Mathematical Programming Approaches for Generating p-Efficient Points

Miguel Lejeune

Department of Decision Sciences, George Washington University, Washington, DC, 20052,
mlejeune@gwu.edu

Nilay Noyan

Manufacturing Systems/Industrial Engineering Program, Sabancı University, Tuzla, 34956 Istanbul, Turkey,
nnoyan@sabanciuniv.edu

Abstract: Probabilistically constrained problems, in which the random variables are finitely distributed, are non-convex in general and hard to solve. The p-efficiency concept has been widely used to develop efficient methods to solve such problems. Those methods require the generation of p-efficient points (pLEPs) and use an enumeration scheme to identify pLEPs. In this paper, we consider a random vector characterized by a finite set of scenarios and generate pLEPs by solving a mixed-integer programming (MIP) problem. We solve this computationally challenging MIP problem with a new mathematical programming framework. It involves solving a series of increasingly tighter outer approximations and employs, as algorithmic techniques, a bundle preprocessing method, strengthening valid inequalities, and a fixing strategy. The method is exact (resp., heuristic) and ensures the generation of pLEPs (resp., quasi pLEPs) if the fixing strategy is not (resp., is) employed, and it can be used to generate multiple pLEPs. To the best of our knowledge, generating a set of pLEPs using an optimization-based approach and developing effective methods for the application of the p-efficiency concept to the random variables described by a finite set of scenarios are novel. We present extensive numerical results that highlight the computational efficiency and effectiveness of the overall framework and of each of the specific algorithmic techniques.

Keywords: Stochastic programming; probabilistic constraints; p-efficiency; outer approximation; valid inequalities

1. Introduction This study is devoted to the concept of p-efficiency (Prékopa, 1990) and proposes a new mathematical programming approach to generate p-efficient points.

Definition 1.1 *Let $p \in [0,1]$. A point $\mathbf{v} \in \mathbb{R}^n$ is called a p-efficient point of the probability distribution function F, if $F(\mathbf{v}) \geq p$ and there is no $\mathbf{y} \leq \mathbf{v}, \mathbf{y} \neq \mathbf{v}$ such that $F(\mathbf{y}) \geq p$.*

Along with mixed-integer programming (MIP) approaches (Kücükyavuz, 2009; Luedtke et al., 2010; Ruszczyński, 2002), robust optimization and approximation (Calafiore and Campi, 2005; Nemirovski and Shapiro, 2006), the concept of p-efficiency has been successfully and extensively used to solve probabilistically constrained stochastic programming problems (Charnes et al., 1958; Prékopa, 1970, 1973), in which the random vector has a multivariate discrete probability distribution (Dentcheva et al., 2001, 2002; Prékopa, 1990; Sen, 1992). The generic formulation of such problems reads:

$$\min \quad g(\mathbf{x}) \tag{1}$$

$$\text{subject to} \quad A\mathbf{x} \geq \mathbf{b} \tag{2}$$

$$\mathbb{P}\left(h_i(\mathbf{x}) \geq \xi_i, i = 1, \ldots, n\right) \geq p \tag{3}$$

$$\mathbf{x} \in \mathbb{R}^{m_1} \times \mathbb{Z}^{m_2}, \tag{4}$$

where $\mathbf{x}$ is the m-dimensional vector of m_1 continuous ($\mathbb{R}$) and m_2 integer ($\mathbb{Z}$) decision variables, $g(\mathbf{x}) : \mathbb{R}^{m_1} \times \mathbb{Z}^{m_2} \rightarrow \mathbb{R}$, $h_i(\mathbf{x}) : \mathbb{R}^{m_1} \times \mathbb{Z}^{m_2} \rightarrow \mathbb{R}$, $i = 1, \ldots, n$, and ξ is a n-dimensional random vector having a multivariate probability distribution. The set of deterministic constraints is represented by (2) with $A \in \mathbb{R}^{t \times m}$ and $\mathbf{b} \in \mathbb{R}^t$, while (3) is a *joint probabilistic* (*chance*) constraint that imposes that the n inequalities $h_i(\mathbf{x}) \geq \xi_i$ $(i = 1, \ldots, n)$ hold jointly with a large probability at least equal to p. The formulation (3) allows us to model dependencies between random variables and it does not assume any restrictions on the type of dependencies between components of the random vector ξ. Probabilistic problems with discretely distributed random variables are non-convex in general. They have been receiving significant attention over the last few years (Kress et al., 2007; Kücükyavuz, 2009, Lejeune, 2008, 2009; Lejeune and Ruszczyński, 2007; Luedtke et al., 2010; Saxena et al., 2010), and have been applied in a variety of fields (see Dentcheva, 2006 and Prékopa, 1995, 2003 for a review and a comprehensive list of references).

Existing solution methods for problem (1)-(4) based on the concept of p-efficiency involve an enumerative phase (Avital, 2005; Beraldi and Ruszczyński, 2002; Lejeune, 2008; Prékopa, 2003; Prékopa et al., 1998) for generating the pLEPs, which are then used to derive a deterministic equivalent reformulation. Preprocessing methods have been proposed to alleviate the enumeration of candidate points (Lejeune, 2008). Cone (Dentcheva et al., 2001), primal-dual (Dentcheva et al., 2004), and column generation (Lejeune and Ruszczyński, 2007) algorithms have been successfully employed and a convexification method (Dentcheva et al., 2001) has been proposed to obtain a tight relaxation of problem (1)-(4).

This study contributes to the literature in the following ways: *(i)* the p-efficiency concept is applied to a random vector whose possible values are discretized with a finite set of scenarios. Other applications of the p-efficiency concepts are generally proposed for random variables following a discrete probability distribution which has finite support; *(ii)* an exact mathematical programming method is proposed to generate pLEPs; *(iii)* a mathematical programming-based heuristic is developed for generating "quasi pLEPs". The term "quasi pLEP" refers to a point that is very close to being a pLEP, i.e., that enforces requirements that are marginally more demanding than those defined by a pLEP; *(iv)* a new preprocessing method is introduced to reduce the number of scenarios and so to reduce the complexity of the proposed mathematical programming formulations.

This paper is related to a recent study of Kress et al. (2007) who consider a specific variant

$$\min \quad \sum_{i=1}^{n} x_i \tag{5}$$

$$\mathbb{P}\left(x_i \geq \xi_i, i = 1, \ldots, n\right) \geq p \tag{6}$$

$$\mathbf{x} \in \mathbb{Z}_+^n, \tag{7}$$

of (1)-(4), in which there is no deterministic constraint besides the non-negativity and integrality restrictions (7), and the coefficients associated with the decision variables $\mathbf{x}$ in the objective function (5) and in the probabilistic constraint (6) are all equal to 1. Kress et al. reformulate this problem as an NP-hard, minmax multidimensional knapsack problem (MKP), and propose an enumerative algorithm to solve it. It can be seen that the optimal solution $\mathbf{x}^*$ of problem (5)-(7) defines a pLEP of the probability distribution function of the random vector ξ, when ξ has integer-valued components.

Some of the key features that differentiate the present study from Kress et al. (2007) are that *(i)* we propose a new formulation and a new mathematical programming based framework (instead of an enumerative one) for generating exact and quasi pLEPs. Both the formulation and the solution approach are applicable to the general probabilistically constrained optimization problem (1)-(4); *(ii)* our approach can be used to generate one as well as a series of pLEPs. The rationale for deriving a new solution framework comes from the observation of Kress et al. that their enumerative algorithm is outperformed by the state-of-the-art MIP solver CPLEX for problems of moderate to large size. The key feature of the proposed solution framework is that it is based on an outer approximation (Duran and Grossmann, 1986; Fletcher and Leyffer, 1994) algorithm, which is enhanced by a new family of valid inequalities, a fixing strategy, and a new preprocessing method that groups possible realizations of the random vector in *bundles* defining identical requirements. The proposed methods constitute an efficient alternative to the sometimes computationally intensive enumerative methods for generating pLEPs, the cardinality of which is finite yet unknown. An extended computational study analyzes the contribution of three specific algorithmic techniques (bundle preprocessing, strengthening valid inequalities and fixing strategy) integrated within the outer approximation method, and investigates the efficiency and effectiveness of the proposed heuristic algorithm. The computational results show that the heuristic approach allows the generation of quasi pLEPs in very short CPU times with significantly small optimality gaps, even when the number of scenarios used to describe the random variables is large.

