A Branch-and-Price Algorithm for Multi-stage Stochastic Integer Programming with Application to Stochastic Batch-Sizing Problems

Guglielmo Lulli and Suvrajeet Sen

Department of Systems and Industial Engineering - University of Arizona
Tucson, USA.

July 31, 2002

Abstract

In this paper we present a branch-and-price method to solve special structured multi-stage stochastic integer programming problems. We validate our method on two different versions of a multi-stage stochastic batch-sizing problem. One version adopts a recourse formulation, and the other is based on probabilistic constraints. Our algorithmic approach is applicable to both formulations. Our computational results suggest that both classes of problems can be solved using relatively few nodes of a branch-and-price tree. The success of our approach calls for extensions in methodology as well as applications.

Keywords: Stochastic Programming, Integer Programming, Batch-sizing, Branch-and-price Algorithm.

1 Introduction

Integer requirements on decision variables are indispensible in many operational and planning models. For instance, resource acquisition decisions that represent factors such as fixed-charges, change-over costs etc. lead to mixed-integer programming models. A variety of applications, such as those arising in scheduling, routing, location, production planning and finance lead to integer and combinatorial optimization models. Traditional approaches to these applications usually assume that all data for these models are known with certainty. However, this is seldom the case. Operations problems often involve parameters (e.g. demand, lead-time etc.) that are unknown at the time of planning, and their values are unveiled over time. The need to explicitly model uncertainty within mixed-integer programming models leads to the so called stochastic mixed-integer programming (SMIP) problems. As in other areas of mathematical optimization, integer requirements in stochastic programming problems have serious consequences on structural properties and algorithm design. SMIP problems are aimed at finding nonanticipative (here-and-now) decisions that must be taken prior to knowing the realization of the random variables. These decisions are required to be made in such a way that total expected costs or revenues (from here-and-now decisions and possible recourse actions) are optimized, and moreover, some of the decisions (including recourse actions) are restricted to be integer. Several classes of SMIP problem may arise depending on when the integer decision are made, relative to the observations of outcomes of the random variables.

In this paper, we are interested in the general case where there can be any number

of integer variables in any and all stages, and where any part of the data can be stochastic. However, in keeping with much of the literature, we deal with discrete random variables with finite support (Birge and Lauveaux [6]).

Interest in SMIP problems is relatively recent. Klein-Haneveld, Stougie and van der Vlerk [12] proposed solution schemes for the two-stage stochastic integer problems with simple integer recourse. Because of the special structure of the second stage, their approach is based upon the construction of the convex hull of the second stage value function. Laporte and Louveaux [15] proposed a decomposition-based approach for stochastic integer program when both the first and second stage variables are binary. They proposed a branch and bound approach in which optimality cuts approximate the non-convex second stage value function for a given binary first stage solution. Carøe [7] addresses a slightly more general mixed-binary SMIP problem, and adapts the lift-and-project methodology of Balas, Ceria and Cornuéjols [4] to this class of problems. Sen and Higle [20] developed a decomposition based algorithm for the solution of the two-stage stochastic integer programming problems emphasizing decomposition among the integer variables that appear in the first and second stages. Sherali and Fraticelli [21] have studied a related approach in which the reformulation-linearization technique has been used within a decomposition scheme. Also Ahmed, Tawarmalani and Sahinidis [3] proposed a branch and bound algorithm for the two-stage stochastic integer programs with mixed-integer first stage variables and pure integer second stage variables. They use a reformulation which exploits the special structure arising from a fixed recourse matrix in a two-stage SMIP. Finally, we should note that the last three papers mentioned above

were announced somewhat simultaneously in the year 2000, and there appears to be significant on-going activity for two stage SMIP problems.

As for multi-stage SMIP, much less is known. Løkketangen and Woodruff [16] applied a heuristic in which the progressive hedging algorithm was combined with a tabu search to solve multi-stage SMIP with binary variables. Carøe and Schultz [8] proposed a Lagrangian relaxation for use within a branch and bound algorithm for multi-stage SMIP. However their computational results were restricted to two-stage problems.

In this paper, we provide an algorithm based on a branch-and-price methodology to solve multi-stage SMIP problems. There are several reasons motivating the use of the branch-and-price methodology. First, it allows us to decompose the original stochastic problem into a collection of deterministic (scenario) subproblems which are coordinated by a master problem. Hence, the method allows us to take advantage of any deterministic special structure. Furthermore it is amenable for parallelization. Another motivation results from noting that the branch-and-price algorithm is applicable to both two-stage SMIP as well as multi-stage SMIP; the only difference is the amount of computational work. Moreover, column generation allows us to get lower bounds of the same quality as those based on Lagrangian relaxation, which are of course tighter than those obtained by simply solving the LP relaxation. Finally, the use of LP software allows efficient implementation because modern LP codes easily deal with the addition of variables and constraints during the solution process, which is essential for branch and bound algorithms.

We begin this paper by developing a general multi-stage branch-and-price algo-

rithm. This method is then specialized to a batch sizing problem under uncertainty. This class of problems is a generalization of the lot sizing problem and allows us to illustrate the applicability of our branch-and-price methodology to special structured problems. We also formulate a probabilistically constrained batch-sizing problem. This formulation is particularly applicable in cases where the cost of accommodating all possible demand realizations is astronomical. In such instances, managers often attempt to satisfy demand with significant reliability. We show that the branch-and-price methodology can also be applied to this new formulation of the stochastic batch sizing problem. The algorithm has been implemented using COIN, an IBM open source software, and CPLEX 7.0. Our results demonstrate the viability of using branch-and-price as a methodology for special structured multi-stage SMIP.

The paper is organized as follows: in § 2, we present a branch-and-price method for multi-stage stochastic integer programming problems for which a general formulation is also given. We verify the validity of the method on the multi-stage stochastic batch-sizing problem. In § 3, we provide a formulation of the problem and some of its extension are discussed. Computational results are reported in § 4. Finally, § 5 contains some modelling considerations and conclusions.

