A finite branch-and-bound algorithm for two-stage stochastic integer programs *

Shabbir Ahmed [a], Mohit Tawarmalani [b], Nikolaos V. Sahinidis [c] †

[a] *School of Industrial & Systems Engineering, Georgia Institute of Technology, 765 Ferst Drive, Atlanta, GA 30332*

[b] *Krannert School of Management, Purdue University, West Lafayette IN 47907*

[c] *Department of Chemical and Biomolecular Engineering, University of Illinois, 600 South Mathews Avenue, Urbana, IL 61801*

June 16, 2000; Revised June 10, 2002, March 31, 2003

Abstract

This paper addresses a general class of two-stage stochastic programs with integer recourse and discrete distributions. We exploit the structure of the value function of the second-stage integer problem to develop a novel global optimization algorithm. The proposed scheme departs from those in the current literature in that it avoids explicit enumeration of the search space while guaranteeing finite termination. Computational experiments on standard test problems indicate superior performance of the proposed algorithm in comparison to those in the existing literature.

Keywords: stochastic integer programming, branch-and-bound, finite algorithms.

1 Introduction

Under the two-stage stochastic programming paradigm, the decision variables of an optimization problem under uncertainty are partitioned into two sets. The first-stage variables are those that have to be decided before the actual realization of the uncertain parameters. Subsequently, once the random events have presented themselves, further design or operational policy improvements

*The authors wish to acknowledge partial financial support from the IBM Research Division, ExxonMobil Upstream Research Company, and the National Science Foundation under awards DMI 95-02722, DMI 00-99726, and DMI 01-15166.

†Corresponding author (email: nikos@uiuc.edu)

can be made by selecting, at a certain cost, the values of the second-stage or *recourse* variables. The goal is to determine first-stage decisions such that the sum of first-stage cost and the expected recourse cost is minimized. A standard formulation of the two-stage stochastic program is as follows (cf., [2]):

$$\begin{aligned} (\text{2SSP}): \quad z = \quad & \min \quad c^T x + \mathrm{E}_{\omega \in \Omega}[Q(x, \omega)] \\ & \text{s.t.} \quad x \in X, \end{aligned} \tag{1}$$

with

$$\begin{aligned} Q(x, \omega) = \quad & \min \quad f(\omega)^T y \\ & \text{s.t.} \quad D(\omega) y \geq h(\omega) + T(\omega) x \\ & \qquad\; y \in Y, \end{aligned} \tag{2}$$

where $X \subseteq \mathbb{R}^{n_1}$, $c \in \mathbb{R}^{n_1}$, and $Y \subseteq \mathbb{R}^{n_2}$. Here, ω is a random variable from a probability space $(\Omega, \mathcal{F}, \mathcal{P})$ with $\Omega \subseteq \mathbb{R}^k$, $f : \Omega \to \mathbb{R}^{n_2}$, $h : \Omega \to \mathbb{R}^{m_2}$, $D : \Omega \to \mathbb{R}^{m_2 \times n_2}$, $T : \Omega \to \mathbb{R}^{m_2 \times n_1}$. Problem (1) with variables x constitute the first-stage which needs to be decided prior to the realization of the uncertain parameters ω; and (2) with variables y constitute the second stage.

In case of linear constraints and variables, (2SSP) is referred to as a two-stage stochastic *linear* program. For a given value of the first-stage variables x, the second-stage problem decomposes into independent linear subproblems, one for each realization of the uncertain parameters. This decomposability property, along with the convexity of the second-stage linear programming value function $Q(\cdot, \omega)$ [24], has been exploited to develop a number of decomposition-based and gradient-based algorithms. For an extensive discussion of stochastic linear programming, the reader is referred to standard text books [2, 10].

In contrast to stochastic linear programming, the study of stochastic *integer* programs, those that have integrality requirements in the second stage, is very much in its infancy. As reviewed recently in [1], these problems arise in many contexts, including the modeling of risk objectives in stochastic linear programming, as well as when the second stage involves scheduling decisions, routing decisions, resource acquisition decisions, fixed-charge costs, and change-over costs. The main difficulty in solving stochastic integer programs is that the value function $Q(\cdot, \omega)$ is not necessarily convex but only lower semicontinuous (l.s.c.) [3]. Thus, standard convex programming based approaches that work nicely for stochastic linear programs, break down when second-stage integer variables are present. As an illustration of the non-convex nature of stochastic integer programs, consider the following example from [21]:

$$\begin{aligned} (\text{EX}): \quad & \min \quad -1.5x_1 - 4x_2 + E[Q(x_1, x_2, \omega_1, \omega_2)] \\ & \text{s.t.} \quad 0 \leq x_1, x_2 \leq 5, \end{aligned}$$

where

$$Q(x_1, x_2, \omega_1, \omega_2) = \quad \min \quad -16y_1 - 19y_2 - 23y_3 - 28y_4$$

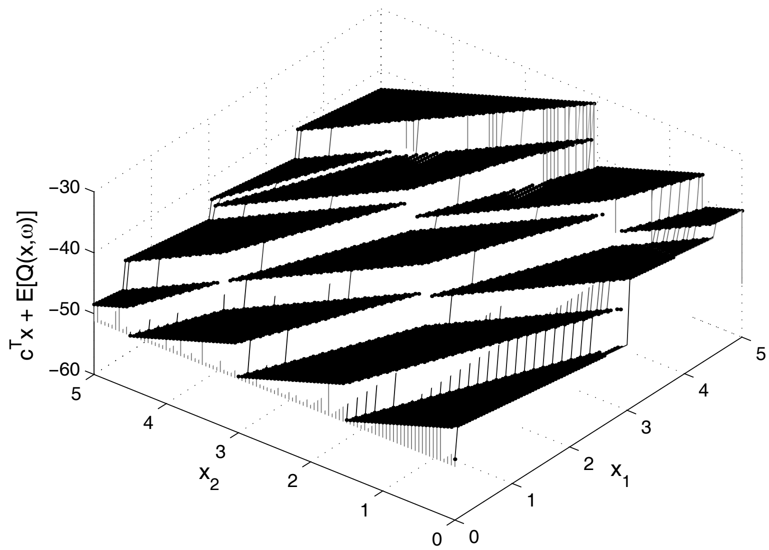

Figure 1: Objective function of (EX)

$$\begin{aligned} \text{s.t.} \quad & 2y_1 + 3y_2 + 4y_3 + 5y_4 \leq \omega_1 - \frac{1}{3}x_1 - \frac{2}{3}x_2 \\ & 6y_1 + y_2 + 3y_3 + 2y_4 \leq \omega_2 - \frac{2}{3}x_1 - \frac{1}{3}x_2 \\ & y_1, y_2, y_3, y_4 \in \{0, 1\}, \end{aligned}$$

and $(\omega_1, \omega_2) \in \{5, 15\} \times \{5, 15\}$ with a uniform probability distribution. Figure 1 shows the objective function of (EX) in the space of the first-stage variables. The highly discontinuous (lower semicontinuous) and multi-extremal nature of the function is clearly observed. Thus, in general, stochastic integer programming constitutes globally minimizing a highly non-convex function.

For problems where the second-stage recourse matrix D possesses a special structure known as *simple recourse*, Klein Haneveld *et al.* [11, 12] proposed solution schemes based upon constructing the convex envelope of the second-stage value function. For more general recourse structure, Laporte and Louveaux [14] proposed a decomposition-based approach for the case when first-stage variables are pure binary. This restriction allows for the construction of optimality cuts that approximate the non-convex second-stage value function at the binary first-stage solutions. The authors proposed a branch-and-bound algorithm to search the space of the first-stage variables for a globally optimal solution, while using optimality cuts to approximate the second-stage value function. Finite termination of the algorithm is obvious since the number of first-stage solutions is finite.

Unfortunately, the algorithm is not applicable if any of the first-stage variables is continuous. Carøe and Tind [7] generalized this algorithm for mixed-integer first- and second-stage variables. Their method uses non-linear integer programming dual functions to approximate the second-stage value function in the space of the first-stage variables. The resulting master problem then consists of non-linear (possibly discontinuous) cuts, and no practical method for its solution is currently known.

Carøe [5, 6] used the scenario decomposition approach of Rockafellar and Wets [17] to develop a branch-and-bound algorithm for stochastic integer programs. Lower bounds were obtained from the Lagrangian dual derived by dualizing the non-anticipativity constraints. The subproblems of the Lagrangian dual correspond to the scenarios and include variables and constraints from both the first and second stage. These subproblems are more difficult to solve than in Benders-based methods, where a subproblem corresponds to only the second-stage problem for a particular scenario. Furthermore, although the Lagrangian dual provides very tight bounds, its solution requires the use of subgradient methods and is computationally expensive. A major limitation of this approach is that finite termination is guaranteed only if the first-stage variables are purely discrete, or if an $\epsilon-$optimal termination criterion with $\epsilon > 0$ is used [5, 6].

Norkin *et al.* [15] proposed a stochastic branch-and-bound algorithm for minimizing the expected value of an arbitrary function over a finite set. To avoid explicit computation of the value function, the authors use Monte Carlo sampling based upper and lower bounds, statistical fathoming rules, and standard branching techniques. Almost sure convergence of the method was established. The authors used this method to solve a class of two-stage stochastic integer programs with pure integer first- and second-stage variables.

