Disjunctive Decomposition for Two-Stage Stochastic Mixed-Binary Programs with Random Recourse

Lewis Ntaimo
Department of Industrial and Systems Engineering, Texas A&M University, 3131 TAMU, College Station, TX 77843, USA, ntaimo@tamu.edu

Abstract

This paper introduces disjunctive decomposition for two-stage mixed 0-1 stochastic integer programs (SIPs) with random recourse. Disjunctive decomposition allows for cutting planes based on disjunctive programming to be generated for each scenario subproblem under a temporal decomposition setting of the SIP problem. A new class of valid inequalities for mixed 0-1 SIP with random recourse is presented. In particular, valid inequalities that allow for sharing cut coefficients among scenario subproblems for SIP with random recourse but deterministic technology matrix and righthand side vector are obtained. The valid inequalities are used to derive a disjunctive decomposition method whose derivation has been motivated by real-life stochastic server location problems with random recourse, which find many applications in operations research. Computational results with large-scale instances to demonstrate the potential of the method are reported.

Keywords: Set convexification, cutting plane, random recourse, disjunctive decomposition, stochastic programming

1 Introduction

Stochastic integer programming (SIP) is a branch of stochastic programming that deals with stochastic programs in which the decision variables involve integrality requirements. By inheriting the properties of stochastic linear programs and integer programs, SIP is among the most challenging classes of optimization problems. In two-stage SIP with recourse, the first-stage decisions have to be made in the face of future uncertainty in the problem data, while the recourse decisions are determined in the second-stage based on the realization of the random outcomes. Recourse decisions allow for taking corrective action after a random event has taken place. In this paper we consider the following two-stage SIP problem:

$$\text{SIP1:} \quad \underset{x \in X \cap \mathcal{B}^{n_1}}{\text{Min}} \; c^\top x + E[f(\tilde{\omega}, x)], \tag{1}$$

where c is a known vector in $\Re^{n_1}$, $X \subseteq \Re^{n_1}$ is a convex polyhedral set, $\mathcal{B} \subset \Re^{n_1}$ is the subset of binary vectors, and $\tilde{\omega}$ is a multi-variate discrete random variable with a realization (scenario) ω with probability of occurrence p_ω and sample space Ω. For any $\omega \in \Omega$,

$$\begin{aligned} f(\omega, x) = \text{Min} \;\; & q(\omega)^\top y(\omega) \\ \text{s.t.} \;\; & W(\omega)y(\omega) \geq r(\omega) - T(\omega)x \\ & y(\omega) \in \mathcal{B}^{n_z} \times \Re_+^{n_2 - n_z}. \end{aligned} \tag{2}$$

Problem (2) is referred to as the second-stage (scenario) subproblem and $f(\omega, x)$ as the recourse function. In this problem $q(\omega)$ is a vector in $\Re^{n_2}$, $r(\omega)$ is a vector in $\Re^{m_2}$, and $W(\omega)$ and $T(\omega)$ are matrices in $\Re^{m_2 \times n_2}$ and $\Re^{m_2 \times n_1}$, respectively. We consider instances of SIP1 under the following assumptions: (A1) Ω is a finite set; (A2) X is a closed set which includes the constraints $0 \leq x \leq 1$ and is given as $X = \{x \in \Re^{n_1}_+ \mid Ax \geq b\}$; (A3) subproblem (2) is dual feasible for all $\omega \in \Omega$; and (A4) for all $(\omega, x) \in X \cap \mathcal{B} \times \Omega$, $f(\omega, x) < \infty$. Assumption (A4) requires that subproblem (2) remains feasible for all $(\omega, x) \in X \cap \mathcal{B} \times \Omega$, a property referred to as relatively complete recourse in stochastic programming. The matrices $T(\omega)$ and $W(\omega)$ are referred to in the literature as the technology matrix and recourse matrix, respectively. If $T(\omega) = T$ for all $\omega \in \Omega$, SIP1 is said to have *fixed tenders*. Similarly, if $W(\omega) = W$ for all ω, the problem is said to have *fixed recourse*. In this paper we consider SIP1 under *random recourse*, where $W(\omega)$ is allowed to vary with ω.

Due to assumption (A1), problem (1-2) can also be formulated as a large-scale problem in extensive form (EF) as follows:

$$\begin{aligned} \text{EF:} \quad \underset{x \in X \cap \mathcal{B}^{n_1}}{\text{Min}} \quad & c^\top x + \sum_{\omega \in \Omega} p_\omega q(\omega)^\top y(\omega) \\ \text{s.t. } & T(\omega)x \ + W(\omega)y(\omega) \geq r(\omega), \ \forall \omega \in \Omega \\ & y(\omega) \in \mathcal{B}^{n_z} \times \Re^{n_2 - n_z}_+, \quad \forall \omega \in \Omega. \end{aligned} \tag{3}$$

Direct mixed-integer programming (MIP) solvers such as CPLEX (ILOG, 2003) can be applied directly to problem EF. However, due to the large-scale nature of the problem coupled with the integrality requirements on the decision variables, this approach is generally hopeless. Therefore, this calls for novel decomposition methods that exploit the special structure inherent in the problem formulation. Note that when the second-stage has continuous variables only, the recourse function is a well-behaved piecewise linear and convex function of the first-stage variables. Benders' decomposition (Benders, 1962) is applicable in this case. However, when integrality requirements are imposed on second-stage variables, the problem becomes computationally challenging. The recourse function is now lower semicontinuous with respect to the first-variables (Blair and Jeroslow, 1982), and is generally nonconvex (Schultz, 1993).

This paper introduces disjunctive decomposition for mixed 0-1 SIP with *random recourse*. In particular, valid inequalities based on disjunctive programming are derived for SIP1. These cuts are used in a cutting plane method, which allows for scenario subproblems to share some cut coefficients when SIP1 has random recourse but deterministic technology matrix and righthand side vector. This work has been motivated by need for algorithms for this class of problems. The remainder of this paper is organized as follows. The next section reviews closely related work. In Section 3 theoretical results for set convexification for two-stage SIP with *random recourse* are derived using results from disjunctive programming. The theoretical results are applied to a cutting plane setting for SIP1 in Section 4 and extended to the special case of deterministic technology matrix and righthand side vector in Section 5. A disjunctive decomposition algorithm for SIP1 is presented in Section 6. Computational results on the application of the algorithm to

several SSLPR instances are reported in Section 7. The paper ends with some concluding remarks in Section 8.

2 Related Work

To date very few optimum-seeking algorithms have been developed for SIP, and many of these algorithms assume the *fixed recourse* property. The reader is referred to surveys on algorithms for SIP (Schultz et al., 1996, Klein Haneveld and van der Vlerk, 1999, Sen, 2003) and to textbooks on stochastic programming (Birge and Louveaux, 1997, Ruszczynski and Shapiro, 2003). However, certain applications such as capacity acquisition and assignment under uncertainty (Ahmed and Garcia, 2003, e.g), require *random recourse*. The branch-and-bound algorithm of Ahmed et al. (2004) and the cutting-plane based algorithm of (Carøe and Tind, 1997, Carøe, 1998) allow for SIP problems with random recourse. The algorithm derived by Ahmed et al. (2004) requires general first-stage and pure integer second-stage variables. This method uses a transformed space in which tender variables ($\chi = Tx$) are used to partition the problem using a hyperrectangular partitioning process. The method by Carøe and Tind (1997) and Carøe (1998) uses disjunctive programming to derive cutting-planes under the EF (problem 3) setting and requires continuous first-stage and mixed-binary second-stage decision variables. An extension of the method for SIP with pure binary first-stage and mixed-binary second-stage decision variables is also proposed.

Disjunctive decomposition (D^2) (Sen et al., 2002, Sen and Higle, 2005) takes advantage of the ideas from disjunctive programming (Balas, 1975, Blair and Jeroslow, 1978, Sherali and Shetty, 1980) and reverse convex programming (Sen and Sherali, 1987) to provide a rather general setting for the study of the convex hull of feasible points of SIP problems under the two-stage setting. Sen and Higle (2005) develop a cutting-plane-based approximation of the feasible set of the second-stage problem (2) under the *fixed* recourse property. The D^2 approach provides a mechanism for transforming cuts derived for one instance of the second-stage problem into cuts that are valid for other scenario instances. Implementation and computational experience with the D^2 method on solving large-scale instances from the literature is reported in a dissertation by Ntaimo (2004) and in Ntaimo and Sen (2005).

One motivating application for this work is the stochastic server location problem with random recourse (SSLPR), which has many real-life applications in operations research. SSLPR deals with the optimal location of a limited number of 'servers' with limited capacity of some resource under uncertainty regarding potential future 'client' resource *demand* and *availability* after the servers are located. Constraints include budgetary requirements on the total server location costs and the requirement that each client be served by servers from specific locations. A world-wide problem that has raised a lot of attention in recent years is wildfires. Fighting wildfires is very challenging and costs about a billion dollars annually in the US alone (NIFC, 2006), not to mention the huge costs associated with the destruction of homes and vegetation. The problem of firefighting resource (e.g. dozers, fire engines, airtankers, crews) management for wildfire

initial attack (Donovan and Rideout, 2003, e.g.) leads to SSLPR (Lee, 2006). In this particular SSLPR decisions regarding firefighting resource deployment to bases ('server' locations) are made in the first stage while decisions regarding the dispatch of resources to wildfires ('clients') are made in the second stage. Uncertainty stems from stochastic wildfire occurrence and dynamic fire behavior, which depend on landscape, fuels and weather conditions. The *random recourse* property in these models is due to the fact that recourse decisions regarding firefighting resource dispatch from bases to wildfires in the second stage, must adapt to scenarios (fire location and fire behavior). Thus instances of these models are generally difficult to solve and require decomposition methods for mixed 0-1 SIP with random recourse.

Closely related to SSLPR is the SSLP with *fixed recourse* studied by Ntaimo and Sen (2005). The SSLP was motivated by the implementation of networks consisting of high speed optical fiber cables, and high speed optical switches which are very capital intensive. Telecommunications service providers are usually unsure of customer demand and under these circumstances network design models lead to SSLPs. In the SSLP, future client *demand* levels are assumed to be known and only client *availability* is unknown and is modeled as a Bernoulli random variable. However, in reality clients may not know there future demand with certainty. So this leads to SSLPR, which is amenable to the solution method introduced in this paper. Other applications of SSLPs include the facility location problem for immobile servers with continuous stochastic demands (Wang et al., 2003), and server location under uncertain demand for the deployment of mobile switching centers in telecommunication networks (Riis et al., 2004). Wang et al. (2003) present several models and provide heuristics for their solutions, while (Riis et al., 2004) uses the stochastic programming approach and apply a dual decomposition procedure to solve scenario subproblems by means of branch-and-cut.

