Math. Program., Ser. B 108, 597–616 (2006)

Digital Object Identifier (DOI) 10.1007/s10107-006-0724-6

Hanif D. Sherali · Xiaomei Zhu

On solving discrete two-stage stochastic programs having mixed-integer first- and second-stage variables

Received: November 9, 2004 / Accepted: August 25, 2005
Published online: June 2, 2006 –

Abstract. In this paper, we propose a decomposition-based branch-and-bound (**DBAB**) algorithm for solving two-stage stochastic programs having mixed-integer first- and second-stage variables. A modified Benders' decomposition method is developed, where the Benders' subproblems define lower bounding second-stage value functions of the first-stage variables that are derived by constructing a certain partial convex hull representation of the two-stage solution space. This partial convex hull is sequentially generated using a convexification scheme such as the Reformulation-Linearization Technique (RLT) or lift-and-project process, which yields valid inequalities that are reusable in the subsequent subproblems by updating the values of the first-stage variables. A branch-and-bound algorithm is designed based on a hyperrectangular partitioning process, using the established property that any resulting lower bounding Benders' master problem defined over a hyperrectangle yields the same objective value as the original stochastic program over that region if the first-stage variable solution is an extreme point of the defining hyperrectangle or the second-stage solution satisfies the binary restrictions. We prove that this algorithm converges to a global optimal solution. Some numerical examples and computational results are presented to demonstrate the efficacy of this approach.

Key words. Two-stage stochastic mixed-integer programs – Benders' decomposition – Convexification – Reformulation-Linearization Technique (RLT)

1. Introduction

Stochastic programs with recourse are optimization problems in which a set of decisions have to be made *a priori* in a context when the related environmental information is not completely available (*nonanticipative*). Given these decisions, the values of subsequent *recourse variables* need to be prescribed based on the realization of certain random events. A *two-stage stochastic program* can be formulated as follows:

$$\textbf{SP: Minimize} \quad c^T x + E[f(x, \tilde{\omega})] \tag{1a}$$

$$\text{subject to} \quad x \in X \subseteq R^n \tag{1b}$$

where $\tilde{\omega}$ is a random variable defined on a probability space $(\tilde{\Omega}, \tilde{\mathcal{A}}, \tilde{\mathcal{P}})$ (with $\tilde{\Omega}$, $\tilde{\mathcal{A}}$, and $\tilde{\mathcal{P}}$ respectively denoting the set of all outcomes, a collection of random variables, and the assigned probabilities), and where for any given realization ω of $\tilde{\omega}$, we have

H. D. Sherali, X. Zhu: Grado Department of Industrial and Systems Engineering (0118),
Virginia Polytechnic Institute and State University, Blacksburg, 24061, VA, USA.
e-mail: {hanifs,xzhu}@vt.edu

Mathematics Subject Classification (1991): 20E28, 20G40, 20C20

$$f(x,\omega) = \text{minimum } g(\omega)^T y$$
$$\text{subject to } W(\omega)y \geq r(\omega) - T(\omega)x \tag{1c}$$
$$y \in Y \subseteq R^m. \tag{1d}$$

Here, x and y respectively denote the nonanticipative first-stage variables and the second-stage recourse variables, and the associated sets X and Y are assumed to be described by linear constraints along with some possible integrality restrictions. In this paper, we permit the model to generally include 0-1 mixed-integer first-stage and second-stage variables. (We assume that any general bounded integer variables are transformed using a binary representation.) Such a problem is accordingly referred to as a *two-stage stochastic mixed-integer program* (**SMIP**). The matrix W is called the *recourse matrix* and is often assumed to be scenario-independent (i.e., $W(\omega) \equiv W$, $\forall\omega$), in which case the model is referred to as having *fixed recourse*. The matrix T is the so-called *technology matrix* and is typically considered to be scenario-dependent, although some analyses do assume a fixed technology matrix, as, for example, in Ahmed et al. [2]. For computational viability, a finite number of scenarios $s = 1, \ldots, S$ is often considered based on some discretization of the realization of $\tilde{\omega}$, each with an associated probability of occurrence p_s, $s = 1, \ldots, S$ (see Schultz [14] for a justification on approximating continuously distributed scenario parameters by a discrete distribution having a finite support). Accordingly, the realizations of $W(\omega)$, $T(\omega)$, and $r(\omega)$ are correspondingly denoted as W_s, T_s, and r_s, respectively, for $s = 1, \ldots, S$. Note that we have assumed a linear objective function as in most studies of stochastic models. However, models that consider variability measures, as for example in the case of robust optimization (Mulvey et al. [12], Takriti and Ahmed [23], and Ahmed [1]), do accommodate nonlinear terms in the objective function. Naturally, the computational difficulty of solving such models increases drastically.

Arguably, SMIPs are among the most challenging of optimization problems because they combine two generally difficult classes of problems: stochastic programs and discrete optimization problems. Researchers have actively studied the properties and solution approaches for such problems for the past decade (see [15, 10, 22] for surveys on some major results in this area and an annotated bibliography). The simplest form of stochastic integer programs contain pure binary first-stage variables and continuous second-stage variables. Laporte and Louveaux [11] provide a branch-and-cut (B&C) procedure for such problems in which feasibility and optimality cuts are applied, similar to those for Benders' partitioning method [4].

When the second stage contains only continuous variables, its objective function is a well-behaved piecewise linear and convex function of the first-stage variables. However, if the second stage contains discrete decisions, not only does the number of integer variables increase as the number of scenarios increases, but also, the second-stage value functions are now only lower semicontinuous with respect to the first-stage variables [5], which makes them generally non-convex [15]. Schultz et al. [15] present a finite-set enumeration framework for solving stochastic models having complete and pure integer recourse and continuous first-stage variables. van der Vlerk [24] provides a mechanism to obtain a (continuous) convex approximation for the expected second-stage value function through a modification of the random right-hand-side vector. Ahmed et al. [2] develop a finite branch-and-bound (B&B) solution approach for stochastic programs

having a fixed technology matrix and general first-stage and pure integer recourse variables by adopting an appropriate partitioning process in the transformed space of the *tender variables* $\chi = Tx$. Carøe and Tind [8] generalize the L-shaped method to deal with two-stage stochastic programs having fixed integer, though not complete, recourse. Carøe and Tind [7] have also investigated applying cutting planes to solve stochastic programs having 0-1 mixed-integer recourse variables and either pure continuous or pure binary first-stage variables.

If the first stage contains pure binary variables, finite termination is readily justified when adopting search procedures that branch over the first-stage variables. Sherali and Fraticelli [21], Sen and Higle [17], and Ntaimo and Sen [13] investigate two-stage stochastic programs having pure binary first-stage variables and 0-1 mixed-integer recourse variables, and propose decomposition algorithms that rely on a sequential partial convexification process that generates cuts as functions of the first-stage variables, where these cuts are shown to be reusable for subsequent subproblems by updating the values of the first-stage variables. Sen and Sherali [18] combine a similar decomposition-based cutting plane approach along with the disjunctive decomposition cuts of Sen and Higle [17] in a branch-and-cut (B&C) framework to derive a class of finite disjunctive decomposition-based B&C (D^2-BAC) approaches.

Carøe and Schultz [6] design a B&B algorithm for problems having mixed-integer variables in both stages. In their algorithm, the bounding process resorts to solving a Lagrangian dual problem predicated on the nonanticipativity condition, and evaluated via mixed-integer separable subproblems. The Lagrangian dual provides tighter bounds than the LP relaxation at each B&B node, yet the price paid is having to solve the nonsmooth Lagrangian dual problem along with the mixed-integer subproblems. Schultz and Tiedemann [16] extend this approach to solve stochastic programs that include an additional objective term based on the probability of a risk function exceeding a prespecified threshold value.

In this paper, we study a wide class of mixed-integer two-stage stochastic programs in the following form:

$$\textbf{SMIP:}\quad \text{Minimize} \quad c^T x + \sum_{s=1}^{S} p_s f_s(x) \tag{2a}$$

$$\text{subject to} \quad x \in X \cap \Omega \tag{2b}$$

where

$$X \equiv \{x \in R^n :\ Ax \geq b,\ x_i \geq 0,\ \forall i \in I_1 \subseteq \{1, \ldots, n\},\ x_i \text{ binary},\ \forall i \in I_2 \equiv \{1, ..., n\} \setminus I_1\} \tag{2c}$$

$$\Omega \equiv \{x \in R^n :\ l \leq x \leq u\} \text{ (with } l_i \equiv 0 \text{ and } u_i \equiv 1, \forall i \in I_2) \tag{2d}$$

and for any scenario $s = 1, \ldots, S$, we have,

$$f_s(x) = \text{minimum} \quad g^T y \tag{2e}$$

$$\text{subject to} \quad W_s y \geq r_s - T_s x \tag{2f}$$

$$y \in Y \equiv \{y \in R^m : y_j \geq 0,\ \forall j \in J_1 \subseteq \{1, \ldots, m\},\ y_j \text{ binary},\ \forall j \in J_2 \equiv \{1, ..., m\} \setminus J_1\}. \tag{2g}$$

We make the following assumptions:

A1. The continuous variables in both stages are bounded. Moreover, the continuous variables in the second stage are scaled onto [0,1], with the corresponding bounding restrictions $y_j \leq 1, \forall j \in J_1$, being absorbed within (or implied by) (2f).

