization of Large Scale Systems*. Springer-Verlag, Berlin-Dahlem, 579–598.

Sen, S. 2005. Algorithms for stochastic mixed-integer programming models. K. Aardal, G. L. Nemhauser, R. Weismantel, eds. *Handbook of Discrete Optimization*. North-Holland Publishing Co., Amsterdam, 515–558.

Singh, K., A. B. Philpott, K. Wood. 2008. Dantzig-Wolfe decomposition for solving multi-stage stochastic capacity planning problems. Under review.

Takriti, S., J. R. Birge, E. Long. 1996. A stochastic model for the unit commitment problem. *IEEE Trans. Power Systems* **11** 1497–1508.