Chapter 9

Algorithms for Stochastic Mixed-Integer Programming Models

Suvrajeet Sen
MORE Institute, SIE Department, University of Arizona, Tucson, AZ 85721, USA

Abstract

In this chapter, we will study algorithms for both two-stage as well as multi-stage stochastic mixed-integer programs. We present stagewise (resource-directive) decomposition methods for two-stage models, and scenario (price-directive) decomposition methods for multi-stage models. The manner in which these models are decomposed relies not only on the specific data elements that are random, but also on the manner in which the integer (decision) variables interact with these data elements. Accordingly, we study a variety of structures ranging from models that allow randomness in all data elements, to those that allow only specific elements (e.g. right-hand-side) to be influenced by randomness. Since the decomposition algorithms presented here are based on certain results from integer programming, the relevant background is also provided in this chapter.

1. Introduction

Integer Programming (IP), and Stochastic Programming (SP) constitute two of the more vibrant areas of research in optimization. Both areas have blossomed into fields that have solid mathematical foundations, reliable algorithms and software, and a plethora of applications that continue to challenge the current state-of-the-art computing resources. For a variety of reasons, these areas have matured independently. A study of SMIP requires that we integrate the methods of continuous optimization (SP) and those of discrete optimization (IP). With the exception of a joint appreciation for Benders’ decomposition (Benders [1962] and Van Slyke and Wets [1969]), the IP and SP communities have, for many years, kept their distance from a large class of stochastic mixed-integer programming (SMIP) models. Indeed, the only class of SMIP models that has attracted its fair share of attention is the one for which Benders’ decomposition is applicable without further mathematical developments. Such models are typically two-stage stochastic
programs in which the first-stage decisions are mixed-integer, and the second-stage (recourse) decisions are obtained from linear programming (LP) models. Research on other classes of SMIP models is recent; some of the first structural results for integer recourse problems are only about a decade old (e.g. Schultz [1993]). The first algorithms also began to appear around the same time (e.g. Laporte and Louveaux [1993]). As for dissertations, the first in the area appears to be Stougie [1985], and a few of the early notable ones may be Takriti [1994], Van der Vlerk [1995], and Caroe [1998], to name a few.

In the last few years there has been a flurry of activity resulting in rapid growth of the area. This chapter is devoted to algorithmic issues that have a bearing on two focal points. First, we focus on decomposition algorithms because they have the potential to provide scalable approaches for large-scale models. For realistic SP models, the ability to handle a large number of potential scenarios is critical. The second focal point deals with integer recourse models (i.e. the integer variables are associated with recourse decisions in stages two and beyond). These issues are intimately related to IP decomposition which is likely to be of interest to researchers in both SP as well as IP. We hope that this chapter will motivate readers to investigate novel algorithms that will be scalable enough to solve practical stochastic mixed-integer programming models.

Problem Setting

A two-stage SMIP model is one in which a subset of both first and second-stage variables are required to satisfy integer restrictions. To state the problem, let \( \tilde{\omega} \) denote a random variable used to model data uncertainty in a two-stage model. (We postpone the statement of a multi-stage problem to section 4.) Since SP models are intended for decision-making, a decision vector \( x \) must be chosen in such a manner that the consequences of the decisions (evaluated under several alternative outcomes of \( \tilde{\omega} \)) are accommodated within an optimal choice model. The consequences of the first-stage decisions are measured through an optimization problem (called the recourse problem) which allows the decision-maker to adapt to an observation of the data (random variable). Suppose that an observation of \( \tilde{\omega} \) is denoted \( \omega \). Then the consequences of choosing \( x \) in the face of an outcome \( \omega \) may be modeled as

\[
\begin{align*}
  h(x, \omega) &= \text{Min} \ g(\omega)^T y \\
  W(\omega)y &\geq r(\omega) - T(\omega)x \\
  y &\geq 0, y_j \text{ integer, } j \in J_2,
\end{align*}
\]

where \( J_2 \) is an index set that may include some or all the variables listed in \( y \in \mathbb{R}^{m_2} \). Throughout this chapter, we will assume that all realizations \( W(\omega) \) are rational matrices of size \( m_2 \times n_2 \). Whenever \( J_2 \) is non-empty, and \( |J_2| \neq n_2 \),
(1.1) is said to provide a model with mixed-integer recourse. Although (1.1) is stated as though the random variable influences all data, most applications lead to models that lead to only some data uncertainty, which in turn lead to certain specialized models.

A typical decision-maker used his/her attitude towards risk to order alternative choices of $x$. In the decision analysis literature, the collection of possible choices are usually a few in number, and for such cases, it is possible to enumerate all the choices. For more complicated decision models, where the choices may be too many to enumerate, one resorts to optimization techniques, and more specifically to stochastic programming.

While several alternative “risk preferences” have been incorporated within SP models recently (see Ogryczak and Ruszczynski [2002], Riis and Schultz [2003], Takriti and Ahmed [2004]), the predominant approach in the SP literature is the “expected value” model. In order to focus our attention on complications arising from integer restrictions on decision variables, we will restrict our study to the “expected value” model. For this setting, the two-stage SMIP model may be stated as follows.

$$\min_{x\in\mathbb{R}^n} c^T x + E[h(x, \omega)],$$

where $\omega$ denotes a random variable defined on a probability space $(\Omega, \mathcal{A}, P)$, $X$ a convex polyhedron, and $X$ denotes either the set of binary vectors $B$, or integer vectors $Z$ or even mixed-integer vectors $M = \{x \mid x \geq 0, x_j \text{ integer, } j \in J_1\}$, where $J_1$ is a given index set consisting of some or all of first-stage variables $x \in \mathbb{R}^{m_1}$. Whenever we refer to the two-stage SMIP problems, we will be referring to (1.1,1.2). Throughout this chapter, we will assume that the random variables have finite support, so that the expectation in (1.2) reduces to a summation.

Within the stochastic programming literature, a realization of $\omega$ is known as a “scenario”. As such, the second-stage problem (1.1) is often referred to as a “scenario subproblem.” Because of its dependence on the first-stage decision $x$, the value function $h(\cdot)$ is referred to as the recourse function. Accordingly, $E[h(\cdot)]$ is called the expected recourse function of the two-stage model. These two-stage models are said to have a fixed recourse matrix (or simply fixed recourse) when the matrix $W(\omega)$ is deterministic; that is, $W(\omega) = W$. If the matrix $T(\omega)$ is deterministic, (i.e., $T(\omega) = T$), the stochastic program is said to have fixed tenders. When the second-stage problem is feasible for all choices of $x \in \mathbb{R}^{m_1}$, the model is said to possess the complete recourse property; moreover, if the second-stage problem is feasible for all $x \in X \cap X$, then it is said to possess the relatively complete recourse property. When the matrix $W$ has the special structure that $W = (I, -I)$, the second-stage decision variables are continuous, and the constraints (1.1b) are equations, then the resulting problem is called a stochastic program with “simple recourse.” In this special case, the second-stage variables simply measure the deviation from an uncertain target. The standard news-vendor problem of perishable
inventory management is a stochastic program with simple recourse. It turns out that the continuous simple recourse problem is one class of models that is very amenable to accurate solutions (Kall and Mayer [1996]). Moreover as discussed subsequently, these models may be used in connection with methods for the solution of simple integer recourse models.

Algorithmic research in stochastic programming has focused on methods that are intended to accommodate a large number of scenarios so that realistic applications can be addressed. This has led to novel decomposition algorithms, some deterministic (e.g. Rockafellar and Wets [1991], Mulvey and Ruszczynski [1995]), and some stochastic (Higle and Sen [1991], Infanger [1992]). In this chapter we will adopt a deterministic decomposition paradigm. Such approaches are particularly relevant for SMIP because the idea of solving a series of small MIP problems to ultimately solve a large SMIP is computationally appealing. Moreover, due to the proliferation of networks of computers, such decomposition methods are likely to be more scalable than methods that treat the entire SMIP as one large deterministic MIP. Accordingly, this chapter is dedicated to decomposition-based algorithms for SMIP.

In this chapter, we will examine algorithms for both two-stage and multi-stage stochastic mixed-integer programs. In section 2, we will summarize some preliminary results that will have a bearing on the development of decomposition algorithms for SMIP. Section 3 is devoted to two-stage models under alternative assumptions that specify the structure of the model. For each class of models, we will discuss the decomposition method that best suits the structure. Section 4 deals with multi-stage models. We remind the reader that the state-of-the-art in this area is still in a state of flux, and encourage him/her to participate in our exploration to find ways to solve these very challenging problems.

2. Preliminaries for Decomposition Algorithms

The presence of integer decisions in (1.1) adds significant complications to designing decomposition algorithms for SMIP. In devising decomposition methods for these problems, it becomes necessary to draw upon results from the theory of IP. Most relevant to this study are results from IP duality, value functions, and disjunctive programming. The material in this section relies mainly on the work of Wolsey [1981] for IP duality, Blair and Jeroslow [1982], Blair [1995] for IP/MIP value functions, and Balas [1979] for disjunctive programming. Of course, some of this material is available in Nemhauser and Wolsey [1988]. We will also provide bridges from the world of MIP into that of SMIP. The first bridge deals with the properties of the SMIP recourse function which derive from properties of the MIP value function. These results were obtained by Schultz [1993]. The next bridge is that provided in the framework of Caroe and Tind [1998].
Structural Properties

Definition 2.1. $f : \mathbb{R}^n \to \mathbb{R}$ is said to be sub-additive if $f(u + v) \leq f(u) + f(v)$. When this inequality is reversed, $f$ is said to be super-additive.

In order to state some results about the value function of an IP/MIP, we restate (1.1) in a familiar form, without the dependence on the data random variable, or the first-stage decision.

$$h(r) = \min y$$
$$Wy \geq r$$
$$y \geq 0, y_j \text{ integer, } j \in J_2.$$ (2.1a, 2.1b, 2.1c)

Proposition 2.2.

a) The value function $(h(r))$ associated with (2.1) is non-decreasing, lower semi-continuous, and sub-additive over its effective domain (i.e. over the set of right hand sides for which the value function is finite).

b) Consider an SMIP as stated in (1.1,1.2) and suppose that the random variables have finite support. If the effective domain of the expected recourse function $E[h()]$ is non-empty, then it is lower semi-continuous, and sub-additive on its effective domain.

c) Assume that the matrix $W$ and the right-hand side vector $r$ are integral, and (2.1) is a pure IP. Let $v$ denote any vector of $m_2$ integers. Then the value function $h$ is constant over sets of the form

$$\{z \mid v - (1, \ldots, 1)^\top < z \leq v\}, \quad \forall v \in \mathbb{Z}^{m_2}.$$  

For a proof of part a), please consult chapter II.3 of Nemhauser and Wolsey [1988]. Of course part b) follows from the fact that the expected recourse function is a finite sum of lower semi-continuous and sub-additive functions. And part c) is obvious since $W$ and $y$ have entries that are integers. This theorem is used in Schultz, Stougie, and Van der Vlerk [1998], as well as Ahmed, Tawarmalani and Sahinidis [2004] (see section 3).

For the case in which the random variables in SMIP are continuous, one may obtain continuity of the recourse function, but at a price. The following result requires that the random variables be absolutely continuous, which as we discuss below, is a significant restriction for constrained optimization problems.

Proposition 2.3. Assume that (1.1) has randomness only in $r(\tilde{\omega})$, and let the probability space of this random variable, denoted $(\Omega, \mathcal{A}, \mathcal{P})$, be such that $\mathcal{P}$
is absolutely continuous with respect to the Lebesgue measure in $\mathbb{R}^{m_2}$. Moreover, suppose that the following hold.

a) (Dual feasibility). There exists $\theta \geq 0$ such that $W^\top \theta \leq g$.

b) (Complete recourse). For any choice of $r$ in (2.1), the MIP feasible set is non-empty.

c) (Finite expectation). $E[|r(\tilde{\omega})|] < \infty$.

Then, the expected recourse function is continuous.

This result was proven by Schultz [1993]. We should draw some parallels between the above result for SMIP and requirements for differentiability of the expected recourse function in SLP problems. While the latter possess expected recourse functions that are continuous, differentiability of the expected recourse function in SLP problems requires a similar absolute continuity condition (with respect to the Lebesgue measures in $\mathbb{R}^{m_2}$). We remind the reader that even when a SLP has continuous random variables, the expected recourse function may fail to satisfy differentiability due to the lack of absolute continuity (Sen [1993]). By the same token, the SMIP expected recourse function may fail to be continuous without the assumption of absolute continuity as required above. It so happens that the requirement of absolute continuity (with respect to the Lebesgue measure in $\mathbb{R}^{m_2}$) is rather restrictive from the point of view of practical optimization models. In order to appreciate this, observe that many practical LP/IP models have constraints that are entirely deterministic; for example, flow conservation/balance constraints often have no randomness in them. Formulations of this type (where some constraints are completely deterministic) fail to satisfy the requirement that the measure $P$ is absolutely continuous with respect to the Lebesgue measure in $\mathbb{R}^{m_2}$. Thus, just as differentiability is a luxury for SLP problems, continuity is a luxury for SMIP problems.

**IP Duality**

We now turn to an application of sub-additivity, especially its role in the theory of valid inequalities and IP duality.

**Definition 2.4.**

a) Let $S$ denote the set of feasible points of an MIP such as (2.1). If $y \in S$ implies $\pi^\top y \geq \pi_0$, then the latter is called a valid inequality for the set $S$.

b) A monoid is a set $M$ such that $0 \in M$, and if $W_1, W_2 \in M$, then $W_1 + W_2 \in M$. 
Theorem 2.5. Let \( Y = \{ y \in \mathbb{R}^n_+ \mid Wy \geq r \} \), and assume that the entries of \( W \) are rational. Consider a pure integer program whose feasible set \( S = Y \cap \mathbb{Z}^n \) is non-empty.

a) If \( F \) is a sub-additive function defined on the monoid generated by the columns \( \{ W_j \}_{j=1}^n \) of \( W \), then

\[
\sum_j F(W_j)y_j \geq F(r)
\]

is valid inequality.

b) Let \( \pi^\top y \geq \pi_0 \) denote a valid inequality for \( S \). Then, there is a sub-additive non-decreasing function \( F \) defined on the monoid generated by the columns \( W_j \) of \( W \) such that \( F(0) = 0, \pi_j \geq F(W_j) \) and \( \pi_0 \leq F(r) \).

The reader may consult the book by Nemhauser and Wolsey [1988] for more on sub-additive duality. Given the above theorem, the sub-additive dual of (2.1) is as follows.

\[
\begin{align*}
\text{Max} & \quad F(r) \\
\text{s.t} & \quad F(W_j) \leq g_j, \quad \forall j \\
& \quad F(0) = 0.
\end{align*}
\]

Several standard notions such as strong-duality and complementary slackness hold for this primal-dual pair. Moreover, Gomory’s fractional cuts lead to a class of sub-additive functions constructed from using the ceiling operation on coefficients of linear valid inequalities; that is, functions of the form

\[
\sum_j [\pi_j]y_j \geq [\pi_0],
\]

where \( \pi^\top y \geq \pi_0 \) is a valid inequality for \( S \) (defined in Theorem 2.5). Such functions, which are referred to as Chvatal functions, are sub-additive and provide the appropriate class of dual price functions for the analysis of Gomory’s fractional cuts. However, it is important to note that other algorithmic procedures for IP develop other dual price functions. For instance, branch-and-bound (B&B) methods generate non-decreasing, piecewise linear concave functions that provide solutions to a slightly different dual problem. In this sense, IP algorithms differ from algorithms for convex programming for which linear price functions are sufficient. For a more in-depth review of
non-convex price functions (sub-additive or others), the reader should refer to Tind and Wolsey [1981]. Because certain algorithms do not necessarily generate sub-additive price functions, Caroe and Tind [1998] state an IP dual problem over a class of non-decreasing functions, which of course, includes the value function of (2.1). Therefore, the dual problem used in Caroe and Tind [1998] is as follows.

\[
\text{Max} \quad F(r) \quad \text{non-decreasing} \quad \quad (2.3a)
\]

\[
\text{s.t} \quad F(Wy) \leq g^T y \quad \quad (2.3b)
\]

\[
F(0) = 0. \quad \quad (2.3c)
\]

We are now in a position to discuss the conceptual framework provided in Caroe and Tind [1998]. Their investigation demonstrates that on a conceptual level, it is possible to generalize the structure of Benders' decomposition (or L-shaped method) to decompose SMIP problems. However as noted in Caroe and Tind [1998], this conceptual scheme does not address practical computational difficulties associated with solving first-stage approximations which contain non-convex functions such as Chvatal functions. Nevertheless, the approach provides a conceptual bridge between MIP and SMIP problems.

In order to maintain simplicity in this presentation, we assume that the second-stage problem satisfies the complete recourse property. Assuming that the random variable modeling uncertainty is discrete, with finite support \( \Omega = \{\omega^1, \ldots, \omega^N\} \), a two-stage SMIP may be stated as

\[
\text{Min} \quad c^T x + \sum_{\omega \in \Omega} p(\omega)g(\omega)^T y(\omega) \quad \quad (2.4a)
\]

\[
\text{s.t} \quad Ax \geq b \quad \quad (2.4b)
\]

\[
T(\omega)x + Wy(\omega) \geq r(\omega), \quad \forall \omega \in \Omega \quad \quad (2.4c)
\]

\[
(x, \{y(\omega)\}_{\omega \in \Omega}) \geq 0, x_j \text{ integer}, j \in J_1, \quad \text{and} \quad y_j(\omega) \text{ integer}, \forall j \in J_2. \quad (2.4d)
\]

Despite the fact that there are several assumptions underlying (2.4), it is somewhat general from the IP point of view since both the first and second stages allow general integer variables.
Following Caroe and Tind [1998], suppose we wish to apply a resource directive decomposition method, similar to Benders’ decomposition. At iteration $k$ of such a method, we solve one second-stage subproblem for each outcome $\omega$, and assuming that we have chosen an appropriate solution method for the second-stage, then we obtain a non-decreasing price function $F^k(r(\omega) - T(\omega)x)$ for each outcome $\omega \in \Omega$. Consequently, we obtain a “cut” of the form

$$\eta \geq \sum_{\omega \in \Omega} p(\omega) F^k(r(\omega) - T(\omega)x).$$

Hence, as the iterations proceed, one obtains a sequence of relaxed master programs of the following form.

Min $c^T x + \eta$ \hfill (2.5a)

s.t $Ax \geq b$ \hfill (2.5b)

$$\eta \geq \sum_{\omega \in \Omega} p(\omega) F^t(r(\omega) - T(\omega)x), \quad t = 1, \ldots, k \hfill (2.5c)$$

$$x \geq 0, x_j \text{ integer}, \ j \in J_1. \hfill (2.5d)$$

As with Benders’ (or L-shaped) decomposition, each iteration augments the first-stage approximation with one additional collection of price functions as shown in (2.5c). The rest of the procedure also mimics Benders’ decomposition in that the sequence of objective values of (2.5) generates an increasing sequence of lower bounds, whereas, the subproblems at each iteration provide values used to compute an upper bound. The method stops when the upper and lower bounds are sufficiently close. Provided that the second-stage problems are solved using Gomory’s cuts, or B&B, it is not difficult to show that the method must terminate in finitely many steps. Of course, finiteness also presumes that (2.5) can be solved in finite time.

