A Class of Stochastic Programs with Decision Dependent Random Elements

Tore W. Jonsbråten [a] Roger J-B Wets [b] David L. Woodruff [c]

[a] *Institute of Theoretical Dynamics, University of California Davis, and Department of Business Administration, Stavanger College, N-4004 Stavanger, Norway*
E-mail: tore.w.jonsbraten@oks.his.no

[b] *Department of Mathematics, University of California Davis, Davis, CA 95616 U.S.A.*
E-mail: rjbwets@ucdavis.edu

[c] *Graduate School of Management, University of California Davis, Davis, CA 95616 U.S.A.*
E-mail: dlwoodruff@ucdavis.edu

In the 'standard' formulation of a stochastic program with recourse, the distribution of the random parameters is independent of the decisions. When this is not the case, the problem is significantly more difficult to solve. This paper identifies a class of problems that are 'manageable' and proposes an algorithmic procedure for solving problems of this type. We give bounds and algorithms for the case where the distributions and the variables controlling information discovery are discrete. Computational experience is reported.

Keywords: stochastic program with recourse, integer stochastic programming, modelling

1. Introduction

Models based on stochastic programming lend valuable solutions to many types of problems. In the 'standard' formulation of a stochastic program with recourse, the distribution of the random parameters is independent of the decisions. When this is not the case, the problem is significantly more difficult to solve. This paper deals with a class of such problems that are 'manageable' and proposes an algorithmic procedure for solving problems of this type.

Before getting down to specifics, the issues can best be laid out in terms of a 'simple,' but general, formulation of this class of stochastic optimization problems where the information that will be provided to the decision maker is decision dependent. In order to introduce our new class of problems in the con-

text of the current literature, we first develop the following 'standard' stochastic programming problem:

$$\min_{x \in \boldsymbol{R}^n} \; E\{f(\boldsymbol{\xi}; x)\} = Ef(x) \tag{1}$$

where $f : \Xi \times I\!R^n \to \overline{I\!R} = [-\infty, \infty]$ is the 'cost' associated with a decision x when the random variable $\boldsymbol{\xi}$ takes on the value ξ; $\boldsymbol{\xi}$ is a $I\!R^k$-valued random variable with possible values in $\Xi \subset I\!R^k$, which is the support of the distribution, μ, of the random variable; $Ef : I\!R^n \to \overline{I\!R}$, the function to be minimized, is defined by

$$Ef(x) = \int_{\Xi} f(\xi; x)\, \mu(d\xi).$$

One can recast two stage stochastic programs with recourse so that they are seen as special cases of the problem just formulated. Indeed, for the two stage recourse problem with stage t variable indexes in the set N^t:

$$\begin{array}{ll} \min\limits_{x \in C^1} & f_{10}(x) + E\{\, \min_{x^2 \in C^2} [\, f_{20}(\xi; x^2) \,\big|\, f_{2i}(\xi; x, x^2) \le 0, i \in N^2\,]\} \\ \text{such that} & f_{1i}(x) \le 0, i \in N^1, \end{array}$$

and the function f is defined as follows:

$$f(\xi; x) = \begin{cases} f_{10}(x) + Q(\xi; x) & \text{if } x \in C^1,\ f_{1i}(x) \le 0 \text{ for } i \in N^1, \\ \infty & \text{otherwise} \end{cases}$$

where

$$Q(\xi; x) = \inf_{x^2 \in C^2} [\, f_{20}(\xi; x^2) \,\big|\, f_{2i}(\xi; x, x^2) \le 0, i \in N^2\,].$$

Even multistage stochastic programs can be recast so as to fit the general formulation. For example, in the case of a three stage problem, redefine the function Q as follows:

$$Q(\xi; x) = \inf_{x^2 \in C^2} [\, f_{20}(\xi; x^2) + E\{Q_2(\xi, \xi^3; x, x^2)|\xi\} \,\big|\, f_{2i}(\xi; x, x^2) \le 0, i \in N^2\,],$$

where ξ^3 stands for the observations made after choosing x^2 (but before selecting x^3), $E\{Q_2|\xi\}$ is the conditional expectation of Q_2 given ξ and

$$Q_2(\xi, \xi^3; x, x^2) = \inf_{x^3 \in C^3} [\, f_{30}(\xi, \xi^3; x^3) \,\big|\, f_{3i}(\xi, \xi^3; x, x^2, x^3) \le 0, i \in N^3\,].$$

The same pattern is followed if there are more than three stages, i.e., in the objective defining Q_2, f_{30} is replaced by $f_{30} + Q_3$, and so on.

This 'standard' formulation of the problem implicitly assumes that the measure μ is not affected by the decision x. This assumption, satisfied for all practical purposes by a very rich class of applications, makes possible the design of rather efficient solution procedures for 'real' size problems.

However, there are important decision making problems that do not fit in this mold, namely cases when the distribution of the random quantities will be affected by the decision selected. This can happen in many ways, but it seems the following formulation would cover all such cases:

$$\min E^{\mu} f(x) = \int_{\Xi} f(\xi; x)\, \mu(d\xi) \quad \text{such that} \quad (\mu, x) \in \mathcal{K} \subset M \times I\!R^n$$

where M is a subset of the probability measures on Ξ and $\mathcal{K}$ are the constraints linking the decision x to the choice of μ. In this formulation, the 'decision' x, affecting the level at which certain activities will be conducted, is viewed as disjoint from the choice of μ which will affect the information received and the time (stage) at which certain realizations of the random quantities will be observed. The constraints linking the choice, or inevitability, of μ to certain decisions x are modeled here through the linking constraints $(\mu, x) \in \mathcal{K}$.

On particular in the literature devoted to discrete event dynamic systems [10,14,16], the dependence of the probability measure on the decision(s) has often received the following formulation:

$$\min_{x \in I\!R^n} \int_{\Xi} f(\xi, x)\, \mu_x(d\xi).$$

In such a situation, the set $\mathcal{K}$ is the graph of the mapping $x \mapsto \mu_x$, i.e., $\mathcal{K} = \{(\mu, x) \,|\, x \in I\!R^n, \mu = \mu_x\}$; observe that our framework also allows for set-valued dependence between decisions and associated measures. In some other problems the measures might depend on certain parameters that are costly to evaluate and one can (should?) allocate some of the available resources to the estimation of these parameters, see e.g. [1]. If we then include among the decision variables those that will affect the estimation process, the problem is of the same type as here above and thus fits the general framework.

One advantage of our formulation is that it allows for a better classification of problems of this types based on the properties of the set $\mathcal{K}$ of the linking constraints and this should be helpful both in the design of computationally schemes and for purposes of analysis (as in Section 6, for example).

If there are no linking constraints, the problem is relatively simple. Since the 'cost' function is linear in μ, it is known that the solution will occur at a

measure μ^{ext} that is an extreme point of the convex hull of the set M. And there are results about generalized moment problems [3,15], and about optimization techniques for finding optimal submeasures [8,9] that are useful for dealing with such cases.