A2. The inherent stochasticity in the problem is discretized into so-called scenarios, $s = 1, \ldots, S$, each with an associated probability of occurrence $p_s, \forall s = 1, \ldots, S$.

A3. For any $x \in \Omega$, the region defined by (2f, 2g) is feasible (relative complete recourse with respect to Ω).

The remainder of this paper is organized as follows. Section 2 introduces several important concepts that lay the foundation for designing a decomposition-based branch-and-bound approach for solving two-stage stochastic programs of the form given in (2). The main algorithmic procedure is described in this section and its convergence to a global optimum is established. Section 3 then presents the supporting algorithmic routines for solving the related subproblems and the lower-bounding master programs. An illustrative example is provided in Section 4, and Section 5 reports some encouraging preliminary computational experience. Finally, Section 6 concludes this paper.

2. Decomposition-based branch-and-bound algorithm (DBAB)

In this section, we begin by presenting some fundamental concepts and results that lead us to design our proposed overall algorithm for solving Problem SMIP. Observe that when the first-stage variables x are purely binary, then for each fixed binary vector $\bar{x}$, the extreme points of $conv\{(x, y) : T_s x + W_s y \geq r_s, x \in \Omega, y \in Y\} \cap \{(x, y) : x = \bar{x}\}$ have binary values for $y_j, \forall j \in J_2$, for each scenario $s = 1, \ldots, S$. This follows because the restriction $x = \bar{x}$ is then facial with respect to Ω. However, this statement is no longer true when the first stage contains continuous variables, whereby, x might be fixed at some nonextremal point within Ω. As a result, the algorithms developed by Sen and Higle [17], Sherali and Fraticelli [21], and Sen and Sherali [18], in particular, all of which rely on this basic construct, are no longer applicable in this case. This necessitates the development of an alternative solution approach.

Toward this end, for any fixed $x \in \Omega$, consider the scenario-based value functions

$$f_s(x) \equiv \min\{g^T y : W_s y \geq r_s - T_s x, \ y \in Y\}, \ \forall s. \tag{3}$$

Now, given any Ω (which will be partitioned in a B&B context in the sequel), define

$$Z_s(\Omega) \equiv conv(\Gamma_s(\Omega)), \text{ where} \tag{4a}$$

$$\Gamma_s(\Omega) \equiv \{(x, y) : T_s x + W_s y \geq r_s, x \in \Omega, y \in Y\}, \ \forall s. \tag{4b}$$

By a convexification process (e.g., see Sherali and Adams [20]), suppose that we have the representation of $Z_s(\Omega)$ in a possibly higher dimensional space (x, y, z), including certain new variables z in the added dimensions, given as follows:

$$Z_s(\Omega) = \{(x, y) : H_{1s} x + H_{2s} y + H_{3s} z \geq h_s, x \in \Omega, (y, z) \geq 0\}, \ \forall s. \tag{4c}$$

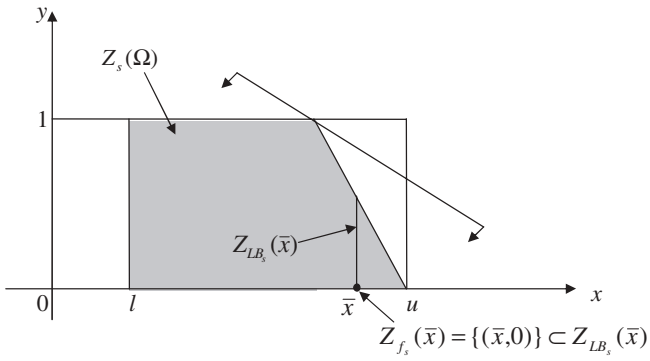

Fig. 1. Illustration of the Concepts of Proposition 1

Based on this representation, define the following function for any fixed $x \in \Omega$:

$$LB_s(x) \equiv \min\{g^T y : H_{2s} y + H_{3s} z \geq h_s - H_{1s} x, (y, z) \geq 0\} = \underset{y:(x,y)\in Z_s(\Omega)}{\text{minimum}} \{g^T y\}, \forall s. \tag{5}$$

Proposition 1. *For any $s \in \{1, \ldots, S\}$, consider the functions $f_s(x)$ and $LB_s(x)$ as defined by* (3) *and* (5)*, respectively, over $x \in \Omega$. Then, we have,*

$$f_s(x) \geq LB_s(x), \forall x \in \Omega. \tag{6a}$$

Furthermore, if $x \in vert(\Omega)$, where $vert(\Omega)$ denote the vertices of Ω, or, more generally, if there exists an optimal solution $\bar{y}$ that evaluates $LB_s(x)$ such that $\bar{y} \in Y$, then we have

$$f_s(x) = LB_s(x). \tag{6b}$$

Proof. Consider any fixed $\bar{x} \in \Omega$. Define

$$Z_{LB_s}(\bar{x}) \equiv \{y : (\bar{x}, y) \in Z_s(\Omega)\}, \text{ and } Z_{f_s}(\bar{x}) \equiv conv\{y : (\bar{x}, y) \in \Gamma_s(\Omega)\}. \tag{7a}$$

Hence, from (3), (4b), and (5), we have,

$$f_s(\bar{x}) = \min\{g^T y : y \in Z_{f_s}(\bar{x})\} \text{ and } LB_s(\bar{x}) = \min\{g^T y : y \in Z_{LB_s}(\bar{x})\}. \tag{7b}$$

Note that $Z_{LB_s}(\bar{x})$ is given by $conv(\Gamma_s(\Omega)) \cap \{(x, y) : x = \bar{x}\}$, while $Z_{f_s}(\bar{x})$ is given by $conv(\Gamma_s(\Omega) \cap \{(x, y) : x = \bar{x}\})$. Hence,

$$Z_{f_s}(\bar{x}) \subseteq Z_{LB_s}(\bar{x}), \forall \bar{x} \in \Omega, \text{ with } Z_{f_s}(\bar{x}) = Z_{LB_s}(\bar{x}), \forall \bar{x} \in vert(\Omega), \tag{8}$$

where the latter statement holds true because $x = \bar{x}$ is facial with respect to $conv(\Gamma_s(\Omega))$ for $\bar{x} \in vert(\Omega)$. The results (6a) and (6b) now follow from (7b) and (8) and the fact that if an optimal solution $\bar{y}$ that evaluates $LB_s(\bar{x})$ is also feasible to $Z_{f_s}(\bar{x})$, then $f_s(\bar{x}) = LB_s(\bar{x})$. □

Figure 1 illustrates the concepts of Proposition 1 for a single binary variable y and a continuous variable x, by displaying a situation where for $\bar{x} \notin vert(\Omega)$, we have $Z_{f_s}(\bar{x}) \subset Z_{LB_s}(\bar{x})$, whereas for either $\bar{x} = l$ or $\bar{x} = u$, it is clear that $Z_{f_s}(\bar{x}) = Z_{LB_s}(\bar{x})$.

Consequently, based on Proposition 1 and (5), we get for any $x \in \Omega$,

$$f_s(x) \geq LB_s(x) = \max\{\phi(h_s - H_{1s}x) : \phi H_{2s} \leq g^T,\ \phi H_{3s} \leq 0,\ \phi \geq 0\},\ \forall s. \quad (9)$$

Define

$$\Lambda_s \equiv \{\phi : \phi H_{2s} \leq g^T,\ \phi H_{3s} \leq 0, \phi \geq 0\}, \quad \forall s = 1, \ldots, S. \quad (10)$$

Then, from (9) and (10), we get for any $x \in \Omega$ (since $LB_s(x)$ is finite by assumptions A1 and A3)

$$f_s(x) \geq \max_{\phi^i \in vert(\Lambda_s)} \{\phi^i(h_s - H_{1s}x)\}, \quad \forall s = 1, \ldots, S. \quad (11)$$

Accordingly, let us define the following *lower bounding master program* (LBMP(Ω)), which is designated based on a specification of Ω for subsequent use.

$$\textbf{LBMP}(\Omega) : \text{Minimize } c^T x + \sum_s p_s \eta_s \quad (12a)$$

$$\text{subject to } \eta_s \geq \phi^i(h_s - H_{1s}x),\ \forall \phi^i \in vert(\Lambda_s),\ \forall s = 1, \ldots, S \quad (12b)$$

$$x \in X \cap \Omega. \quad (12c)$$

Proposition 2. *Problem LBMP(Ω) provides a lower bound for SMIP. Moreover, if $(\bar{x}, \bar{\eta}_s,\ s = 1, \ldots, S)$ solves LBMP(Ω), and either (a) $\bar{x} \in vert(\Omega)$, or more generally, (b) the solution $\bar{y}_s$ obtained when solving $LB_s(\bar{x})$ satisfies $\bar{y}_s \in Y$, $\forall s = 1, \ldots, S$, then $\bar{x}$ solves SMIP with the same objective value given by $c^T\bar{x} + \sum_s p_s\bar{\eta}_s$.*