We now visit the question of computational practicality of the procedure outlined above. The main observation is that the first-stage (master program) can be computationally unwieldy because the Chvatal functions arising from Gomory’s method and piecewise linear concave functions resulting from B&B are nonconvex and are directly imported into the first-stage minimization [see (2.5c)]. These functions render the first-stage problem somewhat intractable. In section 3, we will discuss methods that will convexify such functions, thus leading to a more manageable first-stage problem.
Disjunctive Programming

Disjunctive programming focuses on characterizing the convex hull of disjunctive sets of the form

\[ S = \bigcup_{h \in H} S_h, \tag{2.6} \]

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

\[ S_h = \{ y \mid G_h y \geq r_h, y \geq 0 \}. \tag{2.7} \]

This line of work originated with Balas [1975], and further developed in Blair and Jeroslow [1978]. Balas [1979] and Sherali and Shetty [1980] provide a comprehensive treatment of the approach, as well as its connections with other approaches for IP. Balas, Ceria and Cornuéjols [1993] provide computational results for such methods under a particular reincarnation called “lift-and-project” cuts.

The disjunction stated in (2.6, 2.7) is said to be in disjunctive normal form (i.e., none of the terms \( S_h \) contain any disjunction). It is important to recognize that the set of feasible solutions of any mixed-integer (0-1) program can be written as the union of polyhedra as in (2.6, 2.7) above. However, the number of elements in \( H \) can be exponentially large, thus making an explicit representation computationally impractical. If one is satisfied with weaker relaxations, then more manageable disjunctions can be stated. For example, the lift-and-project inequalities of Balas, Ceria and Cornuéjols [1993] use conjunctions associated with a linear relaxation together with one disjunction of the form: \( y_j \leq 0 \) or \( y_j \geq 1 \), for some \( j \in J_2 \). (Of course, \( y_j \) is assumed to be a binary variable.) For such a disjunctive set, the cardinality of \( H \) is two, with one polyhedron containing the inequalities \( W y \geq r, y \geq 0, -y_j \geq 0 \) and the other polyhedron defined by \( W y \geq r, y \geq 0, y_j \geq 1 \). For binary problems it is customary to include the bound constraint \(-y \geq -1\) in \( W y \geq r \). Observe that in the notation of (2.6, 2.7), the matrices \( G_h \) differ only by one row, since \( W \) is common to both. Since there are only two atoms in the disjunction, it is computationally manageable. Indeed, it is not difficult to see that there is a hierarchy of disjunctions that one may use in developing relaxations of the integer program. Assuming that we have chosen some convenient level within the hierarchy, the index set \( H \) is specified, and we may proceed to obtain convex relaxations of the non-convex set. The idea of using alternative relaxations is also at the heart of the reformulation-linearization technique (RLT) of Sherali and Adams [1990].

The following result is known as the disjunctive cut principle. The forward part of this theorem is due to Balas [1975], and the converse is due to Blair and Jeroslow [1978]. In the following, the column vector \( G_{hj} \) denotes the \( j^{th} \) column of the matrix \( G_h \).
Theorem 2.6. Let $S$ and $S_h$ be defined as in (2.6, 2.7) respectively. If $\lambda_h \geq 0$ for all $h \in H$, then
\[
\sum_j \left\{ \max_{h \in H} \lambda_h^T G_{hj} \right\} y_j \geq \min_{h \in H} \lambda_h^T r_h
\] (2.8)
is a valid inequality for $S$. Conversely, suppose that $\pi^T y \geq \pi_0$ is a valid inequality, and $H^* = \{ h \in H | S_h \neq \emptyset \}$. Then there exist nonnegative vectors $\{ \lambda_h \}_{h \in H^*}$ such that
\[
\pi_j \geq \max_{h \in H^*} \lambda_h^T G_{hj}, \quad \text{and} \quad \pi_0 \leq \min_{h \in H^*} \lambda_h^T r_h.
\] (2.9)

Armed with this characterization of valid inequalities for the disjunctive set $S$, we can develop a variety of relaxations of a mixed-integer linear program. The quality of the relaxations will, of course, depend on the choice of disjunction used, and the subset of valid inequalities used in the approximation. In the process of solving a MIP, suppose that we have obtained a solution to some linear relaxation, and assuming that the solution is fractional, we wish to separate it from the set of IP solutions using a valid inequality. Using one or more of the fractional variables to define $H$, we can state a disjunction such that the IP solutions are a subset of $S = \bigcup_{h \in H} S_h$. Theorem 2.6 is useful for developing convexifications of the feasible mixed-integer solutions of the second-stage MIP.

The strongest (deepest) inequalities that one can derive are those that yield the closure of the convex hull of $S$, denoted $\text{clconv}(S)$. The following result of Balas [1979] provides an important characterization of the facets of $\text{clconv}(S)$.

Theorem 2.7. Let the reverse polar of $S$, denoted $S^\#$, be defined as
\[
S^\# = \{ (\pi, \pi_0) | \text{there are nonnegative vectors} \{ \lambda_h \}_{h \in H} \text{ such that (2.9) is satisfied} \}.
\]

When $\pi_0$ is fixed, we denote the reverse polar by $S^\#(\pi_0)$. Assume that $S$ is full dimensional and $S_h \neq \emptyset$ for all $h \in H$. An inequality $\pi^T y \geq \pi_0$ with $\pi_0 \neq 0$ is a facet of $\text{clconv}(S)$ if and only if $(\pi, \pi_0)$ is an extreme point of $S^\#(\pi_0)$. Furthermore, if $\pi^T y \geq 0$ is a facet of $\text{clconv}(S)$ then $(\pi, \pi_0)$ is an extreme direction of $S^\#(\pi_0)$ for all $\pi_0$.

Balas [1979] observes that for $\pi \neq 0$, if $(\pi, 0)$ is an extreme direction of $S^\#$, then $\pi^T y \geq 0$ is either a facet of $\text{clconv}(S)$ or there exist two facets $\pi_1^T y \geq \pi_1^0$ and $\pi_2^T y \geq \pi_2^0$ such that $\pi = \pi_1 + \pi_2$ and $\pi_1^0 + \pi_2^0 = 0$. In any event, Theorem 2.7 provides access to a sufficiently rich collection of valid inequalities to the permit $\text{clconv}(S)$ to be obtained algorithmically. The
notion of reverse polars will be extensively used in section 3 to develop convexifications of certain non-convex functions, including price functions resulting from B&B methods for the second-stage.

In studying the behavior of sequential cutting plane methods, it is important to recognize that without appropriate safeguards, one may not, in fact, recover the convex hull of the set of feasible integer points (see Jeroslow [1980], Sen and Sherali [1985]). In such cases, the cutting plane method may not converge. We maintain however, that this is essentially a theoretical concern since practical schemes use cutting planes in conjunction with a B&B method, which are of course finitely convergent.

Before closing this section, we discuss a certain special class of disjunctions for which sequential convexification (one variable at a time) does yield the requisite closure of the convex hull of integer feasible points. This class of disjunctions gives rise to facial disjunction sets, which are described next.

A disjunctive set in conjunctive normal form may be stated in the form

\[ S = Y \cap \bigcup_{j \in J} D_j, \]

where \( Y \) is a polyhedron, \( J \) is a finite index set, and each set \( D_j \) is defined by the union of finitely many halfspaces. The set \( S \) is said to possess the facial property for each \( j \), every hyperplane used in the definition of \( D_j \) contains some face of \( Y \). It is not difficult to see that a 0-1 MIP is a facial disjunctive program. For these problems \( Y \) is a polyhedral set that includes the “box” constraints \( 0 \leq y_j \leq 1, j \in J_2 \), and the disjunctive sets \( D_j \) are defined as follows.

\[ D_j = \{ y \, | \, y_j \leq 0 \} \cup \{ y \, | \, y_j \geq 1 \}. \]

Balas [1979] has shown that for sets with the facial property, one can recover the set \( \text{clconv}(S) \) by generating a sequence of convex hulls recursively. Let \( j_1, j_2, \ldots, \) etc. denote the indices of \( J_2 \), and initialize \( j_0 = 0, Q_0 = Y \). Then

\[ Q_{jk} = \text{clconv}(Q_{jk-1} \cap D_{jk}), \quad (2.10) \]

and the final convex hull operation yields \( \text{clconv}(S) \). Thus for a facial disjunctive program, the complete convexification can be obtained by convexifying the set by using disjunctions one variable at a time. As shown in Sen and Higle [2000], this result provides the basis for the convergence of the convex hull of second-stage feasible (mixed-binary) solutions using sequential convexification.
3. Decomposition Algorithms for Two-stage SMIP: Stagewise Decomposition

In this section, we study various classes of two-stage SMIP problems for which stagewise (resource-directive) decomposition algorithms appear to be quite appropriate. Recall that we have chosen to focus on the case of two-stage problems with integer recourse (in the second-stage). Our presentation excludes SMIP models in which the recourse function is defined using the LP value function. This is not to suggest that these problems (with integer first-stage, and continuous second-stage) are well solved. Significant challenges do remain, although they are mainly computational. For instance, the stochastic B&B method of Norkin, Ermoliev and Ruszczynski [1998] raises several interesting questions, especially those regarding its relationship with machine learning. By the same token, computational studies (e.g. Verweij et al [2003]) for this class of problems are of great importance. However, such an excursion would detract from our mission to foster a deeper understanding of the challenges associated with integer recourse models.

Much of this presentation revolves around convexification of the value functions of the second-stage IP. This section is divided into the following subsections.

- Simple Integer Recourse Models with Random RHS
- Binary First-stage, Arbitrary Second-stage
- Binary First-stage, 0-1 MIP Second-stage with Fixed Recourse
- Binary First-stage, MIP Second-stage
- Continuous First-stage, Integer Second-stage and Fixed Tenders
- 0-1 MIP in Both Stages with General Random Data

The heading for the subsections below indicate the above classification, and the subheadings identify the solution approach discussed in that subsection.

Simple Integer Recourse Models with Random RHS: Connections with the Continuum

The Simple Integer Recourse (SIR) model is the pure integer analog of the continuous simple recourse model. Unlike the continuous version of the simple recourse model, this version is intended for “news-vendor”-type models of “large-ticket” items. This class of models introduced by Louveaux and Van der Vlerk [1993], has been studied extensively in a series of papers by Klein Haneveld, Stougie and Van der Vlerk [1995, 1996]. We assume that all data elements except the right-hand side are fixed, and that the matrix $T$ has full row rank. Moreover, assume that $g_i^T, g_i^- > 0, i = 1, \ldots, m_2$. Let $r_i(\omega)$ and $t_i$ denote the $i^{th}$ row of $r(\omega)$ and $T$ respectively, and let $x_i = t_i x$. Moreover, define a scalar function
\[ v^+ = \max\{0, [v]\} \quad \text{and} \quad [v]^-= \max\{0, -[v]\}. \]

Then the statement of the SIR model is as follows.

\[
\begin{align*}
\min_{x \in X \cap X} & \left\{ c^T x + \sum_i E\left[ g_i^+ \left[ r_i(\tilde{\omega}) - \chi_i \right] + g_i^- \left[ r_i(\tilde{\omega}) - \chi_i \right] \right] | x = T x \right\}.
\end{align*}
\]  

(3.1)

This relatively simple problem provides a glimpse at some of the difficulties associated with SMIP problems in general. Under the assumptions specified earlier, Klein Haneveld, Stougie and Van der Vlerk [1995, 1996] have shown that whenever \( r_i(\tilde{\omega}) \) has finite support, and \( T \) has full row-rank, it is possible to compute the convex hull of the expected recourse function by using enumeration over each dimension \( i \). We describe this procedure below. However, it is important to note that since the set \( X \cap X \) will not be used in the convexification process, the resulting optimization problem will only provide a lower bound. Further B&B search may be necessary to close the gap.

The expected recourse function in (3.1) has an extremely important property which relates it to its continuous counterpart. Let the \( i^{th} \) component of the expected recourse function of the continuous counterpart be denoted \( R_i(\chi_i) \), and the \( i^{th} \) component of the expected recourse function in (3.1) be denoted \( \hat{R}_i(\chi_i) \). That is,

\[
\hat{R}_i(\chi_i) = E\left[ g_i^+ \left[ r_i(\tilde{\omega}) - \chi_i \right] + g_i^- \left[ r_i(\tilde{\omega}) - \chi_i \right] \right].
\]

Then,

\[
R_i(\chi_i) \leq \hat{R}_i(\chi_i) \leq R_i(\chi_i) + \max\{g_i^+, g_i^-\}.
\]  

(3.2)

The next result (also proved by Klein Haneveld, Stougie and Van der Vlerk [1995, 1996]) is very interesting.

**Theorem 3.1.** Let \( \hat{R}_i^c \) denote any convex function that satisfies (3.2), and let \( (\hat{R}_i^c)'_+ \) denote its right directional derivative. Then, for \( a \in \mathbb{R} \)

\[
P_i(a) = \frac{(\hat{R}_i^c)'_+(a) + g_i^+}{g_i^+ + g_i^-}
\]

is a cumulative distribution function (cdf). Moreover, if \( \vartheta_i \) is a random variable with cdf \( P_i \), then for all \( \chi_i \in \mathbb{R} \),

\[
\hat{R}_i^c(\chi_i) = g_i^+ E\left[ (\vartheta_i - \chi_i)^+ \right] + g_i^- E\left[ (\chi_i - \vartheta_i)^+ \right] + \frac{g_i^+ c_i^+ + g_i^- c_i^-}{g_i^+ + g_i^-}.
\]  

(3.3)
where \((v)^+ = \max\{0, v\}\), and \(c^+_i, c^-_i\) are asymptotic discrepancies between \(\hat{R}_i\) and \(R_i\) defined as follows:

\[
c^+_i = \lim_{x_i \to \infty} \hat{R}_i(x_i) - R(x_i), \quad \text{and} \quad c^-_i = \lim_{x_i \to -\infty} \hat{R}_i(x_i) - R(x_i).
\]

Note that unlike (3.1), the expectations in (3.3) do not include any ceiling/floor functions. Hence it is clear that if we are able to identify random variables \(\theta_i\) with cdf \(P_i\), then, we may use the continuous counterpart to obtain a tight approximation of the SIR model.

In order to develop the requisite cdf, the authors construct a convex function by creating the convex hull of \(\hat{R}_i\). In order to do so, assume that \(r_i(\omega)\) has finite support \(\Omega = \{\omega^1, \ldots, \omega^N\}\). Then, the points of discontinuity of \(\hat{R}_i\) can be characterized as \(\cup_{\omega \in \Omega} \{r_i(\omega) + \mathbb{Z}\}\), where \(\mathbb{Z}\) denotes the set of integers. Moreover, \(\hat{R}_i\) is constant in between the points of discontinuity. Consequently, the convex hull of \(\hat{R}_i\) can be obtained by using the convex hull of \((\chi_i, \hat{R}_i(\chi_i))\) at finitely many points of discontinuity. This convex hull (in two-space) can be constructed by adopting a method called Graham scan. This method works by first considering a piecewise linear function that joins the points of discontinuity \((\chi_i, \hat{R}_i(\chi_i))\), and then verifying whether the right directional derivative at a point is greater than the left directional derivative at that point, for only such points can belong to the boundary of the convex hull. Proceeding in this manner, the method constructs the convex hull, and hence the function \(\hat{R}_i\). Thereafter, the optimization of a continuous simple recourse problem may be undertaken. This procedure then provides a good lower bound to the optimal value of the SIR model. It is important to bear in mind that there is one additional assumption necessary; the matrix \(T\) must have full rank so that the convex hull of the \((m_2\text{-dimensional})\) expected recourse function may be obtained by adding all of the elements \(\hat{R}_i^j, i = 1, \ldots, m_2\). This lower bounding scheme may also be incorporated within a B&B procedure to find an optimal solution to the problem.

**Binary First-stage, Arbitrary Second-stage: First-stage cuts**

For SMIP problems studied in this subsection, we use \(X = \mathcal{B}\) (binary vectors) in (1.1,1.2). Laporte and Louveaux [1993] provide valid inequalities that can be applied to a wide class of expected recourse functions, so long as the first-stage decisions are binary. In particular, the second-stage problems admissible under this scheme include all optimization problems that have a known lower bound on expected recourse function. As one might expect, such widely applicable cuts rely mainly on the fact that the first-stage decisions are binary. The algorithmic setting within which the inequalities of Laporte and Louveaux [1993] are used follows the basic outline of Benders’ decomposition (or L-shaped method). That is, at each iteration \(k\), we solve one master program, and as many subproblems as there are outcomes of the random
variable. Interestingly, despite the non-convexity of value functions of general optimization problems (including MIPs), the valid inequality provided by Laporte and Louveaux [1993] is linear. As shown in the development below, the linearity derives from a property of the binary first-stage variables.

At iteration \( k \), let the first-stage decision \( x^k \) be given, and let

\[
I_k = \{ i | x_i^k = 1 \}, \quad Z_k = \{ 1, \ldots, n_1 \} - I_k.
\]

Next define the linear function

\[
\delta_k(x) = |I_k| - \left[ \sum_{i \in I_k} x_i - \sum_{i \in Z_k} x_i \right].
\]

It can be easily seen that when \( x = x^k \) (assumed binary), \( \delta_k(x) = 0 \); whereas, for all other binary vectors \( x \neq x^k \), at least one of the components must switch “states.” Hence for \( x \neq x^k \), we have

\[
\left[ \sum_{i \in I_k} x_i - \sum_{i \in Z_k} x_i \right] \leq |I_k| - 1, \quad \text{i.e. } \delta_k(x) \geq 1.
\]

Next suppose that a lower bound on the expected recourse function, denoted \( \tilde{h}_\ell \), is available. Let \( \tilde{h}(x^k) \) denote the value of the expected recourse function for a given \( x^k \). If \( \tilde{h}(x^k) = \infty \) (i.e. the second-stage is infeasible), then (3.4a) can be used to delete \( x^k \). On the other hand, if \( \tilde{h}(x^k) \) is finite, then the following inequality is valid.

\[
\eta \geq \tilde{h}(x^k) - \delta_k(x)[\tilde{h}(x^k) - \tilde{h}_\ell]. \tag{3.4b}
\]

This is the “optimality” cut of Laporte and Louveaux [1993]. To verify its validity, observe that when \( x = x^k \), the second term in (3.4b) vanishes, and hence the master program recovers the value of the expected recourse function. On the other hand, if \( x \neq x^k \), then,

\[
\delta_k(x)[\tilde{h}(x^k) - \tilde{h}_\ell] \geq [\tilde{h}(x^k) - \tilde{h}_\ell].
\]

Hence, for all \( x \neq x^k \), the right-hand side of (3.4b) obeys

\[
\tilde{h}(x^k) - \delta_k(x)[\tilde{h}(x^k) - \tilde{h}_\ell] \leq \tilde{h}(x^k) - \tilde{h}(x^k) + \tilde{h}_\ell = \tilde{h}_\ell.
\]

It is interesting to observe that the structure of the second-stage is not critical to the validity of the cut. For the sake of expositional simplicity,
we state the algorithm of Laporte and Louveaux [1993] under the complete recourse assumption, thus requiring only (3.4b). If this assumption is not satisfied, then one would also include (3.4a) in the algorithmic process. In the following, \( \bar{x} \) denotes an incumbent, \( f \) its objective value, and \( f_u, f_u' \) are lower and upper bounds, respectively, on the entire objective function. We use the notation \( \alpha + \beta \bar{x} \) to denote the right-hand side of (3.4b).