But beyond this, i.e., when there are nontrivial constraints linking μ to x, the problem becomes *significantly* more difficult to analyze and solve, and, not surprisingly, the literature dealing with this class of problems is very sparse. In fact, we only know of one such paper dealing with a 'Markovian' case [14]; consult also [5]. The relationship between x and μ can be quite complex, and although one might be able to write down quite general optimality conditions, at this stage we are going to limit our attention to a class of problems that are 'manageable.' In our case the set M will be finite, and its elements can be indexed by a boolean vector $d = (d_1, \ldots, d_q)$, i.e., each $d_j \in \{0, 1\}$, which will indicate if certain options have or have not been selected. The problem can then be reformulated as follows:

$$\min E^d f(x) = \int_{\Xi} f(\xi; x)\, \mu^d(d\xi) \quad \text{such that} \quad (d, x) \in \mathcal{K} \subset \{0,1\}^q \times I\!R^n. \tag{2}$$

The class of problems covered by this formulation includes those when the choice of d, and consequently of x, will affect the time at which the information about the realizations of certain random elements will become available. In the case of multistage problems, such as those described in the next sections, the choice of d affects the 'timing' of the observation, i.e., at which stage certain random variables will be observed. In contrast with the x-part of the problem which might involve 'staged decisions' that depend on past realizations, the choice of d will be a *first stage* decision only. This is a definite restriction, but it simplifies notation greatly while preserving the important new concepts. The resulting algorithms make inroads into this important class of problems, but full generalization is left as future research.

The following simple example might be helpful in understanding the connection. Suppose a production line must meet the demand for a certain number of products: $P_1, \ldots, P_q$, but if demand for a given product can't be satisfied, it's always possible to substitute a better one. Assuming that the products are linearly ordered, with P_q of the highest quality and P_1 of the lowest, it is thus possible to substitute P_k for P_j if $k > j$. However, there is quite a bit of uncertainty about the actual production costs of these different products, and consequently about the potential profits. Moreover, the production capacity available limits the choice to only a few of these items in the first stage. In the first stage, the

information available about the production costs is 'probabilistic', i.e., all what is known about the random variable $\boldsymbol{\xi}_j$ is their joint distribution μ. The information available in the second stage is then the actual production costs of the items produced in the first stage, and for the remaining ones it remains 'probabilistic.' In this example, the decision vector x fixes the production levels for $P_1, \ldots, P_q$. Each d_j is a boolean variable that indicates if P_j will be produced or not in the first stage, which in turn determines the information that will become available in the second stage.

The overall approach will be one of 'reduced minimization': the overall goal is to choose a 'best' stochastic optimization problem in a certain collection of such problems:

$$\left\{ \min_x E^d f(x) \,\middle|\, (d,x) \in \mathcal{K} \right\} \qquad d \in \{0,1\}^q$$

the one that yields the lowest possible value. Equivalently one could cast the problem as follows,

$$\min_{d \in \{0,1\}^q} \left\{ \min_x E^d f(x) \,\middle|\, (d,x) \in \mathcal{K} \right\}. \tag{3}$$

In the remainder of the paper, we are going to be interested in a class of problems where d identifies the time at which certain information is going to become available. More specifically, the production costs of certain items will be revealed either in stage two or stage three depending on d. The choice of a particular $d \in \{0,1\}^q$ determines

$$\mu^d \text{ and } \mathcal{K}^d = \{x \in I\!R^n \mid (\mu^d, x) \in \mathcal{K}\}$$

where this last set describes the corresponding set of feasible x-decisions.

In §2 specific notation is introduced for (stochastic) linear models that is convenient when dealing with algorithmic issues. Bounds given in §3 are employed in an implicit enumeration algorithm that is described in §4. Computational experience is reported in §5. Certain direction of descent are identified in §6 by means of variational analysis and the paper concludes with a summary and directions for further research.

2. Linear Models

The abstract definitions given in the previous section are useful for indicating the scope of our new models and for analysis. However, for algorithmic

development we need more specific notation. We begin by developing notation for stochastic linear programs, then we extend it to the case where there is decision-dependent information discovery. The bounds in §3 and the implicit enumeration algorithm that we develop in §4 do not rely explicitly on any assumptions of linearity, but linear problems provide a concrete basis for discussion and solvers are available for subproblems.

2.1. Stochastic Linear Programs

In order to give concrete definition to the class of problems for which computational results are obtained, we outline a 'standard' scenario-based, multistage, linear (perhaps integer or mixed integer) stochastic programming version of the problem given in expression (1). This formulation begins with the assumption that each random variable will be realized at a predetermined time and that scenarios are specified giving a full set of random variable realizations. For each scenario s, we are given the probability of occurrence of (or, more accurately, a realization "near") scenario s as $p(s)$. Each scenario s might actually correspond to the aggregation of a certain number of the possible realizations of the random parameters.

Decisions are made at times indexed by $1, \ldots, T$ and random variables in a scenario are realized at times $1, \ldots, T+1$. Our modelling convention is that information obtained up to time t is available for decisions at time t. Some information is available at time $t = 1$ before the first decision is selected, some information becomes available during the decision process, and some information may not become available until time $T+1$ after all decisions have been made.

We assume that scenarios are given in the form of a *scenario tree* denoted usually by $\mathcal{S}$. Together with the probabilities p_s attached to its scenarios, a scenario tree completely specifies the stochastic process associated with the random quantities of the problem. The notion of a scenario tree is important for many methods of solving stochastic programs and the notion that new information becomes available only at certain times is central to the construction of 'implementable' solutions. Each (terminal) leaf identifies a particular scenario. The leaves are grouped for connection to nodes at time T. Each leaf is connected to exactly one time T node and each of these nodes represents a unique realization up to time T. The time T nodes are connected to time $T-1$ nodes so that for each scenario connected to the same node at time $T-1$ has the same realization

up to time $T-1$. This continues back until information available "now" (at time index 1) constitutes the root node of the scenario tree. Variables that are observed at time t generate the *branches at time* $t-1$.

For each scenario s and each stage t we are given a vector $c^t(s)$ of length n^t, a $m^t \times n^t$ matrix $A^t(s)$ and a column vector $b^t(s)$ of length m^t. For notational convenience let $A(s)$ be $(A^1(s), \ldots, A^T(s))$, let $b(s)$ be $(b^1(s), \ldots, b^T(s))$, and let $c(s)$ be $(c^1(s), \ldots, c^T(s))$.

The decision variables are a set of n^t-vectors x^t; one vector for each scenario; notice that we reserve superscripts for the stage (or time) index. Notice also that the solution is allowed to depend on the scenario. Let $X(s) = (x^1(s), \ldots, x^T(s))$.

If we were prescient enough to know which scenario would actually occur, call it s, the problem would be solved by minimizing

$$f(s; X(s)) = \sum_{t=1}^{T} \langle c^t(s), x^t(s) \rangle$$

subject to

$$A(s)X(s) \geq b(s), \tag{4}$$

$$x_i^t(s) \in \{0,1\}, \quad i \in I^t,\ t = 1, \ldots, T \tag{5}$$

$$x_i^t(s) \geq 0, \quad i \in \{1, \ldots, n^t\} \setminus I^t,\ t = 1, \ldots, T \tag{6}$$

where $AX \geq b$ includes the usual sorts of single period and period linking constraints that one typically finds in multistage linear programming formulations. We use I^t to identify the integer variables in each time stage. For most of the literature, this set is empty. However, approaches for some classes of stochastic programming problems with integer variables in the first stage have been discussed (see, e.g., [2]) or for classes of two stage problems with integers [12,17,18,20]. Recently, there has been some work on various forms of multistage integer stochastic problems [4,6,13,19].