Schultz *et al.* [21] proposed a finite scheme for two-stage stochastic programs with discrete distributions and pure-integer second-stage variables. For this problem, the authors of [21] observe that only integer values of the right-hand-side parameters of the second-stage problem are relevant. This fact is used to identify a countable set, called the candidate set, in the space of the first-stage variables containing the optimal solution. In its basic form, the scheme outlined in [21] corresponds to complete enumeration of the candidate set to search for the optimal solution. Evaluation of an element of the set requires the solution of second-stage integer subproblems corresponding to all possible realizations of the uncertain parameters. Thus, explicit enumeration of all elements is, in general, computationally prohibitive. In [21], various ideas are implemented to reduce the number of candidate solutions to be evaluated. Moreover, instead of solving the scenario integer programs individually, the authors exploited the common structure of these problems by using a Gröbner basis strategy.

A detailed discussion on various algorithms for stochastic integer programming can be found in the survey of Klein Haneveld and van der Vlerk [13].

In this paper, we develop a branch-and-bound algorithm for the global optimization of two-stage stochastic integer programs with discrete distributions, mixed-integer first-stage variables, and pure-integer second-stage variables. The main difficulty with applying branch-and-bound to a (semi-)continuous domain is that the resulting approach may not be finite, *i.e.*, infinitely many subdivisions may be required for the lower and upper bounds to become exactly equal. With the exception of [21], all existing practical algorithms for general stochastic integer programming also rely on applying branch-and-bound to the first-stage variables to deal with the non-convexities of the value function. Consequently, finite termination of these algorithms is not guaranteed unless the first-stage variables, *i.e.*, the search space, is purely discrete. For the algorithm proposed in this paper, we prove finite termination. The method differs from the finite algorithm of [21], in that it avoids explicit enumeration of all discontinuous pieces of the value function. Furthermore, the proposed method allows for uncertainties in the cost parameters and the constraint matrix in addition to the right-hand-sides of the recourse problem.

The key concept behind our development is to reformulate the problem via a variable transformation that induces special structure to the discontinuities of the value function. This structure is exploited through: (a) a branching strategy that isolates the discontinuous pieces and eliminates discontinuities, and (b) a bounding strategy that provides an exact representation of the value function of the second-stage integer program in the absence of discontinuities. Finiteness of the method is a consequence of the fact that, within a bounded domain, there is only a finite number of such discontinuous pieces of the value function. The issue of finiteness is not only of theoretical significance—our computational experiments using standard test problems indicate faster convergence of the proposed algorithm in comparison to existing strategies in the literature.

The remainder of the paper is organized as follows. Section 2 specifies the assumptions required for the proposed algorithm. In Section 3, we present the transformed problem and discuss its relation to the original problem. Some structural results on the transformed problem are presented in Section 4. These results are used to develop a branch-and-bound algorithm in Section 5. Section 6 provides the proof of finiteness of the proposed algorithm. Some enhancements and extensions of the algorithm are suggested in Section 7. Finally, computational results are presented in Section 8.

2 Assumptions

In this paper, we address instances of (2SSP) under the following assumptions:

(A1) The uncertain parameter ω follows a discrete distribution with finite support $\Omega = \{\omega^1, \ldots, \omega^S\}$ with $\Pr(\omega = \omega^s) = p^s$.

(A2) The second-stage variables y are purely integer, *i.e.*, $y \in \mathbb{Z}^{n_2}$.

(A3) The technology matrix T linking the first- and second-stage problems is deterministic, *i.e.*, $T(\omega) = T$.

Assumption (A1) is justified by the results of Schultz [20] who showed that, if ω has a continuous distribution, the optimal solution to the problem can be approximated within any given accuracy by the use of discrete distributions.

Extensions of the proposed algorithm when assumptions (A2) and (A3) are not satisfied are briefly discussed in Section 7.

The uncertain problem parameters $(f(\omega), D(\omega), h(\omega))$ associated with a particular realization ω^s (a scenario), will be succinctly denoted by (f^s, D^s, h^s) with associated probability p^s. Without any loss of generality, we assume the first-stage variables to be purely continuous. Mixed-integer first-stage variables can be handled in the framework to follow without any added conceptual difficulty. We can then state the problem as follows:

$$\text{(2SSIP)}: \quad z = \quad \min \quad cx + \sum_{s=1}^{S} p^s Q^s(x) \qquad \text{s.t.} \quad x \in X,$$

with

$$Q^s(x) = \quad \min \quad f^s y \qquad \text{s.t.} \quad D^s y \geq h^s + Tx, \quad y \in Y \cap \mathbb{Z}^{n_2},$$

where $X \subseteq \mathbb{R}^{n_1}$, $c \in \mathbb{R}^{n_1}$, $T \in \mathbb{R}^{m_2 \times n_1}$, and $Y \subseteq \mathbb{R}^{n_2}$. For each $s = 1, \ldots, S$, $f^s \in \mathbb{R}^{n_2}$, $h^s \in \mathbb{R}^{m_2}$, and $D^s \in \mathbb{R}^{m_2 \times n_2}$. Note that the expectation operator has been replaced by a probability-weighted finite sum, and the transposes have been eliminated for simplicity.

We make the following additional assumptions for (2SSIP):

(A4) The first-stage constraint set X is non-empty and compact.

(A5) $Q^s(x) < \infty$ for all $x \in \mathbb{R}^{n_1}$ and all s.

(A6) For each s, there exists $u^s \in \mathbb{R}_+^{m_2}$ such that $u^s D^s \leq f^s$.

(A7) For each s, the second-stage constraint matrix is integral, *i.e.*, $D^s \in \mathbb{Z}^{m_2 \times n_2}$.

As detailed in Section 3, to guarantee the existence of an optimal solution and the convergence of branch-and-bound search, we require assumption (A4). Assumption (A5) is known as the *complete recourse* property [24]. In fact, we only need relatively complete recourse, *i.e.*, $Q^s(x) < \infty$ for all $x \in X$ and all s. Since X is compact, relatively complete recourse can always be accomplished by adding penalty-inducing artificial variables to the second-stage problem. However, we shall assume complete recourse for simplicity of exposition. Assumption (A6) guarantees $Q^s(x) > -\infty$ [20]. Together, (A5) and (A6) imply that $Q^s(x)$ is finite-valued and (2SSIP) is well-defined. Assumption (A7) can be satisfied by appropriate scaling whenever the matrix elements are rational.

For a given value of the first-stage variables x, the problem decomposes into S integer programs with value functions $Q^s(x)$. It is implicitly assumed that these "small" integer subproblems are easier to solve than the deterministic equivalent. Our methodology is independent of the oracle required to solve the integer subproblems. For example, when the second-stage objective function and constraint matrix are deterministic, the Gröbner basis framework described in [21] to solve many similar integer programs, can be used in this context.

Note that, for each s, $Q^s(x)$ is the value function of an integer program, and is well-known to be l.s.c. with respect to x. Blair and Jeroslow [3, 4] showed that such value functions are, in general, continuous only over set-theoretic differences of certain cones in the space of x and the discontinuities lie along the boundaries of these cones. Existing branch-and-bound methods [14, 5, 6] for stochastic integer programs attempt to partition the space of first-stage variables by branching on one variable at a time. In this way, the first-stage variable space is partitioned into (hyper)rectangular cells. Since the discontinuities are, in general, not orthogonal to the variable axes, there would always be some rectangular cell that contains a discontinuity in the interior. Thus, in the case of continuous first-stage variables, it might not be possible for the lower and upper bounds to converge for such a cell, unless the cell is arbitrarily small. This would require considerable partitioning of the first-stage variables and result in a convergent only scheme, *i.e.*, possibly infinite. In general, it is not obvious how one can partition the search space by subdividing *along* the discontinuities within a branch-and-bound framework.

Next, we propose a transformation of the problem that causes the discontinuities to be orthogonal to the variable axes. Thus, a rectangular partitioning strategy can potentially isolate the discontinuous pieces of the value function, thereby allowing upper and lower bounds to collapse finitely. This is the key to the subsequent development of a finite branch-and-bound algorithm.

3 Problem Transformation

Instead of (2SSIP), we propose to solve the following problem:

$$\text{(TP)}: \quad \begin{array}{ll} \min & f(\chi) \\ \text{s.t.} & \chi \in \mathcal{X} \end{array}$$

where

$$\begin{aligned} f(\chi) &= \Phi(\chi) + \overline{\Psi}(\chi), \\ \overline{\Psi}(\chi) &= \sum_{s=1}^{S} p^s \Psi^s(\chi), \\ \Phi(\chi) &= \min\{cx \mid Tx = \chi, x \in X\}, \\ \Psi^s(\chi) &= \min\{f^s y \mid D^s y \geq h^s + \chi, y \in Y \cap \mathbb{Z}^{n_2}\}, \text{ and} \\ \mathcal{X} &= \{\chi \in \mathbb{R}^{m_2} \mid \chi = Tx, x \in X\}. \end{aligned}$$

Variables χ are known as the "tender variables" in the stochastic programming literature. These are the variables that link the first- and second-stage problems. Instead of the first-stage variables, we propose to search the space of the tender variable for global optima. The following results establish the existence of a solution of (TP), and its relation to the original problem (2SSIP).

Theorem 3.1. *There exists an optimal solution to problem* (TP).