2 Branch-and-Price for Multi-stage SMIP

We consider a finite horizon sequential decision process under uncertainty. Denoting with $\mathcal{T} = \{1, \ldots, |\mathcal{T}|\}$ the decision horizon, we assume that the information is given by a discrete time stochastic process $\{\xi_t\}_{t=1}^{|\mathcal{T}|}$ defined on some probability space

$(\Xi, \mathcal{F}, \mathcal{P})$. Decisions are based on the information available at that time, i.e. on the set of decisions already made and on the outcome of the random variable in the previous stages. If we denote the vector of all decisions made from stage 1 to stage t by $\underline{x}_t = (x_1, \ldots, x_t)$ and the vector of the random variable outcomes during the same interval by $\underline{\xi}_t = (\xi_1, \ldots, \xi_t)$ then, a prototypical multi-stage stochastic program is given by the following problem:

$$min\{c_1(\xi_1)x_1 + Q_1(x_1) : W_1 x_1 \leq h_1(\xi_1), x_1 \in \mathbf{X}_1\},$$

where

$$Q_t(\underline{x}_t) = E_{\tilde{\xi}_{t+1}|\underline{\xi}_t} min\{c_{t+1}(\tilde{\xi}_{t+1})x_{t+1} + Q_{t+1}(\underline{x}_{t+1}) :$$

$$T_{t+1}(\tilde{\underline{\xi}}_{t+1})\underline{x}_t + W_{t+1}x_{t+1} \leq h_{t+1}(\underline{\xi}_{t+1}),\ \ x_{t+1} \in \mathbf{X_{t+1}}\}$$

for $t = 1, \ldots, |\mathcal{T}| - 1$ with $Q_{|\mathcal{T}|} \equiv 0$. Here we assume that ξ_1 is known at time $t = 1$ and $E_{\tilde{\xi}_{t+1}|\underline{\xi}_t}$ denotes expectation with respect to the distribution of $\tilde{\xi}_{t+1}$ conditioned on the observation $\underline{\xi}_t$. For all realizations of ξ and time stages, we suppose that $T_t(\underline{\xi}_t), W_t, c_t(\xi_t), h_t(\underline{\xi}_t)$ are matrices and vectors of conformable dimensions. The set $\mathbf{X_t}$ denotes restrictions that require some or all the decision variables to be integer.

In this paper, we assume that the random vector ξ has a finite support; that is $\Xi = (\xi^1, \ldots, \xi^r)$ with probabilities $p^1, \ldots, p^r$. This hypothesis allows us to represent uncertainty by means of scenarios. A scenario is a realization of the random variable $(c(\xi), h(\xi), T(\xi))$ corresponding to an elementary atom $\xi \in \Xi$. The relationship between scenarios is represented via a scenario tree $\Im$, which captures the evolution of all information trajectories over time (Birge and Louveaux [6]). At any node of the tree, there are several branches to indicate future possible outcomes of the random

variable from the current node. Because a scenario includes one node at each stage exactly once, it is represented by a path from the root node (at stage 1) to a leaf node (at stage $|\mathcal{T}|$) of the scenario tree. Note that with the exception of leaf nodes, other nodes of the scenario tree may belong to more than one scenario. Given the set of scenarios $\mathcal{S} = \{1, \ldots, r\}$, the correspondence between nodes of the scenario tree and 2-tuples $(t, s) \in \mathcal{T} \times \mathcal{S}$ is given by the surjective map $\mathcal{H} : \mathcal{T} \times \mathcal{S} \rightarrow \Im$. Furthermore we use $|\Im|$ to denote the number of nodes composing the tree.

A vector of decisions $x(\xi^s) = (x_1(\xi^s), \ldots, x_T(\xi^s))$ is associated with each scenario $s \in \mathcal{S}$. A multi-stage SMIP problem can now be restated as a large-scale MIP of the following form:

$$
\begin{aligned}
min \quad & \sum_{s=1}^{r} p^s [\sum_{t=1}^{T} c_t(\xi_t^s) x_t(\xi^s)] & & \\
s.t. \quad & W_1 x_1(\xi^s) \leq h_1(\xi_1^s) & \forall s \in \mathcal{S}. & \\
& T_{t+1}(\underline{\xi}_{t+1}^s) \underline{x}_t(\xi^s) + W_{t+1} x_{t+1}(\xi^s) \leq h_t(\underline{\xi}_{t+1}^s) & \forall s \in \mathcal{S}\, \forall t \in \mathcal{T}. & (1) \\
& x_t(\xi^s) = [\sum_{u \in \mathcal{B}_t^s} p^u x_t(\xi^u)] / \sum_{u \in \mathcal{B}_t^s} p^u & \forall s \in \mathcal{S}\, \forall t \in \mathcal{T}. & (2) \\
& x_t(\xi^s) \in \mathbf{X_t} & \forall s \in \mathcal{S}\, \forall t \in \mathcal{T}. &
\end{aligned}
$$

where $\mathcal{B}_t^s$ represents the set or bundle of scenarios that are indistinguishable from scenario s at time t, i.e. all scenarios u for which $\xi_\tau^u = \xi_\tau^s$ for all $\tau = 1, \ldots, t$. Therefore, a bundle of scenarios for any node $\mathcal{H}(t, s)$ of the scenario tree $\Im$ includes all those paths passing through node $\mathcal{H}(t, s)$. Constraints (2) are the non-anticipativity constraints which state that all scenarios with same history until the t-th stage should result in the same decision until this stage, i.e. decisions depend only on information

revealed in the past and not in the future. Non-anticipativity constraints can also be represented by associating decision vector $X_{\mathcal{H}(t,s)}$ for each node $\mathcal{H}(t,s)$ of the scenario tree. In this case, constraints (2) are replaced by

$$X_{\mathcal{H}(t,s)} = x_t(\xi^s) \qquad \forall s \in \mathcal{S}\, \forall t \in \mathcal{T}. \tag{3}$$

What separates stochastic programming from deterministic optimization is the presence of non-anticipativity constraints. These couple the decisions associated with different scenarios, thus making the problem "harder" to solve. If we relax the problem, discarding all the non-anticipativity constraints from the formulation, we obtain a fully decoupled block-angular mixed-integer programming problem. Decomposition methods are particularly attractive for problems with this feature, as they allow us to split the mixed-integer programming problem into more manageable pieces corresponding to single scenario subproblems. In this paper, we are going to use a column generation methodology in order to take advantage of this particular feature of multi-stage SMIP problems. Such methods, referred to as "Branch-and-Price" (B&P) algorithms in the MIP literature, have been successfully implemented to solve a broad class of large-scale integer programming problems arising in applications such as routing and distribution (Desrosiers et al. [10]), airline crew scheduling (Vance et al. [23]) and generalized assignment problem (Salvesberg [18]) among others.

The basic idea of B&P consists of combining column generation with branch-and-bound. A comprehensive description of branch-and-price methodology, including a discussion of practical issues concerning efficient implementation, is provided

in Barnhart et al. [5], Johnson et al. [14] and Martin [17]. Vanderbeck [25] and Vanderbeck and Wolsey [24] provide an alternative approach to combining the Dantzig-Wolfe decomposition method with branch-and-bound. The method proposed here can be interpreted in either setting.