Since humans are not prescient, we must require solutions that do not require foreknowledge and that will be feasible no matter which scenario is realized. We refer to solution systems that satisfy constraints with probability one as *admissible*. We refer to a system of solution vectors as *implementable* if for scenario pairs s and s' that are indistinguishable up to time t, it is true that $x^r(s) = x^r(s')$ for all $1 \leq r \leq t$. For a given scenario tree $\mathcal{S}$, the set of implementable solutions will be denoted by $\mathcal{N}_{\mathcal{S}}$; one writes $E^{\mathcal{S}}$ to indicate that expectation is with re-

spect to the measure associated with the scenario tree $\mathcal{S}$. This brings us to the following formulation for stochastic linear programs:

$$\min \sum_{s\in\mathcal{S}} [p(s)f(s;X(s))] = E^{\mathcal{S}}\{f(\cdot, X(\cdot))\}$$

subject to

$$A(s)X(s) \geq b(s), \quad s \in \mathcal{S} \tag{7}$$

$$x_i^t(s) \in \{0,1\}, \quad i \in I^t,\ t = 1,\ldots,T,\ \ s \in \mathcal{S} \tag{8}$$

$$x_i^t(s) \geq 0 \ \ i \in \{1,\ldots,n^t\} \setminus I^t,\ t = 1,\ldots,T,\ \ s \in \mathcal{S} \tag{9}$$

$$X(\cdot) \in \mathcal{N}_{\mathcal{S}}. \tag{10}$$

The expectation here can be expressed as a finite sum since we are dealing with only a finite number of scenarios (in $\mathcal{S}$).

2.2. Discoveries as a Result of Decisions

In §1 the 'standard' stochastic programming model was extended to allow for situations when the probability measure is decision dependent. The problems that we are going to consider are stochastic linear programs that fit into the class of problems identified by the formulation in (3). In such models the decisions have been split into a (boolean) vector d specifying a probability measure μ^d, and the usual vector X that determines the activity levels; the relationship between these decision being expressed through the constraint $(d, X) \in \mathcal{K}$.

In our formulation of (multistage) stochastic linear programs, the probability measure of the random elements was identified with a scenario tree $\mathcal{S}$ with probabilities p_s attached to each individual scenario $s \in \mathcal{S}$. Since each d corresponds to a different measure, this now translates into (different) scenario trees $\mathcal{S}(d)$, indexed by d with $p(d;s)$ as the probabilities attached to the scenarios s of $\mathcal{S}(d)$.

Similarly, the constraints of the stochastic program will depend on the choice of d. For $d \in \{0,1\}^q$, let

$$\mathcal{K}(d) = \{x \mid (d,x) \in \mathcal{K}\}.$$

To stay in the linear framework, we have to assume that for all d, the set $\mathcal{K}(d)$ is itself determined by a finite number of linear equations and linear inequalities, i.e., is a convex polyhedral set.

Thus to each choice of d corresponds a (standard) stochastic linear program of the form:

$$\min \sum_{s\in\mathcal{S}(d)} [p(d;s)f(s;X(s))] = E^d\{f(\cdot, X(\cdot))\}$$

subject to

$$A(s)X(s) \geq b(s), \quad s \in \mathcal{S}(d) \tag{11}$$

$$x_i^t(s) \in \{0,1\}, \quad i \in I^t,\ t = 1,\ldots,T, \quad s \in \mathcal{S}(d) \tag{12}$$

$$x_i^t(s) \geq 0 \ \ i \in \{1,\ldots,n^t\} \setminus I^t,\ t = 1,\ldots,T, \quad s \in \mathcal{S}(d) \tag{13}$$

$$X \in \mathcal{K}(d) \tag{14}$$

$$X(\cdot) \in \mathcal{N}(d) \tag{15}$$

using here the shorthand notation $\mathcal{N}(d)$ to denote $\mathcal{N}_{\mathcal{S}(d)}$.

Since implementability depends on the timing at which information will become available, the constraint imposing implementability must also be formulated in terms of the scenario tree $\mathcal{S}(d)$. As already indicated in §1, in the class of problems we have chosen to illustrate our approach, the vectors d —recall that they are first stage decisions— specify at which time information about the values taken by certain random variables becomes available.

2.3. Example

In the introduction we illustrated our notation with a small example that is an extension of the model given by Jorjani et al. [11] for the optimal selection of subsets of sizes under demand uncertainty. We continue that example with more detail here. A product is available in a finite number of sizes, and demand for a smaller size can be met by substituting a larger size. However, this substitution comes at a certain cost. The objective is to minimize the expected cost of satisfying the demand for all different sizes. Binary variables model the selection of sizes, while production and substitution quantities are represented by continuous variables. Details for the formulation are summarized in Appendix A.

In addition to the demand uncertainty, we introduce uncertainty regarding the production costs. When we choose to produce a size, we will learn about its production costs. So q matches the number of sizes to be produced and vectors in $\{0,1\}^q$ have an element for each size and correspond directly to the decision

variable that indicates production of the size. In this way a production decision can also be viewed as an investment in information as well as production.

In Appendix A, we present the data for an instance with two periods and three different sizes. Because the demand has to be met, we know that the largest size always has to be produced, and in the problem considered here the cost of producing the largest size is deterministic. If we neglect the cost uncertainty and use the expected production costs for the two smallest sizes, the optimal first-stage decision is to produce either size 1 or size 2 (size 3 is always produced). Both these decisions have a total cost of 25140.

When uncertain production costs are introduced, we get different results. The four feasible first-stage production decisions and associated total costs are reported in Table 1. The vector d gives the production decision for the sizes 1 and 2 where "0" represents not produce while "1" is produce. We see that by

d	Minimum $E^d\left\{f\left(\cdot, X(\cdot)\right)\right\}$
0, 0	\$25180
1, 0	25130
0, 1	25115
1, 1	25220

Table 1
Production costs, Fixed first stage decisions

producing size 2 (and 3), the expected cost has been reduced to 25115. Even though the reduction in total cost is not large, it is interesting to see that the optimal production policy has been dramatically altered by introducing the cost uncertainty in the problem.

3. Bounds

In this section we develop bounds to be used in an implicit enumeration algorithm. There is no universal scenario tree attached to a stochastic (linear) program when the distribution of (some of) the random parameters is decision dependent. However, once a timing of discovery for the values realized by the random variables has been specified (or assumed) a scenario tree can be drawn.

Armed with a scenario tree, we can search for solutions for the associated optimization problem and use the result as a bound of some sort.

The bounds that we derive are based on the use of *branching* (or *partial*) vectors, denoted $d^{\#}$, in $\{0,1,\#\}^q$. The components $d_j^{\#}$ of such vectors take on the values 0, 1 or #. The interpretation we attach to these values is as follows:

$d_j^{\#} = 1$ means that information about the j-th (family of) random variable(s) will come as early as possible (here in stage two);

$d_j^{\#} = 0$ means that information about the j-th (family of) random variable(s) will come as late as possible (in our examples, this will be stage three);

$d_j^{\#} = \#$ means that there has been no decision yet about this particular variable.

We refer to vector elements with a 0 or 1 as *determined* and those with # as *undetermined*.