First-Stage Cuts for SP with Binary First Stage

0. Initialize. \( k \leftarrow 0 \) Let \( \epsilon \geq 0, x^1 \in X \cap \mathcal{B} \) and \( \tilde{h}_\ell \) (a lower bound on the expected recourse function) be given. Define \( \eta_0(x) = \tilde{h}_\ell \), \( f_u = \infty \).

1. Obtain a Cut
   \( k \leftarrow k + 1 \). Evaluate the second-stage objective value \( \tilde{h}(x^k) \).
   Use (3.4b) to define the cut \( \alpha + \beta \bar{x} \).

2. Update the Piecewise Linear Approx.
   (a) Define \( \eta_k(x) = \max\{\eta_{k-1}(x), \alpha + \beta \bar{x}\} \), and \( f_k(x) = c^T x + \eta_k(x) \).
   (b) Update the upper bound (if possible): \( f_u \leftarrow \min\{f_u, f_k(x^k)\} \). If a new upper bound is obtained, \( \bar{x} \leftarrow x^k, f_u \leftarrow f_u \).

3. Solve the Master Problem. Let \( x^{k+1} \in \arg\min \{f_k(x) \mid x \in X \cap \mathcal{B}\} \).

4. Stopping Rule. \( f_\ell = f_k(x^{k+1}) \). If \( f_u - f_\ell \leq \epsilon \), declare \( \bar{x} \) as an \( \epsilon \)-optimum and stop. Otherwise, repeat from 1.

The above algorithm has been stated in a manner that mimics the Kelley-type methods of convex programming (Kelley [1960]) since the L-shaped method of Van Slyke and Wets [1969] is a method of this type. The main distinctions are in step 1 (cut formation), and step 3 (the solution of the master problem) which requires the solution of a binary IP. We note however that there are various other ways to implement these cuts. For instance, if the solution method adopted for the master program is a B&B method, then one can generate a cut at any node (of the B&B tree) at which a binary solution is encountered. Such an implementation would have the benefit of generating cuts during the B&B process at the cost of carrying out multiple evaluations of the second-stage objective during the B&B process. We close this subsection with an illustration of this scheme.
Example 3.2. Consider the following two-stage problem

\[
\begin{align*}
\text{Min} & \quad x_1 + 0.25(-2y_1(1) + 4y_2(1)) + 0.75(-2y_1(2) + 4y_2(2)) \\
& \quad -3x_1 - 3y_1(1) + 2y_2(1) \geq -4 \\
& \quad -5x_1 - 3y_1(2) + 2y_2(2) \geq -8 \\
& \quad x_1, y_1(1), y_1(2) \in [0, 1], y_2(1), y_2(2) \geq 0.
\end{align*}
\]

To maintain notational simplicity in this example, we simply use \( \omega = \{1, 2\} \), instead of our regular notation of \( \{\omega^1, \omega^2\} \). From the above data, it is easily seen that 

\[
-2y_1 + 4y_2 \geq -2 \text{ for } y_1 \in [0, 1] \text{ and } y_2 \geq 0.
\]

Hence \( \hat{h}_e = -2 \) is a valid lower bound for the second-stage problems.

0. **Initialization.** \( k = 0 \), and let \( \epsilon = 0 \), \( x_1^1 = 0 \), \( \hat{h}_e = -2 \), \( f_u = \infty \), \( \eta_0(x) = -2 \).

**Iteration 1**

1. **Obtain a cut.** For the given \( x_1^1 \), we solve each second-stage MIP subproblem. We get \( y_1(1) = 1 \), \( y_2(1) = 0 \), \( y_1(2) = 1 \), \( y_2(2) = 0 \), and \( \hat{h}(x_1^1) = -2 \). Moreover, \( \delta(x_1) = x_1 \), so that the cut is \( \eta \geq -2 - (x_1)(-2 + 2) = -2 \).

2. **Update the Piecewise Linear Approximation.** The upper bound is 

\[
\hat{f} = \text{Min}\{\infty, 0 + f(0)\} = -2.
\]

The incumbent is \( \bar{x}_1 = 0, f = -2 \).

3. **Solve the Master Program.**

\[
\text{Min}\{-x_1 + \eta \mid \eta \geq -2, x_1 \in [0, 1]\}.
\]

\( x_1^2 = 1 \) solves this problem, and the lower bound \( f_e = -3 \).

4. **Stopping Rule.** Since \( f_u - f_e > 0 \), repeat from step 1.

**Iteration 2**

1. **Obtain a cut.** For \( x_1^2 = 1 \) solve each second-stage MIP subproblem. We get \( y_1(1) = 0 \), \( y_2(1) = 1 \), \( y_2(2) = 0 \), yielding \( \hat{h}(x_1^2) = -1.5 \). Now, \( \delta(x_1) = 1 - x_1 \), and the cut is \( \eta \geq -1.5 - (1 - x_1)(-1.5 + 2) = -2 + 0.5x_1 \).

2. **Update the Piecewise Linear Approximation.** The upper bound is 

\[
\hat{f} = \text{Min}\{-2, -1 - 1.5\} = -2.5, \text{ hence, } \bar{x}_1 = 1, \hat{f} = -2.5 \).

3. **Solve the Master Program.**

\[
\text{Min}\{-x_1 + \eta \mid \eta \geq -2, \eta \geq -2 + 0.5x_1, x_1 \in [0, 1]\}.
\]

\( x_1^3 = 1 \) solves this problem, and the lower bound \( f_e = -2.5 \).

4. **Stopping Rule.** Since \( f_u - f_e = 0 \), the method stops with \( \bar{x}_1 = 1 \) as the optimal solution.
As in this example, all $2^n_1$ valid inequalities may be generated in the worst case (where $n_1$ is the number of first-stage binary variables). However, the finiteness of the method is obvious.

**Binary First-stage, 0-1 MIP Second-stage with Fixed Recourse:**

Cuts in both stages

In this subsection we impose the following structure on (1.1,1.2): a fixed recourse matrix, binary first-stage variables, and mixed-integer (binary) recourse decisions. The methodology here is one of sequential convexification of the integer recourse problem. The main motivation for sequential convexification is to avoid the need to solve every subproblem from scratch in each iteration. These procedures will be presented in the context of algorithms that operate within the framework of Benders’ decomposition, as in the previous subsection; that is, in iteration $k$, a first-stage decision, denoted $x^K$, is provided to the subproblems, which in turn returns an inequality that provides a linear approximation of the expected recourse function. The cuts derived here use disjunctive programming. This approach has been used to solve some rather large server location problem, and the computational results reported in Ntiamo and Sen [2004] are encouraging. Cuts for this class of models can also be derived using the RLT framework, and has appeared in the work of Sherali and Fraticelli [2002].

We start this development with the assumption that by using appropriately penalized continuous variables, the subproblem remains feasible for any restriction of the integer variables $y_j$, $j \in J_2$. Let $x^K$ be given, and suppose that matrices $W_k$, $T_k(\omega)$ and $r_k(\omega)$ are given. Initially (i.e. $k = 1$) these matrices are simply $W$, $T(\omega)$ and $r(\omega)$, and recall that in our notation, we include the constraints $-y_j \geq -1, j \in J_2$ explicitly in $W y \geq r(\omega) - T(\omega)x$. (Similarly, the constraint $-x \geq 1$ is also included in the constraints $x \in X$.) During the course of solving the 0-1 MIP subproblem for outcome $\omega$, suppose that we happen to solve the following LP relaxation.

$$
\begin{align*}
\text{Min} & \quad g^\top y \\
\text{s.t.} & \quad W_k y \geq r_k(\omega) - T_k(\omega)x \\
& \quad y \in \mathbb{R}^{n_2}_+.
\end{align*}
$$

Whenever the solution to this problem is fractional, we will be able to derive a valid inequality that can be used in all subsequent iterations. Let $y^K(\omega)$ denote a solution to (3.5), and let $j(k)$ denote an index $j \in J_2$ for which $y^K_j(\omega)$ is non-integer for one or more $\omega \in \Omega$. To eliminate this non-integer solution, a disjunction of the following form may be used:

$$
S_k(x^K, \omega) = S_{0,j(k)}(x^K, \omega) \cup S_{1,j(k)}(x^K, \omega),
$$
where
\[ S_{0,j(k)}(x^k, \omega) = \{ y \in \mathbb{R}^n_+ \mid W_k y \geq r_k(\omega) - T_k(\omega)x^k, -y_{j(k)} \geq 0 \} \] (3.6a)
\[ S_{1,j(k)}(x^k, \omega) = \{ y \in \mathbb{R}^n_+ \mid W_k y \geq r_k(\omega) - T_k(\omega)x^k, y_{j(k)} \geq 1 \}. \] (3.6b)

The index \( j(k) \) is referred to as the “disjunction variable” for iteration \( k \). This is precisely the disjunction used in the lift-and-project cuts of Balas, Ceria and Cornuéjols [1993]. To connect this development with the subsection on disjunctive cuts, we observe that \( H = \{0, 1\} \). We assume that the subproblems remain feasible for any restriction of the integer variables, and thus both (3.6a) and (3.6b) are non-empty.

Let \( \lambda_{0,1} \) denote the vector of multipliers associated with the rows of \( W_k \) in (3.6a), and \( \lambda_{0,2} \) denote the scalar multiplier associated with the fixed variable \( y_{j(k)} \) in (3.6a). Let \( \lambda_{1,1} \) and \( \lambda_{1,2} \) be similarly defined for (3.6b). Then Theorem (2.6) implies that if \( \pi, \pi_0(\omega), \omega \in \Omega \) satisfy (3.7), then \( \pi^T y \geq \pi_0(\omega) \) is a valid inequality for \( S_k(x^k, \omega) \).

\[ \pi_j \geq \lambda^T_{0,1} W_{jk} - \bar{f}_j \lambda_{0,2} \ \forall j \] (3.7a)
\[ \pi_j \geq \lambda^T_{1,1} W_{jk} + \bar{f}_j \lambda_{1,2} \ \forall j \] (3.7b)
\[ \pi_0(\omega) \leq \lambda^T_{0,1} (r_k(\omega) - T_k(\omega)x^k) \ \forall \omega \in \Omega \] (3.7c)
\[ \pi_0(\omega) \leq \lambda^T_{1,1} (r_k(\omega) - T_k(\omega)x^k) + \lambda_{1,2} \ \forall \omega \in \Omega \] (3.7d)
\[ -1 \leq \pi_j \leq 1, \ \forall j, -1 \leq \pi_0(\omega) \leq 1, \ \forall \omega \in \Omega \] (3.7e)
\[ \lambda_{0,1}, \lambda_{0,2}, \lambda_{1,1}, \lambda_{1,2} \geq 0 \] (3.7f)

where
\[ \bar{f}_j = \begin{cases} 0, & \text{if } j \neq j(k) \\ 1, & \text{otherwise}. \end{cases} \]

**Remark 3.3.** Several objectives have been proposed in the disjunctive programming literature for choosing cut coefficients (Sherali and Shetty [1980]). One possibility for SMIP problems is to maximize the expected value of the depth of cut: \( E[\pi_0(\omega)] - E[\bar{f}^k(\omega)]\pi \). We should note that the optimal
objective value of the resulting LP can be zero, which implies that the inequality generated by the LP does not delete some of the fractional points \( y^k(\omega), \omega \in \Omega_k \). Here \( \Omega_k \) denotes those \( \omega \in \Omega \) for which \( y^k(\omega) \) does not satisfy mixed-integer feasibility. So long as the cut deletes a fractional \( y^k(\omega) \) for some \( \omega \), we may proceed with the algorithm. However, if we obtain an inequality such that \( (\pi^k)^\top y^k(\omega) > \pi_0^k y^h(\omega) \), for all \( \omega \in \Omega_k \), then one such outcome should be removed from the expectation operation \( \mathbb{E}[y^k(\tilde{\omega})] \), and this vector should be replaced by a conditional expectation over the remaining vectors \( y^k(\omega) \). Since the rest of the LP remains unaltered, the re-optimization should be carried out using a “warm start.” Other objective functions can also be used for the cut generation process. For instance, we could maximize the function \( \min_{\omega \in \Omega} \pi_0(\omega) - y^k(\omega)^\top \pi \).

For vectors \( x \neq x^k \), the cut may need to be modified in order to maintain its validity. Sen and Higle [2000] show that for any other \( x \), one only needs to modify the right-hand side scalar \( \pi_0^k \); in other words, the vector \( \pi^k \) provides valid cut coefficients as long as the recourse matrix is fixed. This result, known as the Common Cut Coefficients (C3) Theorem, was proven in Sen and Higle [2000], and a general version may be stated as follows.

**Theorem 3.4. (The C3 Theorem).** Consider a 0-1 SMIP with a fixed recourse matrix. For \( (x, \omega) \in X \times \Omega \), let \( Y(x, \omega) = \{ y \in \mathbb{R}^n_+ \mid W y \geq r(\omega) - T(\omega)x, y_j \in \{0, 1\}, j \in J_2 \} \), the set of mixed-integer feasible solutions for the second-stage mixed-integer linear program. Suppose that \( \{C_h, d_h\}_{h \in H} \) is a finite collection of appropriately dimensioned matrices and vectors such that for all \( x, \omega \in X \times \Omega \)

\[
Y(x, \omega) \subseteq \bigcup_{h \in H} \{ y \in \mathbb{R}^n_+ \mid C_h y \geq d_h \}.
\]

Let

\[
S_h(x, \omega) = \{ y \in \mathbb{R}^n_+ \mid W y \geq r(\omega) - T(\omega)x, C_h y \geq d_h \},
\]

and let

\[
S = \bigcup_{h \in H} S_h(x, \omega).
\]

Let \( (x, \tilde{\omega}) \) be given, and suppose that \( S_h(x, \tilde{\omega}) \) is nonempty for all \( h \in H \) and \( \pi^\top y \geq \pi_0(x, \tilde{\omega}) \) is a valid inequality for \( S(x, \tilde{\omega}) \). There exists a function \( \pi_0 : X \times \Omega \rightarrow \mathbb{R} \) such that for all \( (x, \omega) \in X \times \Omega \), \( \pi^\top y \geq \pi_0(x, \omega) \) is a valid inequality for \( S(x, \omega) \).
Although the above theorem is stated for general disjunctions indexed by $H$, we only use $H = \{0, 1\}$ in this development. The LP used to obtain the common cut coefficients is known as the $C_3$LP, and its solution $(\pi_k)^T$ is appended to $W_k$ in order to obtain $W_{k+1}$. In order to be able to use these coefficients in subsequent iterations, we will also calculate a new row to append to $T_k(\omega)$, and $r_k(\omega)$ respectively. These new rows will be obtained by solving some other LPs, which we will refer to as RHS-LPs. These calculations are summarized next.

Let $\lambda_{0,1}^k, \lambda_{0,2}^k, \lambda_{1,1}^k, \lambda_{1,2}^k \geq 0$ denote the values obtained from $C_3$LP in iteration $k$. Since these multipliers are non-negative, Theorem 2.6 allows us to use these multipliers for any choice of $(x, \omega)$. Hence by using these multipliers, the right-hand side function $\pi_0(x, \omega)$ can be written as

$$\pi_0(x, \omega) = \text{Min} \left\{ (\lambda_{0,1}^k)^T r_k(\omega) - (\lambda_{0,1}^k)^T T_k(\omega)x, (\lambda_{1,1}^k)^T r_k(\omega) + \lambda_{1,2}^k - (\lambda_{1,1}^k)^T T_k(\omega)x \right\}.$$ 

For notational convenience, we put

$$\bar{v}_0(\omega) = (\lambda_{0,1}^k)^T r_k(\omega), \quad \bar{v}_1(\omega) = (\lambda_{1,1}^k)^T r_k(\omega) + \lambda_{1,2}^k$$

and

$$\left[ \tilde{y}_h(\omega) \right]^T = (\lambda_{h,1}^k)^T T_k(\omega), \quad h \in \{0, 1\},$$

so that

$$\pi_0(x, \omega) = \text{Min} \left\{ \bar{v}_0(\omega) - \left[ \tilde{y}_0(\omega) \right]^T x, \bar{v}_1(\omega) - \left[ \tilde{y}_1(\omega) \right]^T x \right\}.$$ 

Being the minimum of two affine functions, the epigraph of $\pi_0(x, \omega)$ can be represented as the union of the two half-spaces. Hence the epigraph of $\pi_0(x, \omega)$, restricted to the set $X$ will be denoted as $\Pi_X(\omega)$, and represented as

$$\Pi_X(\omega) = \bigcup_{h \in H} E_h(\omega),$$

where $H = \{0, 1\}$ and

$$E_h(\omega) = \{(\eta, x) | \eta \geq \bar{v}_h(\omega) - \tilde{y}_h(\omega)^T x, x \in X\}. \quad (3.8)$$
Here $X = \{ x \in \mathbb{R}^n \mid Ax \geq b, x \geq 0 \}$, and we assume that the inequality $-x \geq -1$ is included in the constraints $Ax \geq b$. It follows that the closure of the convex hull of $\Pi_{\Lambda}(\omega)$ provides the appropriate convexification of $\pi_0(x, \omega)$. This computational procedure is discussed next.