The paper is organized as follows. Section 2 defines the optimization model to generate a single p-efficient point. Section 3 introduces a preprocessing method that reduces the complexity of the described mathematical programming formulation. Section 4 is devoted to the outer approximation solution framework. Section 5 describes the iterative procedure to generate a series of p-efficient points. Section 6 presents the computational results, while concluding remarks are given in Section 7.

2. Mathematical Programming Problem for Generating a Single pLEP We denote by S the finite set of scenarios characterizing the probability distribution of the random vector $\xi = (\xi_1, \ldots, \xi_n)^T$. Let d_i^s denote the realization of the random variable ξ_i under scenario s, $i = 1, \ldots, n, s \in S$, i.e., $\mathbf{d}^s = (d_1^s, \ldots, d_n^s)^T$ is the n-dimensional deterministic vector representing the joint realizations of the components $\xi_i, i = 1, \ldots, n$, under scenario s. The probabilities associated with scenarios are denoted by π_s, $s \in S$, where $\pi_s = \mathbb{P}(\xi = \mathbf{d}^s) > 0$ and $\sum_{s \in S} \pi_s = 1$. Without loss of generality, we assume that $\mathbf{d}^s$ is non-negative for all $s \in S$.

We consider the following formulation to generate a single pLEP of the probability distribution of the random vector ξ represented by the deterministic vectors $\mathbf{d}^s, s \in S$:

$$\min \quad \sum_{i=1}^{n} v_i \tag{8}$$

$$\text{subject to} \quad v_i \geq d_i^s \delta_s, \quad i = 1, \ldots, n, \ s \in S \tag{9}$$

$$\sum_{s \in S} \pi_s \delta_s \geq p \tag{10}$$

$$\delta \in \{0,1\}^{|S|} \tag{11}$$

$$\mathbf{v} \in \mathbb{R}_+^n, \tag{12}$$

where $\mathbf{v}$ represents a n-dimensional vector of decision variables, δ_s is a binary variable that can take value 1 if all constraints $v_i \geq d_i^s,\ i = 1, \ldots, n$, hold and that takes value 0 otherwise. Constraints (9) and (10) require that $\mathbf{v}$ satisfies the requirements of a set of scenarios whose aggregate probability is at least equal to the enforced probability level p.

As in Kress et al. (2007) we refer to the above problem as MKP (*minmax multidimensional knapsack problem*). Let us denote by $(\mathbf{v}^*, \delta^*)$ an optimal solution of MKP and by $S^* = \{s \in S : \delta_s^* = 1\} \subseteq S$ the corresponding optimal set of "selected" scenarios. It is easy to see that $\mathbf{v}_i^* = \max_{s \in S^*} d_i^s,\ i = 1, \ldots, n$, hold true. Thus, the objective function (8) of MKP minimizes the sum of the componentwise *maximum* of $\mathbf{d}^s$ vectors for which the corresponding scenarios are included in S^*.

PROPOSITION 2.1 *The optimal solution* $(\mathbf{v}^*, \delta^*)$ *of* MKP *defines a p-efficient point* $\mathbf{v}^*$.

PROOF. First, constraints (9) and (10) ensure that $\mathbb{P}(\mathbf{v}^* \geq \xi) \geq p$. Second, the optimality of $\mathbf{v}^* = (v_1^*, \ldots, v_n^*)^T$ implies that for any point $\mathbf{v}' = (v_1^* - \alpha_1, \ldots, v_n^* - \alpha_n)^T$ with $\alpha_i \geq 0,\ i = \ldots, n$, and $\sum_{i=1}^{n} \alpha_i > 0$, $\mathbb{P}(\mathbf{v}' \geq \xi) < p$ holds true. Thus, there does not exist any point $\mathbf{v}' \leq \mathbf{v}^*$ and $\mathbf{v}' \neq \mathbf{v}^*$ such that $\mathbb{P}(\mathbf{v}' \geq \xi) \geq p$, and by Definition 1.1 $\mathbf{v}^*$ is p-efficient. □

MKP is an NP-hard, mixed-integer programming problem. Kress et al. (2007) solve this problem using an enumerative algorithm (EA) for problem instances where $40, 60$ or 100 scenarios are used to describe the joint realizations of the random variables. Kress et al. report that their EA algorithm outperforms the branch-and bound (B&B) algorithm of CPLEX when $n \leq 15$ and $|S| \leq 100$, but that the B&B approach is up to 10 times faster than the EA algorithm for $|S| = n = 50$. The motivation for our study comes from the observations that the CPLEX MIP solver outperforms the EA proposed in Kress et al. (2007), and that it is hard to solve MKP for large problem instances using a standard MIP solver such as CPLEX.

3. Preprocessing The difficulty of solving MKP increases with the dimension of the random vector and, in particular, with the number of scenarios used to represent the random vector. In this section, we present a new preprocessing method that can be used to reduce the number of scenarios to which attention must be paid. The idea is to construct *bundles* of scenarios, such that all the scenarios included in a bundle define similar requirements. A preprocessing technique based on the quantile of the marginal probability distribution associated with each component of the random vector ξ was previously used in Lejeune (2008).

DEFINITION 3.1 *The first quantile function* $F_X^{(-1)} : (0, 1] \to \mathbb{R}$ *corresponding to a random variable* X *is the left-continuous inverse of the cumulative distribution function* F_X*:*

$$F_X^{(-1)}(p) = \inf\{\eta \in \mathbb{R} \ : \ F_X(\eta) \geq p\}.$$

PROPOSITION 3.1 *A necessary, although not sufficient, condition for* $\mathbf{v}$ *to be p-efficient is that the set of* n *inequalities*

$$v_i \geq F_{\xi_i}^{(-1)}(p),\ i = 1, \ldots, n, \tag{13}$$

hold, with $F_{\xi_i}^{(-1)}(p)$ *denoting the* $p-$*quantile of the marginal probability distribution function of* ξ_i.

The proof is straightforward and is given in Dentcheva (2001) and Lejeune (2008). While the quantile-based cuts (13) are implicitly taken into account in Luedtke et al. (2010) to obtain a strengthened formulation to solve problem (1)-(4), these cuts are used in Lejeune (2008) for developing a preprocessing method, which discards the scenarios that are not relevant to the attainment of the prescribed probability level p. The quantile-based preprocessing of scenarios is based on Corollary 3.1.

COROLLARY 3.1 *Any scenario s with $d_i^s \leq F_{\xi_i}^{(-1)}(p),\ i = 1, \ldots, n,$ is such that all its requirements are always satisfied by any point* $\mathbf{v}$ *for which* $\mathbb{P}(v_i \geq \xi_i, i = 1, \ldots, n) \geq p$ *holds true.*

The proposed preprocessing method, called *bundle preprocessing* is an extension of the quantile-based preprocessing approach. The reader interested in the aggregation of scenarios into bundles in multi-stage stochastic programming is referred to Rockafellar and Wets (1991).