Proof. Note that from (2), (11), and (12), we have that LBMP(Ω) provides a lower bound on SMIP. Moreover, by Proposition 1, we have that under either of the conditions (*a*) or (*b*) of the proposition, equality holds true in (6b) or (11), and so, $\bar{x}$ solves SMIP with the same objective value as for LBMP(Ω). □

Proposition 2 leads to a decomposition-based B&B (**DBAB**) algorithm for solving SMIP, which is predicated on the partitioning of the hyperrectangle Ω. In this procedure (a more formal statement is given below), starting with iteration $k = 1$ and a list of active nodes $L_1 \equiv \{1\}$, where $\Omega^1 \equiv \Omega$, at any general iteration k, we will have a current index list L_k of active (non-fathomed) nodes, where each $\lambda \in L_k$ corresponds to some hyperrectangle $\Omega^\lambda \equiv \{x : l^\lambda \leq x \leq u^\lambda\} \subseteq \Omega$. For each such node, we will have computed a lower bound LB_λ via the solution of LBMP(Ω^λ). (The algorithm described below, as well as its convergence arguments, remains the same if the lower bounds are computed by instead solving $\overline{\text{LBMP}}(\Omega^\lambda)$, which is given by (12) but with the integrality restrictions on x_i, for $i \in I_2$, relaxed. Naturally, such a lower bound is potentially weaker, although relatively easier to compute.) Note that LBMP(Ω^λ) (or $\overline{\text{LBMP}}(\Omega^\lambda)$) will be solved via a standard row-generation or Benders' scheme, by iterating between the master program (12) and the subproblems (5) or (9), for each $s = 1, \ldots, S$, with $\Omega \equiv \Omega^\lambda$ (see Section 3.2 for details). Whenever the lower boundingsolution x^λ, say,

for any node subproblem λ turns out to satisfy either of the conditions of Proposition 2 with respect to Ω^λ, then the corresponding value of LBMP(Ω^λ) provides the actual value of SMIP restricted to Ω^λ, and we can fathom this node, updating the incumbent solution and its value, x^* and ν^*, respectively, as possible. Additionally, if $LB_\lambda \geq \nu^*$, or whenever the value of the current relaxed master program (12) exceeds ν^* during the solution process, we can fathom node λ. Therefore, at any iteration k, all active nodes $\lambda \in L_k$ will satisfy $LB_\lambda < \nu^*$, and we will select

$$\lambda(k) \in \operatorname{argmin}\{LB_\lambda : \lambda \in L_k\} \tag{13}$$

and proceed by partitioning the corresponding hyperrectangle $\Omega^{\lambda(k)}$ into two sub-hyperrectangles based on a *branching variable* x_p selected according to the following rule, where $x^{\lambda(k)}$ is the optimal solution obtained for LBMP($\Omega^{\lambda(k)}$).

Branching Rule A:

Define

$$\theta_i \equiv \min\{x_i^{\lambda(k)} - l_i^{\lambda(k)},\ u_i^{\lambda(k)} - x_i^{\lambda(k)}\}, \quad \forall i = 1, \ldots, n, \tag{14a}$$

and select

$$p \in \operatorname*{argmax}_{i=1,\ldots,n}\{\theta_i\}. \tag{14b}$$

Accordingly, partition $\Omega^{\lambda(k)}$ into two sub-hyperrectangles by partitioning the interval $[l_p^{\lambda(k)}, u_p^{\lambda(k)}]$ into two sub-intervals as follows

$$\left.\begin{array}{cccc} [l_p^{\lambda(k)}, x_p^{\lambda(k)}] & \text{and} & [x_p^{\lambda(k)}, u_p^{\lambda(k)}], & \text{if} \quad p \in I_1; \\ [0,0] & \text{and} & [1,1], & \text{if} \quad p \in I_2. \end{array}\right\} \tag{14c}$$

(Note that the case $p \in I_2$ arises only when $\overline{\text{LBMP}}(\Omega^\lambda)$ is used to compute lower bounds.) The resulting **DBAB** algorithm for solving SMIP is stated formally below. (As mentioned above, LBMP$(\cdot)$ may be replaced by $\overline{\text{LBMP}}(\cdot)$ throughout the following discussion in this section, with the added premise that the updating of incumbent solutions via Proposition 2 should additionally verify that $\bar{x}$ satisfies the integrality restrictions defining X.)

Algorithm DBAB.

Step 0: Initialization Step. Initialize the incumbent solution x^* to be null and set its objective value as $\nu^* = \infty$. Let the iteration counter $k = 1$, the number of nodes enumerated $N = 1$, and commence with the list of active nodes $L_k = \{1\}$, and set $\lambda(k) = 1$, and $\Omega^1 = \Omega$, with $[l^1, u^1] \equiv [l, u]$. Use the prescribed decomposition algorithm (Section 3.2) to solve LBMP(Ω^1), and let x^1 be the solution obtained of objective value $LB_1 = \nu[\text{LBMP}(\Omega^1)]$. If either of the conditions of Proposition 2 holds true, then stop with $x^* = x^1$ as an optimal solution to SMIP having an objective value $\nu^* = LB_1$. Otherwise, select a branching variable x_p via (14a) and (14b), let $\epsilon \geq 0$ be a chosen optimality tolerance, and proceed to Step 1.

Step 1: Partitioning Step. Partition $\Omega^{\lambda(k)}$ for the selected active node $\lambda(k)$ into two sub-hyperrectangles according to (14c) based on the identified branching variable x_p. Replace $\lambda(k)$ within L_k by these two new node indices, $N + 1$ and $N + 2$.

Step 2: Bounding Step. Solve the problem LBMP(Ω^{N+t}) for $t = 1, 2$, corresponding to each of the two new nodes generated. If x^{N+t} satisfies either of the conditions of Proposition 2 with respect to Ω^{N+t} for any of $t = 1, 2$, then update the incumbent solution x^* and its objective value ν^*, if necessary.
Step 3: Fathoming Step. Fathom any non-improving nodes by setting $L_{k+1} = L_k \setminus \{\lambda \in L_k : LB_\lambda + \epsilon \geq \nu^*\}$. If $L_{k+1} = \emptyset$, then stop with the incumbent solution as ϵ-optimal. Otherwise, select and store branching variable indices using (14a) and (14b) for each of the two new nodes generated if still active (i.e., belong to L_{k+1}), increment k by one, N by two, and proceed to Step 4.
Step 4: Node Selection Step. Select an active node $\lambda(k)$ as in (13), and return to Step 1. □

Proposition 3. (**Convergence result**). *Algorithm DBAB (run with $\epsilon \equiv 0$) either terminates finitely with the incumbent solution being optimal to Problem SMIP, or else, we get $k \to \infty$ such that along any infinite branch of the B&B tree that is associated with the nested sequence of partitions $\{\Omega^{\lambda(k)}\}$, $k \in K_1$, say, any accumulation point of the corresponding solution sequence $\{x^{\lambda(k)}\}_{K_1}$ solves SMIP.*

Proof. The case of finite termination is clear from the derivation of the algorithm. Hence, suppose that $k \to \infty$, and consider any infinite branch of the B&B tree generated as identified by the proposition. Over some convergent subsequence indexed by $K_2 \subseteq K_1$, if necessary (noting the boundedness of the sequence generated), let

$$\left\{x^{\lambda(k)}, l^{\lambda(k)}, u^{\lambda(k)}\right\}_{K_2} \to (x^*, l^*, u^*). \tag{15}$$

We need to show that x^* solves Problem SMIP.