To each branching vector $d^{\#}$ is associated a collection, say $\mathcal{D}(d^{\#})$, of decision vectors d obtained by replacing all # entries in $d^{\#}$ by either 1 or 0; these correspond one-to-one with a set of 'standard' stochastic optimization problems that we will call $\mathcal{P}(d^{\#})$. Within the collection $\mathcal{D}(d^{\#})$, let's denote by $(d^{\#})^e$ the vector in $\{0,1\}^q$ obtained by replacing every (undetermined) entry # in $d^{\#}$ by 1, and by $(d^{\#})^e$ the vector obtained by replacing every (undetermined) entry # in $d^{\#}$ by 0. The two vectors correspond respectively to the cases when information associated with the variables d_j for which $d_j^{\#} = \#$ is received as early as possible (the index e stands for early) or late (the index l stands for late). Since $(d_j^{\#})^e = (d_j^{\#})^l$ for all j for which $d_j^{\#} \neq \#$, it follows that the scenario trees

$$\mathcal{S}^e := \mathcal{S}((d^{\#})^e), \qquad \mathcal{S}^l := \mathcal{S}((d^{\#})^l)$$

have common components, but differ in those parts that correspond to the families of random variables associated with $d_j^{\#} = \#$. Because discovery at a later time imposes more restrictions on implementable solutions, for all $d \in \mathcal{D}(d^{\#})$, one has

$$\mathcal{N}((d^{\#})^e) \subseteq \mathcal{N}(d) \subseteq \mathcal{N}((d^{\#})^l) \quad \forall d \in \mathcal{D}(d^{\#});$$

recall that $\mathcal{N}(d)$ is shorthand for $\mathcal{N}_{\mathcal{S}(d)}$.

In the calculation of bounds for the collection $\mathcal{P}(d^{\#})$ of stochastic programs associated with $d \in \mathcal{D}(d^{\#})$, we want to impose the constraints on X from among

those generated by the restriction $(d, X) \in \mathcal{K}$ only if they come from the already determined components of $d^{\#}$. So, we let

$$\mathcal{K}(d^{\#}) = \bigcup\nolimits_{d \in \mathcal{D}(d^{\#})} \mathcal{K}(d).$$

We shall only deal with problems where this set is a nonempty convex polyhedral set. So, the linear program,

$$\min \sum_{s \in \mathcal{S}^e} [p^e(s) f(s; X(s))]$$

subject to

$$A(s)X(s) \geq b(s), \quad s \in \mathcal{S}^e \tag{16}$$

$$x_i^t(s) \in \{0, 1\}, \quad i \in I^t,\ t = 1, \ldots, T, \ \ s \in \mathcal{S}^e \tag{17}$$

$$x_i^t(s) \geq 0 \ \ i \in \{1, \ldots, n^t\} \setminus I^t,\ t = 1, \ldots, T, \ \ s \in \mathcal{S}^e \tag{18}$$

$$X \in \mathcal{K}(d^{\#}) \tag{19}$$

$$X(\cdot) \in \mathcal{N}^e \tag{20}$$

provides a lower bound for the values of all stochastic programs in $\mathcal{P}(d^{\#})$. Let's denote this *lower bound* by $L(d^{\#})$. It is also important to note that one can always replace $L(d^{\#})$ by quantities that provide lower bounds for the minimization involved in its definition. This is particularly relevant for integer and mixed integer problems where lower bounds are often readily available via relaxation even when exact minimization might be quite difficult.

If the superscripts e in the linear program above is replaced with superscripts l (late branching), we get a program that provides an *upper bound*, $U(d^{\#})$, for the values of all stochastic programs in $\mathcal{P}(d^{\#})$ provided that certain restrictions hold on the structure of $\mathcal{K}(d)$. This fact is not exploited in our algorithms because they use full vectors when calculating upper bounds. An upper bound computed by a full decision vector, $U(d)$, is globally valid, at least as tight and requires no greater computational effort.

3.1. Example

To illustrate the notation, we continue with the example of §2.3. Upper and lower bounds for branching vectors are displayed in Table 2. The two vector elements shown correspond to fixing the decisions to produce or not produce the two parts that have an uncertain, but discoverable, cost. We can see that using the best upper bound given in the table we are able to fathom the partial vector

$d^{\#}$	$U(d^{\#})$	$L(d^{\#})$
0, #	25140	25080
1, #	25130	25056
#, 0	25140	25130
#, 1	25115	25095

Table 2
Bounds for the Example Given in §2.3

(#,0). The upper bounds $U(d^{\#})$ are valid because $\mathcal{K}(\cdot)$ is completely separable. When comparing to the results in Table 1, we see that the upper bound for the vector (#,1) is the same as the minimum cost reported for the full vector (0,1), the optimal d-vector for this problem.

4. An Algorithm

The bounds introduced in the preceding discussion can be directly used to create a branch and bound algorithm. In this section we introduce a more memory intensive implicit enumeration algorithm that relies on an ability to store information about each full vector in $\{0,1\}^q$. The algorithm is developed only for situations where there are two decision stages (three realization times). This algorithm is clearly not appropriate for large q, but given present (and projected near term) computer capabilities, exact solutions for such problems will require further developments. The advantage of retaining information in this fashion is that we are able to make good branching decisions and we are able to quickly reduce the search space for good upper bounds.

In order to compress the space required to present the algorithms, we make use of the following notation:

$\leftarrow$ means assignment. For example, $j \leftarrow j+1$ means that the value of j is incremented.

$d^{\#}_j \leftarrow i$ means that branching vector $d^{\#}$ is modified so that element j takes the value i.

$(d^{\#}_j = i)$ refers to a branching vector for which element j is equal to i and all other elements are undetermined.

$d^{\#} \subseteq d$ is true if the determined values in $d^{\#}$ match the corresponding vector elements in d. For example, $(\#, 1, \#) \subseteq (0, 1, 1)$, but $(\#, 1, 0) \not\subseteq (0, 1, 1)$.

A conceptual version of the algorithm called I1 is shown in Figure 2. This is not written to convey the most efficient computer implementation, but to display the concepts.

Necessary initialization of variables are done in step 1. The list of lower bounds, $\mathcal{L}[d]$ has all elements set to zero, the branching vector $d^{\#}$ is assigned undetermined vector elements, the set D^* is assigned the set of all possible d-vectors, J and J' are index sets with q elements and B is an index set for branching decisions. In step 2 the algorithm calculates lower bounds for all possible branching vectors where only one decision is fixed and the others are undetermined. These calculated bounds are incorporated in the list $\mathcal{L}[d]$, that represents the highest lower bound for each full vector, $d \in \{0,1\}^q$, which serves as the index set for the list. The list V is of length q and is used in step 5 for deciding branching order. In step 3 the full vector with lowest lower bound, $\operatorname{argmin} \mathcal{L}[d]$, is used for computing a global upper bound. Thus $\mathcal{U}$ represents the best feasible solution found so far to the overall minimization problem. This upper bound is in step 4 used to prune full vectors for which the best lower bound is higher than the upper bound. The set D^* thus represents all full vectors in $\{0,1\}^q$ that have not yet been bounded out.