3.1 Bundle Preprocessing The satisfaction of the requirements imposed by scenario s requires that all inequalities $v_i \geq d_i^s, i = 1, \ldots, n$, hold. Also, it follows from Proposition 3.1 that any inequality $v_i \geq d_i^s$ such that $d_i^s \leq F_{\xi_i}^{(-1)}(p)$ is redundant for the derivation of a pLEP. This observation inspired us to represent a scenario s by its actual *requirement path* $\left[\bar{d}_1^s, \ldots, \bar{d}_i^s, \ldots, \bar{d}_n^s\right]$, which is defined as follows:

$$\bar{d}_i^s = \begin{cases} F_{\xi_i}^{(-1)}(p) & \text{if} \quad d_i^s \leq F_{\xi_i}^{(-1)}(p) \\ d_i^s & \text{otherwise} \end{cases}, \ i = 1, \ldots, n, s \in S.$$

Using the requirement path, we create bundles which are clusters of scenarios that have the same requirement path. We consider one representative scenario s for each bundle and compute the aggregate bundle probability π'_s as the sum of the probabilities of all scenarios included in bundle s. Note that all the scenarios, which would be excluded using Corollary 3.1, are grouped in one bundle whose requirement path is $\left[F_{\xi_i}^{-1}(p), \ldots, F_{\xi_n}^{-1}(p)\right]$ and that, by construction, no bundle has less demanding requirements than $\left[F_{\xi_i}^{-1}(p), \ldots, F_{\xi_n}^{-1}(p)\right]$. Thus, the proposed bundle approach ensures the satisfaction of the conditions defined by the marginal distribution of each component ξ_i $(i = 1, \ldots, n)$ and makes the quantile-based cuts redundant.

Denoting the set of bundles by S_1 and introducing the binary variables $\gamma_s, s \in S_1$, which are the complements of δ_s $(\gamma_s = 1 - \delta_s)$, we obtain the strengthened formulation of MKP:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} v_i \\ \text{subject to} \quad & v_i \geq \bar{d}_i^s(1 - \gamma_s), \quad i = 1, \ldots, n, \ s \in S_1 && (14) \\ & \sum_{s \in S_1} \pi'_s \gamma_s \leq 1 - p && (15) \\ & \gamma \in \{0,1\}^{|S_1|} && (16) \\ & \mathbf{v} \in \mathbb{R}_+^n. && (17) \end{aligned}$$

The above problem is thereafter referred to as SMKP. The formulation of SMKP in terms of the γ_s variables (instead of their complements δ_s as in Kress et al. (2007)) makes the presentation of the proposed outer approximation algorithm easier, as it will be explained in Section 4. Since $S_1 \subseteq S$ and, in some

cases, the cardinality of S_1 is significantly smaller than that of S, the computational benefits of the bundle preprocessing method can be very significant. Numerical results supporting this claim are presented in Section 6.2.1.

4. An Outer Approximation Solution Framework In this section, we develop (exact and heuristic) mathematical programming methods to solve SMKP. These methods are based on the *iterative* generation of outer approximation problems obtained by relaxing the integrality restrictions on a subset of the binary variables $\gamma_s, s \in S_1$. The notation OA^t denotes the outer approximation problem solved at iteration t.

We shall first describe the initialization phase which involves the solution of the continuous relaxation of SMKP, the generation of an optimality cut, and its up-front introduction to the formulation of SMKP. Next, we detail the sequential generation of increasingly tighter outer approximations. It rests on the following steps: *(i)* definition of the subset of binary variables for which the integrality restrictions are relaxed; *(ii)* generation of *valid inequalities* that strengthen the incumbent outer approximation; *(iii)* verification of the *stopping criterion*. In addition to the above steps, we can also employ a fixing strategy. Finally, we provide the pseudo-code of the outer approximation approach, which converges in a finite number of iterations. We show that the proposed solution framework is exact if the fixing strategy is not used.

4.1 Initialization The initial outer approximation problem OA^0 is the continuous relaxation of SMKP. We use its optimal solution $(\mathbf{v}^{*(0)}, \gamma^{*(0)})$ to initiate the sequence of outer approximations. If all the variables $\gamma_s, s \in S_1$, have an integer value in the optimal solution of OA^0, then $(\mathbf{v}^{*(0)}, \gamma^{*(0)})$ is also optimal for SMKP and we stop. Otherwise, we use the optimal solution of OA^0 to derive an optimality cut, i.e., an upper bound on the optimal objective function value of SMKP.

The derivation of the optimality cut is based on the following steps. We sort the scenarios in S_1 and the associated scenario probabilities in decreasing order of the optimal values $\gamma^{*(0)}$ of the γ variables. The ordered vector of probabilities is denoted by $\tilde{\pi}'$. We construct a set V comprising the l scenarios with the largest $\gamma^{*(0)}$ values such that $\sum_{s=1}^{l} \tilde{\pi}'_s \leq 1-p$ and $\sum_{s=1}^{l+1} \tilde{\pi}'_s > 1-p$ and set (temporarily) the γ variables associated with the scenarios in V to 1, whereas we set the rest of γ variables to 0. It is easy to see that such a γ vector is feasible for SMKP and that the solution of SMKP under this strategy provides us with an upper bound $\bar{\theta}$ on the optimal objective function value. Thus, the following optimality cut

$$\sum_{i=1}^{n} v_i \leq \bar{\theta}$$

is valid. It is easy to see that $\bar{\theta}$ simply equals to $\sum_{i=1}^{n} \max_{s \in S_1 \setminus V} \bar{d}_i^s$.

The successive outer approximation problems are constructed by partitioning, at iteration t, the set S_1 into two subsets $T_0^{(t)}$ and $T_1^{(t)}$: $T_0^{(t)}$ includes the scenarios for which the integrality restrictions on the binary variables are relaxed, while $T_1^{(t)}$ includes the remaining scenarios. The partitioning is carried out by assigning scenario s to subset $T_0^{(t)}$ (resp., $T_1^{(t)}$) if the corresponding variable γ_s is expected to be 0 (resp., 1) in the optimal solution and the cardinality of these subsets are based on a parameter, denoted by Q. In the initialization phase ($t = 0$), the value of Q is computed as follows. We determine the set of smallest cardinality such that the sum of the probabilities of the scenarios included in S_1 is strictly larger than $1-p$. This comes up to finding the minimum possible cardinality of a minimal cover set (Hammer et al., 1975) for

the knapsack constraint (15). Suppose that we sort the probabilities of all the scenarios in S_1 in descending order and this ordered vector of probabilities are denoted by $\tilde{\pi}''$. Let

$$\hat{Q} = \min\left\{k : \sum_{i=1}^{k} \tilde{\pi}''_k > 1 - p, k \in \{1 \dots |S_1|\}\right\} - 1, \tag{18}$$

where $(\hat{Q}+1)$ is the smallest possible cardinality of a minimal cover, and set $Q = \hat{Q}$. If all the scenarios in S are equally likely, we have $\hat{Q} = \lfloor (1-p)|S| \rfloor$. Note that when there are large probabilities $\tilde{\pi}''_k$, $\hat{Q}$ takes a very small value and the iterative process would require many iterations to find the optimal solution (as explained in the following section). In such situations, one shall set the value of Q based on the number of scenarios. For example, in our computational study we define the parameter Q as follows:

$$Q = \max\left(\min\left(\hat{Q}, \lfloor 0.15|S| \rfloor\right), \lfloor 0.05|S| \rfloor\right). \tag{19}$$

As we demonstrate in Section 6.2.3, the way we set the values of Q permits our iterative process to terminate after a very limited number of iterations.

4.2 Iterative Process In this section, we describe the main steps that are executed at each iteration of the proposed algorithm. We explain how the outer approximations are constructed. We derive a family of valid inequalities to strengthen the outer approximation formulations and present the stopping criterion. We also propose a fixing strategy that is used within a heuristic solution approach.