Let $(c_t^s, h_t^s, T_t^s, x_{t,s}) = (c_t(\xi^s), h_t(\xi^s), T_t(\xi^s), x_t(\xi^s))$. For any scenario define

$$\Gamma_s := \{\{x_{t,s}\}_{t=1}^{|\mathcal{T}|} \ : \ x_{t,s} \in \mathbf{X_t}, \ T_t^s \underline{x}_{t,s} + W_t x_{t+1,s} \leq h_t^s, \ \forall t \in \mathcal{T}\}.$$

Then the deterministic equivalent of SMIP can be written as follows:

$$z = min\{\sum_{s\in\mathcal{S}} p^s \sum_{t\in\mathcal{T}} c_t^s x_{t,s} \ : \ \{x_{t,s}\}_{t=1}^{|\mathcal{T}|} \in \Gamma_s, \ X_{\mathcal{H}(t,s)} - x_{t,s} = 0, \ \forall t \in \mathcal{T}, \ \forall s \in \mathcal{S}\}.$$

In our development, we assume that the set of feasible solutions is nonempty and bounded. Applying the Mixed Integer Finite Basis Theorem (Theorem 4.30 in [17]) to the subset of constraints Γ_s, we note that there exists a finite set of its points $Q_s \subseteq \Gamma_s$ such that all points of Γ_s can be represented as a convex combination of points in Q_s. Let G_s denote the index set of these points. Then the equivalent deterministic formulation, also called master problem, can be restated as follows:

$$[RMP] \qquad \begin{aligned} min \ & \sum_{s\in\mathcal{S}} p^s \sum_{t\in\mathcal{T}, i\in G_s} c_t^s x_{t,s}^i \alpha_s^i & & \\ s.t. \quad & X_{\mathcal{H}(t,s)} - \sum_{i\in G_s} x_{t,s}^i \alpha_s^i = 0 & \forall s \in \mathcal{S}, \ \forall t \in \mathcal{T}. & \quad (4) \\ & \sum_{i\in G_s} \alpha_s^i = 1 & \forall s \in \mathcal{S}. & \quad (5) \\ & \alpha_s^i \geq 0 & \forall i \in G_s, \ \forall s \in \mathcal{S}. & \\ & X \in Z_+^{|\Im|} & & \end{aligned}$$

Constraints (4) and (5) are the non-anticipativity constraints and the convexity constraints respectively. The formulation reported above, is obtained by applying the Dantzig-Wolfe decomposition principle based on the convexification of the integer scenario polyhedron Γ_s. This is the most common approach in branch-and-price applications. In this case, the integer requirements are imposed directly on the convex combinations that arise in the master formulation.

Another approach is to apply the Dantzig-Wolfe decomposition method based on a complete discretization of the subsystem Γ_s, as in Vanderbeck [25]. In this case, the integer restrictions are naturally translated into integrality restrictions on the master problem variables, therefore constraints $X \in Z_+^{|\Im|}$ are replaced by

$$\alpha_s^i \in \{0,1\} \quad \forall i \in G_s,\ \forall s \in \mathcal{S}.$$

However, as we shall discuss subsequently, the two alternatives are not significantly different in the context of SMIP.

Because the number of variables in the master problem is exponential in terms of the original problem size, the reformulation is carried out only implicitly by way of using a column generation procedure. Only a subset of variables is selected, defining a restricted master problem (*RMP*). To verify the optimality of the restricted master problem *RMP* solution, we have to verify if columns not listed in *RMP* can generate an improvement of the objective function, if added to the *RMP*, i.e. price out negative. The reduced cost of the master problem variables α_s^i is given by $(\sum_{t\in\mathcal{T}}(p^s \cdot c_t^s - u_{t,s}) \cdot x_{t,s}^i) - \pi_s$ where $u_{t,s}$ and π_s are the dual variables associated with the non-anticipativity constraints (4) and the convexity constraints (5) respectively. This

task of verifying the *RMP* optimality is accomplished by solving the following pricing program for each scenario $s \in \mathcal{S}$:

$$\begin{aligned} & min \;\; C_s \cdot x_s \\ s.t. \quad & x_s \in \Gamma_s \end{aligned}$$

where $C_s = (p^s \cdot c_1^s - u_{1,s}, \ldots, p^s \cdot c_{|\mathcal{T}|}^s - u_{|\mathcal{T}|,s})$. If the cost obtained from the above program is smaller than π_s then the corresponding column has negative reduced cost and it can be added to the *RMP* for a new iteration.

The $|\mathcal{S}|$ pricing problems are independent of each other, and moreover, they inherit any special structure associated with the scenario problems. This permits the use of parellalization in the computation of new columns. When the LP relaxation of *RMP* is solved, integrality conditions on the decision variables have to be verified. If the master LP solution is not integral, and the LP bound is not higher than the current incumbent SMIP solution, branching must take place. When dealing with the master reformulation, several issues have to be addressed in the branching phase. Special branching schemes have been developed for integer programming column generation. The challenge in designing a branching scheme, is to find the one that excludes the current fractional solution, validly partitions the solution space of the problem and provides a pricing problem that is still tractable.

Henceforth in this section, we denote with *DRMP* and *CRMP* the restricted master problems obtained applying the Dantzig-Wolfe decomposition principle based respectively on discretization and convexification of the integer scenario polyhedron Γ_s. The *CRMP* reformulation of SMIP problems results in the same formulation as

[*RMP*] given earlier in this section. The *DRMP* reformulation is as follows:

$$
\begin{aligned}
& min \ \sum_{s\in\mathcal{S}} p^s \sum_{t\in\mathcal{T}, i\in G_s} c_t^s x_{t,s}^i \alpha_s^i \\
& s.t. \qquad X_{\mathcal{H}(t,s)} - \sum_{i\in G_s} x_{t,s}^i \alpha_s^i = 0 \qquad\qquad \forall s\in\mathcal{S},\ \forall t\in\mathcal{T}. \\
[DRMP] \qquad & \sum_{i\in G_s} \alpha_s^i = 1 \qquad\qquad \forall s\in\mathcal{S}. \\
& \alpha_s^i \in \{0,1\} \qquad\qquad \forall i\in G_s,\ \forall s\in\mathcal{S}. \\
& X \in R_+^{|\Im|}
\end{aligned}
$$

Note that the main distinction between *CRMP* and *DRMP* concerns the integrality conditions. In the former formulation they are imposed on the X variables while in the latter formulation are imposed on the α variables.