In the following, we assume that for all $x \in X, \eta \geq 0$ in (3.8). As long as $X$ is bounded, there is no loss of generality with this assumption, because the epigraph can be translated to ensure that $\eta \geq 0$. Analogous to the concept of reverse polars (see Theorem 2.7), Sen and Higle [2000] define the epi-reverse polar, denoted $\Pi_{\Lambda}^*(\omega)$, as

$$
\Pi_{\Lambda}^*(\omega) = \{ \sigma_0(\omega) \in \mathbb{R}, \sigma(\omega) \in \mathbb{R}^n, \zeta(\omega) \in \mathbb{R} \text{ such that} \}
$$

for $h = 0, 1$, $\exists \tau_h \in \mathbb{R}^n, \tau_{0h} \in \mathbb{R}$

$$
\sigma_0(\omega) \geq \tau_{0h} \forall h \in \{0, 1\}
$$

$$
\sum_h \tau_{0h} = 1
$$

$$
\sigma(\omega) \geq \tau_h^T A_j + \tau_{0h} \hat{y}_h(\omega) \quad \forall h \in \{0, 1\}; j = 1, \ldots, n
$$

$$
\zeta(\omega) \leq \tau_h^T b + \tau_{0h} \tilde{y}_h(\omega) \quad \forall h \in \{0, 1\}
$$

$$
\tau_h \geq 0, \tau_{0h} \geq 0, \quad h \in \{0, 1\}.
$$

The term “epi-reverse polar” is intended to indicate that we are using the reverse polar of an epigraph to characterize its convex hull (see Theorem 2.7). Note that the epi-reverse polar allows only those facets of the closure of the convex hull of $\Pi_{\Lambda}(\omega)$ that have a positive coefficient for the variable $\eta$. From Theorem 2.7, we can obtain all necessary facets of the closure of the convex hull of $\pi_0(x, \omega)$. We can derive one such facet by solving the following problem, which we refer to as the RHS-LP($\omega$).

$$
\text{Max} \quad \zeta(\omega) - \sigma_0(\omega) - (\chi^k)^T \sigma(\omega)
$$

$$
\text{s.t.} \quad (\sigma_0(\omega), \sigma(\omega), \zeta(\omega)) \in \Pi_{\Lambda}^*(\omega).
$$

With an optimal solution to (3.9), $(\sigma^k(\hat{\omega}), \sigma_0^k(\hat{\omega}), \chi^k(\hat{\omega}))$, we obtain $\chi^k(\omega) = \frac{\zeta^k(\omega)}{\sigma_0^k(\omega)}$ and $\gamma^k(\omega) = \frac{\sigma^k(\omega)}{\sigma_0^k(\omega)}$. For each $\omega \in \Omega$, these coefficients are used to update the right-hand-side functions $r_{k+1}(\omega) = [r_k(\omega)^T, \chi^k(\omega)]^T$, and $T_{k+1}(\omega) = [T_k(\omega)^T; \gamma^k(\omega)]^T$.

One can summarize a cutting plane method of the form presented in the previous subsection by replacing step 1 of that method by a new version of step 1 as summarized below. Sen and Higle [2000] provide a proof of convergence of convex hull approximations based on an extension of (2.10). We caution however that as with any cutting plane method, its full benefits can only be realized when it is incorporated...
within a B&B method. Such a branch-and-cut approach is discussed in the following subsection.

### Deriving Cuts for Both Stages

1. **Obtain a Cut**
   \[ k \leftarrow k + 1. \]
   
   (a) (Solve the LP relaxation for all \( \omega \)). Given \( x^k \), solve the LP relaxation of each subproblem, \( \omega \in \Omega \).
   
   (b) (Solve \( C^k \)-LP). Optimize some objective from Remark 3.3, over the set in (3.7). Append the solution \((\pi^k)^T\) to the matrix \( W_k \) to obtain \( W_{k+1} \).
   
   (c) (Solve RHS-LP(\( \omega \)) for all \( \omega \)). Solve (3.9) for all \( \omega \in \Omega \), and derive \( r_{k+1}(\omega), T_{k+1}(\omega) \).
   
   (d) (Solve an enhanced LP relaxation for all \( \omega \)). Using the updated matrices \( W_{k+1}, r_{k+1}(\omega), T_{k+1}(\omega) \), solve an LP relaxation for each \( \omega \in \Omega \).
   
   (e) (Benders’ Cut). Using the dual multipliers from step (d), derive a Benders’ cut denoted \( \alpha + \beta x \).

---

**Example 3.5.** The instance considered here is the same as that in Example 3.2. While this example illustrates the process of cut formation, it is too small to really demonstrate the benefits that might accrue from adding cuts into the subproblem. A slightly larger instance (motivated by the example in Schultz, Stougie and Van der Vlerk [1998]) which requires a few more iterations, and one that demonstrates the advantages of stronger LP relaxations appears in Sen, Higle and Ntaimo [2002], and Ntaimo and Sen [2004]. As in Example 3.2, we use \( \omega = \{1, 2\} \).

**Iteration 1**

The LP relaxation of the subproblem in iteration 1 (see Example 3.2) provides integer optimal solutions. Hence, for its iteration, we use the cut obtained in Example 3.2 (without using the Benders’ cut). In this case, the calculations of this iteration mimic those for iteration 1 in Example 3.2. The resulting value of \( x_1 \) is \( x_1^1 = 1 \).

**Iteration 2**

In the following, elements of the vector \( \lambda_{01} \) will be denoted \( \lambda_{011} \) and \( \lambda_{012} \). Similarly, elements of \( \lambda_{11} \) will be denoted \( \lambda_{111} \) and \( \lambda_{112} \).

1. **Derive cuts for both stages.**
1a) Putting $x_i^2 = 1$, solve the LP relaxation of the subproblems for $\omega = 1, 2$. For $\omega = 1$, we get $y_{1}^{(1)} = 1/3$ and $y_{2}^{(1)} = 0$; similarly for $\omega = 2$, we get $y_{1}^{(2)} = 1$ and $y_{2}^{(2)} = 0$.

1b) Solve the $C^{\text{LP}}$ using $E(y_{1}, y_{2}) = (0.833, 0)$.

Max $0.25\pi_{0}(1) + 0.75\pi_{0}(2) - 0.833\pi_{1}$

s.t. $\pi_{1} + 3\lambda_{011} + \lambda_{012} + \lambda_{02} \geq 0$

$\pi_{1} + 3\lambda_{111} + \lambda_{112} - \lambda_{12} \geq 0$

$\pi_{2} - 2\lambda_{011} \geq 0$

$\pi_{2} - 2\lambda_{111} \geq 0$

$\pi_{0}(1) + \lambda_{011} + \lambda_{012} \leq 0$

$\pi_{0}(1) + \lambda_{111} + \lambda_{112} - \lambda_{12} \leq 0$

$\pi_{0}(2) + 3\lambda_{011} + \lambda_{012} \leq 0$

$\pi_{0}(2) + 3\lambda_{111} + \lambda_{112} - \lambda_{12} \leq 0$

$-1 \leq \pi_{j} \leq 1, \forall j, -1 \leq \pi_{0}(\omega) \leq 1, \forall \omega, \lambda \geq 0$.

The optimal objective value of this LP is 0.083, and the cut coefficients are $(\pi_{1}^1, \pi_{1}^2)^{(\omega)} = (-1, 1)$, and the multipliers $\lambda_{01}^{(\omega)} = (0, 0), \lambda_{02} = 1$, whereas, $\lambda_{11}^{(\omega)} = (0.5, 0), \lambda_{12} = 0.5$.

1c) For $H = \{0, 1\}$ we will now compute $\tilde{\nu}_{\omega}(\omega)$ and $\tilde{\gamma}_{\omega}(\omega)$ so that the sets $E_{\omega}(\omega), h \in H$ can be determined for all $\omega$. Thereafter the union of these sets can be convexified using the RHS-LP (3.9). Using the multipliers $\lambda_{01} = (0, 0), \lambda_{02} = 1$, we obtain $\tilde{\nu}_{0}(1) = 0$, and $\tilde{\gamma}_{0}(1) = 0$. Hence

$E_{0}(1) = \{0 \leq x_{1} \leq 1 | \eta \geq 0\}$.

and similarly by using $\lambda_{11} = (0.5, 0), \lambda_{12} = 0.5$ we have

$E_{1}(1) = \{0 \leq x_{1} \leq 1 | \eta \geq -1.5 + 1.5x_{1}\}$.

Clearly, the convex hull of these two sets is $E_{1}(1)$, and the facet can be obtained using linear programming. In the same manner, we obtain

$E_{0}(2) = \{0 \leq x_{1} \leq 1, \eta \geq 0\}$, and

$E_{1}(2) = \{0 \leq x_{1} \leq 1, \eta \geq -3.5 + 2.5x_{1}\}$.

Once again the convex hull of these two sets is $E_{1}(2)$, and the facet can be derived using linear programming. In any event, the matrices are updated as follows: we obtain $W_{2}$ by appending the row $(-1, 1)$ to $W$; $r_{2}(1)$ is obtained by appending the scalar $-1.5$ to $(r_{1}(1))^\top = (-4, -1)$, $r_{2}(2)$ is obtained by appending the
scalar $-3.5$ to $(r_1(2))^\top = (-8, -1)$. Finally we append the “row” 1.5 to $T_1(1)$ to obtain $T_2(1)$, and the “row” 2.5 is appended to $T_1(2)$, and the resultant is $T_2(2)$.

1d) Solve the LP relaxation associated with each of the updated subproblems using $x_1^1 = 1$. Then we obtain the MIP feasible solutions for each subproblem: $y_1(1) = 0$, $y_2(1) = 0$, $y_1(2) = 1$, $y_2(2) = 0$.

1e) The Benders’ cut in this instance is $\eta = -4.75 + 3.25x_1$.

(Steps 2,3,4). As in Example 3.2, the optimal solution to the first-stage master problem is $x_1^3 = 1$, with a lower bound $f_t = -2.5$, and the algorithm stops.

**Remark 3.6.** In this instance, the Benders’ cut for the first-stage is weaker than that obtained in Example 3.2. The benefit however comes from the fact that the Benders’ cut requires only LP solves in the second-stage, and that the second-stage LPs are strengthened sequentially. Hence if there was a need to iterate further, the cut-enhanced relaxations could be used. In contrast, the cuts of the previous subsection requires the solution of as many 0-1 MIP instances as there are scenarios.

**Binary First-stage, MIP Second-stage: Branch-and-Cut**

We continue with the two-stage SMIP models (1.1,1.2), and the methods of this subsection will accommodate general integers in the second-stage. The methods studied thus far have not used the properties of B&B algorithms in any significant way. Our goal for this subsection is to develop a cut that will convey information uncovered during the stage-two B&B process to the first-stage model. This development appears in Sen and Sherali [2002] who refer to this as the $D^2$-BAC method. While our development proceeds with the fixed recourse assumption, the validity of the cuts are independent of this assumption.

Consider a partial B&B tree generated during a “partial solve” of the second-stage problem. Let $Q(\omega)$ denote the set of nodes of the tree that have been explored for the subproblem associated with scenario $\omega$. We will assume that all nodes of the B&B tree are associated with a feasible LP relaxation, and that nodes are fathomed when the LP lower bound exceeds the best available upper bound. This may be accomplished by introducing artificial variables, if necessary. The $D^2$-BAC strategy revolves around using the dual problem associated with the LP relaxation (one for each node), and then stating a disjunction that will provide a valid inequality for the first-stage problem.

For any node $q \in Q(\omega)$, let $z_{q\ell}(\omega)$ and $z_{qu}(\omega)$ denote vectors whose elements are used to define lower and upper bounds, respectively, on the second-stage
(integer) variables. In some cases, an element of $z_{qu}$ may be $+\infty$, and in this case, the associated constraint may be ignored, implying that the associated dual multiplier is fixed at 0. In any event, the LP relaxation for node $q$ may be written as

$$\begin{align*}
\text{Min} & \quad g^T y \\
W_k y & \geq r_k(\omega) - T_k(\omega)x \\
y & \geq 0 \\
y & \geq z_{ql}(\omega) - y \geq -z_{qu}(\omega),
\end{align*}$$

and, the corresponding dual LP is

$$\begin{align*}
\text{Max} & \quad \theta_q(\omega)^T [r_k(\omega) - T_k(\omega)x] + \psi_{ql}(\omega)^T z_{ql}(\omega) - \psi_{qu}(\omega)^T z_{qu}(\omega) \\
\theta_q(\omega)^T W_k + \psi_{ql}(\omega)^T - \psi_{qu}(\omega)^T & \leq g^T \\
\theta_q(\omega) & \geq 0, \quad \psi_{ql}(\omega) \geq 0, \quad \psi_{qu}(\omega) \geq 0,
\end{align*}$$

where the vectors $\psi_{ql}(\omega)$, and $\psi_{qu}(\omega)$ are appropriately dimensioned. Note also that we assume that the second-stage constraints include cuts that are similar to those developed in the previous subsection, so that $W_k$, $r_k(\omega)$, and $T_k(\omega)$ are updated from one iteration to the next.

We now turn our attention to approximating the value function of the second-stage MIP. As noted in section 2, the IP and MIP value functions are complicated objects. Certain convex approximations have been proposed by perturbing the distribution of the random right-hand-side vector (Van der Vlerk [2004]). For problems with a totally unimodular (TU) recourse matrix, this approach provides an optimal solution. For more general recourse matrices, these approximations only provide a lower bound. Consequently, we resort to a different approach for SMIP problems that do not satisfy the TU requirement.

The B&B tree, together with the LP relaxations at these nodes, provide important information that can be used to approximate MIP value functions. The main observation is that the B&B tree embodies a disjunction, and when coupled with the value functions of LP relaxations of each node, we obtain a disjunctive description of an approximation to the MIP value function. By using the disjunctive cut principle, we will then obtain linear inequalities (cuts) that can be used to build value function approximations. In order to do so, we assume that we have a lower bound $h_\ell$ such that

$$h(x, \tilde{\omega}) \geq h_\ell \quad \text{(almost surely)}$$

for all $x \in X$. Without loss of generality, this bound may be assumed to be 0.

Consider a node $q \in Q(\omega)$ and let $(\theta_q^k(\omega), \psi_{ql}^k(\omega), \psi_{qu}^k(\omega))$ denote optimal dual multipliers for node $q$. Then a lower bounding function may be obtained
by requiring that $x \in X$ and that the following disjunction holds.

$$\eta \geq \theta^k_q(\omega)^T [r_k(\omega) - T_k(\omega)x] + \psi^{k}_{q\ell}(\omega)^T z_{q\ell}(\omega) - \psi^h_{qu}(\omega)^T z_{qu}(\omega)$$

for at least one $q \in Q(\omega)$. (3.10)

Note that each inequality in (3.10) corresponds to a second-stage value function approximation that is valid only when the restrictions (on the $y$-variables) associated with node $q \in Q(\omega)$ hold true. Since any optimal solution of the second-stage must be associated with at least one of the nodes $q \in Q(\omega)$, the disjunction (3.10) is valid. By assumption, we have $\eta \geq 0$. Hence, $x \in X$ and (3.10) leads to the following disjunction:

$$\Pi_X(\omega) = \{ (\eta, x) \in \bigcup_{q \in Q(\omega)} E_q^k(\omega) \},$$

where

$$E_q^k(\omega) = \{ (\eta, x) | \eta \geq \tilde{v}^k_q(\omega) - \tilde{y}^k_q(\omega)^T x, Ax \geq b, x \geq 0, \eta \geq 0 \},$$

with,

$$\tilde{v}^k_q(\omega) = \theta^k_q(\omega)^T r_k(\omega) + \psi^{k}_{q\ell}(\omega)^T z_{q\ell}(\omega) - \psi^h_{qu}(\omega)^T z_{qu}(\omega),$$

and

$$\tilde{y}^k_q(\omega)^T = \theta^k_q(\omega)^T T_k(\omega).$$

The arguments provided above are essentially the same as that used in the previous subsection, although the precise setting is different. In the previous subsection, we convexified the right-hand side function of a valid inequality derived from the disjunctive cut principle. In this subsection, we convexify an approximation of the second-stage value function. Yet, the tools we use are the same. As before, we derive the epi-reverse polar which we denote by $\Pi^\dagger_X(\omega)$.

$$\Pi^\dagger_X(\omega) = \{ \sigma_0(\omega) \in \mathbb{R}, \sigma(\omega), \in \mathbb{R}^{m_1}, \zeta(\omega) \in \mathbb{R} | \forall q \in Q(\omega), \exists \tau_q(\omega) \geq 0, \tau_{0q}(\omega) \in \mathbb{R}_+ \}$$

s.t $\sigma_0(\omega) \geq \tau_{0q}(\omega) \forall q \in Q(\omega)$

$$\sum_{q \in Q(\omega)} \tau_{0q}(\omega) = 1$$

$$\sigma_j(\omega) \geq \tau_q(\omega)^T A_j + \tau_{0q}(\omega) \tilde{v}^k_q(\omega) \forall q \in Q(\omega), j = 1, \ldots, n_1$$

$$\zeta(\omega) \leq \tau_q(\omega)^T b + \tau_{0q}(\omega) \tilde{y}^k_q(\omega) \forall q \in Q(\omega)$$

$$\tau_q(\omega) \geq 0, \tau_{0q}(\omega) \geq 0 \forall q \in Q(\omega).$$ (3.11)
As the reader will undoubtedly notice, the number of atoms in the disjunction here depend on the number nodes available from the B&B tree, whereas, the disjunctions of the previous subsection contained exactly two atoms. In any event, the cut is obtained by choosing non-negative multipliers $\tau^k_{0q}(\omega)$ for all $q$, and then using the “Min” and “Max” operations as follows:

$$
\sigma^k_0(\omega) = \max_q \tau^k_{0q}(\omega)
$$

$$
\sigma^k_j(\omega) = \max_q \left\{ \tau^k_{jq}(\omega)^{\top} A_j + \tau^k_{0q}(\omega) \bar{y}^k_q(\omega) \right\} \forall j
$$

$$
\zeta^k(\omega) = \min_q \left\{ \left[ \tau^k_{jq}(\omega) \right]^{\top} b + \tau^k_{0q}(\omega) \bar{y}^k_q(\omega) \right\}.
$$

These parameters can also be obtained by using an LP of the form (3.9), and the disjunctive cut for any outcome $\omega$ is then given by

$$
\sigma^k_0(\omega) \eta + \sum_j \sigma^k_j(\omega) x_j \geq \zeta^k(\omega),
$$

where the conditions in (3.11) imply that $\sigma^k_0(\omega) \geq \max_q \tau^k_{0q}(\omega) > 0$. Hence, the epi-reverse polar only allows those facets (of the convex hull of $\Pi_X(\omega)$) that have a positive coefficient for the variable $\eta$. The “optimality cut” to be included in the first-stage master in iteration $k$ is given by

$$
\eta \geq E \left[ \frac{\zeta^k(\omega)}{\sigma^k_0(\omega)} \right] - E \left[ \frac{\sigma^k(\omega)}{\sigma^k_0(\omega)} \right]^{\top} x.
$$

It is obvious that one can also devise a multi-cut method in which the above optimality cut is disaggregated into several inequalities (e.g. Birge and Louveaux [1997]). The following asymptotic result is proved in Sen and Sherali [2002].