Proof: It follows from Assumption (A5), and the results in [3, 4, 20] that $\Psi^s(\cdot)$ is finite-valued and l.s.c. $\Phi(\cdot)$ is the value function of a linear program and, hence, piece-wise linear and convex. Thus, $f(\cdot)$ is a positive linear combination of real valued l.s.c functions and, therefore, l.s.c by Fatou's Lemma (cf. [18]). Since X is non-empty compact and T is a linear transformation, $\mathcal{X}$ is nonempty and compact. The claim then follows from Weierstrass' theorem. □

Theorem 3.2. *Let* χ^* *be an optimal solution of* (TP). *Then* $x^* \in \operatorname{argmin}\{cx \mid x \in X, Tx = \chi^*\}$ *is an optimal solution of* (2SSIP). *Furthermore, the optimal objective function values of the two problems are equal.*

Proof: First, note that, for any χ and x such that $Tx = \chi$, we have $\Psi^s(\chi) = Q^s(x)$ for all s. Then, from the definition of x^* and $\overline{\Psi}(\cdot)$, we have

$$\Phi(\chi^*) + \overline{\Psi}(\chi^*) = cx^* + \sum_{s=1}^{S} p^s Q^s(x^*). \tag{3}$$

We shall now prove the claim by contradiction. Suppose that x^* is not an optimal solution to (2SSIP). Then, there exists $x' \in X$ such that

$$cx' + \sum_{s=1}^{S} p^s Q^s(x') < cx^* + \sum_{s=1}^{S} p^s Q^(x^*). \tag{4}$$

Now, construct $\chi' = Tx'$ and note that $\chi' \in \mathcal{X}$. Since $x' \in \{x \mid x \in X, Tx = \chi'\}$, we have $f(\chi') \leq cx'$. Also, $\Psi^s(\chi') = Q^s(x')$ for all s. Equations (3) and (4), then, imply that

$$\Phi(\chi') + \overline{\Psi}(\chi') < \Phi(\chi^*) + \overline{\Psi}(\chi^*).$$

Thus, we have a contradiction. Equation (3) also establishes that the objective values of the two problems are equal. □

Theorem 3.2 implies that we can solve (2SSIP) by solving (TP). Structural properties of the latter problem are discussed next.

4 Structural Properties

Let $\Psi^s_j(\chi_j)$ denote $\Psi^s(\chi)$ as a function of the jth component ($j = 1, \ldots, m_2$) of χ. We use $\mathrm{cl}(X)$, $\partial(X)$, and $\dim(X)$ to denote the closure, the relative boundary, and the dimension of a set X, respectively. The following result is well-known.

Lemma 4.1. *For any $s = 1, \ldots, S$, and $j = 1, \ldots, m_2$, $\Psi_j^s(\chi_j)$ is l.s.c and non-decreasing in χ_j.*

Schultz *et al.* [21] proved that the second-stage value function is constant over certain subsets of the x-space. Next, we prove a similar result in the space of the tender variables.

Lemma 4.2. *For any $k \in \mathbb{Z}$, $\Psi_j^s(\chi_j)$ is constant over the interval $\chi_j \in \big(k - h_j^s - 1, k - h_j^s\big]$ for all $s = 1, \ldots, S$ and $j = 1, \ldots, m_2$.*

Proof: Since by (A7), D^s is integral, the jth constraint $(D^s y)_j \geq h_j^s + \chi_j$ implies $(D^s y)_j \geq \lceil h_j^s + \chi_j \rceil$. Thus, for any $k \in \mathbb{Z}$, $\Psi_j^s(\chi_j)$ is constant over the subset $\{(h_j^s + \chi_j) \mid \lceil h_j^s + \chi_j \rceil = k_j^s\} = \{(h_j^s + \chi_j) \mid k - 1 < h_j^s + \chi_j \leq k\}$. Equivalently, $\Psi_j^s(k)$ is constant over intervals $\chi_j \in (k - h_j^s - 1, k - h_j^s]$, $k \in \mathbb{Z}$. $\square$

Let B be a subset of $\mathbb{R}^n$ and I be a set of indices. The collection of sets $\mathcal{M} := \{M_i \mid i \in I\}$, where $M_i \subseteq B$, is called a *partition* of B if $B = \cup_{i \in I} M_i$ and $M_i \cap M_j = \partial(M_i) \cap \partial(M_j)$ for all $i, j \in I, i \neq j$.

Theorem 4.3. *Let $\mathbf{k} = (k_1^1, \ldots, k_j^s, \ldots, k_{m_2}^S)^T \in \mathbb{Z}^{Sm_2}$ be a vector of integers. For a given $\mathbf{k}$, let*

$$\mathcal{C}(\mathbf{k}) := \{\chi \in \mathbb{R}^{m_2} \mid \chi \in \cap_{s=1}^S \Pi_{j=1}^{m_2} (k_j^s - h_j^s - 1, k_j^s - h_j^s]\}.$$

The following assertions hold:

(i) *if $\mathcal{C}(\mathbf{k}) \neq \emptyset$, then $\mathrm{cl}(\mathcal{C}(\mathbf{k}))$ is a full-dimensional hyper-rectangle,* i.e., $\dim(\mathcal{C}(\mathbf{k})) = m_2$,

(ii) *the collection $\{\mathcal{C}(\mathbf{k}) \mid \mathbf{k} \in \mathbb{Z}^{Sm_2}\}$ forms a partition of $\mathbb{R}^{m_2}$,*

(iii) *if $\mathcal{C}(\mathbf{k}) \neq \emptyset$, then $\overline{\Psi}(\chi)$ is constant over $\mathcal{C}(\mathbf{k})$.*

Proof: Part (i): Note that $\Pi_{j=1}^{m_2}[k_j^s - h_j^s - 1, k_j^s - h_j^s]$ is the Cartesian product of intervals, and hence is a hyper-rectangle. The orthogonal intersection of all such hyper-rectangles is also a hyper-rectangle. The first part of the claim then follows from the well-known facts that for convex sets C_i with $i \in I$, $\mathrm{cl}(\Pi_{i \in I} C_i) = \Pi_{i \in I} \mathrm{cl}(C_i)$, and $\mathrm{cl}(\cap_{i \in I} C_i) = \cap_{i \in I} \mathrm{cl}(C_i)$ (cf. [16]). To see that such a hyper-rectangle is full-dimensional, the reader can verify that any $\mathcal{C}(\mathbf{k}) \neq \emptyset$ can be written as $\mathcal{C}(\mathbf{k}) = \Pi_{j=1}^{m_2} \cap_{s=1}^S (k_j^s - h_j^s - 1, k_j^s - h_j^s]$. For each j, $\cap_{s=1}^S (k_j^s - h_j^s - 1, k_j^s - h_j^s]$ is the finite intersection of unit-length intervals which are left-open and right-closed. Thus, this intersection is itself a positive length interval. $\mathcal{C}(\mathbf{k})$ is then the Cartesian product of such positive length intervals and is hence full-dimensional.

Part (ii): It can be easily verified that for any $\chi \in \mathbb{R}^{m_2}$, there exists $\mathbf{k} \in \mathbb{Z}^{Sm_2}$ such that $\chi \in \mathcal{C}(\mathbf{k})$. Furthermore, for $\mathbf{k} \neq \mathbf{k}'$, $\mathcal{C}(\mathbf{k})$ and $\mathcal{C}(\mathbf{k}')$ are disjoint. Thus, $\{\mathcal{C}(\mathbf{k}) \mid \mathbf{k} \in \mathbb{Z}^{Sm_2}\}$ forms a partition of $\mathbb{R}^{m_2}$.

Part (iii): For a given $\mathbf{k}$, it follows from Lemma 4.2, that for any s, $\Psi^s(\chi)$ is constant over the hyper-rectangle $\Pi_{j=1}^{m_2}(k_j^s - h_j^s - 1, k_j^s - h_j^s]$. Since $\mathcal{C}(\mathbf{k})$ is a non-empty subset of all such hyper-rectangles (for all s), each $\Psi^s(\chi)$ is constant over $\mathcal{C}(\mathbf{k})$, and so is $\overline{\Psi}(\chi)$. □

Theorem 4.3 establishes that the second-stage value function is piece-wise constant over (neither open nor closed) rectangular subsets in the space of the tender variables χ. Thus, the discontinuities can only lie at the boundaries of these subsets and, therefore, are all orthogonal to the variable axes. This is not the case, in general, for the value function in the space of the original first-stage variables. To illustrate this, we plot the second-stage value function for example problem (EX) of Section 1, in the space of the original first-stage variables (Figure 2) and in the space of the tender variables (Figure 3). The change in the orientation of the discontinuities is clear. Notice that the feasible region $\mathcal{X}$ is a linear transformation of X.

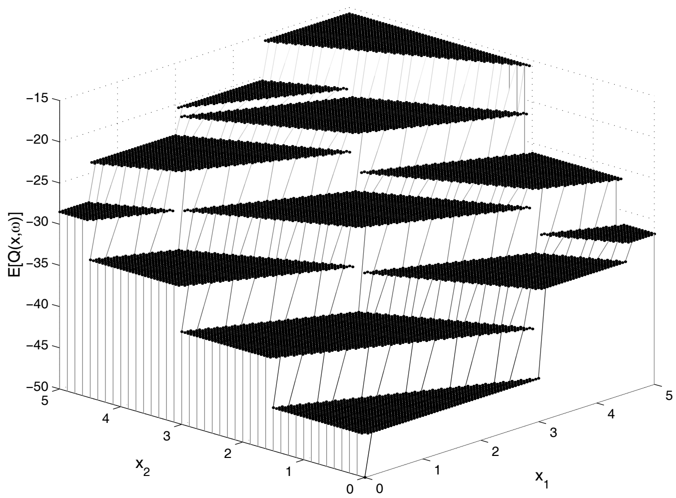

Figure 2: The second-stage Value Function of (EX) over X

Next, we establish the finiteness of the partitioning when the underlying set is compact.