4.2.1 Outer Approximation In order to solve SMKP, we derive a series of outer approximation problems in which the integrality restrictions are relaxed on a subset of binary variables. The selection of binary variables on which integrality restrictions are initially relaxed and then progressively restored is executed in a way that leads to a fast convergence of the procedure. The underlying idea is that solving a limited number of simpler outer approximation problems would be faster than solving the SMKP formulation.

Let $T_0^{(t)}$ (resp., $T_1^{(t)}$) denote the set of scenarios for which the integrality restrictions on the associated binary variables are relaxed (resp., imposed) at iteration t. The parameter $\bar{Q} = \min(|T_0^{(t-1)}|, Q)$ is used to construct the new outer approximation $\text{OA}^{(t)}$ of SMKP. More precisely, $\bar{Q}$ scenarios are selected according to one of the two criteria presented in the next paragraph to build the sets $T_0^{(t)}$ and $T_1^{(t)}$ ($T_0^{(t)} \subseteq T_0^{(t-1)} \subseteq S_1$, $T_1^{(t)} \supseteq T_1^{(t-1)}$ and $T_1^{(t)} = S_1 \setminus T_0^{(t)}$). At each iteration t, $\bar{Q}$ elements of $T_0^{(t-1)}$ are transferred to $T_1^{(t)}$, implying that the cardinality of $T_0^{(t)}$ (resp., $T_1^{(t)}$) decreases (resp., increases) by $\bar{Q}$ as t increases. One may modify the proposed algorithm so that the value of $\bar{Q}$ changes at each iteration.

The first selection criterion, referred to as "Criterion 1", is based on the values of the γ variables in the optimal solution of $\text{OA}^{(t-1)}$. The $\bar{Q}$ scenarios with the largest γ_s, $s \in T_0^{(t-1)}$, values are removed from $T_0^{(t-1)}$ and added to $T_1^{(t-1)}$ to obtain $T_1^{(t)}$. The second criterion, referred to as "Criterion 2", is based on a random selection; $\bar{Q}$ scenarios with the smallest indices are included in $T_1^{(t)}$.

Let $A^{(t)}$ denote the set of scenarios selected to be included in set $T_1^{(t)}$ ($t \geq 1$). The sets $T_0^{(t)}$ and $T_1^{(t)}$ are constructed as follows:

$$\begin{aligned} T_1^{(t)} &= T_1^{(t-1)} \cup A^{(t)} \\ T_0^{(t)} &= T_0^{(t-1)} \setminus A^{(t)} \end{aligned}, \tag{20}$$

with $T_0^{(0)} = S_1$ and $T_1^{(0)} = \emptyset$. The initial sets $T_0^{(1)}$ and $T_1^{(1)}$ are derived from the optimal solution $(\mathbf{v}^{*(0)}, \gamma^{*(0)})$ of the continuous relaxation of SMKP. For example, using Criterion 1, we sort $\gamma_s^{*(0)}, s \in S_1$, and include in

$A^{(1)}$ the scenarios whose corresponding variables $\gamma_s^{*(0)}$ have the $\bar{Q}$ largest values in the optimal solution of the continuous relaxation. Then, $T_1^{(1)} = A^{(1)}$ and $T_0^{(1)} = S_1 \setminus T_1^{(1)}$.

The outer approximation problem $\text{OA}^{(t)}$ is obtained by removing the integrality restrictions on γ variables corresponding to the scenarios in $T_0^{(t)}$, and by replacing constraints (15) and (16) in SMKP by constraints (23), (24), (26)-(28). Setting $u_0 = \min_{s \in T_0^{(t)}} \pi'_s$, the formulation of $\text{OA}^{(t)}$ reads:

$$\min \quad \sum_{i=1}^{n} v_i \tag{21}$$

$$\text{subject to} \quad v_i \geq \bar{d}_i^s (1 - \gamma_s), \quad i = 1, \ldots, n, \ s \in S_1 \tag{22}$$

$$\sum_{s \in T_0^{(t)}} \pi'_s \gamma_s \leq w(1-p) \tag{23}$$

$$u_0 w + \sum_{s \in T_1^{(t)}} \pi'_s \gamma_s \leq 1 - p \tag{24}$$

$$\sum_{i=1}^{n} v_i \leq \bar{\theta} \tag{25}$$

$$0 \leq \gamma_s \leq 1, \quad s \in T_0^{(t)} \tag{26}$$

$$\gamma_s \in \{0,1\}, \quad s \in T_1^{(t)} \tag{27}$$

$$w \in \{0,1\} \tag{28}$$

$$\mathbf{v} \in \mathbb{R}_+^n \ . \tag{29}$$

Notice that, since the enforced probability level p is at least equal to 0.5, and generally is close to 1, the cardinality of $|T_0^{(t)}|$ is in general strictly larger than that of $|T_1^{(t)}|$. Thus, Problem (21)-(29) contains a significantly lower number ($|T_1^{(t)}|$) of binary decision variables than does SMKP and so is easier to solve. The difference of cardinality between $|T_0^{(t)}|$ and $|T_1^{(t)}|$ is a monotone increasing function of the prescribed p value and of the original number of scenarios.

PROPOSITION 4.1 *Problem (21)-(29) is an outer approximation of* SMKP.

PROOF. We have to show that any feasible solution of SMKP is feasible for problem (21)-(29). This implies the following relation between the feasible sets

$$Z = \{\gamma_s : (15), (16)\} \ \subseteq \ Z' = \{\gamma_s : (23), (24), (26), (27), (28)\}\,.$$

Consider a subset of scenarios $V \subseteq S_1$ such that $\sum_{s \in V} \pi'_s \leq 1 - p$ and obtain a feasible solution of SMKP by setting $\gamma_s = 1, s \in V$, and $\gamma_s = 0, s \in S_1 \setminus V$. We refer to this solution as γ^V and it is easy to see that $\gamma^V \in Z$. To prove the claim we show that γ^V also belongs to Z'.

Indeed, if $V \bigcap T_0^{(t)} = \emptyset$, then w can take value 0 in (23), constraint (24) reads $\sum_{s \in T_1^{(t)}} \pi'_s \gamma_s^V \leq 1 - p$ and by (15) we have $\gamma^V \in Z'$. If $V \bigcap T_0^{(t)} \neq \emptyset$, then $w = 1$ by (23). Since $S_1 = T_0^{(t)} \bigcup T_1^{(t)}$ and γ^V satisfies (15), we have

$$\sum_{s \in S_1} \pi'_s \gamma_s^V = \sum_{s \in (T_0^{(t)} \bigcup T_1^{(t)}) \bigcap V} \pi'_s = \sum_{s \in T_0^{(t)} \bigcap V} \pi'_s + \sum_{s \in T_1^{(t)} \bigcap V} \pi'_s \leq 1 - p. \tag{30}$$

Since $\min_{s\in T_0^{(t)}} \pi'_s \leq \sum_{s\in T_0^{(t)}\bigcap V} \pi'_s$, the following inequality is evidently always true

$$\min_{s\in T_0^{(t)}} \pi'_s + \sum_{s\in T_1^{(t)}\bigcap V} \pi'_s \leq \sum_{s\in T_0^{(t)}\bigcap V} \pi'_s + \sum_{s\in T_1^{(t)}\bigcap V} \pi'_s. \tag{31}$$

Then, by (30) and (31) the solution γ^V satisfies (24) and the assertion trivially follows. □

4.2.2 Strengthening Valid Inequalities In this section, we derive a family of valid inequalities that strengthen the successive outer approximation formulations. This contributes to reducing the number of iterations and to improving the computational efficiency of the proposed algorithms. The motivation for the proposed family of valid inequalities rests on the following observation. If one of the $\gamma_s, s \in T_0^{(t)}$, variables does not take value 0 in the optimal solution of $\text{OA}^{(t)}$, then (23) forces w to be equal to 1. This allows any number of the other $\gamma_s, s \in T_0^{(t)}$, variables to take a non-zero value without impacting the optimal value of $\text{OA}^{(t)}$. Preliminary computational results show that many of the variables $\gamma_s, s \in T_0^{(t)}$, take a non-zero value in the optimal solution of $\text{OA}^{(t)}$, whereas they could be equal to 0. In order to circumvent the above issue, we strengthen the formulation of $\text{OA}^{(t)}$ by introducing the set of valid inequalities defined in the next proposition. These valid inequalities are also useful in improving the computational performance of the fixing strategy, which will be discussed in the next section.