For the *CRMP* reformulation of SMIP problems, a straightforward branching scheme is based on the partition of the solution space using the original problem variables for which we impose the integer requirements. Therefore, if a component of the $\tilde{X}$ vector of the master LP solution $(\tilde{\alpha}, \tilde{X})$ is fractional, say the i-th component, then branching takes the form

$$X_i \leq \lfloor \tilde{X}_i \rfloor \qquad \text{or} \qquad X_i \geq \lceil \tilde{X}_i \rceil. \tag{6}$$

Even though it is not mandatory, we eliminate (from the successor nodes) all those columns whose components do not satisfy the new branching constraint. This branching scheme defines a partition of the integer polyhedron Γ_s $(\tilde{\Gamma}_s, \Gamma_s \backslash \tilde{\Gamma}_s)$ for all scenarios belonging to the bundle of scenarios of scenario tree node i.

Next we comment on branching scheme for *DRMP* proposed in Vanderbeck [25] and Vanderbeck and Wolsey [24]. Their scheme is based on the following partition

of the feasible solution set;

$$\sum_{i \in \bar{G}_s} \alpha_s^i \leq \lfloor \lambda \rfloor \qquad \text{or} \qquad \sum_{i \in G_s \setminus \bar{G}_s} \alpha_s^i \geq \lceil \lambda \rceil, \tag{7}$$

where $\lambda = \sum_{i \in \bar{G}_s} \bar{\alpha}_s^i \in [0,1]$ is fractional, and the values $\bar{\alpha}_s^i$ are obtained from a continuous relaxation of *DRMP*. In order to partition G_S (set of columns that have been generated) into $(\bar{G}_s,\ G_s \setminus \bar{G}_s)$ Vanderbeck suggests using inequality of the form $\gamma \cdot x \geq \gamma_0$ (on the original variables). This implies that columns that do not satisfy this inequality can be deleted from the corresponding master program. Of course, the other branch would then include these columns. This branching scheme may provide a little more flexibility, although one would still have to derive appropriate inequalities ($\gamma \cdot x \geq \gamma_0$). One common choice for γ is i-th unit vector, which corresponds to fixing the lower and upper bounds on the i-th component of $X \in \cap_{s \in \mathcal{S}} \Gamma_s$. Because we eliminate those columns that do not satisfy (6), this implementation is equivalent to (7) with $\gamma = e^i$.

The Branch-and-price procedure is summarized as follows.

Initialization set

$\pi_s = \infty$ and $G_s = \{\emptyset\}\ \forall s \in \mathcal{S}$,

$u_{t,s} = 0\ \forall s \in \mathcal{S},\ \forall t \in \mathcal{T}$,

$UB = \infty$ or equal to the objective function of some computed feasible solution.

Step 1 $\forall s \in \mathcal{S}$ solve the pricing problem:

$$min\{\ \Omega_s = C_s x^i \quad s.t. \quad x^i \in \Gamma_s\}$$

If $\Omega_s < \pi_s$ add the column to the current *RMP* and $i \rightarrow G_s$ with total cost g_s^i computed using the original cost vector $(c_1^s, \ldots, c_T^s)$. Otherwise go to Step 5.

Step 2 Solve the linear relaxation of the restricted master problem *RMP*, and update dual variables $u_{t,s}$ and π_s.

Step 3 If $X \in Z_+^{|\Im|}$, update the incumbent solution and go to Step 5.

Step 4 Start a branch phase using the following rule:

find a component of X that is fractional, say i. Define two subproblems adding the constraints $X_i \leq \lfloor \tilde{X}_i \rfloor$ or $X_i \geq \lceil \tilde{X}_i \rceil$ respectively.

Step 5 Select an active node of the branching tree and go to Step 1, otherwise the current solution is optimal, STOP.

3 The stochastic batch-sizing problem

In this section, we present a formulation for the multi-stage stochastic batch-sizing problem. The batch-sizing problem belongs to the class of economic lot-sizing (ELS) models. These models deal with production and/or inventory systems, and in general, their deterministic versions can be stated as follows: given a demand and a cost structure for T time periods, the object of production planning is to minimize the total production and inventory costs. One of the earliest versions of ELS problem is given by the model of Wagner and Whitin [22]. They studied the uncapacitated model with fixed set-up cost and linear inventory and production costs. In addition to the model formulation, Wagner and Whitin's main contribution was

in demonstrating that an optimal replenishment policy is one in which production is undertaken when inventory is zero. Furthermore, they proposed an efficient forward dynamic programming algorithm to solve the problem. A broad variety of ELS models have been studied in the literature. These models include the ones in which backlogging may be allowed, production and inventory capacities may be finite, start-up costs may be non-zero, etc. We have only mentioned a few extensions of the Wagner and Whitin model available in the literature. For a more complete description of the state of the art on the ELS models, see Martin [17], Aggarwal et al. [1] and Kuik et al. [13].

In the literature, two papers recently appear on the stochastic version of the ELS models. Haugen, Løkketangen and Woodruff [11] have proposed a heuristic algorithm casting the progressive hedging algorithm as a meta-heuristic to solve the problem. As a heuristic algorithm to solve the single scenario sub-problem, the authors implement the dynamic programming algorithm proposed by Zangwill. Structural properties of the stochastic ELS have been presented in Ahmed, King and Parija [2]. In particular, the authors show the effectiveness of the Krarup Bilde reformulation of the ELS also in stochastic setting. As an interesting by-product of their work, Ahmed, King and Parija showed that the deterministic optimality condition (i.e. production is undertaken only if inventory is zero) does not apply to the stochastic case.

In this paper we consider the stochastic version of the problem in which the demand, production, inventory and set-up costs are uncertain problem parameters evolving as discrete random variables. Furthermore, production takes place in a

batch mode, i.e. production level is a multiple of the batch size and consequently the corresponding decision variables are discrete. Finally, the problem is capacitated, i.e. capacity constraints on production and/or inventory levels are effective for at least one time period. Under these hypotheses, Wagner and Whitin's optimality conditions are no longer valid. The peculiarity of stochastic optimal solutions is that they may have a non-zero level of production even though inventory levels are non-zero. This is the manner in which a stochastic model helps hedge against future uncertainty. In § 3.1, we will discuss this aspect of the problem in greater detail.

We begin the formulation by providing summary of the notation used in the model.

$\mathcal{S} = \{1, \ldots, S\}$ is the set of scenarios,

$\mathcal{T} = \{1, \ldots, |\mathcal{T}|\}$ is the decision time horizon,

$b \equiv$ batch size,

$C_t \equiv$ production capacity at time period t in terms of number of batches,

$I_t \equiv$ inventory capacity at time period t in terms of number of batches,

$d_{t,s} \equiv$ demand at time period t in scenario s,

$c_{t,s} \equiv$ production cost at time period t in scenario s,

$h_{t,s} \equiv$ holding (or inventory) cost at time period t in scenario s,

$f_{t,s} \equiv$ fixed (or set-up) cost at time period t in scenario s,

$p_s \equiv$ probability of scenario s.