Theorem 4.4. *Let $\mathcal{X} \in \mathbb{R}^{m_2}$ and $\mathcal{K} := \{\mathbf{k} \in \mathrm{Z}^{Sm_2} | \mathcal{C}(\mathbf{k}) \cap \mathcal{X} \neq \emptyset\}$. Then, if $\mathcal{X}$ is compact, $|\mathcal{K}| < \infty$.*

Proof: Since $\mathcal{X}$ is compact, we can obtain finite bounds l_j and u_j such that $l_j \leq \chi_j \leq u_j$ for all $\chi \in \mathcal{X}$. Now, suppose for some $\mathbf{k} = (k_1^1, \ldots, k_j^s, \ldots, k_{m_2}^S)^T$, there exists $\overline{\chi} \in \mathcal{C}(\mathbf{k}) \cap \mathcal{X}$. Then, from the definition of $\mathcal{C}(\mathbf{k})$ and the fact that $\mathcal{X}$ is compact, we must have for each j: $l_j \leq k_j^s - h_j^s$ for all s, which implies $k_j^s \geq \lceil l_j + h_j^s \rceil$. Similarly, we also have $k_j^s - h_j^s - 1 \leq u_j$, which then implies $k_j^s \leq \lfloor u_j + h_j^s + 1 \rfloor$. Thus

$$\lceil l_j + h_j^s \rceil \leq k_j^s \leq \lfloor u_j + h_j^s + 1 \rfloor.$$

We have bounded each component of the vector $\mathbf{k}$ for which $\mathcal{C}(\mathbf{k}) \cap \mathcal{X} \neq \emptyset$. Since $\mathbf{k}$ is an integer vector, there can only be a finite number of these that satisfy the above bounds. Thus, the claim follows. □

The above result along with Theorem 4.3 implies that the compact set $\mathcal{X}$ is completely covered by a finite number of rectangular cells, over each of which the second-stage value function is constant. In Section 5, we exploit this property to develop a finite branch-and-bound algorithm for (TP).

5 A Branch-and-Bound Algorithm

A major issue in applying branch-and-bound over continuous domains is that the resulting approach may not be finite but merely convergent, *i.e.*, infinitely many subdivisions may be required to make the lower bound exactly equal to the upper bound. In addition, for our problem (TP), we need to be able to deal with a discontinuous objective function. The challenge here is to identify

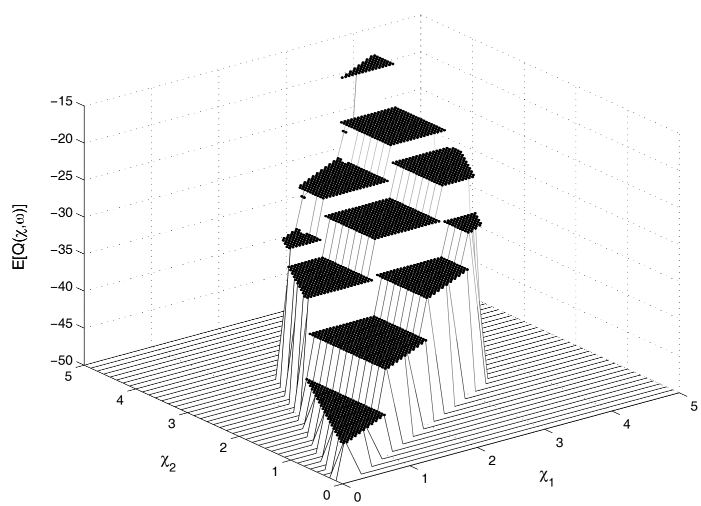

Figure 3: The second-stage Value Function of (EX) over $\mathcal{X}$

combinations of lower-bounding and branching techniques that can handle the discontinuous objective function and yield a finite algorithm. Towards this end, we exploit the structural results of Section 4 by partitioning the search space into subsets of the form $\Pi_{j=1}^{m_2}(l_j, u_j]$, where l_j is a point at which the second-stage value function $\overline{\Psi_j}(\chi_j)$ may be discontinuous. Lemma 4.2 implies that $\overline{\Psi_j}(\chi_j)$ can only be discontinuous at points χ_j where $(h_j^s + \chi_j)$ is integral for some s. Thus, we partition the search space along such values of χ. Branching in this manner, we can isolate subsets over which the second-stage value function is constant, and hence solve the problem exactly.

We shall now present a formal statement of a prototype branch-and-bound algorithm for problem (TP). The words in italic letters constitute the critical operations of the algorithm and will be discussed in subsequent subsections. The following notation is used in the description.

Notation:

$\mathcal{L}$	List of open subproblems
k	Iteration number; also used to indicate the subproblem selected
$\mathcal{P}^k$	Subset corresponding to iteration k
α^k	Upper bound obtained at iteration k
β^k	Lower bound on subproblem k
χ^k	A feasible solution to subproblem k
U	Upper bound on the global optimum
L	Lower bound on the global optimum
χ^*	Candidate globally optimal solution

The Algorithm

Initialization:

Preprocess the problem by constructing the hyper-rectangle $\mathcal{P}^0 := \Pi_{j=1}^{m_1}(l_j^0, u_j^0] \supseteq \mathcal{X}$. Add the problem $\min\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}^0\}$ to the list $\mathcal{L}$ of open subproblems.

Set $U \leftarrow +\infty$ and $k \leftarrow 0$.

Iteration k:

Step k.1: If $\mathcal{L} = \emptyset$, terminate with solution χ^*; otherwise, *select* a subproblem k, defined as $\inf\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}^k\}$, from the list $\mathcal{L}$ of currently open subproblems and set $\mathcal{L} \leftarrow \mathcal{L} \setminus \{k\}$. Note, that the min has been replaced by inf since the feasible region of the problem is not necessarily closed.

Step k.2: *Bound* the infimum of subproblem k from below, *i.e.*, find β^k satisfying $\beta^k \leq \inf\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}^k\}$. If $\mathcal{X} \cap \mathcal{P}^k = \emptyset$, $\beta^k = +\infty$ by convention. Determine a feasible solution $\chi^k \in \mathcal{X}$ and compute an upper bound $\alpha^k \geq \min\{f(\chi) \mid \chi \in \mathcal{X}\}$ by setting $\alpha^k = f(\chi^k)$.

Step k.2.a: Set $L \leftarrow \min_{i \in \mathcal{L} \cup \{k\}} \beta^i$.

Step k.2.b: If $\alpha^k < U$, then $\chi^* \leftarrow \chi^k$ and $U \leftarrow \alpha^k$.

Step k.2.c: Fathom the subproblem list, *i.e.*, $\mathcal{L} \leftarrow \mathcal{L} \setminus \{i \mid \beta^i \geq U\}$. If $\beta^k \geq U$, then goto Step k.1 and select another subproblem.

Step k.3: *Branch*, by partitioning $\mathcal{P}^k$ into $\mathcal{P}^{k_1}$ and $\mathcal{P}^{k_2}$. Set $\mathcal{L} \leftarrow \mathcal{L} \cup \{k_1, k_2\}$, *i.e.*, append the two subproblems $\inf\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}^{k_1}\}$ and $\inf\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}^{k_2}\}$ to the list of open subproblems. For selection purposes, set $\beta^{k_1}, \beta^{k_2} \leftarrow \beta^k$. Set $k \leftarrow k+1$ and goto Step k.1.

5.1 Preprocessing

As mentioned earlier, we shall only consider subsets of the form $\Pi_{j=1}^{m_2}(l_j, u_j]$, where l_j is such that $l_j + h_j^s$ is integral for some s. We can then construct a subset $\mathcal{P}^0 := \Pi_{j=1}^{m_1}(l_j^0, u_j^0] \supset \mathcal{X}$ in the following manner:

- Construct a closed subset $\Pi_{j=1}^{m_2}[l_j, u_j] \supseteq \mathcal{X}$ as follows: for each component j of χ, set $l_j = \min\{\chi_j \mid \chi \in \mathcal{X}\}$ and $u_j = \max\{\chi_j \mid \chi \in \mathcal{X}\}$. Typically, $\mathcal{X}$ is polyhedral and so the above problems are linear programs.

- For each s and j, find $k_j^s \in \mathbb{Z}$ such that $k_j^s - h_j^s - 1 < l_j \leq k_j^s - h_j^s$. If $l_j + h_j^s$ is integral, set $k_j^s = l_j + h_j^s$; otherwise, set $k_j^s = \lfloor l_j + h_j^s + 1 \rfloor$. Let $l_j^0 = \max_{s=1,\ldots,S}\{k_j^s - h_j^s - 1\}$.

- Set $u_j^0 = u_j$.

Above, we have relaxed l_j to l_j^0 such that l_j^0 is the closest point to l_j where $l_j^0 + h_j^s$ is integral for some s. From now on, whenever convenient, we shall denote subsets of the form $\Pi_{j=1}^{m_1}(l_j, u_j]$ by $(l, u]$ with $l = (l_1, \ldots, l_{m_2})^T$, and $u = (u_1, \ldots, u_{m_2})^T$.