Proposition 3.7. Assume that $h(x, \tilde{\omega}) \geq 0 \forall p1$ for all $x \in X$. Let the first-stage approximation solved in iteration $k$ be

$$
\min \{ c^\top x + \eta \mid \eta \geq 0, x \in X \cap B, (\eta, x) \text{ satisfies (3.12.1), \ldots, (3.12.k)} \}.
$$

Moreover, assume that the second-stage subproblem is a mixed-integer linear program whose partial solutions are obtained using a branch-and-bound method in which all LP relaxations are feasible, and nodes are fathomed only when the lower bound (on the second-stage) exceeds the best available upper bound (for the second-stage). Suppose that there exists an iteration $K$ such that for
k \geq K$, the branch-and-bound method (for each second-stage subproblem) provides an optimal second-stage solution for all \( \omega \in \Omega \), thus yielding an upper bound on the two-stage problem. Then the resulting \( D^2 \)-BAC algorithm provides an optimal first-stage solution.

Continuous First-stage, Integer Second-stage and Fixed Tenders: Branch-and-Bound

With the exception of the SIR models, all others studied thus far were restricted to models in which the first-stage decisions are restricted to be binary. For problems in which the first-stage includes continuous decision variables, but the second-stage has mixed-integer variables, the situation is more complex. For certain special cases however, there are some practical B&B methods. We summarize one such algorithm which is applicable to problems with purely integer recourse, and fixed tenders \( T \) (see (1.1, 1.2)). This method is due to Ahmed, Tawarmalani and Sahinidis [2004].

The essential observation in this method is part c) of Proposition 2.2; namely, the value function of a pure IP (with integer \( W \)) is constant over hyper-rectangles ("boxes"). Moreover, if the set \( X = \{ x \mid Ax \geq b, x \geq 0 \} \) is bounded, then there are only finitely many such boxes. This observation was first used in Schultz, Stougie and Van der Vlerk [1998] to design an enumerative scheme for first-stage decisions, while the second-stage decisions were obtained using polynomial ideal theory. However, enumeration in multi-dimensional problems needs far greater care, and this is where the work of Ahmed, Tawarmalani and Sahinidis [2004] makes its contribution. The idea is to transform the original two-stage stochastic integer program into a global optimization problem in the space of "tender variables" \( \chi = Tx \). The transformed problem is as follows.

\[
\min_{\chi \in \mathcal{X}} \varphi(\chi),
\]

where \( \mathcal{X} = \{ \chi \mid Tx = \chi, x \in X \} \) and \( \varphi \) is defined as the sum of

\[
\phi(\chi) = \min \{ c^T x \mid Tx = \chi, x \in X \} \quad \text{and} \quad \Phi(\chi) = \sum_{\omega \in \Omega} p(\omega) h(r(\omega) - \chi),
\]

where \( h(r(\omega) - \chi) \) denotes the value function resulting from the value of a pure IP with right-hand side is \( r(\omega) - \chi \) (see (2.1)). Moreover, the recourse matrix \( W \) is allowed to depend upon \( \omega \). This is one more distinction between the methods of the previous subsections and the one presented here.

Using part c) of Theorem 2.2, the search space of relevance is a collection of boxes of the form \( \prod_{i=1}^n [\ell_i, u_i] \) that may be used to partition the space of tenders. Not having both ends of each interval in the box requires that lower
bounds be computed with some care. Ahmed, Tawarmalani and Sahinidis [2004] provide guidelines so that closed intervals can be used within the optimization calculations. Their method is summarized as follows.

### Branch and Bound for Continuous First Stage with Pure Integers and Fixed Tenders in the Second

0. **Initialize.**
   \[ k \leftarrow 0. \]
   a) Rescale \( \epsilon > 0 \), so that boxes have the form \( \prod_{i=1}^{m_2} [l_i, u_i - \epsilon] \).
   Since this step (choosing \( \epsilon \)) is fairly detailed, we refer the reader to Ahmed, Tawarmalani and Sahinidis [2004].
   b) Identify an initial box \( B^0 \) such that \( X \subseteq B^0 \). Calculate a lower bound \( \phi^0_t, \) and \( y^0(\omega) \) as second-stage solutions during the lower bounding process. If we find \( \chi^0 \in X \) such that \( \phi(\chi^0) = \phi^0_t \), then declare \( \chi^0 \) as optimal and stop.
   c) Initialize \( L \), the list of boxes, with its sole element \( B^0 \), and record \( \phi^0_t \) and \( y^0(\omega) \). Specify an incumbent solution, which may be NULL, and its value (possibly \( +\infty \)). The incumbent solution and its value are denoted \( \bar{X} \) and \( \bar{\phi} \), respectively.

1. **Node Selection and Branching**
   a) If the list \( L \) is empty, then declare the incumbent solution as optimal, unless the latter is NULL, in which case the problem is infeasible.
   b) \( k \leftarrow k + 1. \) Select a box \( B^k \) with the smallest lower bound (i.e. \( \phi^k_t, \forall t \in L \)). Remove \( B^k \) from the list \( L \). Partition \( B^k \) into two boxes by subdividing one edge of the box. Several choices are possible (see below). Denote these boxes as \( B^+ \) and \( B^- \).

2. **Bounding**
   a) (Lower Bounding). For each newly created box, \( B^+, B^- \), calculate a lower bound \( \phi^+_t, \phi^-_t \) (resp.). Include those boxes in \( L \) for which the lower bounds are less than \( \bar{\phi} \). For each box included in \( L \), record the lower bounds \( (\phi^+_t, \phi^-_t) \) as well as associated (non-integer) solutions \( y^+_t(\omega) \) and \( y^-_t(\omega) \). (These second-stage solutions are used for selecting the edge of the box which will be subdivided for partitioning.) Moreover, record \( \chi^+, \chi^- \), the tenders obtained while solving the lower bounding problems for \( B^+ \) and \( B^- \) resp.
b) (Upper Bounding). If $\chi^+ \in \mathcal{X}$ and $\varphi(\chi^+) = \varphi^+_x$, then update the incumbent solution and value ($\tilde{\chi} \leftarrow \chi^+$, $\tilde{\varphi} \leftarrow \varphi(\chi^+)$) provided $\varphi(\chi^+) < \tilde{\varphi}$. Similarly, if $\chi^- \in \mathcal{X}$ and $\varphi(\chi^-) = \varphi^-_x$, then update the incumbent solution and value ($\tilde{\chi} \leftarrow \chi^-$, $\tilde{\varphi} \leftarrow \varphi(\chi^-)$) provided $\varphi(\chi^-) < \tilde{\varphi}$.

3. Fathoming

Remove all those boxes from $\mathcal{L}$ whose recorded lower bounds exceed $\tilde{\varphi}$. Repeat from step 1.

There are two important details to be discussed: a) the lower bounding problem, and b) the choice of the edge for subdivision. Given any box $B$, let $\ell, u$ denote the vector of the upper and lower bounds for $\chi$ admissible to that box. Then, a lower bound on $\varphi(\cdot)$ for $\chi \in B$ can be calculated by evaluating $\Phi(u - \ell)$, and minimizing $\phi(\chi)$ over the set $\chi \in B$. The non-decreasing nature of IP value functions (see Proposition 2.2) imply that $\Phi(u - \ell) \leq \Phi(\chi)$, $\forall \chi \in B$. Hence the lower bounding scheme is easily justified. It is also worth mentioning that this evaluation can be performed without having any interactions between the stages or the scenarios, and hence is very well suited for parallel and/or distributed computing. Finally, there are several possible choices for subdividing an edge; the one suggested by the authors is analogous to a “most fractional” rule (see Remark 4.2).

0-1 MIP in Both Stages with General Random Data: Branch and Cut

Of all the methods discussed in this section, the one summarized here has the most in common with standard deterministic integer programming. One may attribute this to the fact that in the absence of any special structure associated with the random elements, it is easiest to view the entire SMIP as a very large deterministic MIP. This method was studied by Caroe [1998]. In order to keep the discussion simple, we only present the cutting plane version of the method, which essentially mimics any cutting plane method for MIP. The extension to a branch-and-cut method will be obvious.

Consider the deterministic equivalent problem stated in (2.4) under the assumption that the integer variables are restricted to be binary. Suppose that we solve the LP relaxation of this problem, and we obtain an LP optimum point $(\tilde{x}, \tilde{y}(\omega), \omega \in \Omega)$. If these vectors satisfy the mixed-integer feasibility requirement, then the method stops. Otherwise, one derives cuts for those $\omega \in \Omega$ for which the pair $\tilde{x}, \tilde{y}(\omega)$ does not satisfy the mixed-integer feasibility requirement. The new cuts are added to the deterministic equivalent, and the process resumes (by solving the LP relaxation). One could use any cutting plane method to derive the cuts, but Caroe [1998] suggests using the lift-and-project cuts popularized by Balas, Ceria and Cornújols [1993].
Given our emphasis on decomposition, the reader has probably guessed that there is some decomposition lurking in the background here. Of course, the reader is right; note that since each cut is in the space of variables \((x, y(o))\), the cut coefficients maintain the dual-block angular structure of (2.4). Because the cuts maintain this structure, the solution of the LP relaxation within this method relies on two-stage SLP methods (e.g. L-shaped decomposition). We should observe that unlike the IP decomposition methodology of all the previous subsections, this method relies on SLP decomposition, and as a result, convexification (cutting plane) steps are undertaken only at those iterations at which an SLP optimum is found, and when such an optimum is non-integer. Of course, the method is easily generalized to the branch-and-cut setting.

4. Decomposition Algorithms for Multi-stage SMIP: Scenario Decomposition

As with stochastic linear programs (SLP), the Stagewise Decomposition algorithms discussed in the previous section scale well with respect to the number of scenarios in the two-stage case. Indeed for SLP, these algorithms have been extended to the case of arbitrarily many scenarios (e.g. continuous random variables) using sampling in the two-stage case. However, the scalability of stagewise decompositon methods with respect to multiple decision stages may be suspect. In this section we present two scenario decomposition methods for multi-stage SMIP. These methods, based on branch-and-price (B&P) (Lulli andSen [2002]), and Lagrangian relaxation (Caroe and Schultz [1999]), share a lot in common. Accordingly, we will present one of the methods (B&P) in detail, and then show how B&P can be easily adapted for Langrangian relaxation. We mention another method, a heuristic by Lokketangen and Woodruff [1996] which combines a Tabu search heuristic with progressive hedging. As with the Lagrangian relaxation in IP, scenario decomposition methods allow us to exploit special structure while remaining applicable to a wide class of problems.

A Scenario Formulation and a Branch-and-Price Algorithm

There are several alternative ways in which a multi-stage stochastic programming model can be formulated. We restrict ourselves to modeling discrete random variables which evolve over discrete points in time which we refer to as stages. More general SP models have been treated as far back as Olsen [1976], and more recently by Wright [1994], and Dentcheva and Roemisch [2002]. The latter paper is particularly relevant for those interested in multi-stage SMIP, and there, the reader will also find a more succinct measure theoretic (as well as convex analytic) treatment of the problem. Because we restrict ourselves to discrete random variables, the data evolution
process can be described in graph theoretic terms. For this class of models, any possible trajectory of data may be represented as a path that traverses a series of nodes on a graph. Each node is associated with a stage index \( t \), and represents not only the piece of data revealed at stage \( t \), but also the history of data revealed prior to stage \( t \). Thus multi-stage SP models work with “path-dependent” data, as opposed to “state-dependent” data of Markov decision processes. Arcs on this graph represent the process of data (knowledge) discovery with the passage of time (stages). Since a node in stage \( t \) represents the entire history until stage \( t \), it (the node) can only have a unique predecessor. Consequently, the resulting graph is a tree referred to as a scenario tree. A complete path from the root of the tree to a leaf node represents a scenario.

Dynamic deterministic models consider only one scenario and note that for such problems one can associate decisions with each node of the scenario. For SP models, this idea is generalized so that decisions can be associated with every node on the scenario tree, and an SP model is one that chooses decisions for each node in such a manner as to optimize some performance measure. While several papers address other measures of performance (e.g. Ogryczak and Ruszczynski [2002], and Rockafellar and Uryasev [2002]), the most commonly studied measure remains the expected value model. In this case, decisions associated with nodes of the tree must be made in such a way that the expected value of decisions on the entire tree is optimized. (Here the expectation is calculated by weighting the cost of decisions at each node by the probability of visiting that node.) There are several equivalent mathematical representations of this problem, one of which is called the scenario formulation. This is the one we pursue here, although other formulations (e.g. the nodal formulation) may be of interest for the other algorithms.

Let the stages in the model be indexed by \( t \in \mathcal{T} = \{1, \ldots, T\} \), the collection of nodes of the scenario tree be denoted \( \mathcal{N} \), and let \( \Omega \) denote the set of all scenarios. By assumption there are finitely many scenarios indexed by \( \omega \), and each has a probability \( p(\omega) \). Let us associate decisions \( x(\omega) = (x_1(\omega), \ldots, x_T(\omega)) \) with each scenario \( \omega \in \Omega \). The decisions \( x_t(\omega) \) are mixed-integer vectors with \( j \in J_t \) denoting the index (set) of integer components in stage \( t \). It is important to note that since \( \omega \) denotes a complete trajectory (for stages in \( T = \{1, \ldots, T\} \)), these decision vectors are allowed to be clairvoyant. In other words, \( x_t(\omega) \) may use information from the periods \( j > t \) because the argument \( \omega \) is a complete trajectory! Such clairvoyant decisions are unacceptable since they violate the requirement that decisions in stage \( t \) cannot use data revealed in future stages \( (j > t) \). One way to impose this non-clairvoyaunce requirement is to impose the condition that scenarios which share the same history of data until node \( n \), must also share the same history of decisions until that node. In order to model this requirement, we introduce some additional mixed-integer vectors \( z_n, n \in \mathcal{N} \). Let \( \Omega_n \) denote a collection of scenarios (paths) that pass through node \( n \). Moreover, define a mapping \( \mathcal{H} : \mathcal{T} \times \Omega \rightarrow \mathcal{N} \) such that for any 2-tuple \( (t, \omega) \), \( \mathcal{H}(t, \omega) \) provides that node \( n \) in
stage \( t \) for which \( \omega \in \Omega_\omega \). Then, the non-clairvoyance condition (commonly referred to as non-anticipativity) requires that

\[
x_t(\omega) - z_{\mathcal{H}(t, \omega)} = 0 \quad \forall (t, \omega).
\]

(4.1)

Higle and Sen [2002] refer to this as the "state variable formulation;" there are several equivalent ways to state non-anticipativity requirement (e.g. Rockafellar and Wets [1991], Mulvey and Ruszczynski [1995]). We will also use \( J_t \) to index all integer elements of \( z_{\mathcal{H}(t, \omega)} \). The ability to directly address the "state variable" \((z)\) eases the exposition (and even computer programming) considerably, and hence we choose this formulation here. Finally, for a given \( \omega \in \Omega \), we will use \( z(\omega) \) to designate the trajectory of decision states associated with \( \omega \).

(4.1) not only ensures the logical dependence of decisions on data, but also frees us up to use data associated with an entire scenario without having to trace it in a stage-by-stage manner. Thus, we will concatenate all stagewise data into vectors and matrices that can be indexed by \( \omega \). Thus, the trajectory of cost coefficients associated with scenario \( \omega \) will be denoted \( c(\omega) \), the collection of technology matrices by \( A(\omega) \) and the right-hand side by \( b(\omega) \). In the following we use \( x_{jt}(\omega) \) to denote the \( j^{th} \) element of the vector \( x_t(\omega) \), a sub-vector of \( x(\omega) \). Next define the set

\[
X(\omega) = \left\{ x(\omega) \mid A(\omega)x(\omega) \geq b(\omega), x(\omega) \geq 0, x_{jt}(\omega) \text{ integer, } j \in J_t, \forall t \right\}.
\]

Given the above setup, a multi-stage SMIP problem can now be stated as a large-scale MIP of the following form:

\[
\text{Min} \left\{ \sum_{\omega \in \Omega} p(\omega)c(\omega)^\top x(\omega) \mid x(\omega) \in X(\omega), \text{ and } \right\}
\]

\[
\left\{ x(\omega) \right\}_{\omega \in \Omega} \text{ satisfies (4.1) } \forall \omega \in \Omega \}
\]

(4.2)

It should be clear that the above formulation is amenable to solution using decomposition because the only constraints that couple the scenarios together are (4.1). For many practical problems, this collection of constraints may be so large that aggregation schemes may be necessary to solve the large practical problems (see Higle, Rayco and Sen [2002]). However, for moderately sized problems, B&P and similar deterministic decomposition schemes are reasonably effective, and perform better than solving the entire deterministic equivalent using state-of-the-art software like CPLEX (Lulli and Sen [2002]). The following exposition assumes familiarity with standard column generation methods (see e.g. Martin [1999]).

The B&P algorithm may be described as one that combines column generation with branch-and-bound (B&B) or branch-and-cut (B&C). For the
sake of simplicity, we avoid the inclusion of cuts, although this is clearly do-able. The lower bounding scheme within a B&P algorithm requires the solution of an LP master problem whose columns are supplied by a mixed-integer subproblem. Let $e$ denote an event (during the B&B process) at which the algorithm requires the solution of an LP (master). This procedure will begin with those columns that are available at the time of event $e$, and then generate further columns as necessary to solve the LP. We will denote the collection of columns available at the start of event $e$ by the set $I_e^-$, and those at the end of the event by $I_e^+$. For column generation iterations in the interim (between the start and end of the column generation process) we will simply denote the set of columns by $I_e^+$, and the columns themselves by $\{x_i(\omega), i \in I_e^+(\omega)\}_{\omega \in \Omega}$.

Since the branching phase will impose integrality restrictions on the “state variables” $z$ we use the notation $z_\ell$ and $z_u$ to denote lower and upper bounds on $z$ for any nodal problem associated with B&P iteration. (As usual, some of the upper bounds in the vector $z_u$ could be $+\infty$.)