PROPOSITION 4.2 *The following set of inequalities*

$$v_i + \eta_s \geq \bar{d}_i^s, \quad i = 1,\dots,n,\ s \in T_0^{(t)} \tag{32}$$

$$\eta_s \leq \gamma_s\big(\max_{i=1,\dots,n}\{\bar{d}_i^s - F_{\xi_i}^{(-1)}(p)\}\big), \quad s \in T_0^{(t)} \tag{33}$$

$$\gamma_s \leq \eta_s, \quad s \in T_0^{(t)} \tag{34}$$

$$\eta_s \geq 0, \quad s \in T_0^{(t)} \tag{35}$$

is valid for SMKP *(and for* $\text{OA}^{(t)}$*).*

PROOF. First, observe that the bundle preprocessing guarantees that $\bar{d}_i^s \geq F_{\xi_i}^{(-1)}(p)$ for all $i = 1,\dots,n,\ s \in S$, and there always exists a vector η satisfying the above inequalities. If there is no shortfall (i.e., $v_i \geq \bar{d}_i^s, i = 1,\dots,n$) for scenario s, it is easy to see that $\gamma_s = 0$ is a feasible solution. If there is a shortfall (i.e., $v_i < \bar{d}_i^s$ for at least one $i \in 1,\dots,n$) for scenario s, then (32) forces η_s to be at least equal to the maximum ($i = 1,\dots,n$) shortfall amount ($\bar{d}_i^s - v_i$), while (33) ensures that γ_s takes a strictly positive value, at most equal to 1. It follows that the set of inequalities (32)-(35) do not cut any integer solution feasible for SMKP (and for $\text{OA}^{(t)}$).

□

In the computational study described in Section 6, the valid inequalities (32)-(35) are introduced in the formulation of $\text{OA}^{(t)}$. The computational results show that these valid inequalities contribute to forcing the variables $\gamma_s, s \in T_0^{(t)}$, to take value 0 whenever it is possible.

Observation 1 *After adding the valid inequalities (32)-(35), it is easy to see that the constraints* $v_i \geq \bar{d}_i^s(1-\gamma_s),\ i = 1,\dots,n,\ s \in T_0^{(t)}$*, become redundant. Therefore, constraint (22) can be replaced by*

$$v_i \geq \bar{d}_i^s(1-\gamma_s), \quad i = 1,\dots,n,\ s \in T_1^{(t)}, \tag{36}$$

which leads to a reduction in the number of constraints by $n \cdot |T_0^{(t)}|$.

4.2.3 Stopping Criterion The stopping criterion of the algorithm is defined with respect to the value taken by w in the optimal solution of $\text{OA}^{(t)}$. Let $(\mathbf{v}^*, \gamma^*, \mathbf{w}^*)$ be the optimal solution of $\text{OA}^{(t)}$.

PROPOSITION 4.3 *If* $w^* = 0$*, then the optimal solution of* $\text{OA}^{(t)}$ *is also optimal for* SMKP.

PROOF. Consider that the optimal solution $(\mathbf{v}^*, \gamma^*, \mathbf{w}^*)$ of $\text{OA}^{(t)}$ is such that $\mathbf{w}^* = 0$. This implies that $\gamma_s^* = 0$ for all $s \in T_0^{(t)}$ and $\sum_{s \in T_1^{(t)}} \pi'_s \gamma_s \leq 1 - p$. Since $\gamma_s^* = 0$ for all $s \in T_0^{(t)}$ and $T_0^{(t)} \cup T_1^{(t)} = S_1$, the solution $(\mathbf{v}^*, \gamma^*, 0)$ satisfies (15) and it also satisfies (16) by (27). Then, it is easy to see that $(\mathbf{v}^*, \gamma^*)$ is feasible for SMKP. This argument, combined with Proposition 4.1 establishing that problem (21)-(29) is an outer approximation of SMKP, completes the proof. □

We note that Proposition 4.3 remains true when we add the valid inequalities (32)-(35) to the formulation of $\text{OA}^{(t)}$ and replace (22) in $\text{OA}^{(t)}$ by (36). As discussed above, when $\mathbf{w}^* = 0$, we have $\gamma_s^* = 0$ for all $s \in T_0^{(t)}$. Then, (33) implies that $\eta_s = 0$ for all $s \in T_0^{(t)}$ and $v_i \geq \bar{d}_i^s,\ i = 1, \ldots, n,\ s \in T_0^{(t)}$. This observation combined with the satisfaction of (36) implies that $(\mathbf{v}^*, \gamma^*)$ satisfies (14).

In summary, we continue updating the compositions of the sets $T_0^{(t+1)}$ and $T_1^{(t+1)}$ for the $(t+1)$th outer approximation and solving the outer approximation problems until $\mathbf{w}^* = 0$.

4.2.4 Fixing Strategy The fixing strategy relies on the postulate that the values of the γ variables in the optimal solution of the outer approximation problem are very informative to find the optimal set of scenarios $S^* = \{s \in S_1 : \gamma_s^* = 0\}$. Preliminary tests support this idea. Recall that scenario s belongs to the set S^* if its requirements $\mathbf{v}_i \geq \bar{d}_i^s, s \in S_1$, are satisfied in the optimal solution $(\mathbf{v}^*, \gamma^*)$ of SMKP. We identify the scenarios in set $T_0^{(t-1)}$ whose associated γ variables are equal to 0 in the optimal solution of the outer approximation and fix those γ variables to 0. The main idea here if $\gamma_s = 0$ in the optimal solution of $\text{OA}^{(t)}$, it is expected that $\gamma_s = 0$ at some solution that is close to the optimal solution of SMKP. However, for the described greedy approach we cannot guarantee to find an optimal solution.

The fixing strategy reduces very significantly the number of decision variables included in the successive approximation problems and makes the solution method heuristic. There is no guarantee that binary variables with value 0 in the optimal solution of an outer approximation problem will necessarily be equal to 0 in the optimal solution of SMKP. The best solution found by using the fixing strategy is thus sub-optimal for SMKP. This explains why the solution obtained by the fixing heuristic method defines a quasi pLEP. The proximity ("closeness") of a quasi pLEP to a pLEP is measured by the optimality gap of the quasi pLEP with respect to the pLEP (optimal solution). Since the objective function is the sum of the components, the optimality gap is calculated by the sum of the components of the quasi pLEP and the pLEP. When the heuristic method finds a quasi pLEP with a small optimality gap value, this solution can be viewed as a point that is very close to qualifying as a pLEP. Such quasi pLEPs impose marginally more demanding requirements than those defined by pLEPs and can be used to derive a tight inner approximation of problem (1)-(4).