5.2 Selection

In Step k.1, we need to select a subproblem, from the list of open subproblems $\mathcal{L}$, to be considered for bounding and further partitioning. A critical condition for convergence of a branch-and-bound algorithm is that this selection operation be *bound improving* [8]. This is accomplished by choosing the subproblem that attains the least lower bound, *i.e.*, by selecting $k \in \mathcal{L}$ such that $\beta^k = L$.

5.3 Lower Bounding

For a given subset $\mathcal{P}^k := \Pi_{j=1}^{m_2}(l_j^k, u_j^k]$ where each l_j^k is such that $(h_j^s + l_j^k)$ is integral for some s, we can obtain a lower bound on the corresponding subproblem by solving:

$$\text{(LB)}: \qquad f_L(\mathcal{P}^k) = \quad \min \quad cx + \theta \tag{5}$$

$$\begin{aligned} \text{s.t.} \quad & x \in X, Tx = \chi \\ & l^k \leq \chi \leq u^k \\ & \theta \geq \sum_{s=1}^{S} p^s \Psi^s(l^k + \epsilon), \end{aligned} \tag{6}$$

where

$$\begin{aligned} \Psi^s(\chi) = \quad \min \quad & f^s y \\ \text{s.t.} \quad & D^s y \geq h^s + \chi \\ & y \in Y \cap \mathbb{Z}^{n_2}. \end{aligned} \tag{7}$$

In problem (LB), ϵ is sufficiently small such that $\Psi^s(\cdot)$ is constant over $(l^k, l^k+\epsilon]$ for all s. Since we have exactly characterized the subsets over which the $\Psi^s(\cdot)$ is constant, we can calculate this ϵ *a priori* as follows.

Calculation of ϵ:

- Do for $j = 1, \ldots, m_2$:
  - Set $s = 1$, $\Xi = \emptyset$. Choose $k_j^1 \in \mathbb{Z}$.
  - Let $\chi_j^0 = k_j^1 - h_j^1 - 1$ and $\chi_j^1 = \chi_j^0 + 1$.
  - Set $\Xi \leftarrow \Xi \cup \{\chi_j^0, \chi_j^1\}$.
  - Do for $s = 2, \ldots, S$:
    - Choose $k_j^s \in \mathbb{Z}$ such that $\chi_j^0 < k_j^s - h_j^s \leq \chi_j^1$, *i.e.*, set $k_j^s = \lfloor \chi_j^1 + h_j^s \rfloor$.
    - Let $\chi_j^s = k_j^s - h_j^s$.
    - If $\Xi \cap \{\chi_j^s\} = \emptyset$, then set $\Xi \leftarrow \Xi \cup \{\chi_j^s\}$.

    End Do.
  - Order the elements of Ξ such that $\chi_j^0 = \xi_j^0 < \xi_j^1 < \ldots < \xi_j^n = \chi_j^1$, with $n \leq S$.
  - Let $\epsilon_j = \min_{i=1,\ldots,n}\{\xi_j^i - \xi_j^{i-1}\}$.

  End Do.
- Set $\epsilon = 0.5 \times \min_{j=1,\ldots,m_2}\{\epsilon_j\}$.

In the above procedure, we first determine an interval $(\chi_j^0, \chi_j^1]$ such that $\lceil h_j^1 + \chi_j \rceil$, and hence $\Psi_j^1(\chi_j)$, is constant for all $\chi_j \in (\chi_j^0, \chi_j^1]$. Then, for each $s = 2, \ldots, S$, we find χ_j^s such that $\lceil h_j^s + \chi_j \rceil$, and hence $\Psi_j^1(\chi_j)$, is constant for all $\chi_j \in (\chi_j^0, \chi_j^s]$. In this way, all candidate points of discontinuity in $(\chi_j^0, \chi_j^1]$ are identified and collected in set Ξ. The points of discontinuity that appear in $(\chi_j^0, \chi_j^1]$ also repeat to the right of χ_j^1 with a unit period. It then suffices to sort the potential points of discontinuity identified over $(\chi_j^0, \chi_j^1]$ to obtain the length

ϵ_j of the smallest interval along each axis j over which $\Psi_j^s(\chi_j)$ is guaranteed to be constant for all s. We finally choose ϵ to be strictly smaller than each ϵ_j.

We next show that (LB) is a valid lower bounding problem. Note that the feasible region of (LB) is closed, so that a minimizer exists.

Proposition 5.1. *For any subset $\mathcal{P}^k = (l^k, u^k]$,*

$$\beta^k := f_L(\mathcal{P}^k) \leq \inf\{f(\chi) \mid \chi \in \mathcal{P}^k \cap \mathcal{X}\}.$$

Proof: The claim obviously holds if $\mathcal{P}^k \cap \mathcal{X} = \emptyset$. Now consider some $\overline{\chi} \in \mathcal{P}^k \cap \mathcal{X}$. Let $\overline{x} \in \operatorname{argmin}\{cx \mid Tx = \overline{\chi}, x \in X\}$ and $\overline{\theta} = \sum_{s=1}^S p^s \Psi^s(\overline{\chi})$. Thus, $f(\overline{\chi}) = \Phi(\overline{\chi}) + \sum_{s=1}^S p^s \Psi^s(\overline{\chi}) = c\overline{x} + \overline{\theta}$. We shall now show that $(\overline{x}, \overline{\chi}, \overline{\theta})$ is feasible to (LB). $\overline{x}$ and $\overline{\chi}$ are obviously feasible. From the construction of ϵ and definition of l^k, we know that for each s, $\Psi^s(\chi)$ is constant over $(l^k, l^k + \epsilon]$. Then, owing to the monotonicity property of Ψ^s (Lemma 4.1), $\Psi^s(\overline{\chi}) \geq \Psi^s(l^k + \epsilon)$ since $\overline{\chi} > l^k$. Thus, $\overline{\theta} = \sum_{s=1}^S p^s \Psi^s(\overline{\chi}) \geq \sum_{s=1}^S p^s \Psi^s(l^k + \epsilon)$ and the constraint (6) in (LB) is satisfied. Since the solution is feasible, $f_L(\mathcal{P}^k) \leq c\overline{x} + \overline{\theta} = f(\overline{\chi})$. The claim follows from the fact that the above holds for any $\chi \in \mathcal{P}^k \cap \mathcal{X}$. $\square$

To solve (LB), we first need to solve S second-stage subproblems (7) to construct the cut (6). The master problem (5) can then be solved with respect to the variables (x, χ, θ). Note that X is typically polyhedral so that (5) is a linear program. If the first-stage variables have integrality requirements, then (5) is a mixed-integer linear program. Each scenario subproblem and the master problem can be solved completely independently, so complete stage and scenario decomposition is achieved. The variable θ approximates the second-stage value function in the first-stage variable space through constraint (6). In Section 7, we shall discuss how tighter approximations to the value function may be accommodated along with the "lower corner cut"(6).

Proposition 5.2. *Let $\mathcal{P}^k$ be a subset over which the second-stage value function $\overline{\Psi}(\cdot)$ is constant and there exists $\chi^* \in \operatorname{argmin}\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}^k\}$,* i.e., *the infimum is achieved. Let χ^k be an optimal solution to the lower bounding problem* (LB) *over this subset. Then,*

$$f(\chi^k) \leq f(\chi^*).$$

Proof: Let (x^k, χ^k, θ^k) be an optimal solution of the lower bounding problem (LB) for the subset $\mathcal{P}^k = (l^k, u^k]$. Note that $\chi^k \in \mathcal{X} \cap \operatorname{cl}(\mathcal{P}^k)$. Then, $f_L(\mathcal{P}^k) = cx^k + \overline{\Psi}(l^k + \epsilon)$ and $f(\chi^k) = cx^k + \overline{\Psi}(\chi^k)$. If $\chi^k > l^k$, then $\overline{\Psi}(\chi^k) = \overline{\Psi}(l^k + \epsilon)$ since $\overline{\Psi}(\cdot)$ is constant over $\mathcal{P}^k$. Thus, $f(\chi^k) = f_L(\mathcal{P}^k)$. On the other hand, if $\chi_j^k = l_j^k$ for some $j = 1, \ldots, m_2$, then $\overline{\Psi}(\chi^k) \leq \overline{\Psi}(l^k + \epsilon)$, owing to the monotonicity property. Thus $f(\chi^k) \leq f_L(\mathcal{P}^k)$. Since $f_L(\mathcal{P}^k) \leq f(\chi^*)$ by Proposition 5.1, the claim follows. $\square$

5.4 Upper Bounding

For a given subset $\mathcal{P}^k$ such that $\mathcal{P}^k \cap \mathcal{X} \neq \emptyset$, let χ^k be an optimal solution of problem (LB). Note that $\chi^k \in \mathcal{X}$ is a feasible solution. We can then compute an upper bound

$$\alpha^k := f(\chi^k) \geq \min\{f(\chi)|\chi \in \mathcal{X}\}.$$

Proposition 5.3. *If, for a subset $\mathcal{P}^k$, the second-stage value function $\overline{\Psi}(\cdot)$ is constant, then the subset $\mathcal{P}^k$ will be fathomed in the course of the algorithm.*

Proof: From the proof of Proposition 5.2, $\alpha^k = f(\chi^k) \leq f_L(\mathcal{P}^k) = \beta^k$. In the bounding Step k.2.a of the algorithm, we set $U = \min\{U, \alpha^k\}$. Thus, in Step k.2.c, the current subset $\mathcal{P}^k$ satisfies $\beta^k \geq U$ and will be fathomed. □

5.5 Branching

A typical scheme for partitioning $\mathcal{P}^k$ would consist of selecting and bisecting the variable j' corresponding to the longest edge of the hyper-rectangle $\mathcal{P}^k$. Although such a scheme is *exhaustive* [8], it might not be possible to isolate subsets without discontinuities, and take advantage of Proposition 5.3.