Step 5 essentially repeats these steps in a slightly more general way to consider branching with a smaller number of undetermined components. While the number of possible full vectors is greater than one or we have not branched on all possible decisions (in the case with non-unique optimal solution), the algorithm continues the branching. The remaining elements in D^* are investigated, and decisions that are identical for all vectors in D^* are fixed. The determined elements of the branching vector $d^{\#}$, are the decisions proven to be a part of the optimal full vector. The set B represents the variables for which the algorithm branches, and indexes for fixed decisions are removed from B. The index set J' represents all decisions not yet fixed. The order in which the branching is performed (i.e., assignment of j' to B) is determined heuristically, using V_j to estimate the relative efficiency of variable changes. The set B gives the list of indexes for which full factorial branching is to be done. The mechanics of this branching is described by the function Ψ as shown in Figure 1. The function Ψ returns a set of $2^{|B|}$ branching vectors, and this set represents all possible branching vectors that are undetermined for the elements $(J' \setminus B)$, and determined for the fixed decisions and the branching variables. A lower bound is calculated for the branching vectors $\hat{d}^{\#}$ and the list $\mathcal{L}$ is updated. An upper bound is calcu-

$\nu \leftarrow \emptyset$
FORACH $i \in \{0, \ldots, (2^{|B|} - 1)\}$ (i have a binary representation)
  $\tilde{d}^{\#} \leftarrow d^{\#}$
  $j \leftarrow 1$
  FORACH $b \in B$
    $\tilde{d}^{\#}_b \leftarrow i_j$ (i_j is the j-th digit in i)
    $j \longleftarrow j + 1$
  $\nu \leftarrow \nu \cup \{\tilde{d}^{\#}\}$
RETURN ν

Figure 1. Definition of the function $\Psi(d^{\#}, B)$

lated for the full vector with lowest lower bound, and all decision vectors, d, for which the lower bound is higher than the global upper bound, is removed from the set D^*. Algorithm I1 terminates when the set D^* has just one element, the optimal full vector, or we have a non-unique optimal solution (i.e., the V's are all negative). Upon termination, D^* must be the optimal set.

5. Computational Experience

5.1. The Subcontracting Problem

The problem of selecting an optimal subset of sizes that we have used as an example has limited usefulness for computational experiments because we can solve only small instances to optimality. This is due to the fact that the problem has both integer and continuous variables and has complicated constraints. In this section we introduce a problem for which we develop a continuous and a pure integer version.

In the *subcontracting problem* we consider a situation where a manufacturer must use a number of different components in the production of some new items. The components may be produced "in house" or purchased from a foreign subcontractor. The subcontractor offers these components at a given price, but in the future it is possible that an import tax will be added to the price. Hence, the subcontract cost is uncertain and the timing of realization is not influenced by the

1. $\mathcal{L} \leftarrow 0; \quad d^{\#} \leftarrow \#; \quad D^* \leftarrow \{0,1\}^q; \quad J \leftarrow J' \leftarrow \{1,\ldots,q\}; \quad B \leftarrow \emptyset$
2. FORACH $j \in J$

   FORACH $i \in \{0,1\}$

   FORACH $d \in \{0,1\}^q$

   IF [$((d_j^{\#} = i) \subset d)$ AND $(\mathcal{L}[d] < L(d_j^{\#} = i))$]

   $\mathcal{L}[d] \leftarrow L(d_j^{\#} = i)$

   $V_j \leftarrow abs(L(d_j^{\#} = 0) - L(d_j^{\#} = 1))$
3. $\mathcal{U} \leftarrow U(\text{argmin}\, \mathcal{L}[d])$
4. FORACH $d \in \{0,1\}^q$

   IF $\mathcal{U} < \mathcal{L}[d]$
   $D^* \leftarrow D^* \setminus \{d\}$
5. WHILE [$(|D^*| > 1)$ AND $((\max V_j) \geq 0)$]

   FORACH $j \in J'$

   FORACH $i \in \{0,1\}$

   IF $[(d_j^{\#} = i) \subset d \;\forall\; d \in D^*]$
   $d_j^{\#} \leftarrow i$
   IF $(j \subset B)$ $B \leftarrow B \setminus \{j\}$
   $J' \leftarrow J' \setminus \{j\}$

   $\{j'\} = \text{argmax}\, V_j$
   $B \leftarrow \{j'\}$
   $V_{j'} = -1$
   FORACH $\hat{d}^{\#} \in \Psi(d^{\#}, B)$

   FORACH $d \in D^*$

   IF [$(\hat{d}^{\#} \subseteq d)$ AND $(\mathcal{L}[d] < L(\hat{d}^{\#}))$]
   $\mathcal{L}[d] \leftarrow L(\hat{d}^{\#})$

   $\mathcal{U} \leftarrow \min(\mathcal{U}, U(\text{argmin}\, \mathcal{L}[d]))$
   FORACH $d \in D^*$

   IF $\mathcal{U} < \mathcal{L}[d]$
   $D^* \leftarrow D^* \setminus \{d\}$

Figure 2. Definition of Algorithm I1.

values of the decision variables in this model. Since the manufacturer has no cost history for these new components the in-house production costs are uncertain; however, prior probabilities can be given. The uncertainty for a given component can be resolved by producing that component, or a similar one. Each of the components belongs to some *family* of components, with family membership defined by the property that the discovery of cost information is shared within a family, but not between families. We present two models for this problem: one continuous and the other discrete.

5.1.1. A Continuous Problem

To model a learning curve for costs in a very simple way, we assume that if there is a decision to produce at least a certain threshold fraction, α_f, the in-house production costs for family f will be known. There are n different components, and we define e to be a vector of length n whose elements are all 1. Further e_f is a vector of the same length whose elements are 1 for those indexes that are members of family f and 0 for all other elements. The in-house production is constrained by the availabilty of production capacity, and there are m constrained resources (21). There is also a limit on maximum increase in in-house production from one period to the next (22). We also include constraints to enforce $X \in \mathcal{K}(d)$ (24). The problem we want to optimize can formally be written as:

$$\min_{d\in\{0,1\}^q} \min_X E^d \left\{ \sum_{t=1}^{T} \left[\langle c(s), x^t(s)\rangle + \langle h(s), (e - x^t(s)\rangle \right] \right\},$$

subject to:

$$A\left(e - x^t(s)\right) \leq b, \quad t = 1,\ldots,T,\ s \in \mathcal{S}(d) \tag{21}$$

$$\langle e, x^t(s) - x^{t+1}(s)\rangle \leq \beta \quad t = 1,\ldots,T-1, s \in \mathcal{S}(d) \tag{22}$$

$$0 \ \leq \ x_j^t(s) \leq 1, \quad j = 1,\ldots,n,\ t = 1,\ldots,T,\ s \in \mathcal{S}(d) \tag{23}$$

$$\ell_f(d) \leq \langle e_f, (e - x^1(s))\rangle < u_f(d), \quad f = 1,\ldots,q,\ s \in \mathcal{S}(d) \tag{24}$$

$$X \in \mathcal{N}(d) \tag{25}$$

where we have introduced the following notation:

$1,\ldots,n$	the set of component indexes
$1,\ldots,q$	the set of family indexes
$1,\ldots,m$	the set of resource indexes
$x_j^t(s)$	quantity (a decision) of component j to subcontract in time

	t under scenario s
$(1 - x_j^t(s))$	quantity (a decision) of component j to produce in-house
$c_j(s)$	the uncertain cost of subcontracting component j
$h_j(s)$	the uncertain, discoverable cost of producing component j
a_{ij}	capacity requirement for component j on resource i
A	a_{ij} matrix
$b_1, \ldots, b_m$	available production capacities
β	maximum increase in in-house production
$[\ell_f(d), u_f(d))$	equals $[0, \alpha_f)$ or $[\alpha_f, \infty)$ if $d_f = 0$ or $= 1$, respectively