We introduce the following notations. Let us denote by $T^{(t)}$ (resp., $T_F^{(t)}$) the set of scenarios whose binary

variables are not fixed (resp., are fixed to 0) by the end of iteration t. Clearly, $T_F^{(t)}$ is the complement of $T^{(t)}$ with respect to the set S_1. Let $D^{(t)}$ denote the set of scenarios for which the corresponding γ variables are fixed to 0 at iteration t. We set $D^{(t)} = \left\{ s : \gamma_s^* = 0, s \in T_0^{(t-1)} \right\}$ and then for $t \geq 2$ we have

$$\begin{aligned} T_F^{(t)} &= T_F^{(t-1)} \bigcup D^t, \\ T^{(t)} &= S_1 \setminus T_F^{(t)} \end{aligned} \tag{37}$$

with $T_F^{(1)} = \emptyset$. The following set of constraints can be incorporated into to the final proposed formulation at iteration t in order to represent the fixations described above:

$$v_i \geq \max_{s \in T_F^{(t)}} \bar{d}_i^s, \quad i = 1, \ldots, n. \tag{38}$$

The proposed fixing strategy fixes the variables $\gamma_s, s \in T_0^{(t)}$, to 0 when they take value 0 in the optimal solution of $\text{OA}^{(t)}$. Hence, it is critical to set the variables $\gamma_s, s \in T_0^{(t)}$, equal to 0 whenever possible in order to benefit from the fixing strategy in improving the computational performance of the heuristic. The strengthening valid inequalities help us to deal with the above issue and further reduce the number of decision variables included in the outer approximation problems. The numerical results presented in Section 6.2.3 show the efficiency and effectiveness of the heuristic algorithm.

4.3 Pseudo-Code This section presents the pseudo-code of the algorithmic framework proposed to solve SMKP. When the fixing strategy is not employed, the method is exact and the solution it provides, if proven optimal, defines a pLEP. If we implement the fixing strategy, the method is a heuristic. The heuristic algorithm is designed to find a feasible solution which has a small optimality gap, and defines a quasi pLEP (which may be a pLEP).

Since the number of γ variables is finite, it is easy to see that the proposed algorithms stop within finitely many iterations. When the fixing strategy is not employed (Step 12 is skipped), the algorithm finds an optimal solution for SMKP. The exactness of the algorithm easily follows from Propositions 4.1, 4.2 and 4.3.

5. Iterative Generation of a Set of *p*-Efficient Points The satisfaction of all (n) requirements imposed by any pLEP ϑ guarantees to attain the prescribed probability level p: $(\mathbf{v} \geq \vartheta)$ implies that $P(\xi \leq \mathbf{v}) \geq p$. However, the cost triggered by satisfying the requirements imposed by different pLEPs can fluctuate very much. An industrial supply chain management problem described in Lejeune and Ruszczyński (2007) illustrates the differences in the costs associated with the multiple p-efficient points. This observation motivates our iterative solution approach that allows for the elicitation of all or a subset of p-efficient points.

In the proposed iterative approach, the first pLEP is generated by solving SMKP. The other pLEPs are obtained using a slightly "modified" version of SMKP. More precisely, we introduce in SMKP the constraints (42)-(45) that ensure that its optimal solution defines a pLEP that has not already been generated.

We denote by M a parameter taking a sufficiently large positive value, e.g., $M_i = \max_{s \in S} \bar{d}_i^s \; i = 1 \ldots, n$, or simply, $M = \max_{i \in \{1,\ldots,n\}, s \in S} \bar{d}_i^s$ and $M_i = M, \; i = 1 \ldots, n$. Let us denote the pLEP generated at iteration k by $\vartheta^{(k)}$, then $\vartheta^{(1)} = \mathbf{v}^*$ with $\mathbf{v}^*$ being the optimal solution of SMKP. The $(k+1)$th pLEP is obtained through the solution of the MIP problem (39)-(47), which can be solved using the exact solution approach described in Section 4.

Algorithm 1 Algorithms for SMKP (to generate a single pLEP).

(Bundle Preprocessing) Employ the bundle preprocessing method; find set S_1 and the probability vector π'.

Solve the LP relaxation of SMKP to obtain the optimal solution $(\gamma^{*(0)}, \mathbf{v}^{*(0)})$.

if all γ variables are integral **then**

  Stop

else

  Initialize the iterative process and set $t = 1$. Let $T_F^{(1)} = \emptyset$.

  Obtain initial sets $T_0^{(1)}$ and $T_1^{(1)}$, which form a partition of S_1:

  Calculate the parameter Q using (18) and (19) (or using a similar approach) and then set $\bar{Q} = Q$. Using a selection criterion (Criterion 1 or Criterion 2) define $T_1^{(1)}$ as the set of $\bar{Q}$ scenarios from S_1 and let $T_0^{(1)} = S_1 \setminus T_1^{(1)}$.

  Find the upper bound on the objective function value: Construct set V comprising of l scenarios with the largest $\gamma^{*(0)}$ values such that $\sum_{s=1}^{l} \tilde{\pi}''_s \leq 1 - p$ and $\sum_{s=1}^{l+1} \tilde{\pi}''_s > 1 - p$ and set $\bar{\theta} = \sum_{i=1}^{n} \max_{s \in S_1 \setminus V} \bar{d}_i^s$.

  Solve $\text{OA}^{(1)}$ to obtain the optimal solution denoted by $(\mathbf{v}^*, \gamma^*, w^*)$.

  while $w^* \neq 0$ **do** {Iteratively increase the number of binary variables in set $T_1^{(t)}$}

    Let $t := t + 1$

    Execute this step if "fixing strategy" is used, otherwise skip.

    Update the set $T_F^{(t)}$ (fixing some of the $\gamma_s,\ s \in T_0^{(t-1)}$, variables): Let

$$T_F^{(t)} = T_F^{(t-1)} \cup \{s \in T_0^{(t-1)} \mid \gamma_s^* = 0\} \text{ and } T_0^{(t-1)} := T_0^{(t-1)} \setminus \{s \in T_0^{(t-1)} \mid \gamma_s^* = 0\}.$$

    Update sets $T_0^{(t)}$ and $T_1^{(t)}$: Calculate the parameter Q using (18) and (19) (or using a similar approach) and then set $\bar{Q} = \min(|T_0^{(t-1)}|, Q)$.

    Use a selection criterion to pick $\bar{Q}$ scenarios from $T_0^{(t-1)}$ to define $A^{(t)}$.

    Let $T_1^{(t)} = T_1^{(t-1)} \cup A^{(t)}$ and $T_0^{(t)} = T_0^{(t-1)} \setminus A^{(t)}$.

    Solve $\text{OA}^{(t)}$ to obtain the optimal solution denoted by $(\mathbf{v}^*, \gamma^*, w^*)$.

  end while

end if

PROPOSITION 5.1 *If feasible, the optimal solution* $(\mathbf{v}^*, \gamma^*, \mathbf{y}^*)$ *of*

$$\min \quad \sum_{i=1}^{n} v_i \tag{39}$$

$$\textit{subject to} \quad v_i \geq \bar{d}_i^s(1-\gamma_s), \quad i=1,\ldots,n,\ s \in S_1 \tag{40}$$

$$\sum_{s\in S_1} \pi'_s \gamma_s \leq 1-p \tag{41}$$

$$v_i - \vartheta_i^r + 1 \leq M_i y_1(r,i), \quad r=1,\ldots,k,\ i=1,\ldots,n \tag{42}$$

$$y_2(r,i) \leq M_i(1-y_1(r,i)), \quad r=1,\ldots,k,\ i=1,\ldots,n \tag{43}$$

$$\sum_{i=1}^{n} y_2(r,i) \geq 1, \quad r=1,\ldots,k \tag{44}$$

$$y_1(r,i), y_2(r,i) \in \{0,1\}, \quad r=1,\ldots,k,\ i=1,\ldots,n \tag{45}$$

$$\gamma \in \{0,1\}^{|S_1|} \tag{46}$$

$$\mathbf{v} \in \mathbb{R}^n_+ \tag{47}$$

defines the $(k+1)$*th p-efficient point* $\vartheta^{(k+1)} = \mathbf{v}^*$.