Given a collection of columns $\{x_i(\omega), i \in I_e^+(\omega), \omega \in \Omega\}$, the non-anticipativity constraints (4.1) can be expressed as

$$\begin{align*}
\sum_{i \in F(\omega)} \rho_i x_i(\omega) - z(\omega) &= 0, \quad \forall \omega \\
z_\ell &\leq z(\omega) \leq z_u, \quad \forall \omega \\
\sum_{i \in F(\omega)} \rho_i(\omega) &= 1, \quad \forall \omega \\
\rho_i(\omega) &\geq 0, \quad \forall i, \omega
\end{align*}
$$

(4.3a) \hspace{1cm} (4.3b) \hspace{1cm} (4.3c) \hspace{1cm} (4.3d)

Whenever the above set is empty, we assume that a series of “Phase I” iterations (of the column generation scheme) can be performed for those scenarios for which the columns make it infeasible to satisfy the range restrictions on some element of $z(\omega)$. In this case, a “Phase I” problem is solved for each offending scenario and columns are generated to minimize deviations from the box (4.3b). We assume that whenever (4.3) is infeasible, such a procedure is adopted to render a feasible collection of columns in the master program which is stated as follows.

$$\begin{align*}
\text{Min} \left\{ \sum_{\omega \in \Omega} \sum_{i \in F(\omega)} \left[ c(\omega) \trans x_i(\omega) \right] \rho_i(\omega) \right\}
\end{align*}
$$

\{ where $\rho_i(\omega), i \in F(\omega)\}_{\omega \in \Omega} \text{ satisfies (4.3)}. \} \hspace{1cm} (4.4)$
Given a dual multiplier estimate $\mu(\omega)$ for the non-anticipativity constraints (4.3a) in the master problem, the subproblem for generating columns for scenario $\omega \in \Omega$ is as follows.

$$D(\mu(\omega), \omega) = \min \left\{ [p(\omega)c(\omega) - \mu(\omega)]^T x(\omega) \mid x(\omega) \in X(\omega) \right\}.$$  \hfill (4.5)

While each iteration of column generation (LP solve) uses a different vector $\mu(\omega)$, we have suppressed this dependence for notational simplicity. In any case, the column generation procedure continues until $D(\mu(\omega), \omega) - \theta(\omega) \geq 0$ for some $\omega \in \Omega$, where $\theta(\omega)$ is a dual multiplier associated with the convexity constraint (4.3c). Because of the way in which $X(\omega)$ is defined, (4.5) is a deterministic MIP, and one solves as many of these as there are columns generated during the algorithm. As a result, it is best to use the B&P method in situations where (4.5) has some special structure, so that the MIP in (4.5) is solved efficiently. This is the same requirement as in deterministic applications of B&P (e.g. Barnhart et al [1998]). In Lulli and Sen [2002], the structure they utilized for their computational results was the stochastic batch sizing problem. Nevertheless, the B&P method is applicable to the more general problem. The algorithm may be summarized as follows.

**Branch and Price for Multi-Stage SMIP**

0. **Initialize.**
   
   a) $k \leftarrow 0$, $e \leftarrow 0$, $I^e = \emptyset$. $B^0$ denotes a box for which $0 \leq z \leq +\infty$. (The notation $I^e$ includes columns for all $\omega \in \Omega$; the same holds for $I^e_+$.)
   
   b) Solve (4.4) and its optimal value is $f^0$, and a solution $z^0$. If the elements of $z^0$ satisfy the mixed-integer variable requirements, then we declare $z^0$ as optimal, and stop.
   
   c) $I^{e+1} \leftarrow I^e_+$, $e \leftarrow e + 1$. Initialize $L$, the list of boxes, with its sole element $B^0$, and record its lower bound $f^0$, and a solution $z^0$. Specify an incumbent solution, which may be NULL, and its value (possibly $+\infty$). The incumbent solution and its value are denoted $\bar{z}$ and $\bar{f}$ respectively.

1. **Node Selection and Branching**
   
   a) If the list $L$ is empty, then declare the incumbent solution as optimal, unless the latter is NULL, in which case the problem is infeasible.
b) $k \leftarrow k + 1$. Select a box $B^k$ with the smallest lower bound (i.e., $f^k_v \leq f^v_i$, $\forall v \in \mathcal{L}$). Remove $B^k$ from the list $\mathcal{L}$ and partition $B^k$ into two boxes so that $z^k$ does not belong to either box, (e.g. choose the “most fractional” variable in $z^k$, and create two subproblems by partitioning). Denote these boxes as $B^+$ and $B^-$. 

2. Bounding 

a) (Lower Bounding). Let $F^{+1} \leftarrow F_+^e$, $e \leftarrow e + 1$. For the newly created box $B^+$, solve the associated LP relaxation (4.4) using column generation. This procedure provides the lower bound $f^+_i$ and a solution $z^+$. Let $F^{+1} \leftarrow F_+^e$, $e \leftarrow e + 1$. Now solve the LP relaxation (4.4) associated with $B^-$, and obtain a lower bound $f^-_i$, and a solution $z^-$. Include those boxes in $\mathcal{L}$ for which the lower bounds are less than $\bar{f}$. For each box included in $\mathcal{L}$, associate the lower bounds $(f^+_i, f^-_i)$ as well as associated (non-mixed-integer) solutions $z^+$ and $z^-$. 

b) (Upper Bounding). If $z^+$ satisfies mixed-integer requirements and $f^+_i < \bar{f}$, then update the incumbent solution and value ($\tilde{z} \leftarrow z^+, \tilde{f} \leftarrow f^+$). Similarly, if $z^-$ satisfies the mixed-integer requirement, then update the incumbent solution and value ($\tilde{z} \leftarrow z^-, \tilde{f} \leftarrow f^-$). 

3. Fathoming 

Remove all those boxes from $\mathcal{L}$ whose recorded lower bounds exceed $\bar{f}$. Repeat from step 1.

Remark 4.1. While we have stated the B&P method by using $z$ as the branching variables, it is clearly possible to use branching on the original $x$ variables. This is the approach implemented in Lulli and Sen [2002].

Remark 4.2. The term “most fractional” may be interpreted in the following sense: if a variable $z_j$ has a value $\bar{z}_j$, and which is in the interval $z_{l,j} \leq \bar{z}_j \leq z_{u,j}$, then assuming $z_{l,j}, z_{u,j}$ are both integers, the measure of integrality that one may use is $\min(\bar{z}_j - z_{l,j}, z_{u,j} - \bar{z}_j)$. The “most fractional” variable then is the one for which this measure is the largest. Another measure could be based on the “relatively most fractional” index:

$$\min \left\{ \frac{\bar{z}_j - z_{l,j}}{z_{u,j} - z_{l,j}}, \frac{z_{u,j} - \bar{z}_j}{z_{u,j} - z_{l,j}} \right\}.$$
Lagrangian Relaxation and Duality

The algorithmic outline of the previous subsection can be easily adapted to use Lagrangian relaxation as suggested in Caroe and Schultz [1999]. The only modification necessary is in step 2a, where the primal LP (4.4) is replaced by a dual. The exact formulation of the dual problem used in Caroe and Schultz [1999] is slightly different from the one we will use because our branching variables are $z$, whereas, they branch on the $x(\omega)$ variables directly. However, the procedures are essentially the same. We now proceed to the equivalent dual problem that may be used for an algorithm based on the Lagrangian relaxation.

When there are no bounds placed on the “state variables” $z$ (i.e. the root node of the B&B tree), the following dual is equivalent to the Langrangian dual

$$\max_{\mu} \left\{ \sum_{\omega \in \Omega} D(\mu(\omega), \omega) \mid \sum_{\omega \in \Omega} \mu(\omega) = 0, \forall n \in \mathcal{I} \right\}$$

(4.6)

where $\mu = \{\mu(\omega)\}_{\omega \in \Omega}$, and $D(\mu(\omega), \omega)$ is the dual function defined in (4.5). It is not customary to include equality constraints for a Lagrangian dual, but for this particular formulation of non-anticipativity, imposing the dual constraints accommodates the coupling variables $z$ implicitly. There are also some interesting probabilistic and economic features that result from re-scaling dual variables in (4.6) (see Higle and Sen [2002]). Nevertheless, (4.6) will suffice for our algorithmic purposes.

Note that as one proceeds with the branch-and-bound iterations, partitioning the space of “state variables” induces different bound on them. In turn, these bound should be imposed on the primal variables in (4.5). Thus, the dual lower bounds are selectively improved to close the duality gap via the B&B process.

We should note that the dual problem associated with any node results in a nondifferentiable optimization problem, and consequently, Caroe and Schultz [1999] suggest that it be solved using subgradient or bundle based methods (e.g. Kiwiel [1990]). While (4.6) is not the unconstrained problem of Caroe and Schultz [1999], the dual constraints in (4.6) have such a special structure that they do not impede any projection based subgradient algorithm.

In addition to their similarities in structure, B&P and Lagrangian relaxation also lead to equivalent convexifications, as long as the same non-anticipativity constraints are relaxed (see Shapiro [1979], Dentcheva and Roemisch [2002]). Nevertheless, these methods have their computational differences. The master problems in B&P are usually solved using LP software which has become extremely reliable and scalable. It is also interesting to note that B&P algorithms also have a natural criterion for curtailing the size of the master program. In particular, note that we can set aside those columns (in the master) that do not satisfy the bound restrictions
imposed at any given node. While this is not necessary, it certainly reduces the size of the master problem. Moreover, the primal approach leads to primal solutions from which branching is quite easy. For dual-based methods, primal solution recovery is necessary before good branching schemes (e.g. strong branching) can be devised. However, further computational research is necessary for a comparison of these algorithms.

We close this section with a comment of duality gaps for multi-stage SMIP. Alternative formulations of the dual problem may result in different duality gaps for multi-stage SMIP. For example, Dentcheva and Roemisch [2002] compare duality gaps arising from relaxing nodal constraints (in a nodal SP formulation) with gaps obtained from relaxing non-anticipativity constraints of the scenario formulation. They show that scenario decomposition methods, such as the ones presented in this section, provide smaller duality gaps than nodal decomposition. Results of this nature are extremely important in the design of algorithms for SMIP. And a final word of caution regarding duality gaps is that without using algorithms that ensure the search for a global optimum (e.g. branch-and-bound), it is difficult to guarantee that the duality gap for SMIP vanishes, even if the number of scenarios is infinitely large, as in problems with continuous random variables (see Sen, Higle and Birge [2000]).

5. Conclusions

In this chapter, we have studied several classes of SMIP models. However, there are many more models and applications that call for further research. We provide a brief synopsis of some of these areas.

We begin by noting that the probabilistically constrained problem with discrete random variables has been recognized by several authors as a disjunctive program (e.g. Prekopa [1990], Sen [1992]). These authors treat the problem from alternative view points, one of which may be considered a dual of the other. More recently, Dentcheva, Prekopa and Ruszczynski [2000] have proposed extensions that allow more realistic algorithms than previously studied. Nevertheless, there are several open issues, including models with random technology matrices, multi-stage models with stage-dependent probabilistic constraints, and more. Another area of investigation deals with the application of test sets to the solution of SMIP problems (Schultz, Stougie and Van der Vlerk [1998], Hemmecke and Schultz [2003]). The reader will find more on this topic in the recent survey by Louveaux and Schultz [2003]. Another survey of interest is the one by Klein Haneveld and Van der Vlerk [1999].

In addition to the above methods, SMIP models are also giving rise to new applications and heuristics. Network routing and vehicle routing problems have been studied by Verweij et al [2003], and Laporte, Van Hamme and Louveaux [2002]. Another classic problem that has attracted a fair amount
of attention is the stochastic unit-commitment problem (Takriti, Birge and Long [1996], Nowak and Römisch [2000]). Recent applications in supply chain planning have given rise to new algorithms by Alonso-Ayuso et al [2002]. Other related applications include the work on stochastic lot sizing problems (Lokketangen and Woodruff [1996], Lulli and Sen [2002]). It so happens, that all of these applications lead to multi-stage models, which are among the most challenging SMIP problems. Given such complexity, we expect that the study of good heuristics will be of immense value. Papers on multi-stage capacity expansion planning (Ahmed and Sahinidis [2003], MirHassani et al [2000] and others) constitute a step in this direction.

As shown in this chapter, the IP literature has much to contribute to the solution of SMIP problems. Conversely, decomposition approaches studied within the context of SP have the potential to contribute to the decomposition of IP models in general, and of course, SMIP models in particular. As one can surmise, research on SMIP models has picked up considerable steam over the past few years, and we expect this trend to continue. These problems may be characterized as “guard challenge” problems, and we expect modern computer technology to play a major role in the solution of these models. We believe that distributed computing provides the ideal platform for the implementation of decomposition algorithms for SMIP, and expect that vigorous research will overcome this “grand challenge.” The reader may stay updated on this progress through the SIP web site http://mally.eco.rug.nl/spbib.html.

Acknowledgments

I am grateful to the National Science Foundation (DMI-9978780, and CISE-9975050) for its support in this line of enquiry. I wish to thank Guglielmo Lulli, George Nemhauser, and an anonymous referee for their thoughtful comments on an earlier version of this chapter. The finishing touches on this work were completed during my stay as an EPSRC Fellow at the CARISMA Center of the Mathematics Department at the University of Brunel, U.K. My host, Gautam Mitra, was instrumental in arranging this visit, and I thank him for an invigorating stay.

References


Ch. 9. Algorithms for Stochastic Mixed-Integer Programming Models


Sen, S., J.L. Higle and L.A. Ntaimo [2002], “A Summary and Illustration of Disjunctive
Decomposition with Set Convexification,” Stochastic Integer Programming and Network
The Netherlands.
Sen, S. and H.D. Sherali [1985], “On the convergence of cutting plane algorithms for a class of
Sen, S. and H.D. Sherali [2002], “Decomposition with Branch-and-Cut Approaches for Two-
Stage Stochastic Integer Programming” working paper, MORE Institute, SIE Department,
University of Arizona, Tucson, AZ (http://www.sie.arizona.edu/SPEED-CS/raptormore/more/
Sherali, H.D. and W.P. Adams [1990], “A hierarchy of relaxations between the continuous and convex
hull representations for zero-one programming problems,” SIAM Journal on Discrete Mathematics,
3, pp. 411–430.
Sherali, H.D. and B.M.P. Fraticelli [2002], “A modification of Benders’ decomposition algorithm for
discrete subproblems: an approach for stochastic programs with integer recourse,” Journal of
Global Optimization, 22, pp. 319–342.
Sherali, H.D. and C.M. Shetty [1980], “Optimization with Disjunctive Constraints,” Lecture Notes in
Stougie, L. [1985], “Design and analysis of algorithms for stochastic integer programming,” Ph.D.
Department, University of Michigan, Ann Arbor, MI.
Takriti, S., J.R. Birge, and E. Long [1996], “A stochastic model for the unit commitment problem,”
Tind, J. and L.A. Wolsey [1981], “An elementary survey of general duality theory in mathematical
vander Vlerk, M.H. [2004], “Convex approximations for complete integer recourse models,”
Van Slyke, R. and R.J.-B. Wets [1969], “L-Shaped linear programs with applications to optimal
approximation method applied to stochastic routing problems: a computational study,”
Wright, S.E. [1994], “Primal-dual aggregation and disaggregation for stochastic linear programs,”
Enhanced Cut Generation Methods for Decomposition-based Branch-and-Cut for Two-Stage Stochastic Mixed Integer Programs

Yang Yuan, Suvrajeet Sen
Data Driven Decisions Lab, Department of Industrial, Welding and Systems Engineering, The Ohio State University, 1971 Neil Ave, Columbus, Ohio 43210, USA, {yuan.65@osu.edu, sen.22@osu.edu}

This paper is devoted to a study of computational speed-ups that may be possible in cut generation associated with decomposition-based branch-and-cut methods ($D^2$-BAC) for stochastic mixed-integer programs (SMIP). We discuss some bottlenecks in the cut generation process, and suggest several enhancements to speed up this process. Our computational results show that significant improvements (approximately 50% reduction in computation times) may be possible by streamlining the computations associated with the cut generation process. This paper establishes new benchmarks for serial processing of two-stage SMIPs.

Key words: Stochastic mixed-integer programming; disjunctive decomposition; cut generation; stochastic server location.

1 Introduction

Stochastic Mixed Integer Programs (SMIP) are recognized as one of the most formidable classes of mathematical programming problems. Not only are there significant challenges due to potentially large number of scenarios, but, SMIP with integers in the second stage give rise to a non-convex and discontinuous recourse function that may be difficult to optimize. Ahmed et al [2004] provide an illustration of how the presence of integer variables in the second stage leads to extremely complicated (non-convex and discontinuous) recourse functions. Over the past few years, there have been significant advances in the design of algorithms for solving SMIP problems, and recent surveys of these methods include Klein Haneveld and van der Vlerk [1999], Schultz [2003], and Sen [2005]. However, computational implementations, and results for large scale problems have been slow in coming. Some exceptions to this comment are papers by Alonso-Ayuso et al [2003], and Ntaimo and Sen [2006]. In the former paper, the authors present a branch-and-fix algorithm which they use to solve a supply chain planning problem, whereas the latter presents a computational comparison with supply chain as well as large scale stochastic server location instances. The
present paper continues the computational investigation of decomposition-based branch-and-cut methods (D²-BAC) which were initiated by Sen and Sherali [2006]. The main motivation for the enhancements undertaken in this paper arises from the benefits achievable through speed-ups in cut generation. For example, Balas and Perregaard [2003] have shown that significant speed-ups are possible in deterministic MIP problems when the cut generation LP (CGLP) can be implemented by taking advantage of its special structure. In this paper, we are motivated by similar considerations although our approach is intended to overcome two potential bottlenecks which arise from two types of CGLPs solved in any iteration of the D²-BAC algorithm.

In general, our target two-stage SMIP problem may be stated in the following manner:

\[
\min_{x \in X \cap B} \ c^T x + E[f(x, \tilde{\omega})] \quad (1.1)
\]

where \( X \subseteq \mathbb{R}^{n_1} \) is a convex polyhedral set, \( B \subseteq \mathbb{R}^{n_1} \) restricts the first-stage decision variables to be binary, and \( \tilde{\omega} \) is a multi-variate random variable defined on a probability space \((\Omega, \mathcal{A}, P)\). For any \( \omega \),

\[
\begin{align*}
  f(x, \omega) &= \min g(\omega)^T y \\
  &\text{s.t. } Wy \geq r(\omega) - T(\omega)x \\
  &\quad y \geq 0, y_j \text{ binary, } j \in J_2
  \end{align*} \quad (1.2)
\]

Essentially, the second-stage scenario subproblem formulated as (1.2) is a mixed-integer linear program with the index set \( J_2 \) specifying the set of binary variables. For the purposes of this paper, we restrict our attention to the scenario subproblem with fixed recourse by assuming that matrix \( W \) is deterministic. In addition, the D²-BAC algorithm is subject to the following assumptions.

1. \( \Omega \) is a finite set.
2. \( X = \{x \in \mathbb{R}^{n_1}_+ | Ax \leq b\} \).
3. \( \forall (x, \omega) \in X \times \Omega \), the subproblem (1.2) remains feasible with \( f(x, \omega) < \infty \).

The rest of this paper is organized as follows. In the next section, we provide a summary of the D²-BAC algorithm, and demonstrate potential bottlenecks that may be caused by the cut generation process. The subsequent two sections are devoted to the methodology we
adopt in streamlining the cut generation process. Section 5 reports on the computational performance of the revised $D^2$-BAC algorithm with the solution methods implemented. We end the paper with some concluding remarks.

2 Background

Sen and Higle [2005] proposed the Disjunctive Decomposition ($D^2$) method in which the second stage MIP was convexified using the common cut coefficient ($C^3$) theorem. Instead of solving scenario subproblem MIPs directly at every iteration of the algorithm, the $D^2$ method convexifies the second stage MIP by using a sequence of $D^2$ cuts of the form of $\pi^T y \geq \pi_0(x, \omega)$. The following $C^3$ theorem allows the left-hand-side coefficient $\pi$ generated for one scenario subproblem to be used for any other scenario subproblem.