We consider a problem with two decision stages ($T = 2$). Each element of a vector d is a one if there will be enough in-house production (during the first stage) in the family to learn the cost and zero otherwise. To create the lower bounding problem associated with a branching vector $d^{\#}$, the constraints (24) for $X \in \mathcal{K}(d)$ are replaced by:

$$0 \leq \langle e_f, (e - x^1(s)) \rangle < \alpha_f, \quad \text{if } d_f^{\#} = 0, \quad s \in \mathcal{S}(d) \tag{26}$$

$$\alpha_f \leq \langle e_f, (e - x^1(s)) \rangle, \quad \text{if } d_f^{\#} = 1, \quad s \in \mathcal{S}(d). \tag{27}$$

Note that there are no constraints associated with the elements of $d^{\#}$ that are 'undetermined', i.e., for those families f for which $d_f^{\#} = \#$ the interval $[\ell_f(d), u_f(d))$ equals $[0, \infty)$.

5.1.2. A Problem with 5 Families and 12 Components

We consider a problem with 12 components that belong to 5 different families, and $\alpha_f = 0.5$ for all $f \in 1, \ldots, q$. The uncertain costs can be found in the first part of Table 3. The fact that the scenario set depends on the decision values, and furthermore the fact that it is the cross product of the c and h space, makes it difficult to index the scenarios. We have used prime and double prime to distinguish the (equally likely, in this example) possibilities in each space. The in-house capacity is 9 units of each resource and each component's resource requirements are listed in the $a_{j,k}$-columns in Table 3.

When ignoring the decision dependent information discovery and using the expected costs for the in-house production, we find the minimum expected total cost to be 226.53. In the first period components 1 and 3, and 2% of component

f	j	c'_j	c''_j	h'_j	h''_j	a_{1j}	a_{2j}	a_{3j}
1	1	5	6.25	4.8	5.5	1.8	2.0	2.0
	2	6	7.5	5.8	6.5	1.8	1.7	2.0
	3	7	8.75	6.8	7.5	2.0	1.5	1.5
2	4	8	10	7	9.5	1.0	1.3	1.7
	5	9	6.25	4.8	5.5	1.2	1.1	1.1
3	6	8	10	6	10.5	1.0	2.0	1.5
	7	9	11.25	7.5	11	1.2	1.5	1.5
4	8	10	12.5	8.5	12	1.2	1.4	1.9
	9	12	15	11	13.5	1.2	1.7	1.0
5	10	10	12.5	8.5	12	1.2	1.2	2.0
	11	12	15	10.5	14	1.5	1.6	2.0
	12	14	17.5	12	16.5	2.0	1.8	1.4
6	13	6	7.5	5	8	1.2	1.4	1.6
	14	7	8.75	6	8	1.8	1.7	1.3
7	15	8	10	7	10	1.2	1.2	2.0
	16	9	11.25	8	11	1.5	1.6	1.7
	17	10	12.5	9	12	2.0	2.0	1.4

Table 3
Input data, subcontracting, continuous problem

2 are produced in-house. In other words, all components in the families 2, 3, 4 and 5 are subcontracted.

When the uncertain production costs are included in the model, we find it optimal to produce one member from all of the families in the first period. The expected cost of this decision is 222.41. By taking the uncertain production costs into consideration, the optimal first period decisions have changed dramatically. When solving this problem by use of the I1 algorithm, it was necessary to solve 12 subproblems before the provable optimal solution was found. Complete enumeration would have required solution of 32 subproblems.

5.1.3. A Problem with 7 Families and 17 Components

In this example we consider 17 components belonging to 7 different families. The cost coefficients and capacity requirements are those listed in Table 3, and there are 9 units of capacity available of each resource.

When solving the deterministic problem with expected cost coefficients, we find it optimal to produce all of components 3 and 14, and 2 % of component 1 in-house. This decision policy has an expected cost of 311.38. When the uncertain production costs are included in the model, the optimal policy is to produce a member of all but the first family in the first period. This gives an expected cost of 305.81. By using the I1 algorithm, the optimal solution is found after solving 26 subproblems. Complete enumeration would have required solution of 128 subproblems.

5.2. An Integer Problem

In this example we let the production decisions be represented by integer variables, and as soon as one of the family members is produced the production costs for all of the family members are known. The formal model is the same as for the continuous case except that the learning threshold value $\alpha_f = 1$ for all f, and the constraint (23) is written as

$$x_j(s,t) \in \{0,1\}, \quad j \in J, t \in 1,\ldots,T, s \in S(X). \tag{28}$$

5.2.1. A Problem with 5 Families and 12 Components

Also here we first look at a problem with 12 components that belong to 5 different families. The uncertain costs can be found in the first part of Table 3. There is one in-house resource and it has a capacity of 8 units; utilizations (b_j) are 2 for parts in families 1 and 5 and 1 for families 2,3, and 4.

When using the expected costs for the in-house production, we find an optimal solution with an expected cost of 226.32, and in the first period, all members of the first family are produced in-house, but all others are subcontracted. When the uncertain production costs are included in the model, the optimal policy is to produce one member from each of the families 2, 3, 4 and 5 in the first period in-house and subcontract all members of family 1. The expected cost of this decision is 222.41. The optimal first period decisions have been altered considerably. None of the components now produced would have been produced if production uncertainty was not included in the model.

When solving this problem by use of the I1 algorithm, it was necessary to solve 16 subproblems before the provable optimal solution was found (a 50% savings over enumeration.)

5.2.2. A Problem with 7 Families and 17 Components

When considering 17 components belonging to 7 different families we use the same cost coefficients as used in §5.1.3. All components require 1 unit of capacity, and the total capacity is 8 units.

Also in this example we find it optimal to produce only the first family, if only the expected in-house costs are used. This decision policy has an expected cost of 309.58. When the uncertain production costs are included in the model, the optimal policy is to produce a member from all but the first family. This gives an expected cost of 303.58. By using the I1 algorithm, the optimal solution is found after solving 30 subproblems, which is a large saving over enumeration.

5.2.3. Harder Capacity Constraints

The problem we are considering here is rather similar to the one in 5.2.1, but here there is a second capacity constraint for a resource with 9 units of capacity. For this resource the utilizations are 1 for parts in families 1, 2 and 2 for parts in families 3,4, and 5.

This instance is just beyond the envelope of solution using our implementation that solves subproblems using CPLEX version 3 [7] (the subproblems also could not be solved using version 4). Most of the subproblems generated could not be solved to optimality within a reasonable amount of time so the lower bounds were generated using lower bounds given by the relaxations solved by the branch and bound algorithm used to solve the integer subproblems. At termination algorithm I1 was able to bound out all decisions except in-house production of all five families or in-house production of all families except family 1.

6. Variational Analysis

Thus far we have introduced a new modelling concept and outlined methods for finding exact solutions to instances of small to moderate size. In order to enhance our understanding of the model and to facilitate future work on heuristics for it, we describe methods for estimating the effects of perturbations of an optimal d vector, d^*.