3 Set Convexification for SIP with Random Recourse

To set the ground for the ideas developed in this paper, first some key results from disjunctive programming are restated. Consider disjunctive sets of the form

$$\mathcal{S} = \cup_{h \in H} \mathcal{S}_h \tag{4}$$

where H is a finite index set, and the sets S_h are polyhedral sets represented as

$$\mathcal{S}_h = \{y \mid G_h y \geq r_h, y \geq 0\}. \tag{5}$$

Then a convex relaxation of a non-convex set can be stated through a collection of valid inequalities defined as follows:

DEFINITION 3.1. *An inequality $\pi^\top y \geq \pi_0$ is said to be a valid inequality for the set $\mathcal{S}$ if $\mathcal{S} \subseteq \{y \mid \pi^\top y \geq \pi_0\}$.*

A crucial theoretical result in disjunctive programming is the *disjunctive cut principle*. By letting G_{hj} denote the j^{th} column vector of the matrix G_h, the disjunctive cut principle can be stated as follows:

THEOREM 3.2. *Let $\mathcal{S}$ and S_h be defined as in (4) and (5), respectively. If $\lambda_h \geq 0$ for all $h \in H$, then*

$$\sum_j \{\underset{h\in H}{Max}\ \lambda_h^\top G_{hj}\} y_j \geq \underset{h\in H}{Min}\ \lambda_h^\top r_h \tag{6}$$

is a valid inequality for $\mathcal{S}$. Conversely, suppose that $\pi^\top y \geq \pi_0$ is a valid inequality, and $H^* = \{h \in H \mid \mathcal{S}_h \neq \emptyset\}$. There exists nonnegative multipliers $\{\lambda_h\}_{h\in H^*}$ such that

$$\pi_j \geq \underset{h\in H^*}{Max}\ \lambda_h^\top G_{hj},\ \text{and}\ \pi_0 \leq \underset{h\in H^*}{Min}\ \lambda_h^\top r_h. \tag{7}$$

The forward part of this theorem is due to Balas (1975) while the reverse part is due to Blair and Jeroslow (1978). A variety of relaxations of a mixed-integer program can be obtained by using the above characterization of valid inequalities for the disjunctive set $\mathcal{S}$.

Let us now begin with an approximation of the convex hull of feasible integer points for problem (2). Note that the constraints in (2) vary with the first-stage decision x and the scenario ω. Let H be a finite index set and let the sets $\{\mathcal{S}_h(\omega, x)\}_{h\in H}$ be polyhedral sets represented as

$$\mathcal{S}_h(\omega, x) = \{y(\omega) \in \Re_+^{n_2} \mid W_h(\omega)y(\omega) \geq r_h(\omega, x)\},$$

where $r_h(\omega)$ includes $r(w) - T(w)x$. The disjunctive representation of the set depends on $(\omega, x) \in X \times \Omega$, and can be expressed as

$$\mathcal{S}(\omega, x) = \cup_{h\in H}\mathcal{S}_h(\omega, x). \tag{8}$$

A convex relaxation of the non-convex set (8) can be represented by a collection of valid inequalities of the form $\pi^\top(\omega)y(\omega) \geq \pi_0(\omega, x)$. Using Theorem 3.2, we obtain the following result for SIP1.

THEOREM 3.3. *Consider SIP1 as stated in (1). Let $Y(\omega, x) = \{y(\omega) \mid W(\omega)y(\omega) \geq r(\omega) - T(\omega)x,\ y(\omega) \in \{0,1\}^{n_z} \times \Re_+^{n_2 - n_z}\}$ for $(\omega, x) \in X \times \Omega$. Suppose that $\{C_h(\omega), d_h(\omega)\}_{h\in H}$, is a finite collection of appropriately dimensioned matrices and vectors such that for all $(\omega, x) \in X \times \Omega, Y(\omega, x) \subseteq \cup_{h\in H}\{y(\omega) \in \Re_+^{n_2} \mid C_h(\omega)y(\omega) \geq d_h(\omega)\}$. Let $\mathcal{S}_h(\omega, x) = \{y(\omega) \in \Re_+^{n_2} \mid W(\omega)y(\omega) \geq r(\omega) - T(\omega)x, C_h(\omega)y(\omega) \geq d_h(\omega)\} \neq \emptyset$, and let $\mathcal{S}(\omega, x) = \cup_{h\in H}\mathcal{S}_h(\omega, x)$. Then there exists $\pi(\omega) \in \Re^{n_2}$ and $\pi_0 : X \times \Omega \to \Re$ such that for all $(\omega, x) \in X \times \Omega$, $\pi(\omega)^\top y(\omega) \geq \pi_0(\omega, x)$ is a valid inequality for $\mathcal{S}(\omega, x)$.*

Proof. Let $G_{hj}(\omega)$ denote the vector obtained by concatenating $W_j(\omega)$ with $C_{hj}(\omega)$, and let $r_h^\top(\omega, x)$ denote the vector obtained by concatenating $[r(\omega) - T(\omega)x]^\top$ with $d_h(\omega)^\top$. Since $\mathcal{S}_h(\omega, x)$ is non-empty for all $h \in H$, Theorem 3.2 ensures the existence of nonnegative vectors $\{\lambda_h(\omega)\}_{h\in H}$ such that

$$\pi_j(\omega) \geq \underset{h\in H}{\text{Max}}\ \lambda_h(\omega)^\top G_{hj}(\omega);\ \pi_0(\omega, x) \leq \underset{h\in H}{\text{Min}}\ \lambda_h(\omega)^\top r_h(\omega, x).$$

Since $y(\omega) \geq 0$, for all $(\omega, x) \in X \times \Omega$ we have

$$\pi(\omega)^\top y(\omega) \geq \sum_j \underset{h \in H}{\text{Max}}\, \lambda_h(\omega)^\top G_{hj}(\omega) y_j(\omega)$$

and thus

$$\begin{aligned} \pi(\omega)^\top y(\omega) &\geq \lambda_h(\omega)^\top G_h(\omega) y(\omega) \\ &\geq \lambda_h(\omega)^\top r_h(\omega, x) \qquad \forall h \in H \\ \Rightarrow \pi(\omega)^\top y(\omega) &\geq \underset{h \in H}{\text{Min}}\, \lambda_h(\omega)^\top r_h(\omega, x) \end{aligned}$$

Therefore, it follows that $\pi(\omega)^\top y(\omega) \geq \pi_0(\omega, x)$ □

Since both $\pi(\omega)$ and $\pi_0(\omega, x)$ depend on ω, the advantage of Theorem 3.3 is to allow for the generation of valid inequalities for each (ω, x) differently. This allows for cutting off noninteger solutions for all scenarios simultaneously. However, since Ω can be large in general, generating these valid inequalities for all $\omega \in \Omega$ can be a daunting task in terms of both computation and computer memory for storing the cuts. Thus the use of parallel/distributed computing platforms as well as a further exploitation of the problem structure can be considered. Due to the increase in computational power and the emerging distributed middleware platforms (Linderoth and Wright, 2003, e.g.), Theorem 3.3 can be used with potential savings in computation time.

REMARK 3.4. It should be pointed out that Theorem 3.3 is a more general form of the common-cut-coefficients (C3) theorem (Sen and Higle, 2005) for SIP with fixed recourse. Unlike here where we generate valid inequalities in the $y(\omega)$-space, Carøe and Tind (1997) and Carøe (1998) derive a similar theorem but for generating cuts in the $(x, y(\omega))$-space based on the extensive formulation (3). Thus the method developed in this paper is different from that of Carøe and Tind (1997) and Carøe (1998).

COROLLARY 3.5. *Let $H, \{\mathcal{S}_h(\omega, x)\}_{h \in H}$ and $\mathcal{S}(\omega, x)$ be defined as in Theorem 3.3. Suppose $\pi(\omega)^\top y(\omega) \geq \pi_0(\omega, x)$ is a valid inequality for $\mathcal{S}(\omega, x)$. Then for $h \in H$ there exists vectors $(\bar{\nu}_h(\omega), \bar{\gamma}_h(\omega)) \in \Re^{n_1+1}$ such that*

$$\pi_0(\omega, x) \leq \underset{h \in H}{\textit{Min}}\, \{\bar{\nu}_h(\omega) - \bar{\gamma}_h(\omega)^\top x\}.$$

Proof. Since $\pi(\omega)^\top y(\omega) \geq \pi_0(\omega, x)$ is a valid inequality for $\mathcal{S}(\omega, x)$, Theorem 3.2 ensures that there exists $\lambda_h(\omega) \geq 0, h \in H$ such that

$$\pi_0(\omega, x) \leq \underset{h \in H}{\text{Min}}\ \lambda_h(\omega)^\top r_h(\omega, x),$$

where $r_h(\omega, x)$ is obtained by concatenating $[r(\omega) - T(\omega)x]$ with $d_h(\omega)$. Hence

$$\lambda_h(\omega)^\top r_h(\omega, x) = \bar{\nu}_h(\omega) - \bar{\gamma}_h(\omega)^\top x,$$

where

$$\bar{\nu}_h(\omega) = \lambda_h(\omega)^\top [r(\omega); d_h(\omega)] \text{ and } \bar{\gamma}_h(\omega) = \lambda_h(\omega)^\top [T(\omega); 0]$$

and the result follows. □

It should be observed that the function $\pi_0(\omega, .)$ is a piecewise linear concave function on X. Hence, lower bound approximations from the subproblems would be non-convex in general. This requires the convexification of $\pi_0(\omega, x)$ for each $\omega \in \Omega$ (Sen and Higle, 2005) using a strategy from reverse convex programming in which disjunctive programming is used to provide facets of the convex hull of reverse convex sets (Sen and Sherali, 1987).

4 Cutting Plane Generation Setting

Let us now consider the cutting plane generation setting for the scenario subproblems in a stage-wise (temporal) decomposition of SIP1. Let x^k denote the first-stage solution at some iteration k of the algorithm. For $k = 1$ the problem is initialized with $W^1(\omega) = W(\omega)$, $r^1(\omega) = r(\omega)$, and $T^1(\omega) = T(\omega)$. Thus the subproblem LP relaxations are of the form

$$\begin{aligned} f_c^1(\omega, x) = \ & \text{Min } q(\omega)^\top y(\omega) \\ & \text{s.t. } W(\omega)y(\omega) \geq r(\omega) - T(\omega)x \\ & \qquad y(\omega) \in \Re_+^{n_2}. \end{aligned}$$

By letting $\rho_c^k(\omega, x^k) = r^k(\omega) - T^k(\omega)x^k$ in iteration k, the scenario subproblem LP relaxations take the following form:

$$\begin{aligned} f_c^k(\omega, x^k) = \ & \text{Min } q(\omega)^\top y(\omega) \\ & \text{s.t. } W^k(\omega)y(\omega) \geq \rho_c^k(\omega, x^k) \\ & \qquad y(\omega) \in \Re_+^{n_2}. \end{aligned} \tag{9}$$

Then the cutting planes of the $\pi^k(\omega)^\top y(\omega) \geq \pi_c(\omega, x^k)$ are added to the subproblem at iteration k if the solution $y^k(\omega)$ yields a noninteger solution. Otherwise, no cut is added to the subproblem. The sequential addition of the cutting planes refines the approximation of the convex hull of integer solutions. Therefore, for each $\omega \in \Omega$ the vector $\pi^k(\omega)$ is appended to the matrix $W^{k-1}(\omega)$, and the cut coefficients identified by $(\nu_h(\omega), \gamma_h(\omega))$ are appended to $\big(r^{k-1}(\omega), T^{k-1}(\omega)\big)$.