First of all, note that since $LB_{\lambda(k)}$ is the least lower bound among all active nodes at each iteration k, we have that

$$\nu[\text{SMIP}] \geq LB_{\lambda(k)}, \quad \forall k \in K_2. \tag{16}$$

Next, note that we can equivalently view LBMP($\Omega^{\lambda(k)}$) as follows:

$$\text{Minimize } \{c^T x + \sum_s p_s g^T y_s : (x, y_s) \in Z_s(\Omega^{\lambda(k)}),\ \forall s,\ x \in X \cap \Omega^{\lambda(k)}\},$$

where $(x^{\lambda(k)}, y_s^{\lambda(k)}, \forall s)$ solves LBMP($\Omega^{\lambda(k)}$). By (15) and the boundedness of $Z_s(\cdot)$, $\forall s$, and by replacing K_2 by an appropriate subsequence, if necessary, suppose that $\{(x^{\lambda(k)}, y_s^{\lambda(k)}, \forall s)\}_{K_2} \to (x^*, y_s^*, \forall s)$. Since $(x^*, y_s^*, \forall s)$ is feasible to LBMP(Ω^*), where $\Omega^* \equiv \{x : l^* \leq x \leq u^*\}$, we have that

$$c^T x^* + \sum_s p_s g^T y_s^* \geq \nu[\text{LBMP}(\Omega^*)]. \tag{17}$$

We now show that equality must hold true in (17). Suppose on the contrary that

$$c^T x^* + \sum_s p_s g^T y_s^* > c^T \hat{x} + \sum_s p_s g^T \hat{y}_s = \nu[\text{LBMP}(\Omega^*)] \tag{18}$$

where $(\hat{x}, \hat{y}_s, \forall s)$ solves LBMP(Ω^*). Because $\{\Omega^{\lambda(k)}\}_{K_2}$ is a nested sequence, we have that $\Omega^* \subseteq \Omega^{\lambda(k)}$, $\forall k \in K_2$, and so $Z_s(\Omega^*) \subseteq Z_s(\Omega^{\lambda(k)})$, $\forall k \in K_2$. Consequently, $(\hat{x}, \hat{y}_s, \forall s)$ is feasible to LBMP($\Omega^{\lambda(k)}$), $\forall k \in K_2$, and since $\{c^T x^{\lambda(k)} + \sum_s p_s g^T y_s^{\lambda(k)}\}_{K_2} \to c^T x^* + \sum_s p_s g^T y_s^*$, we have that for $k \in K_2$ large enough,

$$c^T x^{\lambda(k)} + \sum_s p_s g^T y_s^{\lambda(k)} > c^T \hat{x} + \sum_s p_s g^T \hat{y}_s,$$

which contradicts the optimality of $(x^{\lambda(k)}, y_s^{\lambda(k)}, \forall s)$ for LBMP($\Omega^{\lambda(k)}$). Hence, equality holds true in (17), and so, viewing LBMP(Ω^*) in the projected x-space, we have that

$$\{\Omega^{\lambda(k)}\}_{K_2} \to \Omega^* \equiv \{x : l^* \le x \le u^*\}, \text{ where } x^* \text{ solves LBMP}(\Omega^*). \tag{19}$$

Now, in the infinite sequence of iterations indexed by $k \in K_2$, there exists some variable x_p, $p \in I_1$, that is branched infinitely often according to (14a)–(14c). Let $K_3 \subseteq K_2$ correspond to iterations at which x_p is selected as the branching variable. By (14c), we have that $x_p^{\lambda(k)} \notin (l_p^{\lambda(k')}, u_p^{\lambda(k')})$, $\forall k' \in K_3, k' > k$, while $x_p^* \in [l_p^*, u_p^*]$. Hence, we must have $x_p^* = l_p^*$ or $x_p^* = u_p^*$. By (14a), this means that $\theta_p \to 0$, which in turn implies from (14b) that

$$\theta_i \to 0, \quad \forall i = 1, \ldots, n, \text{ and so, } x^* \in vert(\Omega^*). \tag{20}$$

By (19) and Proposition 2, we therefore have that the limiting solution x^* and its value LBMP(Ω^*) provide an upper bounding solution and value, respectively, for Problem SMIP. Hence,

$$c^T x^* + \sum_s p_s f_s(x^*) = \nu[\text{LBMP}(\Omega^*)] = \lim_{\substack{k \to \infty \\ k \in K_3}} LB_{\lambda(k)} \ge \nu[\text{SMIP}]. \tag{21a}$$

But since $\{LB_{\lambda(k)}\}_{K_2}$ is monotone increasing and bounded from above, and noting that $K_3 \subseteq K_2$, we get from (16) that

$$\nu[\text{SMIP}] \ge \lim_{\substack{k \to \infty \\ k \in K_3}} LB_{\lambda(k)}. \tag{21b}$$

Hence, from (21a) and (21b), x^* solves SMIP. □

Proposition 4. (**Alternative Branching Rules**). *The result of Proposition 3 continues to hold true under the following two alternative branching rules:*
Branching Rule B. *Define* θ_i, $\forall i$, *as in* (14a), *select* p *as in* (14b), *but replace* (14c) *in Branching Rule A by*

$$\left.\begin{array}{c} \left[l_p^{\lambda(k)}, \frac{l_p^{\lambda(k)} + u_p^{\lambda(k)}}{2}\right], \left[\frac{l_p^{\lambda(k)} + u_p^{\lambda(k)}}{2}, u_p^{\lambda(k)}\right], \text{ if } p \in I_1; \\ [0,0] \text{ and } [1,1], \text{ if } p \in I_2. \end{array}\right\} \tag{22}$$

Branching Rule C. *Select* p *according to*

$$p \in \underset{i=1,\ldots,n}{\text{arglexmax}}\{(u_i^{\lambda(k)} - l_i^{\lambda(k)}, \theta_i)\} \tag{23}$$

where θ_i *is defined by* (14a), *and then replace* (14c) *in Branching Rule A by* (22).

Proof. Following the proof of Proposition 3, Branching Rule B yields that $x_p^* = l_p^* = u_p^*$ by virtue of the bisection process for indices in I_1 as per (22). Hence, $\theta_p \to 0$ in the proof of Proposition 3, which then implies that (20) holds true. The remainder of the proof proceeds identically. Likewise, for Branching Rule C, since (23) selects an index having the largest interval (with the first priority), and adopts the partitioning scheme of (22), we get that $l^* = u^*$ in this case, which again leads to (20) holding true, with the remainder of the proof following that of Proposition 3. □

Remark 1. Note that when $|J_2|$ is relatively small, the entire set (4c) can be generated *a priori* and then used to solve the subproblems (5) that evaluate $LB_s(x)$, $\forall s$, in a decomposition approach for solving LBMP(Ω). Alternatively, when it is not computationally viable to *a priori* generate the entire convex hull representation as in (4c), the subproblems given by (5) can be solved via a sequential convexification procedure that generates cutting planes as necessary, which are valid for $Z_s(\Omega)$ of (4a). The corresponding master program constraint (12b) can then be generated from the resulting linear program at the termination of this scheme. The details for such a finite cutting plane procedure for solving Problem (5) are given in Section 3.1 below. □

3. Algorithmic routines for solving subproblems and master programs

In this section, we present algorithmic procedures for solving the subproblem (5) for any given $\bar{x} \in X \cap \Omega$ (**Algorithm SP**), and for solving the lower-bounding master program LBMP(Ω) (or its relaxation $\overline{\text{LBMP}}(\Omega)$) as given by (12) for any Ω (**Algorithm LBMP**). These routines are described in turn below.

3.1. Cutting plane procedure for solving subproblems of equation (5)

In essence, we can follow the cutting plane scheme described in Sherali and Fraticelli [21] to sequentially construct valid inequalities for $Z_s(\Omega)$ of (4a) in order to solve the subproblem (5). However, the difficulty in directly implementing this scheme lies in the fact that when x is fixed at $\bar{x}$, we may not always have y_j binary for all $j \in J_2$ at the extreme points of $Z_{LB_s}(\bar{x})$ as defined in (7a). We thus need to be able to detect whether Problem (5) is already solved by some solution $\bar{y}$ obtained for a certain relaxation of Problem (5), where $\bar{y}_j$ might not be binary-valued for all $j \in J_2$. This can be achieved as follows. Let

$$\begin{aligned}\hat{Z}_s(\Omega) \equiv \{(x, y) : {} & T_s x + W_s y \geq r_s, \text{ and} \\ & \alpha_s^k x + \beta_s^k y \geq \gamma_s^k, \forall k = 1, \ldots, K, x \in \Omega, y \geq 0\},\end{aligned} \tag{24}$$

where the constraints $y_j \leq 1, \forall j$, are included in $T_s x + W_s y \geq r_s$ and where $\alpha_s^k x + \beta_s^k y \geq \gamma_s^k, \forall k = 1, \ldots, K$, are a set of valid inequalities for $Z_s(\Omega)$. Given any $\bar{x} \in X \cap \Omega$, define the restricted set

$$\hat{Z}_{LB_s}(\bar{x}) \equiv \{y : (\bar{x}, y) \in \hat{Z}_s(\Omega)\}, \tag{25a}$$

along with the associated lower bounding value

$$\widehat{LB}_s(\bar{x}) \equiv \min\left\{g^T y : y \in \hat{Z}_{LB_s}(\bar{x})\right\}. \tag{25b}$$

Proposition 5. *Given any $\bar{x} \in X \cap \Omega$, let $\bar{y}$ be an optimal solution that evaluates $\widehat{LB}_s(\bar{x})$. If $\bar{y}_j \in \{0, 1\}, \forall j \in J_2$, or if $(\bar{x}, \bar{y})$ can be represented as a convex combination of some extreme points of $\hat{Z}_s(\Omega)$ such that these extreme points have binary y_j-variables for all $j \in J_2$, then $\bar{y}$ solves Problem* (5) *with $x \equiv \bar{x}$ fixed, i.e., $\widehat{LB}_s(\bar{x}) = LB_s(\bar{x})$.*