To isolate the discontinuous pieces of the second-stage value function, we are required to partition an axis j' at a point $\chi_{j'}$ such that $\Psi^s(\cdot)$ is possibly discontinuous at $\chi_{j'}$ for some s. While we can do this by selecting $\chi_{j'}$ such that $h^s_{j'} + \chi_{j'}$ is integral for some s, we can do better by determining the value of $\chi_{j'}$ where the current second-stage solution becomes infeasible. Such a point is more likely to be one at which $\Psi^s(\cdot)$ is discontinuous. This scheme is formally stated next. For each s, y^s is the solution of the second-stage IP subproblem in the solution of the lower bounding problem (LB).

The branching scheme

- For each $j = 1, \ldots, m_2$, compute $p_j := \min_{s=1,\ldots,S}\{(D^s y^s)_j - h^s_j\}$.
- Let $j' \in \text{argmax}_j\{\min\{p_j - l^k_j, u^k_j - p_j\}\}$.
- Split $\mathcal{P}^k = \Pi_{j=1}^{m_2}(l^k_j, u^k_j]$ into two subsets $\mathcal{P}^{k_1} = (l^k_{j'}, p_{j'}]\Pi_{j\neq j'}(l^k_j, u^k_j]$, and $\mathcal{P}^{k_2} = (p_{j'}, u^k_{j'}]\Pi_{j\neq j'}(l^k_j, u^k_j]$.

6 Proof of Finiteness

Consider a nested sequence of successively refined subsets $\{\mathcal{P}^{k_q}\}$ such that $\mathcal{P}^{k_{q+1}} \subset \mathcal{P}^{k_q}$.

Definition 6.1. [8] *A bounding operation is called* finitely consistent *if, at every step any unfathomed subset can be further refined, and if any nested sequence of $\{\mathcal{P}^{k_q}\}$ of successively refined subsets is finite.*

Lemma 6.2. *In a branch-and-bound procedure, suppose that the bounding operation is finitely consistent. Then, the procedure terminates after finitely many steps.*
Proof: See Theorem IV.1. in [8]. □

Lemma 6.3. *The bounding operation of the proposed branch-and-bound algorithm is finitely consistent.*

Proof: Consider a subset $\mathcal{P}^k$ that is unfathomed. By Proposition 5.3, the second-stage value function is discontinuous over this subset. Thus, the branching step can further refine it, thereby satisfying the first condition for finite consistency. Branching along the discontinuity on this subset will result in two strictly smaller subsets. By Theorem 4.4, the number of discontinuities in $\mathcal{P}^k$ is finite. Therefore, any nested sequence $\{\mathcal{P}^{k_q}\}$ generated by branching along the discontinuities of $\mathcal{P}^k$ will be finite. □

Theorem 6.4. *The proposed algorithm terminates with a global minimum after finitely many steps.*

Proof: As a consequence of Lemmas 6.2 and 6.3, it follows that the algorithm terminates after finitely many steps. The globality of the solution follows from the validity of the lower and upper bounding procedures used. In particular, let $\chi^* \in \mathcal{P}^0$ be a global minimizer. Then, there exists a finite nested sequence $\{\mathcal{P}^{k_q}\}_{q=1}^{Q}$ of length Q such that $\chi^* \in \mathcal{P}^{k_q}$ for all $q = 1, 2, \ldots, Q$. Clearly, $\mathcal{P}^{k_Q}$ does not contain a discontinuity, otherwise it would be further refined. Furthermore, $\chi^* \in \text{argmin}\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}^{k_Q}\}$. Let χ^k be the solution to the lower bounding problem over $\mathcal{P}^{k_Q}$. Then, by Proposition 5.2, $f(\chi^k) \leq f(\chi^*)$. Since $\chi^k \in \mathcal{X}$, χ^k must also be a global minimizer. □

7 Enhancements and Extensions

The proposed algorithm is valid for any lower bounding scheme that dominates the lower bound obtained by solving problem (LB). In this section, we suggest how such tighter bounds may be obtained. We also discuss the applicability of the proposed algorithm in case of mixed-integer second-stage variables.

Benders cuts

Consider the LP relaxation of the second-stage problem for a given scenario s and a value of the tender variable $\overline{\chi}$:

$$\begin{aligned} \Psi^s_{LP}(\overline{\chi}) = \quad & \min \quad f^s y \\ & \text{s.t.} \quad D^s y \geq h^s + \overline{\chi} \quad (\overline{u}^s) \end{aligned}$$

$$y \geq 0$$

where the $\overline{u}^s$ are the optimal dual solutions. We assume that $y \in Y$ is included in the constraint set $D^s y \geq h^s + \overline{\chi}$. The classical Benders or L-shaped [23] optimality cut for the lower bounding problem (LB) is then given by

$$\theta \geq \sum_{s=1}^{S} p^s[(h^s + \chi)\overline{u}^s].$$

These, along with the "lower corner cuts" (6), may provide a better approximation to the second-stage value function. To illustrate this, consider the example problem (EX) described in Section 1. For the initial subset $\mathcal{P}^0 = [(0,0),(5,5)]$, the lower bound given by problem (LB) is -72.25. If we add a single Benders cut to (LB), the lower bound improves to -60.9.

Bounds from the Lagrangian Dual

Carøe [5, 6] used a scenario decomposition approach to obtain lower bounds for (2SSIP). The idea here is introduce copies $x^1, \ldots, x^S$ and $y^1, \ldots, y^S$ of the first-stage and second-stage variables, corresponding to each scenario, and then rewrite (2SSIP) in the form

$$\begin{array}{lll} \min & \sum_{s=1}^{S} p^s c x^s + p^s f^s y^s & \\ \text{s.t.} & x^s \in X & s = 1, \ldots, S \\ & D^s y^s \geq h^s + T x^s & s = 1, \ldots, S \\ & y^s \in Y \cap \mathbb{Z}^{n_2} & s = 1, \ldots, S \\ & x^1 = \ldots = x^S & \end{array} \qquad (8)$$

Above, the *non-anticipativity* constraint (8) states that the first-stage decision should not depend on the scenario which will prevail in the second stage. This constraint can also be represented as $\sum_{s=1}^{S} H^s x^s = 0$, where H^s are matrices of conformable dimensions (see [5] for details). The Lagrangian relaxation of the above formulation with respect to the non-anticipativity constraints is completely decomposable by scenarios, and is given by

$$L(\lambda) = \sum_{s=1}^{S} \min\{(p^s c + \lambda H^s)x^s + p^s f^s y^s \mid x^s \in X,\ D^s y^s \geq h^s + T x^s,\ y^s \in Y \cap \mathbb{Z}^{n_2}\}.$$

It is well-known that the Lagrangian dual $z_{LD} = \max_\lambda L(\lambda)$ provides a lower bound to (2SSIP). Carøe used this lower bounding scheme within a branch-and-bound framework for solving (2SSIP).

Since we partition the space of tender variables, consider the Lagrangian relaxation of (2SSIP) when these variables are restricted to be $\chi \in \mathcal{P} := (l, u]$:

$$\begin{aligned} L(\lambda, \mathcal{P}) \quad = \quad & \sum_{s=1}^{S} \min \{(p^s c + \lambda H^s)x^s + p^s f^s y^s \mid x^s \in X,\ l \leq T x^s \leq u, \\ & \qquad D^s y^s \geq h^s + T x^s,\ D^s y^s \geq h^s + l + \epsilon,\ y^s \in Y \cap \mathbb{Z}^{n_2}\}, \end{aligned}$$

where ϵ is the same as that considered in problem (LB) in Section 5.3. Note that the above Lagrangian relaxation has additional constraints: $D^s y^s \geq h^s + l + \epsilon$, to deal with the neither open nor closed nature of $\mathcal{P}$. We shall denote the corresponding Lagrangian dual by

$$z_{LD}(\mathcal{P}) := \max_{\lambda} L(\lambda, \mathcal{P}).$$

Proposition 7.1. *Given a subset* $\mathcal{P} := (l, u]$,

$$f_L(\mathcal{P}) \leq z_{LD}(\mathcal{P}) \leq \inf\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}\}.$$

Proof: Let $(\overline{x}^1, \ldots, \overline{x}^S, \overline{y}^1, \ldots, \overline{y}^S)$ be the solutions obtained while computing $L(0, \mathcal{P})$. Thus

$$L(0, \mathcal{P}) = \sum_{s=1}^{S} p^s c \overline{x}^s + p^s f^s \overline{y}^s.$$

Let $\tilde{x}$ be the solution of the master problem (5), and $\tilde{y}^s$ be the solution of the scenario s subproblem (7), while computing $f_L(\mathcal{P})$. Since each $\overline{x}^s$ is feasible to the master problem (5), we have $c\tilde{x} \leq c\overline{x}^s$ for all s. Thus, $c\tilde{x} \leq \sum_{s=1}^{S} p^s c \overline{x}^s$. Recall that the subproblems (7) are solved with $\chi = l + \epsilon$, *i.e.*, with the constraint $D^s y \geq h^s + l + \epsilon$. Since $\overline{y}^s$ also satisfies $D^s \overline{y}^s \geq h^s + l + \epsilon$, $\overline{y}^s$ is feasible to the scenario s subproblems (7), and so $f^s \tilde{y}^s \leq f^s \overline{y}^s$ for each s. Clearly,

$$\begin{aligned} f_L(\mathcal{P}) &= \sum_{s=1}^{S} p^s c \tilde{x}^s + p^s f^s \tilde{y}^s \\ &\leq \sum_{s=1}^{S} p^s c \overline{x}^s + p^s f^s \overline{y}^s \\ &= L(0, \mathcal{P}) \\ &\leq z_{LD}(\mathcal{P}). \end{aligned}$$