Let $y^k(\omega) \in \text{argmin}\{q(\omega)^\top y \mid W^k(\omega)y(\omega) \geq \rho_c^k(\omega, x^k),\ y(\omega) \in \Re_+^{n_2}\}$ for $\omega \in \Omega$. Also, let $\bar{y}_j^k(\omega),\ j = 1, ..., n_z$, denote the optimal values assigned to integer variables in y^k. If $\bar{y}_j^k(\omega),\ j = 1, ..., n_z$, are integer for all ω, then no update is necessary, and $W^{k+1}(\omega) = W^k(\omega), r^{k+1}(\omega) = r^k(\omega)$, and $T^{k+1}(\omega) = T^k(\omega)$. Otherwise, the subproblem LP relaxations yield non-integer optimal solutions. Let $j(k, \omega)$ denote an index j, for which $\bar{y}_j^k(\omega)$ is non-integer for some $\omega \in \Omega$. To eliminate this non-integer solution, a disjunction of the following form is used:

$$\mathcal{S}_k(\omega, x^k) = \mathcal{S}_{0,j(k,\omega)}(x^k, \omega) \cup \mathcal{S}_{1,j(k,\omega)}(\omega, x^k)$$

where

$$\mathcal{S}_{0,j(k,\omega)}(\omega, x^k) = \big\{y \in \Re_+^{n_2} \mid W^k(\omega)y(\omega) \geq r^k(\omega) - T^k(\omega)x^k \tag{10a}$$

$$-y_{j(k,\omega)}(\omega) \geq -\lfloor \bar{y}_{j(k,\omega)} \rfloor \big\} \tag{10b}$$

and

$$\mathcal{S}_{1,j(k,\omega)}(\omega, x^k) = \{y \in \Re^{n_2}_+ \mid W^k(\omega)y(\omega) \geq r^k(\omega) - T^k(\omega)x^k \tag{11a}$$

$$y_{j(k,\omega)}(\omega) \geq \lceil \bar{y}_{j(k,\omega)} \rceil\}. \tag{11b}$$

The variable $y_{j(k,\omega)}(\omega)$ is referred to as the *disjunction variable* for iteration k and scenario ω. Note that since the integer restrictions are binary, the righthand side of (10b) is 0, and that of (11b) is 1.

Let $\lambda_{0,1}(\omega)$ denote the vector of multipliers associated with (10a), and $\lambda_{0,2}(\omega)$ denote the scalar multiplier associated with (10b). Let $\lambda_{1,1}(\omega)$ and $\lambda_{1,2}(\omega)$ be similarly defined for (11a) and (11b), respectively. Also, define

$$I^k_j(\omega) = \begin{cases} 1, & \text{if } j = j(k,\omega) \\ 0, & \text{otherwise.} \end{cases}$$

Then the following LP can be used to generate the cut coefficients $\pi^k(\omega)$ for $\omega \in \Omega$ in iteration k:

$$\begin{aligned}
&\text{Max } \pi_0(\omega) - \bar{y}^k(\omega)^\top \pi(\omega) \\
&\text{s.t. } \pi_j(\omega) - \lambda_{0,1}(\omega)^\top W^k_j(\omega) + I^k_j(\omega)\lambda_{0,2}(\omega) \geq 0, \quad \forall j \\
&\quad \pi_j(\omega) - \lambda_{1,1}(\omega)^\top W^k_j(\omega) - I^k_j(\omega)\lambda_{1,2}(\omega) \geq 0, \quad \forall j \\
&\quad - \pi_0(\omega) + \lambda_{0,1}(\omega)^\top \rho^k_c(\omega, x^k) - \lambda_{0,2}(\omega)\lfloor \bar{y}_{j(k)} \rfloor \geq 0 \\
&\quad - \pi_0(\omega) + \lambda^\top_{1,1}(\omega)\rho^k_c(\omega, x^k) + \lambda_{1,2}(\omega)\lceil \bar{y}_{j(k)} \rceil \geq 0 \\
&\quad - 1 \leq \pi_j(\omega) \leq 1, \quad \forall j \\
&\quad - 1 \leq \pi_0(\omega) \leq 1 \\
&\quad \lambda_{0,1}(\omega),\ \lambda_{0,2}(\omega),\ \lambda_{1,1}(\omega),\ \lambda_{1,2}(\omega) \geq 0.
\end{aligned} \tag{12}$$

The validity of the cut coefficients $\pi(\omega)$ generated by problem (12) follows from Theorem 3.2. Theorem 3.3 allows us to independently generate cut coefficients for each scenario $\omega \in \Omega$. On the contrary, under the *fixed recourse* property the C^3 Theorem (Sen and Higle, 2005) allows for the formulation of a simple recourse stochastic version of problem 12, which is solved only once per iteration to generate the cut coefficients for all $\omega \in \Omega$.

Since the disjunction used for cut formation has $H = \{0, 1\}$, the epigraph of $\pi_0(\omega, x)$ is a union of two polyhedral sets. Therefore, the convex hull approximation $\pi_c(\omega, .)$ of $\pi_0(\omega, .)$ can be represented as

$$\pi_c(\omega, x) = \underset{i \in I}{\text{Max}} \ \{\nu_i(\omega) - \gamma_i(\omega)^\top x\}, \ (\omega, x) \in X \times \Omega,$$

where $\nu_i(\omega)$ and $\gamma_i(\omega)$ are derived from the set I, which denotes the set of extreme points

of the convex hull of the epigraph of $\pi_0(\omega,.)$, denoted $\mathcal{P}_X^k(\omega)$ and defined as follows:

$$\begin{aligned}\mathcal{P}_X^k(\omega) = \big\{ & \sigma_0(\omega) \in \Re,\ \sigma(\omega) \in \Re^{n_1},\ \delta(\omega) \in \Re \mid \\ & \forall h \in H,\ \exists\ \tau_h \in \Re^{m_1}, \tau_{0h} \in \Re \\ & \sigma_0(\omega) \geq \tau_{0h} \quad \forall h \in H \\ & \sum_{h \in H} \tau_{0h} = 1 \\ & \sigma_j(\omega) \geq \tau_h^\top A_j + \tau_{0h} \bar{\gamma}_{hj}^k(\omega) \quad \forall h \in H, j = 1, ..., n_1 \\ & \delta(\omega) \leq \tau_h^\top b + \tau_{0h} \bar{\nu}_h^k(\omega) \quad \forall h \in H \\ & \tau_h \geq 0, \tau_{0h} \geq 0, \quad \forall h \in H \big\}.\end{aligned}$$

To obtain $\pi_c^k(\omega, x)$, the following parameters are derived from an optimal solution of (12) to update $\mathcal{P}_X^k(\omega)$:

$$\begin{aligned}\bar{\nu}_0^k(\omega) &= \lambda_{0,1}^\top r^k(\omega) - \lambda_{0,2} \lfloor \bar{y}_{j(k,\omega)}(\omega) \rfloor, \\ \bar{\nu}_1^k(\omega) &= \lambda_{1,1}^\top r^k(\omega) + \lambda_{1,2} \lceil \bar{y}_{j(k,\omega)}(\omega) \rceil,\end{aligned}$$

and

$$[\bar{\gamma}_h^k(\omega)]^\top = \lambda_{h,1}^\top T^k(\omega),\ \forall h \in H.$$

Then to approximate $\pi_0^k(\omega, x)$ the following LP is solved:

$$\begin{aligned}\text{Max} \quad & \delta(\omega) - \sigma_0(\omega) - (x^k)^\top \sigma(\omega) \\ \text{s.t.} \quad & (\sigma_0(\omega), \sigma(\omega), \delta(\omega)) \in \mathcal{P}_X^k(\omega).\end{aligned} \tag{13}$$

Since $\pi_0(\omega, x)$ is scenario dependent, the convexification is done for each $\omega \in \Omega$. With an optimal solution to (13), $(\sigma_0^k(\omega), \sigma^k(\omega), \delta^k(\omega))$, we obtain $\nu^k(\omega) = \frac{\delta^k(\omega)}{\sigma_0^k(\omega)}$ and $\gamma^k(\omega) = \frac{\sigma^k(\omega)}{\sigma_0^k(\omega)}$. These coefficients define $\pi_c^k(\omega, x) = \nu^k(\omega) - (\gamma^k(\omega))^\top x$ and are used to update the vector $r^{k+1}(\omega) = [r^k(\omega), \nu^k(\omega)]$, and matrix $T^{k+1}(\omega) = [(T^k(\omega))^\top; \gamma^k(\omega)]^\top$. Similarly, the solution $\pi^k(\omega)$ to (12) for each $\omega \in \Omega$ is used to update the constraint matrix, $W^{k+1}(\omega) = [(W^k(\omega))^\top; \pi^k(\omega)]^\top$.

5 SIP with Deterministic Technology Matrix and Righthand Side Vector

Based on the application at hand, SIP1 may have some special structure that can be exploited. For our interest, we consider SIP1 with deterministic technology matrix $T(\omega) = T$ and righthand side vector $r(\omega) = r$, for all $\omega \in \Omega$, and derive valid inequalities that take advantage of this property. Under this setting, problem (1) becomes:

$$\text{SIP2:} \quad \underset{x \in X \cap \mathcal{B}^{n_1}}{\text{Min}}\ c^\top x + E[f(\tilde{\omega}, x)], \tag{14}$$

where for any $\omega \in \Omega$,

$$\begin{aligned} f(\omega, x) = \text{Min } & q(\omega)^\top y, \\ \text{s.t. } & W(\omega)y(\omega) \geq r - Tx \\ & y(\omega) \in \mathcal{B}^{n_z} \times \Re_+^{n_2 - n_z}. \end{aligned} \tag{15}$$

The subproblem LP relaxations take the form

$$\begin{aligned} f_c(\omega, x) = \text{Min } & q(\omega)^\top y(\omega) \\ \text{s.t. } & W(\omega)y(\omega) \geq \rho_c(\omega, x) \\ & y(\omega) \in \Re_+^{n_2}, \end{aligned} \tag{16}$$

where $\rho_c(\omega, x) = r - Tx$. Note that $\rho_c(\omega, x)$ is now independent of ω for all $\omega \in \Omega$.