The decisions variables are:

$x_{t,s}$ production batch level at time t in scenario s,

$i_{t,s}$ inventory level at time t in scenario s,

$$y_{t,s} = \begin{cases} 1 & \text{if production is set-up at time } t \text{ in scenario } s, \\ 0 & \text{otherwise.} \end{cases}$$

$z_{\mathcal{H}(t,s)}$ production quantity at node $\mathcal{H}(t,s)$ of the scenario tree.

The stochastic batch-sizing problem for minimizing total expected cost is given by

$$Min \sum_{s \in \mathcal{S}} \sum_{t \in \mathcal{T}} p_s \cdot (c_{t,s} \cdot x_{t,s} + h_{t,s} \cdot i_{t,s} + f_{t,s} \cdot y_{t,s})$$

subject to

$$i_{t-1,s} + b \cdot x_{t,s} = d_{t,s} + i_{t,s} \qquad \forall t \in \mathcal{T},\ \forall s \in \mathcal{S}. \tag{8}$$

$$x_{t,s} - z_{\mathcal{H}(t,s)} = 0 \qquad \forall t \in \mathcal{T},\ \forall s \in \mathcal{S}. \tag{9}$$

$$x_{t,s} \leq C_t \cdot y_{t,s} \qquad \forall t \in \mathcal{T},\ \forall s \in \mathcal{S}. \tag{10}$$

$$i_{t,s} \leq b \cdot I_t \qquad \forall t \in \mathcal{T},\ \forall s \in \mathcal{S}. \tag{11}$$

$$y_{t,s} \in \{0,1\},\ \ x_{t,s} \in Z^+,\ \ i_{t,s} \in R^+ \qquad \forall t \in \mathcal{T},\ \forall s \in \mathcal{S}. \tag{12}$$

Constraints (8), are the collection of inventory balance constraints. They define the relation between demand, production level and inventory level for each time period t of any scenario s. Constraints (9) are the non-anticipativity constraints on production level decisions. We make decisions before realizing the random outcomes of demand. Note that non-anticipativity constraints are enforced only on the production decision variables, since the non-anticipativity constraints on the set-up

variables are automatically satisfied once they are satisfied by the production level variables. Constraints (10) and (11) are capacity constraints on production and inventory levels respectively. Constraints (10) are also set-up forcing constraints. The basic idea is to force variable y_t to be one if production takes place.

We propose to solve the multi-stage stochastic batch-sizing problem applying a branch-and-price methodology as described in § 2. For each scenario, the pricing problem will be a deterministic batch-sizing problem solved using a dynamic programming algorithm. The branch-and-price algorithm follows the same procedure as outline in § 2, with step 4 (branching) specialized as follows.

Step 4 a If $\exists\, i \in \Im : \; z_i \notin Z^+$ define two subproblems adding the constraints

$$x_{t,s} \leq \lfloor z_i \rfloor \quad \text{or} \quad x_{t,s} \geq \lceil z_i \rceil \qquad\qquad \forall t, s : \mathcal{H}_{t,s} = i.$$

Go to Step 5.

Step 4 b If

$$\exists\, s' \in \mathcal{B}_t^s \;\; : \;\; x_{t,s} \cdot x_{t,s'} = 0 \quad \text{and} \quad x_{t,s} + x_{t,s'} > 0$$

then branch on $y_{\mathcal{H}(t,s)}$. For each newly generated node of the search tree define an appropriate *RMP*, consistent with values assumed by the branching variables, i.e. eliminate all the columns whose entries are not consistent with the branching variable. Go to Step 5.

3.1 The stochastic batch-sizing problem with probabilistic constraints

In the stochastic batch-sizing model developed above in § 3, we require that demand is satisfied at any time period t, and no backlogging is allowed. According to the demand balance constraints (constraint (8)), at any stage of the system the production level should be large enough to cover all the possible demand outcomes in the next stage. Operation managers often consider these situations uneconomical. To overcome this potential drawback it may be appropriate to enforce a probabilistic constraint, which limits the number of scenarios accommodated in the optimal solution. Let q denote the acceptable level of service and μ_s a binary variable, which assumes value one if scenario s is included in the solution and zero otherwise. Mathematically, the condition therefore is

$$\sum_{s \in \mathcal{S}} p_s \cdot \mu_s \geq q.$$

The master formulation of *SBSP* with probabilistic constraint can be represented as follows:

$$Min \sum_{s \in \mathcal{S},\ i \in G_s} p_s \cdot \bar{c}_{i,s} \cdot \mu_s$$

$s.t.$

$$\sum_{i \in G_s} x^i_{t,s} \cdot \alpha^i_s - z_{\mathcal{H}(t,s)} \geq -M \cdot (1 - \mu_s) \qquad \forall t \in \mathcal{T},\ \forall s \in \mathcal{S}.$$

$$\sum_{i \in G_s} x^i_{t,s} \cdot \alpha^i_s - z_{\mathcal{H}(t,s)} \leq M \cdot (1 - \mu_s) \qquad \forall t \in \mathcal{T},\ \forall s \in \mathcal{S}.$$

$$\sum_{s \in \mathcal{S}} p_s \cdot \mu_s \geq q$$

$$\sum_{i \in G_s} \alpha^i_s - \mu_s = 0 \qquad \forall s \in \mathcal{S}.$$

$$\alpha^i_s \geq 0 \qquad \forall i \in G_s,\ \forall s \in \mathcal{S}.$$

$$\mu_s \in \{0, 1\} \qquad \forall s \in \mathcal{S}.$$

where $\tilde{c}_{i,s}$ is the total cost associated to the i^{th} column of scenario s. Note that the non-anticipativity constraints are imposed only on those scenarios that are accommodated when μ_s is equal to 1 while they are ignored when $\mu_s = 0$. The above formulation is somewhat unique in that it uses both continuous and discrete decision variables to enforce non-anticipativity constraints within a probabilistically constrained model.

4 Computational results

In this section, we present the computational results on both the stochastic batch-sizing problem and its probabilistic version, described in § 3.

In our computational analysis we used randomly generated instances. We generated different sets of problems by varying the number of time periods (stages) of the system. These instances are also characterized by the scenario tree. In our experiments we generated binary trees with conditional probability for any branch being p_n, and the other branch having conditional probability $1-p_n$ where n denotes a node of the tree. Here p_n is chosen from the uniform [0,1] distribution. By choos-

ing alternative values of p_n, as well as cost and demands we can generate different problem instances corresponding to any given planning horizon. Such classes are classified according to the length of the planning horizon (number of stages). For each set of instances, the characteristics of the scenario tree are given in Table 1.