It is easier at this point to return to the 'abstract' formulation of the problem featured in the Introduction. More precisely,

$$\min_{x,d} E^d f(x) = \int_{\Xi} f(\xi; x)\, \mu^d(d\xi) \text{ such that } (\mu^d, x) \in \mathcal{K} \subset M \times I\!R^n,$$

where $d = (d_1, \ldots, d_q) \in \{0,1\}^q$ is a boolean vector identifying certain options fixing in the process the probability measure μ^d, and $\mathcal{K}$ determines the constraints linking the decision x to the choice of μ. For a given d, or equivalently μ^d, let

$$x^d \in \operatorname{argmin}\{ E^d f(x) \mid x \in \mathcal{K}^d\}$$

where $\mathcal{K}^d = \{x \,|\, (\mu^d, x) \in \mathcal{K}\}$.

Without going through a detailed analysis of the structure of $\mathcal{K}$ it is not possible to obtain computationally useful optimality conditions for x^d that would identify x^d as the optimal solution of the overall problem. Our goal here will be much more limited, viz. to state necessary optimality conditions that can also be used to pass from x^d to a potentially better option/solution combination. Let's begin with the following observation.

Lemma 1. Let M^0 be the space of (nonnegative) measures defined on Ξ, and suppose that for all $\mu \in M^0$, the function

$$\mu \;\mapsto\; E^\mu f = \int_\Xi f(\xi;\cdot)\,\mu(d\xi)$$

is well-defined; set $E^\mu f(x) = \infty$ whenever the function $\xi \mapsto f(\xi; x)$ is not bounded above by a μ-summable function. Then $\mu \mapsto E^\mu f$ is linear.

This is an immediate consequence of the properties of the integral. We are going to exploit this as follows: Let M as before denote the space of probability measure on Ξ. This set M is convex, i.e., given any pair of probability measures μ^0, μ^1 and $\lambda \in [0,1]$, one has

$$\mu^\lambda := (1-\lambda)\mu^0 + \lambda\mu^1 \;\in\; M$$

and

$$E^{\mu^\lambda} f \text{ is well-defined.}$$

This will allow us to compute a directional derivative of $E^\mu f$ at a point μ^d in a direction $\mu^c - \mu^d$ where μ^c is another probability measure. It is convenient to introduce the following notation:

$$F(\mu, x) := E^\mu f(x)$$

and, with $\mu^\lambda = (1-\lambda)\mu^0 + \lambda\mu^1$,

$$dF(\mu^0, x)(\mu^1 - \mu^0) = \liminf_{\lambda \searrow 0} \lambda^{-1}\, F(\mu^\lambda, x)$$

as the '*directional derivative*' of F at (μ^0, x) in direction $\mu^1 - \mu^0$. This directional derivative reflects the incremental change in the optimal value of the problem if rather than working with μ^0 we are going to let μ^1 dictate the choice of an optimal x. This (sub) derivative is computed for a fixed x.

Usually it is not too difficult to compute this directional derivative. Indeed, under the integrability conditions specified in Lemma 1,

$$dF(\mu^0, x)(\mu^1 - \mu^0) = \int_\Xi \lim_{\lambda \searrow 0} f(\xi; x)\lambda^{-1}[\,(1-\lambda)\mu^0(d\xi) + \lambda\mu^1(d\xi)\,].$$

If μ^0, μ^1 are absolutely continuous, i.e., one can associate with μ^0, μ^1 density functions h^0, h^1 defined on Ξ, then

$$dF(\mu^0, x)(\mu^1 - \mu^0) = \int_\Xi f(\xi; x)(h^1(\xi) - h^0(\xi))\, d\xi.$$

On the other hand, if μ^0, μ^1 are discretely distributed, say $\Xi = \{\,\xi^\ell, \ell = 1, \dots\}$ and $\mu^0(\xi^\ell) = p^0_\ell$, $\mu^1(\xi^\ell) = p^1_\ell$, then

$$dF(\mu^0, x)(\mu^1 - \mu^0) = \sum_\ell f(\xi^\ell; x)(p^1_\ell - p^0_\ell).$$

Thus, if $x^d \in \operatorname{argmin} E^d f$ on $\mathcal{K}^d$ and

$$\text{for all } c \in \{0,1\}^q : \quad dF(\mu^d, x^d)(\mu^c - \mu^d) \geq 0$$

it follows from the linearity of $\mu \mapsto E^\mu f$ that x^d is locally an optimal solution. On the other hand, if

$$\text{for some } c \in \{0,1\}^q : \quad dF(\mu^d, x^d)(\mu^c - \mu^d) < 0$$

there is a potential decrease that could result from going from option 'd' to option 'c'. Note however that this cost reduction might not be realizable because the constraints $(\mu, x) \in \mathcal{K}$ might actually exclude (μ^c, x^d) from the feasible set. Nonetheless, the calculation of the directional derivative suggests directions of descent, and thus can be exploited algorithmically.

7. Conclusions and Directions for Further Research

In this paper we have extended the range of models considered by researchers in stochastic programming by explicitly recognizing that a scenario specifies not only the realized values of random variables, but also that the timing of the realization and the timing of information discovery can be influenced by decisions.

We have proposed bounds and algorithms for the case where the distributions and the variables controlling information discovery are all zero-one and only affect first stage decisions. We then illustrated that these algorithms can be used to solve instances of integer, mixed integer, and continuous problems of moderate size. The instances also demonstrated using three different examples the intuitively obvious idea that inclusion of the information discovery effects of decisions can have a dramatic qualitative effect on the optimal decision.

Development of good heuristics will improve the performance of exact algorithms such as I1 by providing better upper bounds. Also, heuristics must be used in order to attack larger instances. This is particularly true for integer and mixed integer problems. For the problem introduced in §2.3 we are not able to solve realistic sized instances to optimality, and for the integer version of the subcontracting problem introduced in §5, only small to moderate sized instances can be solved. The development of heuristics is left as future research, but we have provided assistance in the form of variational analysis that can be used to guide the search.

In terms of applications, one can quickly imagine many possibilities in addition to the production planning examples that we have given here. The abstract model, solution methods, and variational analysis open up these possibilities, which we leave as our primary contribution to modelling stochastic programs.

Acknowledgments

This work was supported in part by grants from the National Science Foundation (Division Mathematical Sciences) and the Norwegian Research Council.

Appendix A — The Sizes Problem

Complete details of the problem and larger instances are provided by Jorjani et al. [11]; here we give a summary and the data that we used. The notation from that paper is retained to the extent possible.

Single Period Deterministic Model

Suppose a product is available in a finite number N of sizes where 1 is the smallest size and N is the largest size. Further, suppose size i is substitutable

for size j if $i > j$, i.e., larger sizes may fulfill demand for a smaller size. Let D_i, $i = 1, \ldots, N$ denote the demand for size i.

We introduce a cost structure for the production of the product. Let p_i, $i = 1, \ldots, N$ be the unit production cost for size i. Generally $p_i > p_j$ for $i > j$. Let σ be the set up cost for producing units of any size and ρ be the unit penalty cost of meeting demand for size j with a larger size i.