As developed in the previous section, we would like to generate and add cutting planes of the form $\pi(\omega)^\top y \geq \pi_c(\omega, x)$ if the subproblem LP solution $\bar{y}(\omega)$ is noninteger. Let $j(\omega)$ denote an index for which $\bar{y}_{j(\omega)}(\omega)$ is noninteger for some $\omega \in \Omega$, and define

$$I_j = \begin{cases} 1, & \text{if } j = j(\omega) \\ 0, & \text{otherwise.} \end{cases}$$

To eliminate $\bar{y}_{j(\omega)}(\omega)$, a disjunction of the following form is used:

$$\mathcal{S}(\omega, x) = \mathcal{S}_{0,j(\omega)}(x, \omega) \cup \mathcal{S}_{1,j(\omega)}(\omega, x)$$

where

$$\mathcal{S}_{0,j(\omega)}(\omega, x) = \big\{ y(\omega) \in \Re_+^{n_2} \mid W(\omega)y(\omega) \geq r - Tx \tag{17a}$$

$$-y_{j(\omega)}(\omega) \geq -\lfloor \bar{y}_{j(\omega)}(\omega) \rfloor \big\} \tag{17b}$$

and

$$\mathcal{S}_{1,j(\omega)}(\omega, x)\{ y(\omega) \in \Re_+^{n_2} \mid W(\omega)y(\omega) \geq r - Tx \tag{18a}$$

$$y_{j(\omega)}(\omega) \geq \lceil \bar{y}_{j(\omega)}(\omega) \rceil \}. \tag{18b}$$

Let $\varpi \in \Omega$ be a scenario on which the disjunction is based. Such a scenario will be referred to as the *disjunction scenario*. Then using (17) and (18) we get the following LP for generating coefficients $\pi(\varpi)$ for ϖ:

$$\begin{aligned} \text{Max } & \pi_0(\varpi) - \bar{y}(\varpi)^\top \pi(\varpi) \\ \text{s.t. } & \pi_j(\varpi) - \lambda_{0,1}(\varpi)^\top W_j(\varpi) + I_j \lambda_{0,2}(\varpi) \geq 0, \quad \forall j \\ & \pi_j(\varpi) - \lambda_{1,1}(\varpi)^\top W_j(\varpi) - I_j \lambda_{1,2}(\varpi) \geq 0, \quad \forall j \\ & -\pi_0(\varpi) + \lambda_{0,1}(\varpi)^\top \rho_c(\varpi, x) - \lambda_{0,2}(\varpi) \lfloor \bar{y}_j(\varpi) \rfloor \geq 0 \\ & -\pi_0(\varpi) + \lambda_{1,1}(\varpi)^\top \rho_c(\varpi, x) + \lambda_{1,2}(\varpi) \lceil \bar{y}_j(\varpi) \rceil \geq 0 \\ & -1 \leq \pi_j(\varpi) \leq 1, \quad \forall j \\ & -1 \leq \pi_0(\varpi) \leq 1 \\ & \lambda_{0,1}(\varpi),\ \lambda_{0,2}(\varpi),\ \lambda_{1,1}(\varpi),\ \lambda_{1,2}(\varpi) \geq 0. \end{aligned} \tag{19}$$

Using the solution to (19) we obtain $\pi_0(\varpi, x)$ via Corollary 3.5 to get the valid inequality $\pi(\varpi)^\top y(\varpi) \geq \pi_0(\varpi, x)$ for ϖ.

THEOREM 5.1. *Consider problem SIP2 stated in (14) with random recourse matrix $W(\omega)$ but deterministic $T(\omega) = T$ and $r(\omega) = r$, for all $\omega \in \Omega$. Suppose that $\pi(\varpi)^\top y(\varpi) \geq \pi_0(\varpi, x)$ is a valid inequality for $S(\varpi, x) \neq \emptyset$ for $\varpi \in \Omega$ generated using $(\pi(\varpi),\ \pi_0(\varpi))$ from an optimal solution to problem (19). If for $\omega \in \Omega,\ \omega \neq \varpi$, the problem*

$$
\begin{aligned}
&\textit{Min } \bar{y}(\omega)^\top \pi(\omega) \\
&\textit{s.t. } \pi_j(\omega) - \lambda_{0,1}(\omega)^\top W_j(\omega) + I_j \lambda_{0,2}(\omega) \geq 0, \quad \forall j \\
&\qquad \pi_j(\omega) - \lambda_{1,1}(\omega)^\top W_j(\omega) - I_j \lambda_{1,2}(\omega) \geq 0, \quad \forall j \\
&\qquad \lambda_{0,1}(\omega)^\top \rho_c(\omega, x) - \lambda_{0,2}(\omega) \lfloor \bar{y}_j(\omega) \rfloor \geq \pi_0(\varpi) \\
&\qquad \lambda_{1,1}(\omega)^\top \rho_c(\omega, x) + \lambda_{1,2}(\omega) \lceil \bar{y}_j(\omega) \rceil \geq \pi_0(\varpi) \\
&\qquad -1 \leq \pi_j(\omega) \leq 1, \ \forall j \\
&\qquad \lambda_{0,1}(\omega),\ \lambda_{0,2}(\omega),\ \lambda_{1,1}(\omega),\ \lambda_{1,2}(\omega) \geq 0.
\end{aligned}
\tag{20}
$$

is feasible, then $\pi(\omega)^\top y(\omega) \geq \pi_0(\varpi, x)$ is valid for $S(\omega, x) \neq \emptyset$. If $\pi_0(\varpi, x) - \pi(\omega)^\top y(\omega) > 0$, then $\pi(\omega)^\top y(\omega) \geq \pi_0(\varpi, x)$ cuts off the point $\bar{y}(\omega)$.

Proof. Since $\rho_c(\omega, x) = \rho_c(\varpi, x)$ for $\omega \in \Omega$, $\pi_0(\omega)$ should satisfy

$$
\begin{gathered}
-\pi_0(\omega) + \lambda_{0,1}(\omega)^\top \rho_c(\omega, x) - \lambda_{0,2}(\omega) \lfloor \bar{y}_j \rfloor \geq 0 \\
-\pi_0(\omega) + \lambda_{1,1}(\omega)^\top \rho_c(\omega, x) + \lambda_{1,2}(\omega) \lceil \bar{y}_j \rceil \geq 0 \\
-1 \leq \pi_0(\omega) \leq 1 \\
\lambda_{0,1}(\omega),\ \lambda_{0,2}(\omega),\ \lambda_{1,1}(\omega),\ \lambda_{1,2}(\omega) \geq 0
\end{gathered}
$$

for all $\omega \in \Omega$. Therefore, if for $\omega \in \Omega, \omega \neq \varpi$ we fix $\pi_0(\omega)$ in (19) with the corresponding value obtained for ϖ, and can find coefficients $\pi(\omega)$ and multipliers $(\lambda_{0,1}(\omega),\ \lambda_{0,2}(\omega),\ \lambda_{1,1}(\omega),\ \lambda_{1,2}(\omega))$ satisfying

$$
\begin{gathered}
\pi_j(\omega) - \lambda_{0,1}(\omega)^\top W_j(\omega) + I_j(\omega) \lambda_{0,2}(\omega) \geq 0, \quad \forall j \\
\pi_j(\omega) - \lambda_{1,1}(\omega)^\top W_j(\omega) - I_j(\omega) \lambda_{1,2}(\omega) \geq 0, \quad \forall j \\
-1 \leq \pi_j(\omega) \leq 1, \quad \forall j
\end{gathered}
$$

then the result follows. □

Theorem 5.1 ensures that with both a translation and rotation, valid inequalities derived for one pair (ϖ, x) may be used to derive valid inequalities for another pair (ω, x). Note that since $\pi_0(\varpi, x)$ is fixed for all $\omega \in \Omega,\ \omega \neq \varpi$ for which $\pi(\omega)^\top y(\omega) \geq \pi_0(\varpi, x)$ is a valid inequality, the convexification $\pi_c(\varpi, x)$ of $\pi_0(\varpi, x)$ is obtained only once and used for the other scenarios. This sharply differs from the convexification under SIP1, where the convexification is done for each scenario $\omega \in \Omega$.

Observe that Theorem 5.1 allows for each scenario to determine its own multipliers $(\lambda_{0,1}(\omega),\ \lambda_{0,2}(\omega),\ \lambda_{1,1}(\omega),\ \lambda_{1,2}(\omega))$ while fixing $\pi_0(\omega)$ to $\pi_0(\varpi)$. By imposing further restrictions on the multipliers, we obtain the following result.

THEOREM 5.2. *Consider problem SIP2 as stated in (14) with random recourse matrix* $W(\omega)$ *but* $T(\omega) = T$, *and* $r(\omega) = r$ *for all* $\omega \in \Omega$. *Suppose that* $\pi(\varpi)^\top y(\varpi) \geq \pi_0(\varpi, x)$ *is a valid inequality for* $S(\varpi, x) \neq \emptyset$ *for* $\varpi \in \Omega$ *generated using* ($\pi(\varpi)$, $\pi_0(\varpi)$, $\lambda_{0,1}(\varpi)$, $\lambda_{0,2}(\varpi)$, $\lambda_{1,1}(\varpi)$, $\lambda_{1,2}(\varpi)$), *the solution to problem (12). If for* $\omega \in \Omega$, $\omega \neq \varpi$, *the problem*

$$\begin{aligned} &\textit{Min } \bar{y}(\omega)^\top \pi(\omega) \\ &\textit{s.t.} \quad \pi_j(\omega) \geq \lambda_{0,1}(\varpi)^\top W_j(\omega) - I_j \lambda_{0,2}(\varpi), \quad \forall j \\ &\qquad \pi_j(\omega) \geq \lambda_{1,1}(\varpi)^\top W_j(\omega) + I_j \lambda_{1,2}(\varpi), \quad \forall j \\ &\qquad -1 \leq \pi_j(\omega) \leq 1, \quad \forall j \end{aligned} \tag{23}$$

is feasible, then $\pi(\omega)^\top y(\omega) \geq \pi_0(\varpi, x)$ *is valid for* $S(\omega, x) \neq \emptyset$. *If* $\pi_0(\varpi, x) - \pi(\omega)^\top y(\omega) > 0$, *then* $\pi(\omega)^\top y(\omega') \geq \pi_0(\varpi, x)$ *cuts off the point* $\bar{y}(\omega)$.