Class of Instances	Number of Stages	Number of Scenarios	Scenario Tree Nodes
3-4	3	4	7
4-8	4	8	15
5-16	5	16	31
6-32	6	32	63

Table 1: Test instances' characteristics

We implemented the column generation method using BCP, which is part of COIN-OR, the Common Optimization INterface for Operations Research [9], a free-ware open source software. BCP is a framework for Branch, Cut and Price algorithms. It handles all bookkeeping related to search tree management, column and cut generation, and so on. We used CPLEX 7.0 as the LP engine to solve the linear relaxation of RMP. The subproblems were solved using a customized dynamic programming algorithm. Our experiments were conducted on a workstation SUN Ultra 80 with two processors and 1 GB RAM.

For the computational results presented below, we compare the performance of our algorithm with CPLEX Branch-and-Bound method (CPLEX-MIP), implemented using AMPL as modeling language. A time limit of 3600 sec. was imposed for both the solvers. Moreover, CPLEX default relative tolerance of 0.0001 was used.

4.1 Computational results on the Batch-Sizing Problem

In this subsection, we present computational results with the batch-sizing problem. In the multi-stage model for CPLEX-MIP, we used a "node" formulation of the stochastic programming problem which handles the non-anticipativity constraints implicitly. This is a common approach to multi-stage SLP formulations because it results in a more compact formulation (Birge and Louveaux [6]). In fact, the total number of variables and constraints in the formulation solved by Branch-and-Price is three times the number of variables and constraints for the formulation solved by CPLEX.

The results for the batch-sizing problem are given in Table 2, which reports the name of the instance, and associated solution statistics both for SP-COIN-BCP and for CPLEX-MIP for each instance solved.

In this table, the total running time for SP-COIN-BCP includes the time spent in generating the scenario tree and all the concerning data structures, in reading the input and in writing the output. Note that these times are not considered in the solution time statistic of CPLEX-MIP. However, this difference does not impugn the essence of our analysis because there are very few instances that have comparable times [1] (total running time for SP-COIN-BCP, and solution time for CPLEX-MIP). The remaining columns in the table are interpreted as follows:

Time in VG $\equiv$ time spent in variables generation,

[1] Instances with comparable times include: 3-4*g*, instances *a, b, f, g, h* and *i* of the set 4-8 and instances *b* and *l* belonging to the set 5-16.

	SP-COIN-BCP solution						CPLEX solution		
Instance	Total running time	B&B nodes	Time in VG	Time in HEU	Time for LPs	Time in SB	B&B nodes	CPU time	MIP Iter.ns
3-4a	0.45	3	0.00	0.00	0.08	0.10	10	0.20	55
3-4b	0.64	2	0.00	0.00	0.22	0.03	19	0.20	53
3-4c	0.80	3	0.01	0.00	0.21	0.03	26	0.20	46
3-4d	0.41	2	0.00	0.00	0.12	0.02	6	0.20	40
3-4e	0.06	0	0.00	0.00	0.01	0.00	2	0.00	24
3-4f	0.09	0	0.00	0.00	0.02	0.00	0	0.00	10
3-4g	0.08	0	0.00	0.00	0.03	0.00	27	0.04	64
3-4h	0.69	3	0.00	0.00	0.17	0.03	23	0.01	53
3-4i	0.29	3	0.00	0.00	0.06	0.02	2	0.00	22
3-4l	0.17	0	0.00	0.00	0.03	0.00	7	0.02	31
4-8a	0.32	0	0.00	0.00	0.07	0.00	219	0.20	319
4-8b	0.24	0	0.00	0.00	0.07	0.00	196	0.17	331
4-8c	1.28	3	0.04	0.01	0.27	0.20	356	0.26	501
4-8d	0.41	0	0.02	0.00	0.11	0.00	233	0.19	462
4-8e	0.36	0	0.01	0.00	0.11	0.00	57	0.10	168
4-8f	0.24	0	0.00	0.00	0.07	0.00	62	0.09	154
4-8g	0.37	0	0.00	0.00	0.08	0.00	161	0.15	283
4-8h	0.42	0	0.02	0.00	0.08	0.00	156	0.13	286
4-8i	0.40	0	0.02	0.01	0.10	0.00	385	0.30	604
4-8l	1.59	4	0.10	0.00	0.32	0.07	1775	1.00	2069
5-16a	1.13	0	0.04	0.01	0.27	0.00	1722	1.80	3623
5-16b	1.15	0	0.04	0.00	0.24	0.00	569	0.71	1203
5-16c	11.68	8	0.45	0.02	1.72	0.26	1626	1.80	3689
5-16d	16.37	11	0.67	0.04	2.32	0.31	5585	6.30	11195
5-16e	13.29	8	0.55	0.05	1.73	0.28	761	0.87	1487
5-16f	9.96	8	0.27	0.03	1.45	0.19	1401	1.50	2554
5-16g	18.35	9	0.85	0.06	2.59	0.28	7816	7.60	21421
5-16h	4.18	3	0.24	0.03	0.64	0.04	1820	1.90	3206
5-16i	1.30	0	0.00	0.00	0.26	0.00	149	0.28	360
5-16l	1.59	0	0.19	0.00	0.39	0.00	1722	1.50	2731
6-32a	156.31	11	7.84	0.25	12.16	1.19	$1.74e^5$	T	$4.51e^6$
6-32b	2945.03	103	38.40	1.58	165.90	19.83	$1.85e^6$	T	$4.93e^6$
6-32c	91.07	7	2.75	0.12	8.09	0.71	$5.49e^5$	$1.11\ 10^3$	$1.27e^6$
6-32d	1064.91	71	12.15	0.49	73.63	8.85	$1.6e^6$	T	$6.09e^6$
6-32e	403.64	29	7.86	0.34	28.26	2.90	$1.61e^6$	$2.8\ 10^3$	$3.98e^6$
6-32f	530.66	35	12.17	0.49	37.14	3.98	$1.62e^6$	T	$4.89e^6$
6-32g	233.60	19	5.80	0.26	17.86	1.95	$7.08e^5$	$1.4\ 10^3$	$1.70e^6$
6-32h	134.83	11	4.43	0.17	9.46	1.12	$7.20e^5$	$1.4\ 10^3$	$1.56e^6$
6-32i	294.74	19	6.79	0.24	22.51	1.84	$1.74e^6$	T	$4.85e^6$
6-32l	215.37	17	5.82	0.28	16.98	2.02	$1.59e^6$	T	$5.40e^6$

Table 2: Computational results for the Batch-Sizing Problem

Time in HEU $\equiv$ time spent in restoring feasibility (i.e. computing heuristic solutions when the search tree nodes' solutions are not feasible),

Time for LPs $\equiv$ time spent in solving the linear relaxation of *RMPs*,

Time in SB $\equiv$ time spent in strong branching.