PROOF. Constraints (40)-(41) guarantee that any feasible solution $\mathbf{v}$ is such that $\mathbb{P}(\xi \leq \mathbf{v}) \geq p$. By Proposition 2.1 and Definition 1.1, $\mathbf{v}^*$ is p-efficient if it is different from and *not dominated* by any of the pLEPs generated up to iteration k. In other words, we generate *a new pLEP* at iteration $(k+1)$ if there is no $\vartheta^{(r)}, r=1,\ldots,k$, such that

$$\vartheta^{(r)} \leq \mathbf{v}^* \text{ and } \vartheta^{(r)} \neq \mathbf{v}^*. \tag{48}$$

Equivalently, if for all $r=1,\ldots,k$, there exists an index $i_r,\ i_r \in \{1,\ldots,n\}$, such that $v^*_{i_r} < \vartheta^{(r)}_{i_r}$ (at least one component of $\mathbf{v}^*$ is strictly smaller than that of the pLEPs generated at previous iterations), then $\mathbf{v}^*$ is *not dominated* by the pLEPs generated up to iteration k. It can be seen that constraints (42)-(45) enforce (48). Indeed, (44) requires that for at least one index $i,\ i \in \{1,\ldots,n\}$, $y_2(r,i)$ is equal to 1 for all $r=1,\ldots,k$. Let us denote one of those indices associated with $r,\ r=1,\ldots,k$, by i_r, i.e., $y_2(r,i_r)=1$. Then it follows from (43) that $y_1(r,i_r)=0$, which, combined with (42), ensures that $v^*_{i_r} \leq \vartheta^r_{i_r} - 1$ for all $r=1,\ldots,k$. □

Proposition 5.1 indicates that, each time we solve problem (39)-(47) to optimality, we obtain a new pLEP ($\vartheta^{(k+1)} = \mathbf{v}^*$). The mathematical programming based generation of a set of pLEPs is one of the significant contributions of this paper.

The algorithms proposed to solve SMKP are here applied to a "modified" version of SMKP, which is obtained by adding constraints (42)-(45) to the SMKP formulation. The algorithm discovers an empty feasible set when there is no more pLEP left to be identified. Note that solving problem (39)-(47) using the heuristic approach, described in Section 4.2.4, provides a quasi pLEP at each iteration. The computational efficiency of the heuristic algorithm allows its recursive application to generate a set of quasi pLEPs.

6. Computational Results In Section 6.1, we present the problem instances used in our computational study. In Section 6.2, we first assess the individual contribution of the three specific algorithmic techniques (preprocessing, valid inequalities, fixing strategy) integrated within the outer approximation method. We

then analyze the efficiency and effectiveness of the outer approximation method proposed for generating quasi pLEPs.

The optimization problems are modeled with the AMPL mathematical programming language (Fourer et al., 2003) and solved with the 11.2 CPLEX solver (ILOG, 2008). Each problem instance is solved on a `64-bit HP` workstation running on `Linux` with `2 quad- core 1.6GHz CPU`, and `16GB` of `RAM`. All the reported CPU times are in seconds. In our computational study, we terminate CPLEX when the prescribed CPU time limit of 3600 seconds is reached.

6.1 Testing Set The computational evaluation is carried out with respect to three families of problem instances for a total of 423 instances. For the first family, we use random sampling to generate a set of scenarios representing a Poisson probability distribution. The problem instances of the second family are obtained by using a stratified sampling approach to represent a given distribution. In the third family, we consider all the possible values that a random variable following a discrete distribution with finite support can take. All the data instances are generated with `MATLAB R2006`.

6.1.1 Family 1: Random Sampling The specifics of the first family of problem instances are that they concern random vectors of large dimensionality and that a discretized representation of the random vectors is obtained using random sampling. More precisely, we consider the random vector ξ comprising n components $\xi_i,\ i = 1, \ldots, n$, each of which follows a Poisson distribution with parameter λ_i. The values of the arrival rate parameters $\lambda_i,\ i = 1, \ldots, n$, are sampled from the uniform distribution on one of the following intervals: $[50, 100], [50, 150], [50, 200]$ and $[100, 150]$. We use random sampling to extract $|S|$ scenarios. For each of the problem instances, we consider two different probability settings. The first one assumes that all scenarios are equally likely, while the second one assigns to each scenario a value, which is sampled from the uniform distribution on the interval $[0.2, 0.7]$. Those values are then normalized to obtain the probability π_s associated with each scenario s.

We consider 58 types of instances and generate 6 problem instances per each type to take the randomness in data generation into account. Each instance type is defined with respect to the tuple $(n, |S|, p, D)$, where $n = 10, 20, 50,\ |S| = 500, 1000, 3000, 5000,\ p = 0.8, 0.9, 0.95$, and D indicates whether the scenarios are assumed to be equally likely or not.

6.1.2 Family 2: Stratified Sampling For this family of instances, the scenarios representing the random variables are obtained using a stratified sampling approach. More precisely, the possible values of a multivariate random variable that follows an identified probability distribution are separated into four strata that correspond to the fractiles of the multivariate probability distribution.

To generate the set of $|S|$ scenarios used to discretize the random variables, we define a number $f_i, i = 1, 2, 3, 4$, with $\sum_{i=1}^{4} f_i = |S|$, of scenarios that are extracted from each stratum. The first fractile includes the realizations whose cumulative probability does not exceed 0.25. By changing the values of $f_i,\ i = 1, \ldots, 4$, we generate

- approximately symmetric: $f_i = 0.25 \cdot |S|,\ i = 1, \ldots, 4$,
- left-skewed: $f_1 = 0.1 \cdot |S|, f_2 = 0.2 \cdot |S|, f_3 = 0.3 \cdot |S|, f_4 = 0.4 \cdot |S|$, and

- right-skewed: $f_1 = 0.6 \cdot |S|, f_2 = 0.2 \cdot |S|, f_3 = 0.1 \cdot |S|, f_4 = 0.1 \cdot |S|$

sample probability distributions. Each scenario is assigned the same probability equal to $1/|S|$.

For this family of problem instances, we consider 21 types of instances and generate 5 problem instances per type. The type of an instance is defined with respect to the tuple $(n, |S|, p, D)$, where $n = 5$, 10, $|S| =$ 1000, 2000, and $p = 0.7, 0.8, 0.9, 0.95$.

Note that considering random variables of moderate ($n = 5, 10$) dimensionality is most relevant in many applications (see, e.g., Henrion, 2004; Kress et al., 2007; Lejeune and Ruszczyński, 2007; Prékopa, 1995). Enforcing a large probability level on a multivariate random variable is close to setting many components of a pLEP equal to the largest value that the corresponding component of the random vector can take. Note also that it is not easy to generate a representative set of realizations for a large-dimensional random vector, since most of the sampled realizations tend to have small cumulative probabilities.

6.1.3 Family 3: Discrete Probability Distribution The problem instances are generated by assuming that each random variable follows a discrete probability distribution with finite support. We consider 12 types of instances and generate 5 problem instances per type. The type of an instance is defined with respect to the tuple $(n, |S|, p, D)$, where $n = 6, 8$, $|S|$ can take different very large (up to $65,000$) values and $p = 0.9, 0.95$. The probabilities of the scenarios are defined by the discrete distribution.

6.2 Contributions of Specific Algorithmic Techniques In this section, we evaluate the contribution of each novel algorithmic technique integrated in the outer approximation solution framework.