Proof. Since $\rho_c(\omega, x) = \rho_c(\varpi, x)$ for all $\omega \in \Omega$, $\pi_0(\omega)$ should satisfy

$$\begin{gathered} -\pi_0(\omega) + \lambda_{0,1}(\omega)^\top \rho_c(\omega, x) - \lambda_{0,2}(\omega) \lfloor \bar{y}_j \rfloor \geq 0 \\ -\pi_0(\varpi) + \lambda_{1,1}(\omega)^\top \rho_c(\omega, x) + \lambda_{1,2}(\omega) \lceil \bar{y}_j \rceil \geq 0 \\ -1 \leq \pi_0(\omega) \leq 1 \end{gathered}$$

for all $\omega \in \Omega$. Therefore, if for $\omega \in \Omega, \omega \neq \varpi$ we fix $(\pi_0(\omega), \lambda_{0,1}(\omega), \lambda_{0,2}(\omega), \lambda_{1,1}(\omega), \lambda_{1,2}(\omega))$ in (12) with the corresponding values obtained for ϖ, and can find coefficients $\pi(\omega)$ satisfying

$$\begin{gathered} \pi_j(\omega) - \lambda_{0,1}(\varpi)^\top W_j(\omega) + I_j \lambda_{0,2}(\varpi) \geq 0, \quad \forall j \\ \pi_j(\omega) - \lambda_{1,1}(\varpi)^\top W_j(\omega) - I_j \lambda_{1,2}(\varpi) \geq 0, \quad \forall j \\ -1 \leq \pi_j(\omega) \leq 1, \quad \forall j \end{gathered}$$

then the result follows. □

Even though Theorems 5.1 and 5.2 do not provide a guarantee on the quality of the cuts, they allow for generating cuts using smaller cut generation LPs than in the case of SIP1. Furthermore, since the convexification $\pi_c(\varpi, x)$ of $\pi_0(\varpi, x)$ has to be carried out only once for the scenarios sharing $\pi_0(\varpi, x)$, this reduces the effort and computer memory for computing and storing the data associated with $\pi_c(\varpi, x)$. In the case of SIP1, however, the convexification has to be carried out separately for each $\omega \in \Omega$.

In the algorithmic context for Theorems 5.1 and 5.2, in iteration $k = 1$ the problem is initialized with $W^1(\omega) = W(\omega)$, $r^1 = r$, and $T^1 = T$, and $\rho_c^1(\omega, x^1) = r - Tx^1$, for all $\omega \in \Omega$. By letting $\rho_c^k(\omega, x^k) = r^k(\omega) - T^k(\omega)x^k$ in iteration $k > 1$, we allow for T^k and r^k to depend on ω. However, as a consequence of Theorems 5.1 and 5.2, when generating a cut for some $\omega \in \Omega$ based on another scenario $\varpi \in \Omega$, $\varpi \neq \omega$, it is necessary that $\rho_c^k(\omega, x^k) = \rho_c^k(\varpi, x^k)$. Also, in Theorems 5.1 and 5.2 the *disjunction variable* index in iteration k, $j(k)$, is determined by the *disjunction scenario*. Thus we lose the flexibility of having each scenario determine its own disjunction variable as in Theorem 3.3. Next we give a disjunctive decomposition algorithm for SIP1 and SIP2.

6 A Disjunctive Decomposition Algorithm

We first consider a disjunctive decomposition algorithm for SIP1 based on Theorem 3.3 and then provide a modification for SIP2 based on Theorems 5.1 and 5.2. At the k-th iteration of the algorithm we consider the following LP relaxation of the original problem:

$$\underset{x\in X\cap\mathcal{B}^{n_1}}{\text{Min}}\ c^\top x + E[f_c^k(\tilde{\omega}, x)], \tag{26}$$

where for $\omega \in \Omega$,

$$\begin{aligned} f_c^k(\omega, x) = \text{Min}\ & q(\omega)^\top y(\omega), \\ \text{s.t. } & W^k(\omega)y(\omega) \geq \rho_c^k(\omega, x) \\ & y(\omega) \in \Re_+^{n_2}. \end{aligned} \tag{27}$$

Adopting a Benders' decomposition (Benders, 1962) setting, let the master program be defined as follows:

$$\underset{x\in X\cap\mathcal{B}^{n_1}}{\text{Min}}\ c^\top x + F^k(x) \tag{28}$$

where $F^k(\cdot)$ is a piecewise linear approximation of $E[f(\tilde{\omega}, x)]$ in the k-th iteration. Note that $F^k(x)$ is convex while $E[f(\tilde{\omega}, x)]$ is not. Then an algorithm for SIP with random recourse can be stated as follows:

D^2-R **Algorithm:**

Step 0. Initialization.
Set $k \leftarrow 1$, $V_1 \leftarrow \infty$, $v_1 \leftarrow -\infty$, $\epsilon > 0$, $W^1(\omega) \leftarrow W(\omega)$, $T^1(\omega) \leftarrow T(\omega)$, and $r^1(\omega) = r(\omega)$, for all $\omega \in \Omega$.

Step 1. Solve LP Relaxation.
Solve LP relaxation (26) (e.g. use the L-shaped method (Van Slyke and Wets, 1969)) to get $x^1 \in X \cap \mathcal{B}$. Set v_1 to the optimal value of (26). If $\{y^k(\omega)\}_{\omega\in\Omega}$ satisfy the integer restrictions, $V_{k+1} \leftarrow \text{Min}\{c^\top x^k + E[f(x^k, \tilde{\omega})], V_k\}$, set incumbent solution $x_\epsilon^* \leftarrow x^1$ and go to Step 5. Otherwise, go to Step 3.

Step 2. Solve Subproblem LPs.
For each $\omega \in \Omega$, use the matrix $W^k(\omega)$ and righthand side vector $r^k(\omega) - T^k(\omega)x^k$ to solve (27). If $\{y^k(\omega)\}_{\omega\in\Omega}$ satisfy the integer restrictions, $V_{k+1} \leftarrow \text{Min}\{c^\top x^k + E[f(x^k, \tilde{\omega})], V_k\}$. Set incumbent solution $x_\epsilon^* \leftarrow x^k$ if V_{k+1} is updated and go to Step 5.

Step 3. Solve Scenario Cut Generation LPs and Perform Updates.
(a) Choose a disjunction variable $j(k, \omega)$ for each $\omega \in \Omega$ such that $\bar{y}(\omega)$ is noninteger. Formulate and solve (12) to obtain $\pi^k(\omega)$. Define $W^{k+1}(\omega) = [(W^k(\omega))^\top; \pi^k(\omega)]^\top$.
(b) Use the multipliers $(\lambda_{0,1}^k(\omega), \lambda_{0,2}^k(\omega), \lambda_{1,1}^k(\omega), \lambda_{1,2}^k(\omega))$ and the value $\bar{y}_{j(k)}(\omega)$ obtained in (a) to form and solve (13) for each outcome $\omega \in \Omega$. The solution defines $\nu^k(\omega)$ and $\gamma^k(\omega)$ which are used to update $r^{k+1}(\omega) = [r^k(\omega), \nu^k(\omega)]$ and $T^{k+1}(\omega) = [(T^k(\omega))^\top; \gamma^k(\omega)]^\top$.

Step 4. Update and Solve Subproblem LPs.
For each $\omega \in \Omega$ solve (27) using $W^{k+1}(\omega)$ and $r^{k+1}(\omega) - T^{k+1}(\omega)x^k$. If $y^k(\omega)$ satisfies the integer restrictions for all $\omega \in \Omega$, $V_{k+1} \leftarrow \text{Min}\{c^\top x^k + E[f(\omega, x^k)], V_k\}$ and set incumbent solution $x^*_\epsilon \leftarrow x^k$ if V_k is updated. Otherwise, $V_{k+1} \leftarrow V_k$.

Step 5. Update and Solve the Master Problem.
Update the approximation F^k via Benders' decomposition using the dual multipliers from the most recently solved subproblem for each $\omega \in \Omega$. Let $x^{k+1} \in \text{argmin}\{c^\top x + F^{k+1}(x) \mid x \in X\}$, and let v_{k+1} denote the optimal value of the master problem. If $V_{k+1} - v_{k+1} \leq \epsilon$, stop and declare x^*_ϵ ϵ-optimal. Otherwise, $k \leftarrow k+1$ and repeat from Step 2.

Selection of the disjunction variable in Step 3(a) for a given $\omega \in \Omega$ can be done using a heuristic. For example, one can always select the first fractional component of $\bar{y}^k(\omega)$, or the largest fractional component. Also note that we avoid computing the upper bound V_k since it requires the solution of all scenario subproblem MIPs. However, it may be necessary to compute V_k in Step 2 at some iteration k when the first stage solution x^k stabilizes (stops changing). At this point, one can also generate the optimality cut for binary first stage SIP proposed by Laporte and Louveaux (1993) to add to the master program. Now turning to SIP2 (deterministic matrix T and vector r), we can modify Step 0 and Step 3 of the D^2-R algorithm by applying Theorems 5.1 and 5.2 as follows:

Modification of Step 0 and Step 3 for SIP2:

Step 0. Initialization.
Set $k \leftarrow 1$, $V_1 \leftarrow \infty$, $v_1 \leftarrow -\infty$, $\epsilon > 0$, $W^1(\omega) \leftarrow W(\omega)$, $T^1(\omega) \leftarrow T$, and $r^1(\omega) = r$, for all $\omega \in \Omega$.

Step 3. Solve Scenario Cut Generation LPs and Perform Updates.
(a) Choose a disjunction scenario $\varpi \in \Omega$ and disjunction index $j(k)$. Formulate and solve (19) to obtain $\pi^k(\varpi)$ and $\pi^k_0(\varpi)$. Define $W^{k+1}(\varpi) = [(W^k(\varpi))^\top; \pi^k(\varpi)]^\top$.
(b) Use the multipliers $(\lambda^k_{0,1}(\varpi), \lambda^k_{0,2}(\varpi), \lambda^k_{1,1}(\varpi), \lambda^k_{1,2}(\varpi))$ and the value $\bar{y}_{j(k)}(\varpi)$ obtained in Step 3(a) to form and solve (13). The solution defines $\nu^k(\varpi)$ and $\gamma^k(\varpi)$ which are used to update $r^{k+1}(\varpi) = [r^k(\varpi), \nu^k(\varpi)]$ and $T^{k+1}(\varpi) = [(T^k(\varpi))^\top; \gamma^k(\varpi)]^\top$.
(c) For each $\omega \in \Omega \setminus \varpi$ such that $\bar{y}_{j(k)}(\omega)$ is noninteger and $\rho^k_c(\omega, x^k) = \rho^k_c(\varpi, x^k)$, use the solution obtained in Step 3(a) to form and solve (20) (or (23)) to obtain $\pi^k(\omega)$. Define $W^{k+1}(\omega) = [(W^k(\omega))^\top; \pi^k(\omega)]^\top$ and use $\nu^k(\varpi)$ and $\gamma^k(\varpi)$ obtained in Step 3(b) to define $r^{k+1}(\omega) = [r^k(\omega), \nu^k(\varpi)]$ and $T^{k+1}(\omega) = [(T^k(\omega))^\top; \gamma^k(\varpi)]^\top$.

If an initial solution x^1 is known at initialization, Step 1 of the D^2-R algorithms can be skipped and one can simply go to Step 2 of the algorithm and avoid solving the LP relaxation (26). In the modified Step 3 one can generate a cut for a given scenario if $\rho^k_c(\omega, x^k) = \rho^k_c(\varpi, x^k)$ using Theorem 3.3 as in the original step. A heuristic can be employed to determine the disjunction scenario ϖ and the disjunction variable index

$j(k)$ in Step 3. For example, ϖ can be chosen as the scenario with the largest number of fractional components in $\bar{y}$, while $j(k)$ can be chosen as the index with the largest number of fractional components among all the scenarios. A variety of implementation options can be considered to improve the computation time of the algorithm. For example, the implementer can use different procedures for initiating and terminating the generation of cuts. Different normalizations for the cut generation LPs (12), (19),(20) and (23) can also be considered. One can also incorporate rounding and lifting to strengthen the cuts as described in (Balas et al., 1993) and (Balas et al., 1996).