**Theorem 1** (The $C^3$ Theorem). Consider the stochastic program with fixed recourse as stated in (1.1), (1.2). For $(x, \omega) \in X \times \Omega$, let $Y(x, \omega) = \{y | Wy \geq r(\omega) - T(\omega)x, y \geq 0, y_j \text{ binary}, j \in J_2\}$ denote the set of mixed-integer feasible solutions for the second stage MIP. Suppose that $\{C_h, d_h\}_{h \in H}$, is a finite collection of appropriately dimensioned matrices and vectors such that for all $(x, \omega) \in X \times \Omega$,

$$Y(x, \omega) \subseteq \bigcup_{h \in H} \{y \in \mathbb{R}^n_+ | C_h y \geq d_h\}.$$ 

Let

$$S_h(x, \omega) = \{y \in \mathbb{R}^n_+ | Wy \geq r(\omega) - T(\omega)x, C_h y \geq d_h\},$$

and $S(x, \omega) = \bigcup_{h \in H} S_h(x, \omega)$.

Let $(\bar{x}, \bar{\omega})$ be given, and suppose that $S_h(\bar{x}, \bar{\omega})$ is nonempty for all $h \in H$ and $\pi^T y \geq \pi_0(\bar{x}, \bar{\omega})$ is a valid inequality for $S(\bar{x}, \bar{\omega})$. Then there exists a function, $\pi_0 : X \times \Omega \to \mathbb{R}$ such that for all $(x, \omega) \in X \times \Omega$, $\pi^T y \geq \pi_0(x, \omega)$ is a valid inequality for $S(x, \omega)$.

**Proof** See Sen and Higle [2005].

The common-cut-coefficient $\pi$ can be obtained by using linear programming which is referred to as the $C^3$-LP. Sen and Higle [2005] also showed that while the cuts $\pi^T y \geq \pi_0(x, \omega)$ maintain common cut coefficients across all scenarios $\omega$, the function $\pi_0(x, \omega)$ is a piecewise linear concave function of the form

$$\pi_0(x, \omega) = \min_{h \in H} \lambda^T_h [r(\omega) - T(\omega)x].$$
In the $D^2$ algorithm, this function is convexified using the disjunctive cut principle (DCP). Since this case of DCP generates the right-hand-side of the $D^2$ cut, it is referred to as the RHS-LP, and the generated right-hand-side function, which is affine, is denoted $\pi_c(x, \omega)$.

The $D^2$-BAC algorithm (Sen and Sherali [2006]) is essentially a combination of value function convexification and set convexification of the second stage inherited from the $D^2$ method. Although the $D^2$ cuts are not necessary for asymptotic convergence of the $D^2$-BAC algorithm, one generates these cuts to accelerate the algorithm by convexifying the second-stage integer feasible solution set. The finite termination of $D^2$-BAC is guaranteed by the value function convexification procedure. We summarize this process as follows.

At every iteration of the algorithm, a branch-and-bound procedure is initialized for every scenario subproblem MIP, which allows a partial solve of the second stage subproblem by truncating the number of nodes explored by the branch-and-bound tree. For a leaf node $q \in Q(\omega)$, let $(\Theta_q(\omega), \psi_{ql}(\omega), \psi_{qh}(\omega))$ denote optimal dual multipliers for node $q$, where $\psi_{ql}(\omega), \psi_{qh}(\omega)$ are associated with the corresponding branching constraints. Then, the following disjunction provides a lower bound of the MIP value function.

$$\eta \geq \Theta_q(\omega)^T[r(\omega) - T(\omega)x] + \psi_{ql}(\omega)^Tz_{ql}(\omega) - \psi_{qh}(\omega)^Tz_{qh}(\omega) \text{ for at least one } q \in Q(\omega), \tag{2.2}$$

where $z_{ql}(\omega), z_{qh}(\omega)$ are the lower and upper bounds respectively of second stage decision variables at node $q$.

Note that for each $q \in Q(\omega)$, one inequality in the form of (2.2) represents an epigraph of the value function of the node corresponding to $q \in Q(\omega)$. Therefore for each scenario, the second stage value function can be bounded from below by an epigraph which is actually the union of epigraphs for all $q \in Q(\omega)$ (Sen and Sherali [2006]). According to the theory of disjunctive programming, we then construct and solve a cut generation LP (epi-reverse polar LP) to derive a facet of the convex hull of the disjunctive set, which would essentially provide us a convex lower bounding approximation of the value function. We integrate these lower bounding functions by using the expectation operator, and the resulting cut is added to the master problem. The algorithm iterates until the termination condition is satisfied. After combining the value function convexification with the $D^2$ cuts, the entire algorithm may be summarized by a flowchart, as shown in Figure 1.

Figure 1 shows that the approximations generated during the $D^2$-BAC algorithm require the solutions of several linear programs in connection with cut generation. Simplifying
Initialization

For each $\omega \in \Omega$, solve the LP relaxation of scenario subproblem

Integer solutions? for all $\omega \in \Omega$.

- yes
  - Update & Solve first-stage master MIP
  - no

- no
  - Terminated?

- yes
  - Optimum Output

For each $\omega \in \Omega$

Form and Solve the RHS-LP to generate $\pi_c(x, \omega)$

Update the subproblem and resolve it using truncated B&B

Form and Solve the ERP-LP to convexify the value function

Figure 1: A Basic $D^2$-BAC Algorithm

this process by using the special structure of these problems will speed up the method. Specifically, two performance bottlenecks of the $D^2$-BAC algorithm are identified:

- The method requires the solution of a large number of RHS-LPs when the number of scenarios is large.

- The cut generation LP denoted $C^3$-LP in Figure 1 can be streamlined significantly by using a decomposition method.

In the following sections, corresponding solution methods will be derived for resolving these two bottlenecks.

3 Simplified Cut Generation

The $D^2$-BAC algorithm inherits the set convexification technique from the $D^2$ algorithm. However, what makes $D^2$-BAC computationally unique and attractive is its ability to provide
value function convexification. Therefore, it may be possible to reduce the workload of the set convexification process in $D^2$-BAC by applying some weaker cuts, instead of the original $D^2$ cut. Note that the $D^2$ cuts at the $k^{th}$ iteration are of the form

$$(\pi^k)^T y \geq \pi_c^k(x, \omega), \quad (3.1)$$

where the right-hand-side $\pi_c^k(x, \omega)$ is essentially a convexification of $\pi_0^k(x, \omega)$, and should satisfy the following to ensure that (3.1) is a valid inequality.

$$\pi_c^k(x, \omega) \leq \pi_0(x, \omega) = \min\{ (\lambda_{01}^k)^T r_k(\omega) - (\lambda_{01}^k)^T T_k(\omega)x, (\lambda_{11}^k)^T r_k(\omega) + \lambda_{12}^k - (\lambda_{11}^k)^T T_k(\omega)x \} \quad (3.2)$$

where $r_k(\omega)$ and $T_k(\omega)$ denote the concatenated vectors at the $k^{th}$ iteration, and consists of the original $r(\omega)$, $T(\omega)$ together with the coefficients of the $D^2$ cuts generated at previous iterations. Moreover, the multipliers $\lambda_{01}^k, \lambda_{11}^k, \lambda_{12}^k$ can be obtained after solving the $C^3$-LP.

Without loss of generality, suppose that we know a lower bound on the affine pieces defining the ‘min’ in (3.2). Then, for any $\omega \in \Omega$, we can rewrite the ‘min’ function in (3.2) as the union of epigraphs $S_1(\omega)$ and $S_2(\omega)$ of these affine functions.

$$S_1(\omega) = \{(x, \zeta) : \zeta \geq (\lambda_{01}^k)^T r_k(\omega) - (\lambda_{01}^k)^T T_k(\omega)x, (x, \zeta) \geq 0\}$$

$$\Leftrightarrow \{(x, \zeta) : \zeta \geq \alpha_1(\omega) + \beta_1(\omega)^T x, (x, \zeta) \geq 0\} \quad (3.3a)$$

$$S_2(\omega) = \{(x, \zeta) : \zeta \geq (\lambda_{11}^k)^T r_k(\omega) + \lambda_{12}^k - (\lambda_{11}^k)^T T_k(\omega)x, (x, \zeta) \geq 0\}$$

$$\Leftrightarrow \{(x, \zeta) : \zeta \geq \alpha_2(\omega) + \beta_2(\omega)^T x, (x, \zeta) \geq 0\} \quad (3.3b)$$

$$S(\omega) = S_1(\omega) \cup S_2(\omega) \quad (3.3c)$$

where $\alpha(\omega)$ and $\beta(\omega)$ are introduced for notational simplicity.

The disjunction stated in (3.3) is usually referred to as ‘a simple disjunction’, in which we assume the data $\alpha_1(\omega), \alpha_2(\omega) > 0$ and the variable $\zeta \geq 0$. Because $x \in X \cap B^{n_1}$, one can assume, without loss of generality, that a lower bound for $\zeta$ is known. By translating the simple disjunction by the lower bound, we can always ensure a non-negative $\zeta$. Similar transformation can be applied to ensure $\alpha_1(\omega), \alpha_2(\omega) > 0$. In the rest of this section, we will present two methods for calculating $\pi_c$.

### 3.1 Method-V: valid inequality coefficients

The idea behind the first method is straightforward. It is referred to as Method-V, since this method will generate a valid inequality of the simple disjunction (3.3), which consists of two
affine functions of $x$. Because both $x$ and $\zeta$ are assumed to be non-negative, it follows that

$$\zeta \geq \pi^k_c(x, \omega) = \min\{\alpha_1(\omega), \alpha_2(\omega)\} + \sum_i (\min\{\beta_{1,i}(\omega), \beta_{2,i}(\omega)\} x_i)$$

(3.4)

where $\beta_{1,i}(\omega)$ and $\beta_{2,i}(\omega)$ represents the $i^{th}$ element of $\beta_1(\omega)$ and $\beta_2(\omega)$, respectively.

Basically, method-V can generate a relatively weak valid inequality with negligible computational cost as above, which saves some computational effort by eliminating the RHS-LP. On the other hand, it may not provide as strong a linear approximation as that provided by the $D^2$ cut.

### 3.2 Method-F: facet inequality coefficients

The valid inequality of (3.3) can be strengthened using facet inequalities suggested in Sen and Sherali [1986]. They utilize the duality between facets of the convex hull of disjunctive sets $S$ and the extreme points of reverse polars $S^\#$ of these sets to establish simple rules for deriving all facet cuts for simple disjunctions, where $S^\#$ is defined as follows

$$S^\# = \{\mu \in \mathbb{R}^n : \mu^T x \geq 1, x \in clconv(S)\}.$$  

Assuming that the origin $(x, \zeta) = (0, 0)$ is the point to be deleted by the valid inequality, the simple disjunction stated in (3.3) can be rewritten as

$$S_1(\omega) = \{(x, \zeta) : \frac{\zeta}{\alpha_1(\omega)} - \frac{\beta_1(\omega)^T}{\alpha_1(\omega)} x \geq 1, (x, \zeta) \geq 0\}$$

$$\Leftrightarrow \{(x, \zeta) : a_0^1(\omega)\zeta + \sum_j a_j^1(\omega)x_j \geq 1, (x, \zeta) \geq 0\}$$

(3.5a)

$$S_2(\omega) = \{(x, \zeta) : \frac{\zeta}{\alpha_2(\omega)} - \frac{\beta_2(\omega)^T}{\alpha_2(\omega)} x \geq 1, (x, \zeta) \geq 0\}$$

$$\Leftrightarrow \{(x, \zeta) : a_0^2(\omega)\zeta + \sum_j a_j^2(\omega)x_j \geq 1, (x, \zeta) \geq 0\}$$

(3.5b)

$$S(\omega) = S_1(\omega) \cup S_2(\omega).$$

(3.5c)

Now the facet inequality can be derived by finding a vertex of $S^\#(\omega)$, where $S(\omega)$ is in the form of (3.5). Consider the coefficients of affine functions in both $S_1(\omega)$ and $S_2(\omega)$, and let $N(\omega)$ denote the index set of decision variables. Denote the subsets as

$$J^+(\omega) = \{j \in N(\omega) : \exists h = 1, 2, a_j^h(\omega) > 0\},$$

$$J^+_h(\omega) = \{j \in N(\omega) : a_j^h(\omega) > 0\},$$

$$J^-(\omega) = \{j \in N(\omega) : a_j^h(\omega) < 0, \forall j, h = 1, 2\},$$

7
Following Sen and Sherali [1986], define \( \rho_j(\omega) \) as

\[
\rho_j(\omega) = \max\{a^h_j(\omega) : h = 1 \text{ or } 2, j \in J^+(\omega)\}, \forall j \in J^+(\omega)
\]

Define \( \beta_{jk}(\omega) \) as

\[
\beta_{jk}(\omega) = \min\{-a^h_k(\omega) / a^h_j(\omega) : h = 1 \text{ or } 2, j \in J^+(\omega), k \in J^-(\omega)\}
\]

Sen and Sherali [1986] show that the reverse polar can then be computed as follows.

\[
S^\#(\omega) = \{\mu : \mu_j \geq \rho_j(\omega), j \in J^+(\omega); \beta_{jk}(\omega)\mu_j + \mu_k \geq 0, j \in J^+(\omega), k \in J^-(\omega); \\
\mu_j \geq 0, j \in J^0(\omega)\} 
\] (3.6)

With this description of \( S^\#(\omega) \), we can identify an extreme point as follows.

\[
\mu_j^*(\omega) = \rho_j(\omega), j \in J^+(\omega); \\
\mu_j^*(\omega) = \max_{i \in J^+(\omega)} \{-\rho_j(\omega)\beta_{ij}(\omega)\}, j \in J^-(\omega); \\
\mu_j^*(\omega) = 0, j \in J^0(\omega)
\] (3.7a,b,c)

Given the dual correspondence between vertices of \( S^\#(\omega) \) and facets of \( cl\text{conv}(S(\omega)) \), the coefficients obtained from (3.7) are exactly the facet cut coefficients for the simple disjunction \( S \), which will essentially provide a suitable realization of \( \pi_c(x,\omega) \). While the computational effort to generate such a cut is greater than that for the cut provided by Method-V, one can also expect it to be stronger than that cut. In any case, both cuts are expected to reduce the computational cost of \( D^2\text{-BAC} \), since all \( RHS-LP \) solves are eliminated. In section 5, a comparative study will be presented to test the performance of these two newly generated sets of cut coefficients.
4 Decomposition of $C^3$-SLP

The cut generation LP used for the disjunctive decomposition algorithms may be stated as follows.

$$\begin{align*}
\text{Max } & \quad E[\pi_0(\tilde{\omega})] - E[y^k(\tilde{\omega})]^T \pi \\
\text{s.t. } & \quad \pi_j \geq \lambda_{01}^T W_j^k - I_j^k \lambda_{02} \quad \forall j \\
& \quad \pi_j \geq \lambda_{11}^T W_j^k + I_j^k \lambda_{12} \quad \forall j \\
& \quad \pi_0(\omega) \leq \lambda_{01}^T \rho_k^c(x^k, \omega) - \lambda_{02} [\tilde{y}_j(k)] \quad \forall \omega \in \Omega \\
& \quad \pi_0(\omega) \leq \lambda_{11}^T \rho_k^c(x^k, \omega) + \lambda_{12} [\tilde{y}_j(k)] \quad \forall \omega \in \Omega \\
& \quad -1 \leq \pi_j \leq 1, \forall j; -1 \leq \pi_0(\omega) \leq 1, \forall \omega \in \Omega \\
& \quad \lambda_{01}, \lambda_{02}, \lambda_{11}, \lambda_{12} \geq 0
\end{align*}$$

(4.1a - 4.1g)

where $\pi$, $\pi_0(\omega)$, $\lambda$ are decision variables, and all other quantities constitute LP data. Problem (4.1) is essentially the $C^3$-LP at the $k^{th}$ iteration of the disjunctive decomposition algorithms (Sen and Higle [2005]). The indicators $I_j^k$ in (4.1) are parameters, which are chosen according to the specific disjunction variable used for cut formation in the current iteration.

It is easily seen that (4.1) can be interpreted as a stochastic linear program (SLP) in which the common cut coefficients form the first stage variables, and are chosen in such a way that the expected depth of the disjunctive cut is maximized (see also Sen and Higle [2005]). Indeed, this observation leads to a stochastic program with a very special structure which can be utilized to great advantage. In order to do so, let us rewrite (4.1) as the following two stage SLP:

$$\begin{align*}
\text{Min } & \quad E[y^k(\tilde{\omega})]^T \pi + E[-\pi_0(\lambda, \tilde{\omega})] \\
\text{s.t. } & \quad \pi_j \geq \lambda_{01}^T W_j^k - I_j^k \lambda_{02} \quad \forall j \\
& \quad \pi_j \geq \lambda_{11}^T W_j^k + I_j^k \lambda_{12} \quad \forall j \\
& \quad -1 \leq \pi_j \leq 1, \forall j; \\
& \quad \lambda_{01}, \lambda_{02}, \lambda_{11}, \lambda_{12} \geq 0
\end{align*}$$

(4.2a - 4.2e)
where \( \pi, \lambda \) are the first-stage decision variables, and \( z \) is the second-stage decision variable. Based on the \( C^\beta - SLP \) structure identified above, we decide to apply Benders’ decomposition by specializing the formation of feasibility and optimality cuts.