To see that $z_{LD}(\mathcal{P}) \leq \inf\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}\}$, consider a feasible solution $(\chi, x, y^1, \ldots, y^S)$ such that $\chi = Tx$, $x \in X$, $\chi \in \mathcal{X} \cap \mathcal{P}$, and $D^s y^s \geq h^s + \chi$ for all s. Construct a solution $(\overline{x}^1, \ldots, \overline{x}^S, \overline{y}^1, \ldots, \overline{y}^S)$ to the Lagrangian relaxation $L(\lambda, \mathcal{P})$, by setting $\overline{x}^s = x$, and $\overline{y}^s = y^s$ for all s. To see that such a solution is feasible to $L(\lambda, \mathcal{P})$, we only need to verify that $D^s \overline{y}^s \geq h^s + l + \epsilon$. Since $T\overline{x}^s = \chi > l$, and from the definition of ϵ, $\lceil h^s + T\overline{x}^s \rceil$ is constant whenever $T\overline{x}^s \in (l, l+\epsilon]$, we have that $D^s \overline{y}^s \geq h^s + T\overline{x}^s$ implies $D^s \overline{y}^s \geq h^s + l + \epsilon$. Thus the solution $(\overline{x}^1, \ldots, \overline{x}^S, \overline{y}^1, \ldots, \overline{y}^S)$ is feasible to $L(\lambda, \mathcal{P})$. Since $\overline{x}^1 = \ldots = \overline{x}^S$, we have $\sum_{s=1}^{S} H^s \overline{x}^s = 0$, and $\sum_{s=1}^{S} \{(p^s c + \lambda H^s)\overline{x}^s + p^s f^s \overline{y}^s\} = c(\sum_{s=1}^{S} p^s \overline{x}^s) + \sum_{s=1}^{S} p^s f^s \overline{y}^s = cx + \sum_{s=1}^{S} p^s f^s y^s$. Thus, $L(\lambda, \mathcal{P}) \leq \inf\{f(\chi) \mid \chi \in \mathcal{X} \cap \mathcal{P}\}$. Since the λ was arbitrary, the inequality is true for $z_{LD}(\mathcal{P})$. □

Thus, we can use the Lagrangian dual to obtain tighter bounds than those obtained by solving (LB).

Mixed-integer Second Stage

In the presence of continuous variables in the second stage, the orthogonality of the discontinuities in the space of the tender variables may be lost. Consider, for example, a variant of (EX) where the second-stage problem (in the space of the tender variables) is given by:

$$\begin{aligned} Q(\chi_1, \chi_2, \omega_1, \omega_2) = \quad \min \quad & -16y_1 - 19y_2 - 23y_3 - 28y_4 \\ \text{s.t.} \quad & 2y_1 + 3y_2 + 4y_3 + 5y_4 \le \omega_1 - y_5 \\ & 6y_1 + y_2 + 3y_3 + 2y_4 \le \omega_2 - y_6 \\ & -1.9706y_5 + 0.9706y_6 \le -\chi_1 \\ & 0.9706y_5 - 1.9706y_6 \le -\chi_2 \\ & y_1, y_2, y_3, y_4 \in \{0, 1\} \\ & y_5, y_6 \in [0, 5]. \end{aligned}$$

As detailed in [1], the discontinuities of $Q(\chi_1, \chi_2, \omega_1, \omega_2)$ are not orthogonal to the χ_1 and χ_2 axes. As a result, in case of mixed-integer second stage, finiteness of the proposed algorithm is not guaranteed. One can reformulate the problem by including all the continuous variables from the second-stage subproblems in the first stage, thereby generating a pure-integer second stage. The proposed algorithm is finite when applied to this reformulation. Since such a scheme involves branching on continuous variables from the first-stage problem as well as those from the scenario subproblems, the method is expected to be viable only when there are a few continuous variables per second-stage subproblem. Alternatively, in addition to branching on the tender variables, we can allow branching on the second-stage variables similar to what one would do while solving the large-scale deterministic equivalent integer program. Even though explicit second-stage branching will be computationally intensive, it will guarantee finite termination. As the monotonicity property of the second-stage value function with respect to the tender variables still hold, a valid lower bound over a subset in the tender variable space is obtained by evaluating the second-stage value function at the lower-corner.

Random Technology Matrix

The proposed algorithm can be extended to problems with a scenario-dependent technology matrix T by introducing tender variables corresponding to each scenario, *i.e.*, $\chi^s = T^s x$. However, in this case, the algorithm would require branching on $S \times m_2$ variables as opposed to m_2 variables when T is deterministic.

8 Computational Results

In this section, we report our computational experience with the proposed algorithm on instances of two-stage stochastic integer programs from the literature.

Scenarios	Integer Variables	Binary Variables	Constraints
4	2	16	8
9	2	36	18
36	2	144	72
121	2	484	242
441	2	1764	882

Table 1: Sizes of instances in Test Set 1

Test Set 1

The first set of test problems involves pure-integer first-stage variables and is taken from Carøe [5]. Since the first-stage variables are pure-integer, Carøe's method terminates finitely. However, our computational results indicate that the proposed method is much faster than Carøe's algorithm even for this problem class. The test problems are generated from the following basic model:

$$\begin{array}{rll}(\text{EX1}): & \min & -1.5x_1 - 4x_2 + E[Q(x_1, x_2, \omega_1, \omega_2)] \\ & \text{s.t.} & x_1, x_2 \in [0, 5] \cap \mathbb{Z},\end{array}$$

where

$$\begin{array}{rll}Q(x_1, x_2, \omega_1, \omega_2) = & \min & -16y_1 - 19y_2 - 23y_3 - 28y_4 \\ & \text{s.t.} & 2y_1 + 3y_2 + 4y_3 + 5y_4 \leq \omega_1 - x_1 \\ & & 6y_1 + y_2 + 3y_3 + 2y_4 \leq \omega_2 - x_2 \\ & & y_1, y_2, y_3, y_4 \in \{0, 1\},\end{array}$$

where (ω_1, ω_2) is uniformly distributed on $\Omega \subseteq [5, 15] \times [5, 15]$. Five test problems are generated from (EX1) by varying the number of scenarios by taking Ω as equidistant lattice points in $[5, 15] \times [5, 15]$ with equal probability assigned to each point. The resulting instances have 4, 9, 36, 121, and 441 scenarios. The sizes of the corresponding deterministic equivalents are shown in Table 1.

Carøe [5] reports attempts to directly solve the deterministic equivalent of the above instances using the MIP solver of CPLEX 5.0. With 36 or more scenarios, CPLEX 5.0 MIP could not solve the problem instances within resource usage limits. For example, the instance with 121 scenarios could not be solved within 300,000 nodes, yielding an optimality gap of more than 10%. These results clearly motivate the need for decomposition-based approaches.

The proposed algorithm was applied to solve this small example. The computations were carried out on a 332 MHz IBM RS/6000 PowerPC. Table 2 compares the performance of the proposed algorithm to that of the Lagrangian decomposition approach of Carøe [5]. A major part of the computational effort is spent on solving the second-stage IP subproblems. From Table 2, it is clear that the number of IPs solved is smaller for the proposed method. Note that, for the proposed method, the IP subproblems correspond to the second-stage problem involving four binary variables and two constraints, whereas, for

Scenarios	Carøe [5]			Proposed		
	CPU* s.	IPs solved	Obj.	CPU† s.	IPs solved	Obj.
4	0.2	52	57.00	0.01	48	57.00
9	0.4	189	59.33	0.01	135	59.33
36	1.4	720	61.22	0.01	540	61.22
121	4.8	2783	62.29	0.02	1815	62.29
441	25.1	9702	61.32	0.06	6615	61.32

∗ Digital Alpha 500 MHz. (LINPACK DP ($n = 100$): 235.3 Mflop/s)
† IBM RS/6000 43P 332 MHZ. (LINPACK DP ($n = 100$): 59.9 Mflop/s)

Table 2: Computational results for Test Set 1

	CPLEX 5.0 [21]		Schultz *et al.* [21]		Proposed	
	Nodes	Gap	Evaluations	Obj.	Evaluations	Obj.
Problem 1 ($T = I$)	50000	24%	121	61.32	28	61.32
Problem 2 ($T \neq I$)	50000	27%	19	61.44	10	61.44

Table 3: Comparative performance for Test Set 2

Carøe's method, these include the first-stage variables as well and therefore involve two general integer variables, four binary variables, and two constraints. Note also that Carøe branches to enforce integrality of the first-stage solutions, while our branching scheme is geared towards isolating the discontinuities of the second-stage value function. Table 2 indicates that the CPU requirement of the proposed method is significantly smaller than that of Carøe's.