A few comments on the theoretical (finite) convergence of the algorithm are now in order. Balas (1979) has shown that one can recover the closure of the convex hull (clconv) for sets with the facial property. For example, a mixed 0-1 integer program is a facial disjunctive program. In our case we are dealing with the mixed 0-1 sets clconv($\mathcal{S}(\omega, x)$) for all $\omega \in \Omega$. Recovering clconv($\mathcal{S}(\omega, x)$) (if necessary) can be achieved by generating a sequence of convex hulls recursively for each disjunction variable $j = 1, ..., n_z$. Thus finite convergence of the algorithm can be obtained under assumptions (A1-A4) if the algorithm identifies extreme point solutions of problem (12) (or problems (20) and (23)) and further conditions on x^k described in the next paragraph are satisfied.

Note that since Ω is finite, there are finitely many inequalities in (12), (20) and (23). Also, $x^k \in X \cap \mathcal{B}$ are finite, leading to finitely many polyhedra of the form in (12), (20) and (23) from which cut coefficients are drawn. These cut coefficients are by assumption extreme points of the polyhedra. Consequently, after finitely many iterations, all necessary extreme points, and thus all necessary cuts, will have been generated for each disjunction variable. Hence in finitely many iterations all valid inequalities for any scenario subproblem IP can be generated as a result of the facial property of the scenario subproblem. It should also be pointed out that since for any $x^k \in vert(X)$ or $x^k \in \mathcal{B}^{n_1}$, $\pi_c(\omega, x^k) = \pi_0(\omega, x^k)$, after a finite number of iterations $f_c^k(\omega, x^k) = f(\omega, x^k)$ for all $\omega \in \Omega$. From a practical point of view, however, finiteness of the algorithm should be viewed as a desirable property and not a measure of how the algorithm will perform in practice.

7 Application

The proposed method was implemented and a applied to the stochastic server location problem with random recourse (SSLPR). The aim was to study the computational performance of the method relative to a state-of-the-art direct solver applied to the extensive form (EF) of the problem. Next we give a generic model formulation of SSLPR, describe problem instance generation, and then present the computational results.

7.1 Stochastic Server Location Problem with Random Recourse

SSLPR deals with the optimal location of a limited number of 'servers' with limited capacity of some resource under uncertainty regarding potential future 'client' resource *demand* and *availability* after the servers are located. The objective is to minimize the

server location costs in the first stage plus the expected future costs minus the expected revenue from serving the clients in the second stage. Several motivating applications for SSLPR are described in Section 1. Assuming linear objective costs (or revenue), SSLPR can be formulated as follows:

Sets:

$\mathcal{I}$: set of potential server locations.
$\mathcal{J}$: set of potential future clients.
$\mathcal{Z}$: set of server zones.
Ω: set of all possible scenarios.

Decision Variables:

x_i: takes a value of 1 if a server is located at site i, and 0 otherwise.
y_{ij}^{ω}: takes a value of 1 if client j is served by a server at site i under scenario $\omega \in \Omega$, and 0 otherwise.
y_{i0}^{ω}: continuous variable representing unsatisfied client demand by server at site i under scenario $\omega \in \Omega$.

Problem Data:

c_i: cost of locating a server at site i.
τ_i: capacity of Server at site i.
u: upper bound on the total number of servers that can be located.
w_z: minimum number of servers to be located in zone $z \in \mathcal{Z}$.
$\mathcal{I}_z$: subset of server locations that belong to zone z.
g_{ij}: revenue from client j being served by server at site i.
g_{i0}: penalty (overflow cost) for unsatisfied demand due to server at site i.
d_{ij}^{ω}: client j resource demand from server at site i under scenario $\omega \in \Omega$.
$r^j(\omega)$: takes a value of 1 if client j is present under scenario $\omega \in \Omega$, and 0 otherwise.
p_{ω}: probability of occurrence of scenario $\omega \in \Omega$.

Model:

$$\text{Min} \quad \sum_{i \in \mathcal{I}} c_i x_i - \sum_{\omega \in \Omega} p_{\omega} f(x, \omega) \tag{29a}$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{I}} x_i \leq u \tag{29b}$$

$$\sum_{i \in \mathcal{I}_z} x_i \geq w_z, \ \forall z \in \mathcal{Z} \tag{29c}$$

$$x_i \in \{0, 1\}, \ \forall i \in \mathcal{I}, \tag{29d}$$

where for any x satisfying the constraints (29b-29d) and scenario outcome $\omega \in \Omega$,

$$f(x,\omega) = \text{Min} \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}(-q_{ij})y^{\omega}_{ij} + \sum_{i\in\mathcal{I}} g_{i0}y^{\omega}_{i0} \tag{30a}$$

$$\text{s.t.} \sum_{j\in\mathcal{J}} d^{\omega}_{ij}y^{\omega}_{ij} - y^{\omega}_{i0} \leq \tau_i x_i,\ \forall i \in \mathcal{I} \tag{30b}$$

$$\sum_{i\in\mathcal{I}} y^{\omega}_{ij} = r^j(\omega),\ \forall j \in \mathcal{J} \tag{30c}$$

$$y^{\omega}_{ij} \in \{0,1\},\ \forall i \in \mathcal{I}, j \in \mathcal{J} \tag{30d}$$

$$y^{\omega}_{i0} \geq 0,\ \forall i \in \mathcal{I}. \tag{30e}$$

The objective (29a) is to maximize expected profit from providing some limited resource to the clients. Constraint (29b) limits the number of available servers that can be located to u. The zonal requirements are given by constraint (29c) and enforce the requirement that at least a specified number of servers w_z be located in a given zone $z \in \mathcal{Z}$. Constraints (29d) are the binary restrictions on the server location decision variables. In the second stage, constraints (30b) dictate that a server installed at site i can serve only up to its capacity τ_i. The continuous variable y_{i0} accommodate any overflows that cannot be served due to limitations in server capacity. These overflows result in a loss of revenue at a rate of q_{i0} but for cases in which there is sufficient server capacity, the overflows are zero due to the high penalty costs. The requirement that each available client be served by only one server is given by constraints (30c). Binary restrictions on the tactical decision variables are given by constraints (30d), while the nonnegativity restrictions on the overflow variables are given by constraints (30e).

Note that valid inequalities or specialized cuts can also be derived for SSLPR a priori to strengthen the LP relaxation of the model. These inequalities can be derived based on simple polyhedral considerations or using minimal (or lifted) cover inequalities. For example, since a client j can only be assigned to a server that has been located at site i, then for a given $\omega \in \Omega$ the simple constraints $x_i - y^{\omega}_{ij} + y^{\omega}_{j0} \geq 0, \forall i \in \mathcal{I},\ \forall j \in \mathcal{J}$, are valid for SSLPR. Such constraints tighten the LP relaxation of the model. However, the total number of these constraints may be very large in general leading to larger problem sizes. It should also be seen that SSLPR formulation (29-30) has *random recourse*, deterministic technology matrix, and random righthand side vector. If future client availability is assumed to be known (e.g. via client-server contracts), then SSLPR has deterministic righthand side vector, that is, $r^j(\omega) = r^j = 1, \forall j \in \mathcal{J}, \forall \omega \in \Omega$. Thus SSLPR has the same form as SIP2 and thus the disjunctive cuts for SIP2 derived in Section 5 are applicable to this special case of SSLPR.

7.2 Instance Generation

Several instances of SSLPR with random recourse but deterministic technology matrix and righthand side vector were randomly generated. The instances are available online on the author's website at http://ise.tamu.edu/people/faculty/Ntaimo and will also be

made available on the SIP library (SIPLIB). The instances were generated as follows. The maximum number of servers (m) was arbitrarily fixed at 5 and 10, respectively, while the server capacities (τ_i's) were arbitrarily set at 2000 units for all servers. The maximum number of servers to be located (u) was set at the value of m. No zonal constraints were included. The number of clients (n) was arbitrarily fixed at 50 and 100, respectively, and all clients were assumed to be available for future service. Server location costs (c_i's) were generated from the uniform distribution $U(7.5 \times 10^5, 1 \times 10^6)$, while client-server per unit revenue (q_{ij}'s) were fixed at one per unit of demand.

To generate random client-server demands, the ratio of the total server capacity to the total average client demand ($\bar{d}$) for each instance, $\phi = \sum_{i=1}^{m} \tau_i / \bar{d}$, was first computed. Then client-server demands were generated from $U(0, \mu)$, where μ was fixed at a value such that $\phi \approx 2$. This was done through experimentation by varying the values for μ until $\phi \approx 2$. The ratio ϕ provides a measure of instance difficult and was fixed at $\phi \approx 2$ for all the instances to allow for sufficient server capacity to satisfy all client demand. In general, decreasing ϕ towards one results in instances that are more difficult to solve since total server capacity may not be sufficient to satisfy overall client demand. Overflow (penalty) costs were arbitrarily set at 2×10^6, which was high enough for the range of objective cost values considered. The number of scenarios ($S = |\Omega|$) was set at 5, 10, 25, 50 and 100, and all the scenarios were assumed to have equal probability of occurrence. Three replications were made for each instance size to account for variability in the instance data as well as guard against pathological cases. Finally, different initial random seeds were used in generating each instance's problem data to allow for independence among the instances.

Table 1 gives the problem characteristics. The naming convention used for all the instances is 'case$m.n.S.a$', where m is the potential number of servers, n is the number of clients, S is the number of scenarios, and a is the replication number. The columns in the table are as follows: 'Constrs' is the number of constraints, 'Bins' is the number of binary decision variables, 'Cvars' is the number of continuous decision variables, and 'Dens' is the density of the constraint matrix in the extensive form (EF) formulation.

7.3 Computational Results

The D^2-R algorithm was implemented using the CPLEX 9.0 Callable Library (ILOG, 2003) in Microsoft Visual C++ 6.0 and applied to then SSLPR instances. In the current implementation of the algorithm, cuts are generated and added to the subproblems until the first-stage decision stabilized. Then the subproblems are solved to integer optimality and an optimality cut by Laporte and Louveaux (1993) generated and added to master program. CPLEX MIP and LP solvers were used to solve the subproblem MIPs and LPs. All the computational experiments were conducted on a DELL Optiplex GX620 with a Pentium D processor running at 3.0GHz with 3.5GB RAM. The instances were run to optimality or stopped when a CPU time limit of 3600 seconds (one hour) was reached. D^2-R stopping tolerance was set at $\epsilon = 0.005$.