Suppose at the \( t^{th} \) iteration of Benders’ decomposition, it is necessary to derive a feasibility cut given the first stage decision \( \lambda^t \). Let us define \( \phi(\lambda^t, \omega) \) and \( \psi(\lambda^t, \omega) \) as follows.

\[
\phi(\lambda^t, \omega) = (-\lambda^T_{01} \rho^k_c(x, \omega) + \lambda_{02} \bar{y}(k))
\]

\[
\psi(\lambda^t, \omega) = (-\lambda^T_{11} \rho^k_c(x, \omega) - \lambda_{12} \bar{y}(k))
\]

If \( \phi(\lambda^t, \omega) \geq \psi(\lambda^t, \omega) \), the feasibility cut is in the form of

\[
\lambda^T_{01} \rho^k_c(x, \omega) - \lambda_{02} \bar{y}(k) \geq -1.
\]

Otherwise, it is as follows,

\[
\lambda^T_{11} \rho^k_c(x, \omega) + \lambda_{12} \bar{y}(k) \geq -1.
\]

For simplicity, a binary parameter \( \sigma \) can be introduced to combine (4.5) and (4.6) into a unified form.

\[
\sigma (\lambda^T_{01} \rho^k_c(x, \omega) - \lambda_{02} \bar{y}(k)) + (1 - \sigma) (\lambda^T_{11} \rho^k_c(x, \omega) + \lambda_{12} \bar{y}(k)) \geq -1
\]

In order to construct the optimality cut at the \( t^{th} \) iteration, we consider the subgradient of \( E[-\pi_0(\lambda, \bar{\omega})] \) at \( \lambda^t \), denoted by \( \xi(\lambda^t) \). It consists of four subvectors: \( \xi(\lambda^t) = (\xi_{01}(\lambda^t), \xi_{11}(\lambda^t), \xi_{02}(\lambda^t), \xi_{12}(\lambda^t)) \). These subvectors can be calculated as follows.

\[
\xi_{01}(\lambda^t) = - \sum_{\omega: \phi(\lambda^t, \omega) \geq \psi(\lambda^t, \omega)} p(\omega) \rho^k_c(x, \omega)
\]

\[
\xi_{11}(\lambda^t) = - \sum_{\omega: \phi(\lambda^t, \omega) \leq \psi(\lambda^t, \omega)} p(\omega) \rho^k_c(x, \omega)
\]

\[
\xi_{02}(\lambda^t) = \sum_{\omega: \phi(\lambda^t, \omega) \geq \psi(\lambda^t, \omega)} p(\omega) [\bar{y}(k)]
\]

\[
\xi_{12}(\lambda^t) = - \sum_{\omega: \phi(\lambda^t, \omega) \leq \psi(\lambda^t, \omega)} p(\omega) [\bar{y}(k)]
\]
Therefore, the optimality cut has the form
\[ \theta \geq E[-\pi_0(\lambda^t, \tilde{\omega})] + \xi(\lambda^r)(\lambda - \lambda^t). \] (4.9)

After adding (4.7) and (4.9) into the master problem, the updated master problem is as follows:

\[
\begin{align*}
\text{Min} \quad & \theta + E[y^k(\tilde{\omega})]^T \pi \\
\text{s.t.} \quad & \pi_j \geq \lambda_{01}^T W_j^k - I_j^k \lambda_0 \quad \forall j \\
& \pi_j \geq \lambda_{11}^T W_j^k + I_j^k \lambda_{12} \quad \forall j \\
& \sigma_d(\lambda_{01}^T \rho_c(x^k, \omega)) - \lambda_{02}^T \tilde{y}_{j(k)} + (1 - \sigma_d)(\lambda_{11}^T \rho_c(x^k, \omega) + \lambda_{12}^T \tilde{y}_{j(k)}) \geq -1, d = 1, \ldots, D \\
& \theta \geq E[-\pi_0(\lambda^r, \tilde{\omega})] + \xi(\lambda^r)(\lambda - \lambda^r), r = 1, \ldots, R \\
& -1 \leq \pi_j \leq 1, \forall j; \theta \geq -1 \\
& \lambda_{01}, \lambda_{02}, \lambda_{11}, \lambda_{12} \geq 0
\end{align*}
\] (4.10a-g)

where \(d\) and \(r\) are indices for feasibility cuts and optimality cuts, respectively.

In general, the advantage of decomposing the original \(C^3\)-LP is reflected by the significant decrease of the problem size. In particular, if we pay attention to the problem structure of (4.3), it is not difficult to realize that the scenario subproblem can be solved directly by simply maximizing the RHS elements of (4.3b) and (4.3c), without invoking any formal LP solver. The formal procedure of the revised Benders’ decomposition for solving the \(C^3\) stochastic linear program is summarized as follows:

**Step 0. Initialization**

Initialize \(t = 0\), a upper bound \(V_0 = \infty\), and a lower bound \(v_0 = -\infty\).

**Step 1. Master problem solve**

Set \(t \leftarrow t + 1\).

Solve the master problem (4.10), record the optimal solution \(\lambda^t, \pi^t\), the objective \(v_t\) (the lower bound). If \(V_t - v_t < \epsilon\), terminate;

Else, continue to step 2.

**Step 2. Scenario subproblem feasibility check**

Given \(\lambda^t\), consider the scenario subproblem (4.3) \(\forall \omega \in \Omega\).

If \(\exists \omega \in \Omega\), such that (4.3) is not feasible,
add the feasibility cut in the form of (4.7) to (4.10) and repeat from step 1.

Else, continue to step 3.

**Step 3. Scenario subproblem solve**

Calculate the objective of the scenario subproblem \(-\pi_0(\lambda^t, \omega)\) for all \(\omega \in \Omega\).

\[-\pi_0(\lambda^t, \omega) = \max \{\phi(\lambda^t, \omega), \psi(\lambda^t, \omega), -1\}\]  
(4.11)

And the value of the expected recourse function is calculated by

\[E[-\pi_0(\lambda^t, \hat{\omega})] = \sum_{\omega \in \Omega} -p(\omega)\pi_0(\lambda^t, \omega).\]  
(4.12)

Update the upper bound \(V_t \leftarrow \min \{V_{t-1}, E[y^k(\hat{\omega})]^T \pi^t + E[-\pi_0(\lambda^t, \hat{\omega})]\}\).

**Step 4. Optimality cut**

Formulate the optimality cut in the form of (4.9) and add it to (4.10).

Repeat from step 1.

The structure of the above procedure is borrowed from Benders’ decomposition, although the special structure of the subproblem allows us to generate both feasibility and optimality cuts without using any LP solver.

**5 Computational Experiments**

Ntiamo and Sen [2006] investigated the computational performance of the original \(D^2\)-BAC algorithm by solving stochastic server location instances. The preliminary success with \(D^2\)-BAC for SMIP problems has been demonstrated by the performance comparison with the CPLEX MIP solver 7.0 (ILOG CPLEX [2000]). In this section, we report computational results using the enhancements discussed in the previous sections.

The experimental plan was aimed at exploring the performance improvement gained by the revised \(D^2\)-BAC algorithm, and as a consequence establishing new benchmarks for serial processing. All experiments were conducted on a Sun 208R with 2 UltraSPARC-III+ CPUs running at 900 MHz. The stopping criteria for each run was based on a declaration that either the difference between the upper and lower bound was small enough (a gap of 0.001%) or the CPU time limit of 10800 seconds (3hrs) was reached. In addition, as in the original \(D^2\)-BAC algorithm, CPLEX 7.0 LP/MIP solver was adopted to optimize all LPs/MIPs in the revised algorithm.
To ensure reliability of the experiment, all computational results reported in this study were averaged based on three replications. Moreover, in our implementation, we solved the original scenario subproblem MIPs without the $D^2$ cuts when it was necessary to tighten the upper bound. The maximum number of nodes to explore in the truncated branch and bound tree was set to 3, and the branch and bound scheme was activated when the optimality gap was below 10%.

5.1 Stochastic server location instances

The objective of stochastic server location problems is to minimize the total expected cost subject to certain server capacity and availability constraints by choosing locations of servers (the first-stage strategic decision) and server-client assignments (the second-stage tactical decision). In the following analysis, for convenience, the problem instances are named $SSLP\_m\_n\_S$, where $m$ is the number of potential server locations, $n$ is the number of potential clients, and $S$ is the number of scenarios. Table 1 gives the dimensions of the deterministic equivalent problems and their subproblems for comparative purposes. Here the column ‘Bins’ reports number of binary variables and the column ‘Cvars’ reports number of continuous variables. For a description of the generation of SSLP instances, the reader may refer to Ntaimo and Sen [2005].

<table>
<thead>
<tr>
<th>DEP</th>
<th>Subproblem</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Constrs</td>
</tr>
<tr>
<td>SSLP_5_25_50</td>
<td>1,501</td>
</tr>
<tr>
<td>SSLP_5_25_100</td>
<td>3,001</td>
</tr>
<tr>
<td>SSLP_5_50_100</td>
<td>5,501</td>
</tr>
<tr>
<td>SSLP_5_50_500</td>
<td>2,7501</td>
</tr>
<tr>
<td>SSLP_5_50_1000</td>
<td>55,001</td>
</tr>
<tr>
<td>SSLP_5_50_2000</td>
<td>110,001</td>
</tr>
<tr>
<td>SSLP_10_50_50</td>
<td>3,001</td>
</tr>
<tr>
<td>SSLP_10_50_100</td>
<td>6,001</td>
</tr>
<tr>
<td>SSLP_10_50_500</td>
<td>30,001</td>
</tr>
<tr>
<td>SSLP_10_50_1000</td>
<td>60,001</td>
</tr>
<tr>
<td>SSLP_10_50_2000</td>
<td>120,001</td>
</tr>
<tr>
<td>SSLP_15_45_5</td>
<td>301</td>
</tr>
<tr>
<td>SSLP_15_45_10</td>
<td>601</td>
</tr>
<tr>
<td>SSLP_15_45_15</td>
<td>901</td>
</tr>
</tbody>
</table>
5.2 Experiment with simplified cut generation

The computational results of the revised $D^2$-BAC algorithm for the SSLP instances are reported in Table 2, with only new right-hand-side cut generation embedded, instead of the $RHS-LP$ solve. Recall that two types of cuts have been derived: one is a valid inequality by comparing the corresponding coefficient pair in a simple disjunction, as represented by the column ‘$D^2$-BAC-V’; another is a facet inequality whose coefficients can be calculated using (3.7), as represented by the column ‘$D^2$-BAC-F’.

Table 2: $D^2$-BAC with Simplified Cut Generation

<table>
<thead>
<tr>
<th></th>
<th>CPU time (secs)</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$D^2$-BAC</td>
<td>$D^2$-BAC-V</td>
</tr>
<tr>
<td>SSLP_5,25,50</td>
<td>0.70</td>
<td>0.54</td>
</tr>
<tr>
<td>SSLP_5,25,100</td>
<td>1.73</td>
<td>1.31</td>
</tr>
<tr>
<td>SSLP_5,50,100</td>
<td>3.70</td>
<td>3.35</td>
</tr>
<tr>
<td>SSLP_5,50,500</td>
<td>23.05</td>
<td>19.91</td>
</tr>
<tr>
<td>SSLP_5,50,1000</td>
<td>64.17</td>
<td>57.59</td>
</tr>
<tr>
<td>SSLP_5,50,2000</td>
<td>274.40</td>
<td>251.09</td>
</tr>
<tr>
<td>SSLP_10,50,50</td>
<td>373.98</td>
<td>343.12</td>
</tr>
<tr>
<td>SSLP_10,50,100</td>
<td>452.31</td>
<td>688.71</td>
</tr>
<tr>
<td>SSLP_10,50,500</td>
<td>2772.22</td>
<td>2550.00</td>
</tr>
<tr>
<td>SSLP_10,50,1000</td>
<td>5677.80</td>
<td>5485.49</td>
</tr>
<tr>
<td>SSLP_10,50,2000</td>
<td>&gt; 10,800</td>
<td>&gt; 10,800</td>
</tr>
<tr>
<td>SSLP_15,45,5</td>
<td>232.30</td>
<td>341.52</td>
</tr>
<tr>
<td>SSLP_15,45,10</td>
<td>222.41</td>
<td>240.85</td>
</tr>
<tr>
<td>SSLP_15,45,15</td>
<td>1988.26</td>
<td>1784.44</td>
</tr>
</tbody>
</table>

As expected, the revised version of $D^2$-BAC inherits all superior features, including scalability, from the original $D^2$-BAC method. In addition, because of the elimination of $RHS-LP$, both cut generation procedures generally improve upon the computational times, relative to the times required by the original $D^2$-BAC algorithm.

Two observations should be made from the tabulated results. First, the average percentage enhancement (measured by CPU time savings) compared to the original $D^2$-BAC algorithm is around 5%, which is not particularly significant. Because the total number of $RHS-LP$ solves is the product of the number of scenarios and the number of iterations, intuitively one expects a significant reduction in CPU times, especially for instances with large $|S|$. However, in general, $RHS-LP$ problems are relatively small. As a matter of fact, the time spent in solving ‘small-scale’ $RHS-LPs$ is negligible compared to the total CPU time.
Hence, the performance improvement by removing these $RHS-LP$ solves is not significant. The second observation is that simple valid inequalities provide slightly better performance than facet inequalities in general, especially when the problem size increases.

On the other hand, note that even with the assistance of new cuts, the largest instance $SSLP_{10,50,2000}$ still could not be solved by the revised $D^2$-BAC algorithm within the predefined time limit, which means so far, the performance enhancement has stayed at the level of quantitative (i.e. marginal) change, not qualitative change.

### 5.3 Experiment with decomposed $C^3$-SLP

The computational experiment analyzed in this section involves the application of the $D^2$-BAC algorithm coupled with the decomposition of $C^3$-SLP to the SSLP instances. Table 3 gives the computational results. For consistency, we maintained the same environment and parameter settings as the foregoing experiment. For comparison, the decomposition of $C^3$-SLP was implemented on the $D^2$-BAC with valid inequality cut generation (Method-V) embedded. Analogous to Table 2, the columns ‘$D^2$’, ‘$D^2$-BAC’, ‘$D^2$-BAC-V’ provide average CPU times for the corresponding algorithms. Similarly, the column ‘$D^2$-BAC-V-$C^3$’ reports the CPU seconds required by the algorithm that used decomposition for the $C^3$-SLP. The last two columns in Table 3 respectively report the percentages in CPU time saving of $D^2$-BAC-V-$C^3$ for all instances, compared to the performances of two standard algorithms $D^2$ and $D^2$-BAC. The negative percentages in saving for $SSLP_{10,50,100}$ and $SSLP_{15,45,5}$ represent increased computational times.

**Remark.** Since the number of feasibility cuts increases rapidly and frequently in our implementation, we did not solve an updated master program after each feasibility cut was discovered. Instead, the updated master problem was solved only after the necessary feasibility cuts for all $\omega \in \Omega$ were generated and added at any iteration. This provided better computational performance.

From Table 3, we observe that the CPU time differences between the original $D^2$-BAC and $D^2$-BAC-V are insignificant. On comparing these two algorithms with $D^2$, we discern no specific pattern because the table shows slightly increased computational times for some instances and decreased computational times for others. However, when one compares the columns ‘$D^2$-BAC-V’ and ‘$D^2$-BAC-V-$C^3$’, the decomposition approach for $C^3$-SLP is the clear winner. Moreover, when comparing the latter with $D^2$ and $D^2$-BAC the differences
Table 3: $D^2$-BAC with Decomposed $C^3$-SLP Solve and Valid Inequality Cuts for SSLP Instances

<table>
<thead>
<tr>
<th>Instance</th>
<th>$D^2$ (a)</th>
<th>$D^2$-BAC (b)</th>
<th>$D^2$-BAC-V (c)</th>
<th>$D^2$-BAC-V-$C^3$ (d)</th>
<th>$(1 - \frac{d^4}{a})100%$</th>
<th>$(1 - \frac{d^2}{b})100%$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSLP_5,25,50</td>
<td>1.64</td>
<td>0.70</td>
<td>0.54</td>
<td>0.36</td>
<td>78%</td>
<td>49%</td>
</tr>
<tr>
<td>SSLP_5,25,100</td>
<td>2.15</td>
<td>1.73</td>
<td>1.31</td>
<td>0.89</td>
<td>59%</td>
<td>49%</td>
</tr>
<tr>
<td>SSLP_5,50,100</td>
<td>7.10</td>
<td>3.70</td>
<td>3.35</td>
<td>1.56</td>
<td>78%</td>
<td>58%</td>
</tr>
<tr>
<td>SSLP_5,50,500</td>
<td>34.50</td>
<td>23.05</td>
<td>19.91</td>
<td>12.36</td>
<td>64%</td>
<td>46%</td>
</tr>
<tr>
<td>SSLP_5,50,1000</td>
<td>140.47</td>
<td>64.17</td>
<td>57.59</td>
<td>22.77</td>
<td>84%</td>
<td>65%</td>
</tr>
<tr>
<td>SSLP_5,50,2000</td>
<td>603.37</td>
<td>274.40</td>
<td>251.09</td>
<td>42.74</td>
<td>93%</td>
<td>84%</td>
</tr>
<tr>
<td>SSLP_10,50,50</td>
<td>295.95</td>
<td>373.98</td>
<td>343.12</td>
<td>262.13</td>
<td>11%</td>
<td>30%</td>
</tr>
<tr>
<td>SSLP_10,50,100</td>
<td>396.76</td>
<td>452.31</td>
<td>688.71</td>
<td>486.99</td>
<td>-23%</td>
<td>-8%</td>
</tr>
<tr>
<td>SSLP_10,50,500</td>
<td>1902.2</td>
<td>2772.22</td>
<td>2528.45</td>
<td>1313.38</td>
<td>31%</td>
<td>53%</td>
</tr>
<tr>
<td>SSLP_10,50,1000</td>
<td>5410.1</td>
<td>5677.80</td>
<td>5485.49</td>
<td>2139.47</td>
<td>60%</td>
<td>62%</td>
</tr>
<tr>
<td>SSLP_10,50,2000</td>
<td>9055.29</td>
<td>&gt; 10,800</td>
<td>&gt; 10,800</td>
<td>3916.47</td>
<td>57%</td>
<td>64%</td>
</tr>
<tr>
<td>SSLP_15,45,5</td>
<td>110.34</td>
<td>232.30</td>
<td>341.52</td>
<td>211.79</td>
<td>-92%</td>
<td>9%</td>
</tr>
<tr>
<td>SSLP_15,45,10</td>
<td>1494.89</td>
<td>222.41</td>
<td>240.85</td>
<td>153.41</td>
<td>90%</td>
<td>31%</td>
</tr>
<tr>
<td>SSLP_15,45,15</td>
<td>7210.63</td>
<td>1988.26</td>
<td>1784.44</td>
<td>803.56</td>
<td>89%</td>
<td>60%</td>
</tr>
</tbody>
</table>

are significant. Finally, using the decomposition approach, all instances can be solved to optimality within the predefined time limit.

Moreover, the CPU time savings reported in the last two columns demonstrate the superiority of $D^2$-BAC-V-$C^3$ over the original $D^2$ and $D^2$-BAC algorithms for most test instances, except for SSLP_10,50,100 and SSLP_15,45,5. On average, compared to the original $D^2$ algorithm, the new implementation ($D^2$-BAC-V-$C^3$) results in a 48.50% savings in CPU time, while comparisons with the original $D^2$-BAC results in CPU time saving of approximately 46.50%.

6 Conclusion

In this paper, performance bottlenecks of the current implementation of the $D^2$-BAC algorithm were identified. After embedding some specialized solution methods to enhance cut generation, we investigated the computational performance of the revised $D^2$-BAC algorithm. Our computational experiments demonstrate that the new streamlined decomposition algorithm results in far superior performance, cutting computational time by nearly 50% on average. While the algorithms discussed in this paper are applicable to a variety of SMIP problems, our computational experiments were conducted for several stochastic server loca-
tion instances. Given that SSLP instances arise in a variety of applications (Ntaimo and Sen [2005]), we believe that our computational results may be applicable to a variety of practical problems. As for future research, one open area that deserves study is the parallelization of decomposition-based algorithm for SMIP.

Acknowledgments

The research reported in this paper was supported, in part, by grants from the Air Force Office of Scientific Research, first through the MURI grant at the University of Arizona (F49620-03-1-0477) and then at the Ohio State University (FA9550-07-1-0447).

References