Proof. Since $\hat{Z}_s(\Omega) \supseteq Z_s(\Omega)$, we have that

$$Z_{LB_s}(\bar{x}) \subseteq \hat{Z}_{LB_s}(\bar{x}), \text{ and hence, } LB_s(\bar{x}) \geq \widehat{LB}_s(\bar{x}). \tag{26}$$

Now, if $\bar{y}_j \in \{0, 1\}, \forall j \in J_2$, then $(\bar{x}, \bar{y}) \in Z_s(\Omega)$. Alternatively, if $(\bar{x}, \bar{y})$ can be represented as a convex combination of some extreme points (x^p, y^p), $p \in P$, of $\hat{Z}_s(\Omega)$, where $y_j^p \in \{0, 1\}, \forall j \in J_2, \forall p \in P$, we have that $(x^p, y^p) \in Z_s(\Omega), \forall p \in P$, and so, $(\bar{x}, \bar{y}) \in Z_s(\Omega)$. In either case, this yields that $\bar{y} \in Z_{LB_s}(\bar{x})$, or that,

$$\widehat{LB}_s(\bar{x}) = g^T \bar{y} \geq \min_{y \in Z_{LB_s}(\bar{x})} g^T y = LB_s(\bar{x}). \tag{27}$$

The proof now follows from (26) and (27). □

We now present an algorithm that embeds the cutting plane game of Jeroslow [9] along with a purification strategy within its operation to solve Problem (5).

Algorithm SP (*for solving Problem* (5)*, given $\bar{x} \in X \cap \Omega$*).
Initialization: Let $\alpha_s^0 x + \beta_s^0 y \geq \gamma_s^0$ be an initial set of valid inequalities for $Z_s(\Omega)$, which might be possibly empty or might be inherited from parent nodes (see Remark 2 below). Set

$$k = 0,\ T_s^0 = \begin{bmatrix} T_s \\ \alpha_s^0 \end{bmatrix},\ W_s^0 = \begin{bmatrix} W_s \\ \beta_s^0 \end{bmatrix}, \quad \text{and } r_s^0 = \begin{bmatrix} r_s \\ \gamma_s^0 \end{bmatrix}. \tag{28}$$

Step 1: Solve the LP relaxation

$$\widehat{LB}_s^k(\bar{x}) \equiv \min\{g^T y : W_s^k y \geq r_s^k - T_s^k \bar{x},\ y \geq 0\}, \tag{29}$$

and let $\bar{y}$ be an extreme point optimal solution. If $\bar{y}_j \in \{0, 1\}, \forall j \in J_2$, then by Proposition 5, $\bar{y}$ solves Problem (5) and we can stop. Otherwise, denote $\hat{Z}_s(\Omega) \equiv \{(x, y) : T_s^k x + W_s^k y \geq r_s^k,\ x \in \Omega,\ y \geq 0\}$, and proceed to Step 2.
Step 2: Use the polynomial-time (purification) algorithm described in Sherali [19] to represent $(\bar{x}, \bar{y})$ in terms of the extreme points of $\hat{Z}_s(\Omega)$. Let P be the index set of these extreme points, so that $(\bar{x}, \bar{y}) = \sum_{p \in P} \lambda_p (x^p, y^p)$, where $(x^p, y^p) \in vert(\hat{Z}_s(\Omega))$, $\forall p \in P$, and where $\sum_{p \in P} \lambda_p = 1, \lambda_p > 0, \forall p \in P$. If $y_j^p \in \{0, 1\}, \forall j \in J_2, p \in P$, then again by Proposition 5, $\bar{y}$ solves Problem (5), and we can stop. Otherwise, replace $P \leftarrow P \setminus \{p \in P : y_j^p \in \{0, 1\},\ \forall j \in J_2\} \neq \emptyset$, and go to Step 3.

Step 3: Denote

$$q = \max\{j \in J_2 : \exists p \in P, \quad \text{such that } 0 < y_j^p < 1\}. \tag{30}$$

Extract a subsystem $\tilde{Z}_s(\Omega)$ from $\hat{Z}_s(\Omega)$, by removing from $\hat{Z}_s(\Omega)$ those cuts that were previously generated based on indices $j \geq q$ selected according to (30). Using this subsystem, apply the RLT cutting plane procedure described in Sherali and Fraticelli [21] (also, see Sherali and Adams [20]) to generate a cut

$$\beta_s^{k+1} y \geq \gamma_s^{k+1} - \alpha_s^{k+1} x, \tag{31}$$

which is a facet of $conv(\tilde{Z}_s(\Omega) \cap \{(x, y) : y_q \text{ is binary}\})$, to delete the selected fractional extreme point (x^t, y^t), where $t \in \underset{p \in P}{\operatorname{argmin}}\{|0.5 - y_q^p| : 0 < y_q^p < 1\}$. Let

$$T_s^{k+1} = \begin{bmatrix} T_s^k \\ \alpha_s^{k+1} \end{bmatrix}, \; W_s^{k+1} = \begin{bmatrix} W_s^k \\ \beta_s^{k+1} \end{bmatrix}, \quad \text{and } r_s^{k+1} = \begin{bmatrix} r_s^k \\ \gamma_s^{k+1} \end{bmatrix}, \tag{32}$$

and increment k by one.

If $(\bar{x}, \bar{y})$ violates (31), return to Step 1. Otherwise, let $P \leftarrow P \setminus \{p \in P : (x^p, y^p)$ violates $(31)\}$. If $P \neq \emptyset$, then there exist other fractional vertices of $\hat{Z}_s(\Omega)$ that are not cut off by (31); hence, repeat Step 3. Otherwise, $P = \emptyset$, and the fractional extreme points of $\hat{Z}_s(\Omega)$ that were used to represent $(\bar{x}, \bar{y})$ in Step 2 are all cut off. Update $\hat{Z}_s(\Omega) = \{(x, y) : T_s^k x + W_s^k y \geq r_s^k, \; x \in \Omega, \; y \geq 0\}$. Since $(\bar{x}, \bar{y}) \in \hat{Z}_s(\Omega)$, it is still reproducible as a convex combination of the vertices of the updated $\hat{Z}_s(\Omega)$, and so, return to Step 2. □

Let K be the index for the last iteration, and denote by $\phi_s \geq 0$ an optimal dual solution to (29) where $k \equiv K$. We therefore obtain

$$\eta_s \geq \phi_s(r_s^K - T_s^K x) \tag{33}$$

as a Benders' cut of the type (12b) for scenario s, $s = 1, \ldots, S$.

Proposition 6. *Algorithm SP finitely solves Problem* (5) *for any fixed* $x = \bar{x} \in X \cap \Omega$.

Proof. Algorithm SP terminates when the conditions stated in Proposition 5 are satisfied, whence, Problem (5) is solved. By the adopted RLT cutting plane procedure of Sherali and Fraticelli [21] and Theorem 3.1 of Balas et al. [3], it follows that there are only a finite number of such cuts that can be generated before we ultimately construct $Z_s(\Omega)$ in the worst case. At this point, the conditions of Proposition 5 must necessarily be satisfied, and so, the above procedure terminates finitely. □

Remark 2. For a given Ω, cuts that are generated as above can be reused in the next call of the subproblem (5) while solving a given problem LBMP(Ω). Moreover, as shown below, the information generated while solving LBMP(Ω) for one Ω can be advantageously reused based on the structure of (4) and (5) for a subsequent Ω. To see this, notice first of all that $\Omega^{N+t} \subset \Omega^N$, $\forall t = 1, 2, \forall N$. Hence, inductively, for each $s = 1, \ldots, S$, we have that the cuts (31) that are generated for $Z_s(\Omega^N)$ are valid for all subsequent sets $Z_s(\Omega^M)$, where $M > N$ and M is a node of the subtree that is rooted at node N

(so that, $\Omega^M \subset \Omega^N$). Consequently, these cuts can be included in (28) when applying Algorithm SP to solve Benders' subproblems (5) under scenario s in the process of solving LBMP(Ω^M). Similarly, for any $\Omega^M \subset \Omega^N$, any dual multiplier $\phi \in \Lambda_s^{\Omega_N}$ (appropriately augmented) corresponds to a feasible, though not necessarily extremal, solution to $\Lambda_s^{\Omega_M}$, where $\Lambda_s^{\Omega_N}$ and $\Lambda_s^{\Omega_M}$ denote the sets of dual multipliers feasible to Problem (5) when solving a subproblem under the restrictions $x \in \Omega_N$ and $x \in \Omega_M$, respectively. (Imagine as if the restrictions representing Ω^M are written as those present in Ω^N plus any additional constraints, and let the dual multipliers associated with the RLT convexification constraints generated off these additional constraints be zeros.) Benders' cuts (33) developed for Ω^N are therefore also valid for Ω^M. □

Remark 3. Under the additional assumption of fixed recourse, for any given Ω, it is also advantageous if the multipliers for the dual projection cone used for generating the cuts for one scenario can be shared with other scenarios to generate valid cuts. While we could explicitly check appropriate conditions under which a cut generated for one scenario could be used to obtain a valid inequality for another scenario, this idea would require a careful implementation and experimental evaluation. We therefore recommend this specialization for exploiting a fixed recourse structure for future research. □

3.2. Benders' Scheme for Solving LBMP(Ω)

In this sub-section, we combine a decomposition/relaxation scheme with a B&B procedure to solve LBMP(Ω) for use in Algorithm DBAB (for any specified "Ω"). In this process, the master program is solved using B&B, and whenever a feasible solution that satisfies the integrality requirements of the first-stage variables is obtained, the corresponding subproblem is solved for each scenario to verify feasibility to the overall master program, and to generate Benders' cuts when needed. Note that in Algorithm DBAB, when we use $\overline{\text{LBMP}}(\cdot)$ instead of LBMP$(\cdot)$ to compute lower bounds, the same algorithm described below can be used, except that the integrality restriction on the first stage variables are relaxed, i.e., in essence, we assume that $I_2 \equiv \emptyset$.