We need the following decision variables:

$$z_i = \begin{cases} 1 \text{ if we produce size } i \\ 0 \text{ otherwise} \end{cases}$$

$y_i =$ number of units of size i produced

$x_{ij} =$ number of units of size i cut to meet demand for size j, $j < i$.

Hence, in order to find the optimal subset of the N sizes to produce so as to satisfy demand, we solve the following integer linear program.

$$\min \sum_{i=1}^{N} (\sigma z_i + p_i y_i) + \rho \sum_{j<i} x_{ij}$$

subject to

$$y_i = D_i - \sum_{k>i} x_{ki} + \sum_{l<i} x_{il} \quad i = 1, \ldots, N, \tag{29}$$

$$y_i - M z_i \leq 0 \quad i = 1, \ldots, N, \tag{30}$$

$$z_i \in \{0, 1\} \quad i = 1, \ldots, N, \tag{31}$$

$$x_i, y_i \in \{\text{non-negative integers}\} \quad i = 1, \ldots, N, \tag{32}$$

Multiperiod, Stochastic Formulation

To produce a multiperiod formulation variables and data are subscripted with a period index t and we add an index for the scenario.

To model the idea that items produced in one period can be used as-is or reduced in subsequent periods, we use x_{ijt} to indicate that product i is to be used without reduction in period t if $i = j$ and with reduction otherwise. The y vector gives production quantities for each product in each period without regard to the period in which they will be used (and perhaps reduced for use). The formulation is essentially an extension of the single period formulation except that a capacity constraint must be added in the multiple period formulation.

$$\min \sum_{s\in\mathcal{S}} \Pr(s) \sum_{t=1}^{\tau} \left[\sum_{i=1}^{N} (\sigma z_{its} + p_i y_{its}) + \rho \sum_{j<i} x_{ijts} \right]$$

subject to

$$\sum_{j\geq i} x_{ijts} \geq D_{its} \quad s \in \mathcal{S},\ i = 1,\ldots,N,\ t = 1,\ldots,\tau \tag{33}$$

$$\sum_{t'\leq t} \left[\sum_{j\leq i} x_{ijt's} - y_{it's} \right] \leq 0 \quad s \in \mathcal{S},\ i = 1,\ldots,N,\ t = 1,\ldots,\tau \tag{34}$$

$$y_{its} - M z_{its} \leq 0 \quad s \in \mathcal{S},\ i = 1,\ldots,N,\ t = 1,\ldots,\tau \tag{35}$$

$$\sum_{i=1}^{N} y_{its} \leq c_{ts} \quad s \in \mathcal{S},\ t = 1,\ldots,\tau \tag{36}$$

$$z_{its} \in \{0,1\} \quad s \in \mathcal{S},\ i = 1,\ldots,N,\ t = 1,\ldots,\tau \tag{37}$$

Data

For illustration purposed we made use of a small version of this problem, where we consider two decision stages (and a third stage for realization of undiscovered costs) and production of three different sizes. The setup cost for producing each size is \$ 453. The deterministic unit production costs for size 3 , the largest size, is \$ 0.54. For the smallest size the discrete probability distribution has two values: \$ 0.48 and \$ 0.52, each with a 0.5 probability. For size number two the uncertain costs are \$ 0.50 and \$ 0.54, also here each has a 0.5 probability. The unit cutting cost, for substituting a smaller size with a larger, is \$ 0.008. The demand in the first period is 7500 for each of the three sizes. The uncertain demand is modeled by use of two scenarios, each with a probability of 0.5. The scenarios are 5000 of each size in the low demand case, and 10000 of each size in the high demand case. The total production capacity in each period is 30000 units.

References

[1] Z. Artstein, and R.J-B Wets, "Sensors and information in optimization under stochastic uncertainty," *Mathematics of Operations Research*, 28, 1993, 523-547.

[2] I.L. Averbakh, "An Additive Method for Optimization of Two-Stage Stochastic Systems with Discrete Variables," *Soviet Journal of Computer and System Sciences*, 28, 1990, 161-165.

[3] J.R. Birge, and R.J-B Wets, "Computing bounds for stochastic programming problems by means of a generalized moment problem," *Mathematics of Operations Research*, **12**, 1987, 149-162.

[4] S. Bjørnestad, Å. Hallefjord, K. Jörnsten "Discrete optimization under uncertainty: the scenario and policy aggregation technique," Working Paper 89/06, Chr. Michelsen Institute, Bergen, Norway 1989.

[5] H. Bjørstad, Å. Hallefjord, T. Hefting, "Decision trees with coupling constraints," Report Ref.CMI-No.30151-2, Chr. Michelsen Institute, Bergen, Norway 1988.

[6] C.C. Carøe, and R. Schultz, "Dual Decomposition in Stochastic Integer Programming," Technical Report, Institute of Mathematics, Universitetsparken 5, DK-2100, Copenhagen, Denmark, 1996.

[7] CPLEX Optimization, Inc. *Using the CPLEX Callable Library*, Suite 279, 930 Tahoe Blvd. Building 802, Incline Village, NV, 89451-9436, 1994.

[8] A. Gaivoronski, "Linearization methods for optimization of functionals which depend on probability measures," *Mathematical Programming Study* 28, 1986, 157-181.

[9] A. Gaivoronski, "Stochastic optimization techniques for finding optimal submeasures," in *Stochastic Optimization*, V.I. Arkin, A. Shiraev and R. Wets (Eds.), Springer-Verlag Lecture Notes in Control and Information Sciences 81, 351-363, Berlin, 1986.

[10] Y.-C. Ho, X.-R. Cao, *Perturbation analysis of discrete event dynamic systems*, Kluwer Academic Publisher, Boston, 1991.

[11] S. Jorjani, C.H. Scott, D.L. Woodruff, "Optimal Selection of a Subset of Sizes," working paper, GSM, UC Davis, Davis CA 95616, 1996.

[12] G. Laporte, and F.V. Louveaux, "The Integer L-Shaped Method for Stochastic Integer Programs with Complete Recourse," *OR Letters*, 13, 1993, 133-142.

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

[14] G.Ch. Pflug, "On-line optimization of simulated Markovian processes," *Mathematics of Operations Research* **15** 1990, 381-395.

[15] A. Prékopa, *Stochastic Programming*, Kluwer Academic Publisher, Dordrecht, 1995.

[16] R.Y. Rubinstein, and A. Shapiro, *Discrete event systems: sensitivity analysis and stochastic optimization by the score function*, J. Wiley & Sons, New York, 1993.

[17] R. Schultz, L. Stougie, and M.H. van der Vlerk, "Solving stochastic programs with complete integer recourse: a framework using Gröbner bases, Discussion Paper 9562, CORE, Louvain-la-Neuve, Belgium, 1995.

[18] R. Schultz, L. Stougie, and M.H. van der Vlerk, "Two-stage stochastic integer programming: a survey," *Statistica Neerlandica* 50, 1996, 404-416.

[19] S. Takriti, J.R. Birge, and E. Long, "Lagrangian Solution Techniques and Bounds for Loosely-Coupled Mixed-Integer Stochastic Programs," Technical Report, Department of IEOR, University of Michigan, Ann Arbor, MI, USA, 1994.

[20] S.R. Tayur, R.R. Thomas, and N.R. Natraj, "An algebraic geometry algorithm for scheduling in the presence of setups and correlated demands," *Mathematical Programming* 69, 1995,

369-401.