Test Set 2

The second test set is taken from Schultz *et al.* [21]. It consists of two variants of (EX1) with continuous first-stage variables. The first of these problems (Example 7.1 in [21]) is the 441 scenario version of (EX1) with the integrality restrictions removed from the first-stage variables. Schultz *et al.* [21] identified the set of candidate solutions to be the finite set $\{(k_1/2, k_2/2) \mid k_1, k_2 \in \mathbb{Z}\} \cap [0, 5] \times [0, 5]$. This set has a cardinality of 121, and the authors evaluated the second-stage value function $E[Q(x, \omega)]$ corresponding to each of these points to determine the optimal solution $x_1 = 0, x_2 = 4$ with value 61.32. Note that, a single evaluation of $E[Q(x, \omega)]$ amounts to solving 441 second-stage integer programs involving 4 binary variables and 2 constraints. This represents a key bottleneck in solving this class of problems. In [21], the authors used an efficient Gröbner basis-based solution strategy to exploit the common structure in the second-stage problems. In [21], the authors also report their attempt to solve this problem by CPLEX 5.0 with a node limit of $50,000$. After exploring all $50,000$ nodes, CPLEX ended up with an optimality gap of 24%. No CPU times were reported in Schultz *et al.* [21].

We solved the problem to global optimality using the proposed branch-and-bound algorithm. The algorithm required only 28 evaluations of $E[Q(x, \omega)]$ in contrast to the 121 required by the method of [21]—a reduction of 76%. As

	Problem 1 ($T = I$)			Problem 2 ($T \neq I$)		
Scenarios	CPU† s.	Evaluations	Obj.	CPU† s.	Evaluations	Obj.
4	0.01	15	57.00	0.00	6	57.75
9	0.01	19	59.33	0.00	7	59.56
36	0.02	19	61.22	0.01	11	60.28
121	0.03	19	62.29	0.02	11	61.01
441	0.15	28	61.32	0.05	10	61.44

† IBM RS/6000 43P 332 MHZ

Table 4: Computational results for Test Set 2

in [21], each function evaluation requires solving the 441 second-stage integer programs, and the Gröbner basis-based solution strategy suggested in [21] could be used here.

Note that, in (EX1), the technology matrix T is the identity. To illustrate the effect of the variable transformation, we next solve another variant of (EX1) with a more interesting T matrix, namely problem (EX) (described in Section 1) with 441 scenarios. Schultz *et al.* [21] solved this problem (Example 7.3 in [21]) by characterizing the solution set and identifying 53 candidate points. Using some problem-specific results, they were able to reduce the number of candidate points to 19. The second-stage value function was evaluated at each of these points, and the optimal solution of $x_1 = 0, x_2 = 4.5$ with objective value 61.44 was identified. In contrast, the proposed branch-and-bound algorithm required the evaluation of $E[Q(x, \omega)]$ at only 10 points—a 47% reduction. For this problem, Schultz *et al.* [21] reports that CPLEX with a node limit of $50,000$ ended up with an optimality gap of 27%.

The comparative performances discussed above are summarized in Table 3. Table 4 presents the CPU times and the number of evaluations of $E[Q(x, \omega)]$ required by the proposed algorithm for solving various scenario instances of the two problems in Test set 2.

Test Set 3

The final test set is a collection of two-stage stochastic product substitution problems described in Jorjani *et al.* [9]. The problem involves mixed-integer variables in the first and second stage. The set includes three problems, SIZES3, SIZES5, and SIZES10, having 3, 5, and 10 scenarios, respectively. The size of the deterministic equivalent integer program for each of these test problems is presented in Table 5.

A direct attempt to solve the deterministic mixed-integer program using the CPLEX 5.0 MIP solver was reported in Jorjani *et al.* [9]. These results are summarized in Table 6. The authors put a node limit of 20,000 for the two smaller problems, and 250,000 for the larger problem. Even after exploring such large number of nodes, CPLEX could not solve these problems and yielded optimality gaps in the range of 2-4%. From this table, it is clear that, although the problems are of modest size (no more than 110 binary variables), they are

Problem	Binary Variables	Continuous Variables	Constraints
SIZES3	40	260	142
SIZES5	60	390	186
SIZES10	110	715	341

Table 5: Sizes of Test Set 3

not amenable to state-of-the-art integer programming techniques, and one must rely on decomposition methods.

Carøe [5] attempted to solve these problems using Lagrangian decomposition based branch-and-bound algorithm. A CPU limit of 1000 seconds was imposed. Until now, Carøe's results were the best available for these problems.

Having gained some insight regarding the applicability of the proposed method on problems with mixed-integer second stage (Section 7), we attempted to solve these problems by explicitly branching on the second-stage integer variables. The implementation was carried out using BARON [19, 22] to maintain the branch-and-bound tree. CPLEX 6.0 was used as the LP solver. The computations were carried out on a 332MHz IBM RS/6000 PowerPC. A CPU limit of 100,000 seconds was imposed.

Problem	LB	UB	Nodes	CPU‡
SIZES3	218.2	224.7	20000	1859.8
SIZES5	220.1	225.6	20000	4195.2
SIZES10	218.2	226.9	250000	7715.5

‡ DEC alpha 3000/700

Table 6: Performance of CPLEX 5.0 on Test Set 3 as reported in [9]

	Carøe [5]			Proposed		
Problem	*LB*	*UB*	CPU* s	*LB*	*UB*	CPU† s
SIZES3	224.384	224.544	1,000	224.433		70.7
SIZES5	224.354	224.567	1,000	224.486		7,829.1
SIZES10	224.336	224.735	1,000	224.236	224.717	10,000.0

∗ Digital Alpha 500 MHz
† IBM RS/6000 133 MHZ

Table 7: Computational results for Test Set 3

Problem	Total Nodes	Max. Nodes in memory	Nodes until best UB
SIZES3	1885	260	906
SIZES5	108,782	13,562	41,642
SIZES10	36,700	23,750	20,458

Table 8: Nodes in branch-and-bound tree

Table 7 compares the performance of the proposed method with that of [5]. As a reference, the node information required by our branch-and-bound algorithm is presented in Table 8. From Table 7, we observe that Carøe's method was not able to close the gap for these problems on a computer much faster than ours. The proposed approach successfully closed the gap for two of these three very difficult problems. For all three test problems, we were able to identify better upper bounds (feasible solutions) than those known earlier.

References

[1] S. Ahmed. *Strategic Planning under Uncertainty: Stochastic Integer Programming Approaches.* Ph.D. Thesis, University of Illinois, Urbana, IL, 2000.

[2] J. R. Birge and F. V. Louveaux. *Introduction to Stochastic Programming.* Springer, New York, NY, 1997.

[3] C. E. Blair and R. G. Jeroslow. The value function of an integer program. *Mathematical Programming*, 23:237–273, 1982.

[4] C. E. Blair and R. G. Jeroslow. Constructive characterizations of the value-function of a mixed-integer program I. *Discrete Applied Mathematics*, 9:217–233, 1984.

[5] C. C. Carøe. *Decomposition in stochastic integer programming.* PhD thesis, University of Copenhagen, 1998.

[6] C. C. Carøe and R. Schultz. Dual decomposition in stochastic integer programming. *Operations Research Letters*, 24:37–45, 1999.

[7] C. C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming*, 83:451–464, 1998.

[8] R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches.* Springer-Verlag, Berlin, 3rd edition, 1996.

[9] S. Jorjani, C. H. Scott, and D. L. Woodruff. Selection of an optimal subset of sizes. Technical report, University of California, Davis, CA, 1995.

[10] P. Kall and S. W. Wallace. *Stochastic Programming.* John Wiley and Sons, Chichester, England, 1994.

[11] W. K. Klein Haneveld, L. Stougie, and M. H. van der Vlerk. On the convex hull of the simple integer recourse objective function. *Annals of Operational Research*, 56:209–224, 1995.

[12] W. K. Klein Haneveld, L. Stougie, and M. H. van der Vlerk. An algorithm for the construction of convex hulls in simple integer recourse programming. *Annals of Operational Research*, 64:67–81, 1996.

[13] W. K. Klein Haneveld and M. H. van der Vlerk. Stochastic integer programming: General models and algorithms. *Annals of Operational Research*, 85:39–57, 1999.

[14] G. Laporte and F. V. Louveaux. The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13:133–142, 1993.

[15] V. I. Norkin, Y. M. Ermoliev, and A. Ruszczyński. On optimal allocation of indivisibles under uncertainty. *Operations Research*, 46(3):381–395, 1998.

[16] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, New Jersey, 1970.

[17] R. T. Rockafellar and R. J.-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147, 1991.

[18] H. L. Royden. *Real Analysis*. Macmillan Publishing Co., London, 3rd edition, 1988.

[19] N. V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8:201–205, 1996.

[20] R. Schultz. On structure and stability in stochastic programs with random technology matrix and complete integer recourse. *Mathematical Programming*, 70(1):73–89, 1995.

[21] R. Schultz, L. Stougie, and M. H. van der Vlerk. Solving stochastic programs with integer recourse by enumeration: A framework using Gröbner basis reductions. *Mathematical Programming*, 83:229–252, 1998.

[22] M. Tawarmalani and N. V. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Kluwer Academic Publishers, Nonconvex Optimization and Its Applications Series, Vol 65, Dordrecht, 2002.

[23] R. Van Slyke and R. J.-B. Wets. L-Shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17:638–663, 1969.

[24] R. J-B. Wets. Programming under uncertainty: The solution set. *SIAM Journal on Applied Mathematics*, 14:1143 − 1151, 1966.