Algorithm LBMP (*for solving LBMP(Ω) using a B&B and row-generation scheme*).
Step 0: Initialization Step. Initialize the incumbent solution (x^*, η^*) as null and set its objective value as $\nu^* = \infty$. Let the iteration counter $r = 1$, the number of nodes enumerated $N = 1$, the current node $\lambda(r) = 1$, the current node restrictions $PS_{\lambda(r)} = \emptyset$, and commence with the list of active nodes $L_r = \{1\}$. Let $k = 1, \ldots, K_s, \forall s = 1, \ldots, S$, be the indices of the Benders' cuts inherited from the enclosing partition $\Omega' \supset \Omega$ ($K_s = 0$ if no such cut exists).
Step 1: Branching Step. If an optimal solution has been derived for the current node $\lambda(r)$ at some previous iteration, then denote it as $(\hat{x}^{\lambda(r)}, \hat{\eta}_s^{\lambda(r)}, \forall s)$. If no new Benders' cuts have been added since that solution was obtained, then set the optimal solution $(x^{\lambda(r)}, \eta_s^{\lambda(r)}, \forall s) \equiv (\hat{x}^{\lambda(r)}, \hat{\eta}_s^{\lambda(r)}, \forall s)$ and let $\nu^{\lambda(r)}$ be its objective value; otherwise, solve

$$\text{Minimize} \quad c^T x + \sum_s p_s \eta_s \tag{34a}$$

$$\text{subject to} \quad PS_{\lambda(r)} \tag{34b}$$

$$\eta_s \geq \xi_s^k - \zeta_s^k x, \quad \forall k = 1, \ldots, K_s, \forall s = 1, \ldots, S \tag{34c}$$

$$\eta_s \geq -M, \quad \forall s = 1, \ldots, S : K_s = 0 \tag{34d}$$

$$x \in \bar{X} \cap \Omega, \tag{34e}$$

where M is chosen as a large number to bound (34) in case $K_s = 0$ for any s, and $\bar{X}$ is the LP relaxation of X defined in (2c). Let $(x^{\lambda(r)}, \eta_s^{\lambda(r)}, \forall s)$ be an optimal solution and let $\nu^{\lambda(r)}$ be its objective value. (If (34) is infeasible, then set $\nu^{\lambda(r)} = \infty$.)

If $x_i^{\lambda(r)} \in \{0, 1\}, \forall i \in I_2$, then proceed to Step 2 if $x^{\lambda(r)} \neq \hat{x}^{\lambda(r)}$, and go to Step 3 if $x^{\lambda(r)} = \hat{x}^{\lambda(r)}$. Otherwise, let $q \in \text{argmin}\{|0.5 - x_i^{\lambda(r)}| : i \in I_2\}$, $PS_{N+1} = PS_{\lambda(r)} \cap \{x_q = 0\}$ and $PS_{N+2} = PS_{\lambda(r)} \cap \{x_q = 1\}$. Replace $\lambda(r)$ within L_r by the new node indices $N + t, \forall t = 1, 2$. Re-solve (34) with (34b) replaced by PS_{N+t}, for $t = 1, 2$, respectively, to obtain $\nu^{N+t}, \forall t = 1, 2$. Increment N by two, and go to Step 3.

Step 2: Cut Generation Step. Apply Algorithm SP to evaluate $LB_s(x^{\lambda(r)}), \forall s = 1, \ldots, S$, using (any of) the previously generated cuts (31) with their right-hand-sides modified according to the current solution $x^{\lambda(r)}$. For each scenario s, $s = 1, \ldots, S$, if $\eta_s^{\lambda(r)} < LB_s(x^{\lambda(r)})$, then derive the Benders' cut (33), denote this cut as $\eta_s \geq \xi_s^{K_s+1} - \zeta_s^{K_s+1} x$, and increment K_s by 1. Furthermore, if $c^T x^{\lambda(r)} + \sum_s p_s LB_s(x^{\lambda(r)}) < \nu^*$, then update the incumbent solution $(x^*, \eta_s^*, \forall s)$ and objective value ν^* with $(x^{\lambda(r)}, LB_s(x^{\lambda(r)}), \forall s)$ and $c^T x^{\lambda(r)} + \sum_s p_s LB_s(x^{\lambda(r)})$, respectively.

Step 3: Fathoming and Node Selection Step. Fathom any non-improving nodes by setting $L_{r+1} \leftarrow L_r \setminus \{\lambda \in L_r : \nu^\lambda + \epsilon \geq \nu^*\}$, where $\epsilon \geq 0$ is a chosen optimality tolerance. If $L_{r+1} = \emptyset$, then stop with the incumbent solution as (ϵ-)optimal. Otherwise, increment r by one, and select an active node $\lambda(r) \in \text{argmin}\{\nu^\lambda : \lambda \in L_r\}$. Return to Step 1. □

4. Illustrative Example

Consider the following example:

$$\textbf{SMIP:} \quad \text{Minimize} \quad -5x_1 - x_2 + \sum_{s=1}^{2} 0.5 f_s(x) \tag{35a}$$

$$\text{subject to} \quad -x_1 - x_2 \geq -1.5 \tag{35b}$$

$$0 \leq x_1 \leq 1, \quad x_2 \text{ binary}, \tag{35c}$$

$$\text{where} \quad f_s(x) = \text{minimum} \; -16y_1 - 19y_2 - 23y_3 - 28y_4 \tag{35d}$$

$$\text{subject to} \begin{bmatrix} -2y_1 - 3y_2 - 4y_3 - 5y_4 \\ -6y_1 - y_2 - 3y_3 - 2y_4 \end{bmatrix} \geq [r_s - T_s x] \tag{35e}$$

$$y_j \leq 1, \quad \forall j \in J \tag{35f}$$

$$y_1, y_2 \geq 0, \quad y_3, y_4 \text{ binary}, \tag{35g}$$

and where $[r_1 - T_1 x] = \begin{bmatrix} -5 + 0.3x_1 \\ 10 + 0.3x_2 \end{bmatrix}$ and $[r_2 - T_2 x] = \begin{bmatrix} -10 + 0.2x_1 \\ 5 + 0.2x_2 \end{bmatrix}$. Note that we have $\Omega = \{x \in R^2 : 0 \leq x \leq e\}$.

4.1. Applying Algorithm DBAB while using LBMP(ω) for computing bounds

At the root of the DBAB tree, we solve LBMP(Ω) for the original set Ω using Algorithm LBMP. The first iteration of LBMP yields $x_2 = 0.5$. We branch on x_2 to create two new nodes, Node 2 and Node 3, having respective additional restrictions of $x_2 = 0$ and $x_2 = 1$, where Node 2 is selected for the second iteration.

In the second iteration, the subproblem for Scenario 1 yields binary values of y_3 and y_4 directly, while the subproblem for Scenario 2 yields a fractional solution, which is cut off by the following generated RLT cuts:

$$-0.25y_2 - 0.25y_3 - 0.5y_4 \geq -0.75, \tag{36a}$$

$$-0.366y_2 - 0.488y_3 - 0.268y_4 \geq -0.878 + 0.024x_1, \text{ and} \tag{36b}$$

$$-0.167y_2 - 0.167_3 - 0.167y_4 \geq -0.333. \tag{36c}$$

Two Benders' cuts are derived for these two scenario subproblems, respectively:

$$\eta_1 \geq -35 + 1.9x_1, \text{ and} \tag{37a}$$

$$\eta_2 \geq -52.333 + 0.533x_2, \tag{37b}$$

and the incumbent objective value is updated to $v^* = -47.717$. Node 2 is again selected and its associated problem is re-solved after appending the Benders cuts (37a) and (37b). This produces the same objective value as the incumbent value v^*, and so, Node 2 is fathomed.