The CPLEX MIP solver was also applied to the extensive form (EF) to provide a

Table 1: Problem Instance Characteristics

	EF				SUBPROBLEM		
Name	Constrs	Bins	Cvars	Dens	Constrs	Bins	Cvars
case5.25.5	151	630	25	0.05719	30	125	5
case5.25.10	301	1255	50	0.02872	30	125	5
case5.25.25	751	3130	125	0.01152	30	125	5
case5.25.50	1501	6255	250	0.00577	30	125	5
case5.25.100	3001	12505	500	0.00288	30	125	5
case10.50.5	301	2510	50	0.05637	60	500	10
case10.50.10	601	5010	100	0.02824	60	500	10
case10.50.25	1501	12510	250	0.01131	60	500	10
case10.50.50	3001	25010	500	0.00566	60	500	10
case10.50.100	6001	50010	1000	0.00283	60	500	10
case10.100.5	551	5010	50	0.03323	110	1000	10
case10.100.10	1101	10010	100	0.01663	110	1000	10
case10.100.25	2751	25010	250	0.00666	110	1000	10
case10.100.50	5501	50010	500	0.00333	110	1000	10
case10.100.100	11001	100010	1000	0.00167	110	1000	10

benchmark for comparison. The following CPLEX MIP solver settings were used and provided better results than the default values: set mip emphasis 1 (emphasizes looking for feasible solutions), set mip strategy start 4 (uses barrier at the root), and branching priority order on x. Also, due to the large size of the instances, CPLEX node files were used to prevent CPLEX from running out of memory prematurely.

Table 2 gives a summary of the results for instances with $m = 5$ and $n = 25$. The columns of the table are as follows: $Name$ is the instance name; v^1_{LP} is the optimal objective value for the LP relaxation; v^2_{LP} is the optimal objective value for the LP relaxation with IP master program (binary restrictions on x enforced); $\%\Delta v^1_{LP}$ is the percentage difference between the best found SIP objective value and v^1_{LP}; $\%\Delta v^2_{LP}$ is the percentage difference between the best found SIP objective value and v^2_{LP}; $v_{IP}^{D^2\text{-}R}$ is the best SIP objective value found by the D^2-R algorithm; v_{IP}^{CPLEX}, is the best SIP objective value found by the CPLEX MIP solver applied to EF; and $\%\Delta v_{IP}$' is the percentage difference between $v_{IP}^{D^2\text{-}R}$ and v_{IP}^{CPLEX}. The parameters $\%\Delta v^1_{LP}$, $\%\Delta v^2_{LP}$, and $\%\Delta v_{IP}$ are defined as follows: $\%\Delta v^1_{LP} = 100(v_{IP}^{D^2\text{-}R} - v^1_{LP})/v_{IP}^{D^2\text{-}R}$; $\%\Delta v^2_{LP} = 100(v_{IP}^{D^2\text{-}R} - v^2_{LP})/v_{IP}^{D^2\text{-}R}$; and $\%\Delta v_{IP} = 100(v_{IP}^{CPLEX} - v_{IP}^{D^2\text{-}R})/|v_{IP}^{D^2\text{-}R}|$, where $|.|$ denotes the absolute value. Note that value of $\%\Delta v_{IP} > 0$ means that the D^2-R algorithm obtained a better solution value than the CPLEX MIP solver within the time limit of one hour.

As can be seen in Table 2,the D^2-R algorithm solved all the instances to optimality while CPLEX could not. CPLEX only managed to solve two of the smaller instances but was unable to close the optimality gap. As indicated by the last column of the table, CPLEX performance deteriorates as the instance size increases, an indication of increasing problem difficult. However, the performance of the D^2-R algorithm is fairly linear with the number of scenarios, a desirable property for the algorithm. It is interesting to observe that these instances have very large LP gaps (v^1_{LP}) in general. However, the corresponding v^2_{LP} values are not as large. Since the D^2-R algorithm works

Table 2: D^2-R Algorithm Summary for $m = 5$ and $n = 25$

Name	v^1_{LP}	v^2_{LP}	$\%\Delta v^1_{LP}$	$\%\Delta v^2_{LP}$	$v^{D^2\text{-}R*}_{IP}$	v^{CPLEX}_{IP}	$\%\Delta v_{IP}$
case5.25.5.1	-156412.3	-137375.0	21.07	6.34	-129190.0	-129190.0	0.00
case5.25.5.2	-172439.1	-155307.0	16.60	5.01	-147893.0	-147768.0	0.08
case5.25.5.3	-156904.1	-78956.3	149.30	25.45	-62937.4	-60570.0	3.76
case5.25.10.1	-164497.0	-84520.5	126.05	16.15	-72769.5	-68489.6	5.88
case5.25.10.2	-168596.0	-150017.0	17.70	4.73	-143243.0	-143240.0	0.00
case5.25.10.3	-166028.0	-149276.0	17.77	5.89	-140972.0	-140593.0	0.27
case5.25.25.1	-174079.0	-92447.7	120.03	16.85	-79116.6	-65638.6	17.04
case5.25.25.2	-168172.0	-87974.3	125.57	18.00	-74553.7	-62152.2	16.63
case5.25.25.3	-164791.0	-89489.1	117.60	18.17	-75730.9	-63911.9	15.61
case5.25.50.1	-168463.0	-91648.6	113.80	16.31	-78795.8	230520.0	392.55
case5.25.50.2	-166434.0	-82971.9	138.02	18.66	-69923.7	336834.0	581.72
case5.25.50.3	-165692.0	-87562.4	122.23	17.44	-74558.4	289917.0	488.85
case5.25.100.1	-168676.0	-87688.2	127.92	18.49	-74005.8	269612.0	464.31
case5.25.100.2	-167724.0	-86341.1	130.80	18.81	-72671.9	305102.0	519.83
case5.25.100.3	-168152.0	-89487.1	122.23	18.27	-75664.2	291245.0	484.92

Number of Cuts and Computation Time (seconds)

	D^2-R				CPLEX	
	Cuts	L^2 Cuts	Iters	CPU	CPU	%Gap
case5.25.5.1	5	1	23	1.24	>3600	5.17
case5.25.5.2	5	1	27	1.69	>3600	4.46
case5.25.5.3	5	1	25	41.11	>3600	26.75
case5.25.10.1	10	1	28	44.34	>3600	22.93
case5.25.10.2	10	1	27	1.60	>3600	4.40
case5.25.10.3	10	1	27	2.31	>3600	5.70
case5.25.25.1	25	1	30	104.58	>3600	40.67
case5.25.25.2	25	1	30	144.26	>3600	41.35
case5.25.25.3	25	1	30	118.14	>3600	39.88
case5.25.50.1	50	1	30	262.95	>3600	139.74
case5.25.50.2	50	1	30	282.69	>3600	124.63
case5.25.50.3	300	1	46	397.44	>3600	130.19
case5.25.100.1	100	1	30	472.24	>3600	132.52
case5.25.100.2	100	1	30	633.54	>3600	128.29
case5.25.100.3	100	1	29	486.27	>3600	130.72

* optimality reached.

with an IP master, it is the latter gap that the cuts have to close.

The lower part of Table 2 gives a summary of the number of cuts and computation time. The columns of this part of the table are as follows: 'Cuts' gives the number of disjunctive decomposition cuts, L^2 gives the number of optimality cuts of Laporte and Louveaux (1993), 'Iters' gives the number of iterations, 'CPU' gives the computation time in seconds, and '%Gap' gives the percentage gap reported by the CPLEX MIP solver at termination. As can be seen in the table, the D^2-R algorithm was able to solver all the instances to optimality within 3 minutes, while CPLEX could not. Furthermore, the number of cuts generated by the D^2-R algorithm in each instance is relatively small. Notice that the number of disjunctive cuts increases with the instance size in general. In this case, only one L^2 cut was generated during the last iteration of the algorithm.

Table 3: D^2-R Algorithm Summary for $m = 10$ and $n = 50$

Name	v_{LP}^1	v_{LP}^2	$\%\Delta v_{LP}^1$	$\%\Delta v_{LP}^2$	$v_{IP}^{D^2\text{-}R}$	v_{IP}^{CPLEX}	$\%\Delta v_{IP}$
case10.50.5.1	-74871.3	-33592.0	192.26	31.13	-25618.2	-20163.4	21.29
case10.50.5.2	-71559.2	-38333.6	155.90	37.08	-27964.0	-20506.4	26.67
case10.50.5.3	-63694.5	-25759.2	273.75	51.15	-17041.8	-14283.4	16.19
case10.50.10.1	-73612.3	-32262.7	224.22	42.10	-22704.6	36020.5	258.65
case10.50.10.2	-68738.9	-32375.7	204.42	43.38	-22580.2	-10120.8	55.18
case10.50.10.3	-70632.0	-29858.6	246.63	46.53	-20376.5	35323.9	273.36
case10.50.25.1	-77322.9	-41292.3	147.73	32.29	-31212.6	34166.4	209.46
case10.50.25.2	-67961.4	-33529.7	194.83	45.46	-23051.0	48005.6	308.26
case10.50.25.3	-69738.5	-32006.4	221.08	47.36	-21720.2	166712.0	867.54
case10.50.50.1	-69562.9	-32963.8	216.55	50.00	-21975.6	105138.0	578.43
case10.50.50.2	-68954.2	-33456.3	196.54	43.88	-23252.9	129100.0	655.20
case10.50.50.3	-70747.3	-30466.3	263.55	56.56	-19459.9	153067.0	886.58
case10.50.100.1	-71170.8	-33375.3	206.99	43.96	-23183.7	113908.0	591.33
case10.50.100.2	-70071.0	-33306.7	213.86	49.19	-22325.7	116034.0	619.73
case10.50.100.3	-72080.2	-37280.0	170.57	39.94	-26639.7	110157.0	513.51

Number of Cuts and Computation Time (seconds)

	D^2-R				CPLEX	
	Cuts	L^2 Cuts	Iters	CPU	CPU	%Gap
case10.50.5.1	15	24	296	>3600	>3600	65.30
case10.50.5.2	20	24	269	>3600	>3600	84.80
case10.50.5.3	15	24	263	>3600	>3600	78.03
case10.50.10.1	50	12	437	>3600	>3600	189.04
case10.50.10.2	30	12	422	>3600	>3600	217.75
case10.50.10.3	40	12	370	>3600	>3600	183.73
case10.50.25.1	100	5	706	>3600	>3600	220.56
case10.50.25.2	100	5	697	>3600	>3600	169.59
case10.50.25.3	100	5	670	>3600	>3600	119.13
case10.50.50.1	150	3	848	>3600	>3600	131.26
case10.50.50.2	150	3	856	>3600	>3600	125.86
case10.50.50.3	150	3	826	>3600	>3600	119.88
case10.50.100.1	100	1	926	>3600	>3600	129.28
case10.50.100.2	100	1	913	>3600	>3600	128.70
case10.50.100.3	200	2	839	>3600	>3600	133.83

Table 3 give results for instances with $m = 10$ and $n = 50$. These are relatively

larger instances compared to the previous set and even the D^2-R algorithm could not solve them to optimality within the time limit of one hour. However, the D^2-R algorithm clearly outperforms the direct solver on all the instances as shown by the large values in the last column of the upper part of the table. Notice that the LP gaps $\%\Delta v^2_{LP}$ are very large (between 30 - 60%) for this set of instances. This is evident in the large number of cuts generated by the D^2-R algorithm shown in the lower part of the table.