To verify the effectiveness of our proposed methodology, we first compare the total running time of SP-COIN-BCP and the CPU time requested by CPLEX-MIP. From the performance reported in Table 2, it is apparent that for smaller instances (i.e. less than five stages) CPLEX-MIP solution times are shorter than those obtained for SP-COIN-BCP. However, for larger instances, and precisely instances of class 6-32, the SP-COIN-BCP procedure provides better running times. It turns out that we were able to solve larger instances using SP-COIN-BCP, but the same was not true for CPLEX. For example, we solved an instance of the seven-stage batch-sizing problem with 64 scenarios. For this instance, the statistics are as follows: 2133.6 sec. total running time, 43.2 sec. in variables generation, 1.2 sec. for restoring feasibility, 98.3 sec. in solving LPs and 6.4 sec. strong branching. However, CPLEX-MIP was unable to provide a solution after several hours of computing. Similarly we compare the number of nodes of the search tree visited by the two methods. In this case the differences are much more evident. For the largest instances we have a difference in several orders of magnitude. Indeed for the instances solved, CPLEX visits hundreds of thousands of nodes while SP-COIN-BCP visits at most 103. These results demonstrate the quality of our method and the viability of using branch-and-price as a methodology for special structured multi-stage SMIP

problems.

4.1.1 Computational comments on SP-COIN-BCP

One of the intriguing points of the results with SP-COIN-BCP is that the computational time required for the entire process is dominated by the amount of time required in traversing the search tree. We investigate this dominance in the Table 3. The search tree time reported in the table, is the overall time spent waiting for

Instance	Number of Scenarios	Total Running Time	Search Tree Time	Ratio
5-1	16	43.20	27.65	0.64
5-2	16	55.43	40.30	0.73
5-3	16	51.62	33.81	0.66
5-4	16	58.75	41.13	0.70
5-5	16	99.36	64.39	0.65
6-1	32	3451.34	2823.20	0.82
6-2	32	666.16	497.62	0.75
6-3	32	3617.84	2888.15	0.82
6-4	32	1233.12	853.32	0.69
6-5	32	690.48	456.41	0.66

Table 3: Search tree time

a new search node of the search tree, to be processed, after the previous node has been fathomed. The "Ratio" is the percentage of the total running time spent in visiting the search tree. Clearly, an inordinate amount of time is spent in traversing the search tree. Moreover it has the tendency to increase as the dimension of the test problem and the depth of the search tree increase as well. In comparison, CPLEX-MIP is able to handle a far greater number of nodes of its search tree. With improved tree management routines within COIN-BCP, we expect to improve upon the running times reported for SP-COIN-BCP.

4.2 Computational results on the Batch-Sizing Problem with probabilistic constraints

The computational results on the stochastic batch-sizing problem with probabilistic constrained are discussed here. For this version of the problem, we implement a scenario formulation for the CPLEX-MIP too. In our analyses, we use q = 0.75. The computational results are reported in Table 4.

Due to the combinatorial structure of the non-anticipativity constraints, the probabilistically-constrained version of the stochastic batch-sizing problem requires a greater number of search nodes. This irremediably affects the running times of our implementation of SP-COIN-BCP. While the number of nodes still remains manageable, the inefficiencies of the tree management routines (see § 4.1.1), result in higher running times than those obtained through CPLEX-MIP.

5 Modelling Considerations and Conclusions

In this paper, we have accomplished two objectives: we have provided a methodology for special structured multi-stage stochastic integer programming problems, and used this methodology for the stochastic version of the well known batch-sizing problem. To the best of our knowledge, this paper appears to be the first one to accommodate cost/demand uncertainty in a dynamic batch-sizing problem. In addition, we have proposed a probabilistically constrained formulation. One of the main advantages of our approach is that we handle the recourse formulation, and the probabilistically-constrained formulation within the same framework.

	SP-COIN-BCP solution						CPLEX solution		
Instance	Total running time	B&B nodes	Time in VG	Time in HEU	Time for LPs	Time in SB	B&B nodes	CPU time	MIP Iter.ns
3-4a	1.39	9	0.03	0	0.61	0.06	28	0.1	143
3-4b	1.15	7	0.02	0	0.51	0.04	46	0.1	176
3-4c	0.78	5	0.01	0.01	0.3	0.03	36	0.11	173
3-4d	1.43	9	0.01	0.01	0.55	0.05	43	0.09	162
3-4e	0.36	3	0.01	0	0.14	0.01	3	0.03	65
3-4f	0.55	5	0.01	0	0.24	0.01	22	0.06	127
3-4g	0.52	5	0.02	0	0.18	0.02	36	0.11	195
3-4h	0.53	5	0	0	0.19	0.03	126	0.18	286
3-4i	0.83	7	0.02	0	0.31	0.06	10	0.06	98
3-4l	0.67	5	0.02	0	0.28	0.03	38	0.11	153
4-8a	5.16	23	0.08	0	1.31	0.07	7350	43	66517
4-8b	2.4	13	0.05	0	0.61	0.07	1401	3.8	7712
4-8c	3.27	13	0.07	0	0.98	0.1	363	1.3	1863
4-8d	8.05	21	0.14	0.01	2.51	0.26	8953	18	30119
4-8e	12.82	31	0.29	0.01	3.76	0.34	143	0.65	985
4-8f	10.22	25	0.24	0.01	3.28	0.3	262	0.75	1440
4-8g	2.8	9	0.06	0.03	0.91	0.08	657	1.6	2370
4-8h	3.6	9	0.09	0	1.24	0.08	1000	2.5	4495
4-8i	2.79	9	0.06	0.01	0.83	0.09	1296	3.4	7248
4-8l	3.86	11	0.11	0	1.25	0.13	11356	22	30732
5-16a	73.87	65	1.08	0.08	13.36	1.16	41483	$3.8\ 10^2$	453063
5-16b	1466.64	309	9.98	0.26	428.84	10.04	5789	29	40641
5-16c	3205.37	447	27.65	0.87	946.3	23.37	59477	$2.6\ 10^2$	270258
5-16d	1400.57	343	16.37	0.4	286.38	13.86	50823	$3.0\ 10^2$	521205
5-16e	1308.85	381	11.88	0.4	194.39	15.67	123357	$6.7\ 10^2$	800819
5-16f	156.07	91	2.57	0.07	36.82	2.07	60037	$3.4\ 10^2$	358439
5-16g	3037.26	571	41.69	1.33	638.68	35.2	7350	43	66517
5-16h	128.94	77	2.61	0.07	27.44	1.66	1632	9.6	14032
5-16i	82.42	77	1.22	0.06	16.07	1.4	2591	14	26458