Incorporating the Benders' cuts (37a) and (37b) for the relaxed master program, and inheriting the RLT cuts (36a)–(36c) for the Scenario 2 subproblem, Node 3 of the LBMP tree yields y-solutions that satisfy the binary restrictions for both scenarios. Note that had we not inherited the RLT cuts (36a)–(36c), Scenario 2 would have produced a fractional y-solution, which turns out to require the generation of three other RLT cuts that are different from (36a)–(36c) in order to yield a binary-feasible solution.

No additional Benders' cuts are generated, and Node 3 is non-improving and is thus fathomed. The resulting solution to LBMP(Ω) is given by $x^* = (1, 0)$, with $y^* = (1, 0.9, 0, 0)$ for Scenario 1, $y^* = (0.333, 1, 0, 1)$ for Scenario 2, and $v^* = -47.717$. Since $x^* \in vert(\Omega)$, we have by Proposition 2 that this also solves the original SMIP; hence, no branching on Ω in DBAB is needed.

4.2. Applying Algorithm DBAB while using $\overline{LBMP}(\omega)$ for computing bounds

In lieu of solving LBMP(Ω) for computing bounds, suppose that we solve its LP relaxation $\overline{\text{LBMP}}(\Omega)$. The resulting computations proceed as follows.

DBAB - Iteration $k = 1$**:** We start implementing DBAB with the original hyperrectangle Ω.

At the first iteration when solving $\overline{\text{LBMP}}(\Omega)$ via Algorithm LBMP, the master program yields $x_1 = 1$, $x_2 = 0.5$, $\eta_1 = \eta_2 = -M$, and then, Scenario 1 yields $y = (1, 0.9, 0, 0)$, and generates the Benders' cut (37a). Scenario 2 yields a fractional solution $y = (0.122, 1, 0.389, 1)$, which is represented by two extreme points of $\hat{Z}_2(\Omega)$: $(0.144, 1, 0.378, 1)$ and $(0.1, 1, 0.4, 1)$. The second extreme point is used to generate an RLT cut. This process continues and sequentially generates five fractional solutions, each of which is representable by two extreme points of $\hat{Z}_2(\Omega)$, and is cut off by an RLT cut derived based on one of these extreme points. The final solution obtained is $y = (0.317, 1, 0, 1)$, and the Benders' cut generated using the original constraints (35e) and (35f) for $s = 1$ and the aforementioned five RLT cuts is given by (37b).

At the second iteration of LBMP, upon re-solving the master program after appending the two Benders' cuts (37a) and (37b), we obtain an optimal solution to $\overline{\text{LBMP}}(\Omega)$, as the resulting η-values coincide with the subproblem objective values obtained in the previous iteration.

DBAB - Iteration $k = 2$**:** We continue Algorithm DBAB by partitioning Ω into $\Omega_1 = \{x \in R^2 : 0 \le x_1 \le 1, x_2 = 0\}$ and $\Omega_2 = \{x \in R^2 : 0 \le x_1 \le 1, x_2 = 1\}$, based on the fractional solution $x_2 = 0.5$ obtained for $\overline{\text{LBMP}}(\Omega)$.

The master programs for $\overline{\text{LBMP}}(\Omega_1)$ and $\overline{\text{LBMP}}(\Omega_2)$ are re-solved after inheriting the Benders' cuts (37a) and (37b) from Iteration 1 of Algorithm DBAB, and yield solutions $(1, 0, -33.1, -52.333)$ and $(0.5, 1, -34.05, -51.8)$, respectively, with the corresponding objective values -47.717 and -46.425. For each of $\overline{\text{LBMP}}(\Omega_1)$ and $\overline{\text{LBMP}}(\Omega_2)$, the subproblem solutions obtained for both the scenarios satisfy the binary restrictions with $LB_s(x^{\lambda(1)}) = \eta_s^{\lambda(1)}$, $\forall s$, and hence, we have obtained optimal solutions via the corresponding master problems. For $\overline{\text{LBMP}}(\Omega_1)$, since $x^{\lambda(1)} \in vert(\Omega_1)$, it also provides an incumbent solution x^* for the DBAB procedure, with an objective function value $\nu^* = -47.717$.

The node corresponding to Ω_2 in the DBAB tree is now fathomed since $\nu(\overline{\text{LBMP}}(\Omega_2)) = -46.425 > \nu^*$. No other active nodes exist; the incumbent solution x^* and the y-solutions found when solving $\overline{\text{LBMP}}(\Omega_1)$ yield an optimal solution, which is the same as that obtained in Section 4.1.

5. Computational Results

The proposed decomposition-based B&B algorithm DBAB was implemented in C++ using the CPLEX Callable Library 8.1. The computations were carried out on a Sun-Blade-1000 (UltraSPARC-III) Workstation having 512 MB RAM and a cpu speed of 750 MHz. In our implementation, the maximum number of RLT cuts inherited by each node from its "ancestor" nodes was set to be 20–150 in order to limit the growth in the size of the problem. Whenever this limit was reached, the newly generated cuts were

used to replace the cuts that were derived earliest, because the RLT cuts obtained higher in the tree are presumably weaker than the ones generated more recently. For benchmarking purposes, we also directly solved the deterministic equivalent problem (**DEP**) using CPLEX 8.1.

Our test problems comprise two groups of instances. The first group of instances have four first-stage variables and six to eight second-stage variables. They are similar to the example used in Section 4, with some additional variables and constraints. The first-stage problem has the following form:

$$\begin{aligned}
\text{Minimize} \quad & -2x_1 - 2.5x_2 - 2x_3 - 1.5x_4 + E[f_s(x)] && \text{(38a)}\\
\text{subject to} \quad & -x_1 - x_2 - x_3 - 0.5x_4 \geq -1.5 && \text{(38b)}\\
& -x_1 - 2x_3 - x_4 \geq -3 && \text{(38c)}\\
& -x_1 - 2x_2 - x_3 - 2x_4 \geq -5 && \text{(38d)}\\
& -2x_1 - x_3 - x_4 \geq -4 && \text{(38e)}\\
& 0 \leq x_1, x_2 \leq 1, \quad x_3, x_4 \text{ binary}, && \text{(38f)}
\end{aligned}$$

where constraints (38d) and (38e) were optionally excluded from some problems. The second-stage problems were constructed as follows (having fixed recourse):

$$f_s(x) = \text{minimum} - 16y_1 - 19y_2 - 23y_3 - 28y_4 - 15y_5 - 12y_6 - 10y_7 - 20y_8 \qquad \text{(38g)}$$

subject to

$$-2y_1 - 3y_2 - 4y_3 - 5y_4 - y_5 - 2y_6 - y_7 - 2y_8 \geq -r_{s1} - \sum_{i \in I} T_{s1i} x_i \qquad \text{(38h)}$$

$$-6y_1 - y_2 - 3y_3 - 2y_4 - 2y_5 - y_6 - 2y_7 \geq -r_{s2} - \sum_{i \in I} T_{s2i} x_i \qquad \text{(38i)}$$

$$-3y_1 - 2y_2 - 5y_3 - y_4 - 3y_5 - y_6 - 3y_7 - 2y_8 \geq -r_{s3} - \sum_{i \in I} T_{s3i} x_i \qquad \text{(38j)}$$

$$-y_1 - 2y_2 - y_3 - 2y_4 - 3y_5 - 2y_6 - y_7 \geq -r_{s4} - \sum_{i \in I} T_{s4i} x_i \qquad \text{(38k)}$$

$$0 \leq y_j \leq 1, \quad \forall j \in J_1, \quad y_j \text{ binary}, \forall j \in J_2. \qquad \text{(38l)}$$

Constraint (38k) and variables y_7 and y_8 were not included in all problems, and we defined $J_1 \equiv \{1, 2, 3\}$ for the problems having six second-stage variables, and $J_1 \equiv \{1, 2, 3, 4\}$ for the problems having eight second-stage variables. The technology matrices and right-hand-side values for the second-stage problems of this group were generated using uniform distributions over $[-0.3, 0]$ and $[-15, -5]$, respectively. For a given number of scenarios and a chosen number of variables and constraints, we generated twenty problem instances.