Table 4: D^2-R Algorithm Summary for $m = 10$ and $n = 100$

Name	v^1_{LP}	v^2_{LP}	$\%\Delta v^1_{LP}$	$\%\Delta v^2_{LP}$	$v^{D^2\text{-}R}_{IP}$	v^{CPLEX}_{IP}	$\%\Delta v_{IP}$
case10.100.5.1	-112633.0	-84255.1	38.21	3.39	-81494.0	-80497.0	1.22
case10.100.5.2	-110050.0	-92439.2	22.60	2.98	-89760.8	-88228.4	1.71
case10.100.5.3	-106503.0	-84056.1	30.56	3.04	-81573.6	-79994.2	1.94
case10.100.10.1	-107552.0	-81777.4	36.60	3.87	-78732.9	-73131.5	7.11
case10.100.10.2	-106668.0	-80449.6	36.92	3.27	-77902.6	-71668.7	8.00
case10.100.10.3	-106339.0	-83143.9	32.26	3.41	-80402.1	-75630.3	5.93
case10.100.25.1	-106831.0	-84699.1	30.99	3.85	-81556.5	3723.5	104.57
case10.100.25.2	-108508.0	-85527.6	30.87	3.16	-82910.5	29203.8	135.22
case10.100.25.3	-107530.0	-82351.2	35.93	4.10	-79104.2	-19637.2	75.18
case10.100.50.1	-106895.0	-81660.9	36.20	4.05	-78481.6	84278.7	207.39
case10.100.50.2	-109822.0	-87972.1	29.91	4.07	-84534.4	2617.8	103.10
case10.100.50.3	-106293.0	-81759.9	35.50	4.22	-78446.2	-5507.3	92.98
case10.100.100.1	-107571.0	-85015.5	31.71	4.09	-81674.8	114703.0	240.44
case10.100.100.2	-108518.0	-85164.4	32.85	4.26	-81684.2	70525.9	186.34
case10.100.100.3	-107674.0	-85506.1	30.80	3.87	-82320.2	130875.0	258.98

Number of Cuts and Computation Time (seconds)

	D^2-R				CPLEX	
	Cuts	L^2 Cuts	Iters	CPU	CPU	%Gap
case10.100.5.1	25	24	394	>3600	>3600	4.62
case10.100.5.2	25	24	291	>3600	>3600	4.74
case10.100.5.3	40	24	365	>3600	>3600	5.05
case10.100.10.1	20	12	381	>3600	>3600	11.78
case10.100.10.2	40	12	383	>3600	>3600	12.24
case10.100.10.3	50	12	490	>3600	>3600	9.91
case10.100.25.1	50	5	772	>3600	>3600	†
case10.100.25.2	50	5	726	>3600	>3600	392.86
case10.100.25.3	100	5	781	>3600	>3600	319.35
case10.100.50.1	100	2	925	>3600	>3600	196.89
case10.100.50.2	100	2	939	>3600	>3600	†
case10.100.50.3	100	2	919	>3600	>3600	†
case10.100.100.1	100	1	969	>3600	>3600	174.12
case10.100.100.2	100	1	988	>3600	>3600	220.76
case10.100.100.3	100	1	971	>3600	>3600	165.33

† CPLEX MIP solver could not compute the %Gap.

The final results are reported in Tables 4 for the largest size set of instances with $m = 10$ and $n = 100$. The performance of the D^2-R algorithm is relative good even in this case. Notice that for this set of instances, however, even though the LP gaps $\%\Delta v^1_{LP}$ are relative large, the LP gaps $\%\Delta v^2_{LP}$ are much smaller (around 3-5%). CPLEX performs close to the D^2-R algorithm on the smallest size instances. However, as the number of

scenarios increases (problem size increases), the performance of CPLEX quickly deteriorates. Finally, it was observed that unlike the direct solver, the decomposition method provides better objective values in general within the first few minutes. Therefore, in practical cases where quick decisions are needed, an early termination of the algorithm would still provide fairly good quality solutions.

8 Conclusion

This paper introduces disjunctive decomposition for two-stage mixed 0-1 SIP with *random recourse*. Disjunctive decomposition allows for cutting planes based on disjunctive programming to be generated for each scenario subproblem under a temporal decomposition setting of the SIP problem. New valid inequalities for mixed 0-1 SIP with random recourse are presented. In the case of random recourse but deterministic righthand side vector and technology matrix, valid inequalities that allow for sharing cut coefficients among scenario subproblems are obtained. The valid inequalities are used to derive a disjunctive decomposition algorithm for mixed 0-1 SIP. The derivation of the proposed method has been motivated by real-life stochastic server location problems with random recourse, which have many applications in operations research. A computational study with large-scale instances demonstrate the potential of the method towards solving this class of problems. Even though the primary focus of the paper is on the generation of cutting planes within a temporal decomposition of the SIP problem, it should be pointed out that cutting planes alone may not be adequate for solving large-scale instances to optimality. Thus the proposed cuts can be incorporated within a branch-and-cut setting, which may prove to be even more effective. Furthermore, the computational and computer memory demands of this class of problems calls for the use of high performance parallel/distributed computing platforms on which the subproblems are solved on separate computers.

Acknowledgments. This work was supported in part by NSF grant CNS-0540000. The author wishes to thank an anonymous reviewer for their helpful comments on an earlier draft of this paper, and his student Matthew W. Tanner for an initial implementation of the D^2-R algorithm.

References

Ahmed, S. and Garcia, R.: 2003, Dynamic capacity acquisition and assignment under uncertainty, *Annals of Operational Research* **124**, 267–283.

Ahmed, S., Tawarmalani, M. and Sahinidis, N. V.: 2004, A finite branch and bound algorithm for two-stage stochastic integer programs, *Mathematical Programming* **100**, 355–377.

Balas, E.: 1975, Disjunctive programming: cutting planes from logical conditions, *in*

O. Mangasarian, R. Meyer and S. Robinson (eds), *Nonlinear Programming 2*, Academic Press.

Balas, E.: 1979, Disjunctive programming, *Annals of Discrete Mathematics* **5**, 3–51.

Balas, E., Ceria, E. and Cornuéjols, G.: 1996, Mixed 0-1 programming by lift-and-project in a branch-and-cut framework, *Management Science* **42**, 1229–1246.

Balas, E., Ceria, E. S. and Cornuéjols, G.: 1993, A lift-and-project cutting plane algorithm for mixed 0-1 integer programs, *Mathematical Programming* **58**, 295–324.

Benders, J. F.: 1962, Partitioning procedures for solving mixed-variable programming problems, *Numerische Mathematik* **4**, 238–252.

Birge, J. R. and Louveaux, F. V.: 1997, *Introduction to Stochastic Programming*, Springer, New York.

Blair, C. and Jeroslow, R.: 1978, A converse for disjunctive constraints, *Journal of Optimization Theory and Applications* **25**, 195–206.

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

Carøe, C. C.: 1998, *Decomposition in Stochastic Integer Programming*, Ph.D. thesis, Dept. of Operations Research, University of Copenhagen, Denmark.

Carøe, C. and Tind, J.: 1997, A cutting-plane approach to mixed 0-1 stochastic integer programs, *European Journal of Operational Research* **101**, 306–316.

Donovan, G. and Rideout, D.: 2003, An integer programming model to optimize resource allocation for wildfire containtment, *Forest Science* **49(2)**, 331–335.

ILOG, I.: 2003, *CPLEX 9.0 Reference Manual*, ILOG CPLEX Division.

Klein Haneveld, W. and van der Vlerk, M.: 1999, Stochastic integer programming: general models and algorithms, *Annals of Operations Research* **85**, 39–57.

Laporte, G. and Louveaux, F. V.: 1993, The integer L-shaped method for stochastic integer programs with complete recourse, *Operations Research Letters* **1**, 133–142.

Lee, W. J.: 2006, *A Stochastic Mixed Integer Programming Approach to Wildfire Management Systems*, M.S. Thesis, Dept. of Industrial and Systems Engineering, Texas A&M University, USA.

Linderoth, J. and Wright, S.: 2003, Decomposition algorithms for stochastic programming on a computational grid, *Computational Optimization and Applications* **24**, 207–250.

NIFC: 2006, *Wildland Fire Statistics*, National Interagency Fire Center, http://www.nifc.gov/stats/wildlandfirestats.html.

Ntaimo, L.: 2004, *Decomposition Algorithms for Stochastic Combinatorial Optimization: Computational Experiments and Extensions*, Ph.D. thesis, Dept. of Systems and Industrial Engineering, University of Arizona, USA.

Ntaimo, L. and Sen, S.: 2005, The million-variable 'march' for stochastic combinatorial optimization, *Journal of Global Optimization* **32(3)**, 385–400.

Riis, M., Skriver, A. and Lodahl, J.: 2004, Deployment of mobile switching centers in a telecommunications network: A stochastic programming approach, *Telecommunication Systems* **26**, 93–109.

Ruszczynski, A. and Shapiro, A. (eds): 2003, *Stochastic Programming*, Vol. 10 of *Handbooks in Operations Research and Management Science*, Elsevier.

Schultz, R.: 1993, Continuity properties of expectation functions in stochastic integer programming, *Mathematics of Operations Research* **18**, 578–589.

Schultz, R., Stougie, L. and van der Vlerk, M.: 1996, Two-stage stochastic integer programming: A survey, *Statistica Neerlandica* **50**, 404–416.

Sen, S.: 2003, Algorithms for stochastic mixed-integer programming models, *in* K. Aardal, G. Nemhauser and R. Weismantel (eds), *Stochastic Integer Programming Handbook*, Dordrecht, The Netherlands, chapter 18.

Sen, S. and Higle, J.: 2005, The C3 theorem and a D2 algorithm for large scale stochastic mixed-integer programming: Setconvexification, *Mathematical Programming* **104(1)**, 1–20.

Sen, S., Higle, J. L. and Ntaimo, L.: 2002, A summary and illustration of disjunctive decomposition with set convexification, *in* D. L. Woodruff (ed.), *Stochastic Integer Programming and Network Interdiction Models*, Kluwer Academic Press, chapter 6, p. 105.

Sen, S. and Sherali, H. D.: 1987, Nondifferentiable reverse convex programs and facetial cuts via a disjunctive characterization, *Mathematical Programming* **37**, 169–183.

Sherali, H. and Shetty, C.: 1980, Optimization with disjunctive constraints, *Lecture Notes in Economics and Mathematical Systems* **181**, 411–430.

Van Slyke, R. and Wets, R.-B.: 1969, L-shaped linear programs with application to optimal control and stochastic programming, *SIAM Journal on Applied Mathematics* **17**, 638–663.

Wang, Q., Batta, E. and Rump, C. M.: 2003, Facility location models for immobile servers with stochastic demand, *Naval Research Logistics* **51**, 137–152.