Table 4: Computational results for the Probabilistic formulation with q=.75

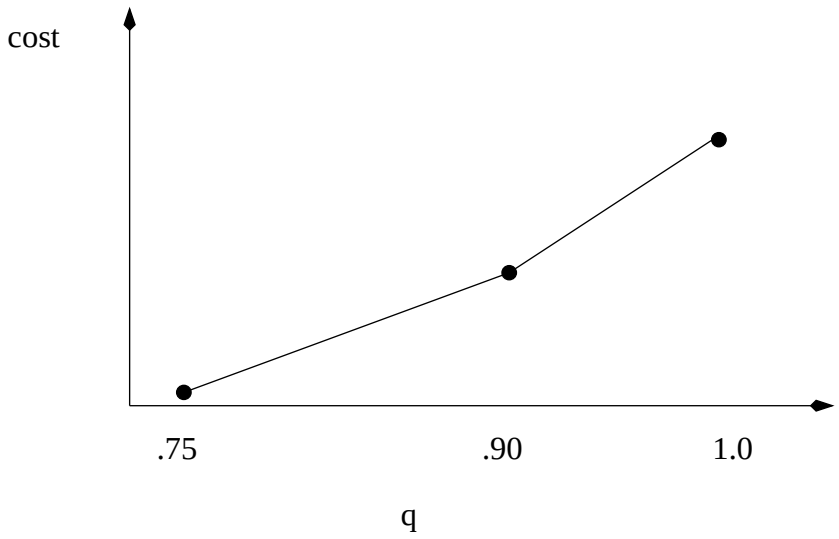

Figure 1: *Reliability vs. Cost*

In order to help the reader appreciate the value of such a unified framework, we present some modelling considerations arising from varying the level of service embodied by q. Solutions with higher q represent fewer stock-outs and hence more reliable. Obviously, higher reliability comes at a cost. A trade-off between reliability and costs may be depicted as shown in Figure 1. The data for which it was computed, used the average value of instances in class 5-16.

From an operational point of view, we may choose the level of service corresponding an acceptable cost structure. Such insights are made possible by the models and algorithm studied in this paper.

Acknowledgment

This research has been partially supported by NSF and the work was performed in the Raptor Lab. The authors are grateful to Laszlo Ladányi and Ted Ralphs for the

valuable suggestions and comments in the implementation of SP-COIN-BCP.

References

[1] Aggarwal A. and J.K. Park: *" Improved Algorithms for Economic Lot Size Problems"*, Operations Research **41** (1993) 549-571.

[2] Ahmed S., A.J. King and G. Parija. A Multi-stage Stochastic Integer Programming Approach for Capacity Expansion under Uncertenty, SPSS E-print Seeries (2001).

[3] Ahmed S., M. Tawarmalani and N.V. Sahinidis: *"A Finite Branch and Bound Algorithm for Two-stage Stochastic Integer Programs"*, preprint University of Illinios, 2000.

[4] E. Balas, S. Ceria and G. Cornuèjols, *"A lift-and-project cutting plane algorithm for mixed 0-1 programs"*, Mathematical Programming **58**, (1993) 295-324.

[5] Barnhart C., E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsberg and P.H. Vance: *" Branch-and-Price: Column Generation for Solving Huge Integer Programs"*, Operations Research **46** (1998) 316-329.

[6] Birge J.R. and F. Louveaux: *"Introduction to Stochastic Programming"*, Springer, 1997.

[7] Carøe C.C.: *"Decomposition in Stochastic Integer Programming"*, PhD thesis, University of Copenhagen (1998).

[8] Carøe C.C. and R. Schultz: *" Dual Decomposition in Stochastic Integer Programming"*, Operations Research Letters **24** (1999) 37-45.

[9] COIN - *"Common Optimization INterface for Operations Research"*, http//www.coin-or.org

[10] Desrosiers J., Y. Dumas, M.M. Solomon and F. Soumis: *"Time Constrained Routing and Sheduling"*, Handbook in Operations Research and Management Science: Networks, M.O.Ball et al. editors, Elsevier Science Publishers B.V., 1995.

[11] Haugen K.K., A. Løkketangen and D.L. Woodruff: *"Progressive hedging as a meta-heuristic applied to stochastic lot-sizing"*, European Journal of Operational Research**132** (2001) 116-122.

[12] Klein-Haneveld W.K., L. Stougie and M.H. van der Vlerk: *"An Algorithm for the Construction of Convex Hulls in Simple Integer Recourse Programming"*, Annals of Operational Research **64** (1996) 67-81.

[13] Kuik R., M.Solomon and L.N. van Wassenhove: *" Batching decisions: Structure and models"*, European J. of Operational Research **75** (1994) 243-263

[14] Johnson E.L., G.L. Nemhauser and M.W.P. Savelsbergh: *"Progress in Linear Programming Based Branch-and-Bound Algorithms: An Exposition"*, INFORMS J. on Computing. (1997)

[15] Laporte G. and F.V. Louveaux: *"The integer L-shaped Method for Stochastic Integer Programs with Complete Recourse"*, Operations Research Letters **13** (1993) 133-142.

[16] Løkketangen A. and D.L. Woodruff: *"Progressive Hedging and Tabu Search Applied to Mixed Integer (0,1) Multi-stage Stochastic Programming"*, Journal of Heuristics **2** (1996) 111-128.

[17] Martin R.K.: *"Large Scale Linear and Integer Optimization"*, Kluwer Academic Publishers, 1999.

[18] Savelsberg M.W.P.: *"A Branch-and-price Algorithm for the Generalized Assignment Problem"*, Operations Research **45** (1997) 831-841.

[19] Sen S.: *"A Branch and Price Algorithm for Multi-stage Stochastic Integer Problems"*, INFORMS 2000, San Antonio.

[20] Sen S. and J.L.Higle: *"The C^3 Theorem and a D^2 Algorithm for Large Scale Stochastic Integer Programming: Set Convexification"*, submitted for publication.

[21] Sherali H.D. and B.M.P Fraticelli: *"A modified Benders' partitioning approach for problems having discrete subproblems with application to stochastic programs with integer recourse"* Working paper, Virginia Polytechnic Institute and State University, Blacksburg, VA (2000).

[22] Wagner H.M. and T.M. Whitin: *"Dynamic Version of the Lot Size Model"*, Management Science **48** (1958) 578-590.

[23] Vance P.H., C. Barnhart, E.L. Johnson and G.L. Nemhauser: *" Airline Crew Scheduling: A New Formulation and Decomposition Algorithm"*, Operations Research **45** (1997) 188-200.

[24] Vanderbeck F. and L.A. Wolsey: *"An Exact Algorithm for IP Column Generation"* Operations Research Letters **19** (1996) 151-159.

[25] Vanderbeck F.: *"On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm"*, Operations Research **48** (2000) 111-128.