The second group of instances either have four first-stage variables and nine second-stage variables, with two binary variables in the first stage and four binary variables in the second stage, or have six first-stage variables and thirteen second-stage variables, with three binary variables in the first stage and six binary variables in the second stage. The

Table 1. Computational Results

Stage 1	Stage 2	S	DEP cpu (s.)			DBAB_b cpu (s.)			DBAB_r cpu (s.)		
$\lvert I_1\rvert, \lvert I_2\rvert, \lvert C_1\rvert$	$\lvert J_1\rvert, \lvert J_2\rvert, \lvert C_2\rvert$		max	min	avg.†	max	min	avg.†	max	min	avg.†
2, 2, 2	3, 3, 3	128	7288*	2	1461	491	13	123	1203	55	377
2, 2, 4	4, 4, 4	128	699	0	64	189	22	86	1422	59	543
2, 2, 2	3, 3, 3	196	7550*	40	4059	1563	28	413	3724	64	1312
2, 2, 2	3, 3, 3	256	7490*	7240*	7389*	1991	63	749	3571	115	998
2, 2, 4	5, 4, 4	128	7638*	6716	7351*	1013	64	439	2304	101	925
3, 3, 4	7, 6, 4	128	–‡	–‡	–‡	1193‡	366‡	732‡	624‡	624‡	624‡
3, 3, 4	7, 6, 4	256	7534*	7505*	7518*	1950	58	747	3223	462	1733

(1) $|I_1|$, $|I_2|$, $|J_1|$, and $|J_2|$ denote the number of continuous and binary variables, respectively, in each of the two stages; $|C_1|$ and $|C_2|$ denote the respective number of constraints in the two stages, and S denotes the number of scenarios.

(2) * A two-hour time limit was imposed on the computational time, which was checked at the end of each iteration loop.

(3) † Average of all the computational times at termination, including those obtained at the specified two-hour limit, but excluding those for the problems that terminated due to the memory limit.

(4) ‡ When the memory limit of the computer was reached, the best feasible solution, if available, was obtained. In this case, the computational times do not represent the optimal solution times. When solved using DEP, all instances in this group reached the memory limit. Only one instance was solved within the memory limit using DBAB_r; hence, the numbers shown represent the result for the same instance.

coefficients and right-hand-side values are defined so that the solutions are non-trivial. For a given number of scenarios and a chosen number of variables and constraints, we generated eight problem instances.

Table 1 summarizes the number and the sizes of the test problems, along with the maximum, minimum, and average cpu times (in seconds) consumed when using CPLEX to directly solve the deterministic equivalent problem and when applying Algorithm DBAB. In DBAB_b, the binary restrictions on the first-stage variables are enforced when solving $\text{LBMP}(\cdot)$, and in DBAB_r, these restrictions are relaxed, i.e., $\overline{\text{LBMP}}(\cdot)$ is solved instead.

When solving DEP directly via CPLEX, for these 104 test problems using a relative tolerance level $\epsilon = 0.1\%$, 43 instances were not solved to ϵ-optimality within the specified two-hour computational time limit and 12 instances were not solved due to insufficient computer memory. Using DBAB, however, only five instances were not solved by DBAB_b and seven instances were not solved by DBAB_r, due to memory limitations. None of the solution times reached the two-hour time limit in either version of DBAB. As evident from Table 1, Algorithm DBAB exhibits a more robust and efficient performance than solving DEP directly via CPLEX, with DBAB_b being relatively faster as compared with DBAB_r. We add the caveat here that while CPLEX is a commercial package, our implementation of DBAB was rather crude in terms of data structure and memory usage. Hence, the performance of DBAB can be further enhanced by a more sophisticated implementation.

6. Summary and conclusion

This paper focuses on solving two-stage stochastic programs having mixed-integer first- and second-stage variables. The proposed decomposition-based branch-and-bound

algorithm (DBAB) adopts a hyperrectangular partitioning process in the projected space of the first-stage variables. Lower bounds for the nodal problems in the branch-and-bound tree are computed by applying a modified Benders' approach that coordinates a master program with lower-bounding scenario-based second-stage subproblems. Each of these subproblems is derived by sequentially constructing a certain partial convex hull representation of the two-stage solution space. We show that the convexification (RLT) cuts derived for a given hyperrectangle Ω at any node are reusable in subsequent solutions of the subproblems for each scenario at this node by updating the first-stage variable values. Furthermore, these cuts can be inherited by the subproblems of the children nodes of Ω. Likewise, the Benders' cuts derived for a given Ω can also be inherited by the lower bounding master programs solved for the children nodes of Ω in the enumeration tree for Algorithm DBAB. The overall process is proven to converge to a global optimum for the underlying stochastic mixed-integer problem.

We have illustrated the proposed algorithm using a numerical example, and have reported encouraging preliminary computational results using some randomly generated instances. The results clearly exhibit the relative robustness and effectiveness of applying the proposed algorithm in contrast with using a commercial package (CPLEX 8.1) on a deterministic equivalent formulation of the stochastic program. This is a groundbreaking effort, and further investigation is needed to both improve the implementation efficiency, as well as to explore alternative algorithmic strategies such as using Sen and Sherali's [18] branch-and-cut approach for solving subproblems, and devising mechanisms to share cuts among scenario subproblems in the special case of fixed recourse in order to be able to handle relatively larger sized problem instances.

Acknowledgements. This research is supported by the *National Science Foundation* under Grant Number DMI-0552676 and DMI 0245643.

References

1. Ahmed, S.: Mean-risk objectives in stochastic programming. School of Industrial & Systems Engineering, Georgia Tech, Atlanta, GA (2004). Working Paper, available at http://www.optimization-online.org/DB_HTML/2004/04/859.html
2. Ahmed, S., Tawarmalani, M., Sahinidis, N.V.: A finite branch and bound algorithm for two-stage stochastic integer programs. Mathematical Programming **100** (2), 355–377 (2004)
3. Balas, E., Ceria, S., Cornuéjols, G.: A lift-and-project cutting plane algorithm for mixed 0-1 programs. Mathematical Programming **58**, 295–324 (1993)
4. Benders, J.F.: Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik **4**, 238–252 (1962)
5. Blair, C., Jeroslow, R.: The value function of an integer program. Mathematical Programming **23**, 237–273 (1982)
6. Carøe, C.C., Schultz, R.: Dual decomposition in stochastic integer programming. Operations Research Letters **24** (1-2), 37–45 (1999)
7. Carøe, C.C., Tind, J.: A cutting-plane approach to mixed 0-1 stochastic integer programs. European Journal of Operational Research **101** (2), 306–316 (1997)
8. Carøe, C.C., Tind, J.: L-shaped decomposition of two-stage stochastic programs with integer recourse. Mathematical Programming **83a** (3), 451–464 (1998)
9. Jeroslow, R.G.: A cutting plane game for facial disjunctive programs. SIAM Journal of Control and Optimization **18**, 264–280 (1980)
10. Klein Haneveld, W.K., van der Vlerk, M.H.: Stochastic integer programming: general models and algorithms. Annals of Operations Research **85**, 39–57 (1999)

11. Laporte, G., Louveaux, F.V.: The integer L-shaped method for stochastic integer programs with complete recourse. Operations Research Letters **13** (3), 133–142 (1993)
12. Mulvey, J.M., Vanderbei, R.J., Zenios, S.A.: Robust optimization of large-scale systems. Operations Research **43** (2), 264–281 (1995)
13. Ntaimo, L., Sen, S.: The million-variable "march" for stochastic combinatorial optimization. Journal of Global Optimization **32** (3), 385–400 (2005)
14. Schultz, R.: On structure and stability in stochastic programs with random technology matrix and complete integer recourse. Mathematical Programming **70** (1), 73–89 (1995)
15. Schultz, R., Stougie, L., van der Vlerk, M.H.: Two-stage stochastic integer programming: a survey. Statistica Neerlandica **50** (3), 404–416 (1996). Also see http://mally.eco.rug.nl/index.html?biblio/SPlist.html
16. Schultz, R., Tiedemann, S.: Risk aversion via excess probabilities in stochastic programs with mixed-integer recourse. SIAM Journal on Optimization **14** (1), 115–138 (2003)
17. Sen, S., Higle, J.L.: The C^3 theorem and a D^2 algorithm for large scale stochastic mixed-integer programming: set convexification. Mathematical Programming **104** (1), 1–20 (2005)
18. Sen, S., Sherali, H.D.: Decomposition with branch-and-cut approaches for two stage stochastic mixed-integer programming. Mathematical Programming **106**(2), 203–223 (2006)
19. Sherali, H.D.: A constructive proof of the representation theorem for polyhedral sets based on fundamental definitions. American Journal of Mathematical and Management Sciences **7** (3/4), 253–270 (1987)
20. Sherali, H.D., Adams, W.P.: A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems. Kluwer Academic Publishing, Boston, MA (1999)
21. Sherali, H.D., Fraticelli, B.M.P.: A modification of Benders' decomposition algorithm for discrete subproblems: An approach for stochastic programs with integer recourse. Journal of Global Optimization **22**, 319–342 (2002)
22. Stougie, L., van der Vlerk, M.H.: Stochastic integer programming. In: Dell'Amico, M., Maffioli, F., Martello, S., (eds.) Annotated Bibliographies in Combinatorial Optimization, Chapter 9:127–141 (1997)
23. Takriti, S., Ahmed, S.: On robust optimization of two-stage systems. Mathematical Programming **99a**, 109–126 (2004)
24. van der Vlerk, M.H.: Convex approximations for complete integer recourse models. Mathematical Programming **99a**, 297–310 (2004)