First, we would like to emphasize that solving the MKP formulation directly using an MIP solver such as CPLEX is hard for large problem instances. Since the optimal solution cannot be reached or proven within the prescribed time limit for many instances, we compute an upper bound on the optimality gap by using a lower bound on the objective value. Let Obf_T denote the best lower bound on the objective function value found by the B&B algorithm of CPLEX and Obf_T^* denote the best objective function value reached within the time limit T ($T = 3600$ seconds). We define the upper bound on the optimality gap (UBOP) as follows:

$$\text{UBOP} = \frac{\text{Obf}_T^* - \text{Obf}_T}{\text{Obf}_T}.$$

Table 1 reports upper bounds on the optimality gaps when the MKP formulation is solved directly by using the B&B algorithm of CPLEX. The results in Table 1 are obtained for a set of Family 1 problem instances (Section 6.1). It can be seen from Table 1 that when applied to the MKP formulation, the B&B algorithm of CPLEX does not find an optimal solution within the time limit of 3600 seconds and the UBOP values are not very small even for problem instances of moderate size. This highlights the difficulty of solving MKP.

6.2.1 Bundle Preprocessing Results pertaining to the reduction in the number of scenarios that remain under consideration after the bundle preprocessing phase are reported for Family 1 (resp., Family 2 and family 3) in Table 2 (resp., Table 3 and Table 4). The notation $|B|$ refers to the average number (i.e., we consider 5 data sets per type of instances) of bundles into which the $|S|$ scenarios have been aggregated, while $R = (|S| - |S_1|)/|S|$ indicates the percentage by which the number of scenarios to be considered has

Problem Instances			Upper Bound on Optimality Gap	
n	$\|S\|$	p	D: Equal Prob.	D: General Prob.
10	500	0.9	4.47%	4.24%
20	500	0.9	4.46%	4.45%
50	500	0.9	3.76%	3.73%
10	1000	0.9	8.81%	8.29%
20	1000	0.9	7.55%	6.94%
50	1000	0.9	6.26%	6.43%

Table 1: Using the direct formulation of MKP

been reduced. Table 3 distinguishes the approximately symmetric, left-skewed, and right-skewed probability distributions.

Tables 2-4 demonstrate that the reduction in the number of scenarios increases as *(i)* the enforced probability level p increases, and as *(ii)* the dimension n of the random vector decreases. Table 3 highlights that the bundle approach performs very well regardless of the skewness of the probability distributions. Tables 3 and 4 show that the average reduction in the number of scenarios varies between 60.46% and 99% of the initial number of scenarios. Clearly, the number of binary variables included in MKP (or SMKP) decreases in the same proportion. Thus, the bundle preprocessing approach has a dramatic impact on reducing the complexity of the MKP formulation and on improving the efficiency of any solution method. Moreover, the bundle algorithm takes few seconds of CPU time for most of the problem instances. For the really large problem instances from Family 1, the bundle algorithm is slightly slower. For example, it takes on average almost 8, 22 and 70 seconds for the problem instances presented in Table 2 with $n = 10$, $|S| = 5000$, $n = 20$, $|S| = 5000$ and $n = 50$, $|S| = 5000$, respectively.

It appears that the contribution of the bundle preprocessing technique is even more conclusive when it is applied to discrete probability distributions (Table 4) than when it is applied to sample probability distributions obtained with random sampling (Table 2) and stratified sampling (Table 3). The average reduction in the number of scenarios is at least equal to 93.78% of the initial number of scenarios. Finally, note that the bundle preprocessing approach is versatile enough to be used within any solution method.

6.2.2 Strengthening Valid Inequalities The evaluation of the strengthening valid inequalities is carried out through the comparison of the solution times obtained by including them or not within the outer approximation framework. The relative CPU time reduction consecutive to the incorporation of the valid inequalities is reported in Table 5 for problem instances belonging to Family 1. The numerical results are obtained for each selection criterion discussed in Section 4.2.1 and indicate that the strengthening valid inequalities significantly contribute to improving the computational performance of the outer approximation algorithm.

6.2.3 Fixing Strategy We solve a set of problem instances from Family 1 with the proposed outer approximation algorithm with and without employing the fixing strategy (the algorithm uses all the other proposed algorithmic techniques). It can be seen from Table 6 that the use of the fixing strategy results in a significant CPU time reduction.

Problem Instances			D: Equal Probabilities		D: General Probabilities	
n	$\|S\|$	p	$\|B\|$	R	$\|B\|$	R
10	500	0.95	116.6	76.68%	115.8	76.84%
20	500	0.95	239.2	52.16%	233.8	53.24%
50	500	0.95	439.8	12.04%	437.6	12.48%
10	1000	0.95	178.4	82.16%	177.4	82.26%
20	1000	0.95	412	58.80%	414.2	58.58%
50	1000	0.95	856.6	14.34%	852.2	14.78%
10	2000	0.95	281.4	85.93%	277.2	86.14%
20	2000	0.95	693.4	65.33%	697	65.15%
50	2000	0.95	1643.8	17.81%	1636.8	18.16%
10	3000	0.95	358.8	88.04%	354.4	88.19%
20	3000	0.95	948	68.40%	951.2	68.29%
50	3000	0.95	2399.2	20.03%	2392.8	20.24%
10	5000	0.95	525.8	89.48%	529.6	89.41%
20	5000	0.95	1433.4	71.33%	1435.4	71.29%
50	5000	0.95	3844.4	23.11%	3820.4	23.59%
10	500	0.9	216.2	56.47%	214.4	56.83%
20	500	0.9	394.8	20.90%	392.8	21.33%
50	500	0.9	496.2	0.73%	496.2	0.73%
10	1000	0.9	355.2	63.98%	352.8	64.13%
20	1000	0.9	727.8	27.13%	723.6	27.68%
50	1000	0.9	989.8	0.97%	989.4	0.98%
10	2000	0.9	605.4	69.50%	603.0	69.61%
20	2000	0.9	1325.4	33.19%	1321.2	33.40%
10	3000	0.9	842.8	71.72%	838.2	71.84%
10	5000	0.9	1268.8	74.43%	1268.8	74.43%
10	500	0.8	375.6	24.43%	375.8	24.47%
20	500	0.8	490.8	1.77%	490.4	1.80%
10	1000	0.8	706.0	29.13%	699.6	29.62%
20	1000	0.8	975.0	2.65%	974.4	2.70%

Table 2: Efficiency of Bundle Preprocessing Approach: Random Sampling

Problem Instances				$\|B\|$	R
Skewness	n	$\|S\|$	p		
`Symmetric`	5	1000	0.95	7	99.30%
	5	1000	0.9	55.8	94.42%
	5	1000	0.8	227.8	77.22%
	5	1000	0.7	239.6	76.04%
	10	2000	0.95	19	97.15%
	10	2000	0.9	199.8	90.01%
	10	2000	0.8	790.8	60.46%
`Left-skewed`	5	1000	0.95	2.6	99.74%
	5	1000	0.9	24.2	97.58%
	5	1000	0.8	111.4	88.86%
	5	1000	0.7	275.2	72.48%
	10	2000	0.95	3.2	99.84%
	10	2000	0.9	3.8	99.81%
	10	2000	0.8	268.2	86.59%
`Right-skewed`	5	1000	0.95	8.6	99.14%
	5	1000	0.9	9.6	99.04%
	5	1000	0.8	105.4	89.46%
	5	1000	0.7	264.0	73.60%
	10	2000	0.95	50	97.50%
	10	2000	0.9	378	81.10%
	10	2000	0.8	639.2	68.04%

Table 3: Efficiency of Bundle Preprocessing Approach: Stratified Sampling Instances (D: Equal Probabilities)

Problem Instances			$\|B\|$	R
n	$\|S\|$	p		
6	9375	0.9	432	95.39%
6	12500	0.9	729	94.17%
6	15625	0.9	972	93.78%
8	36864	0.9	768	97.91%
8	61440	0.9	864	98.59%
8	65536	0.9	1152	98.24%
6	9375	0.95	144	98.46%
6	12500	0.95	324	97.41%
6	15625	0.95	288	98.16%
8	36864	0.95	576	98.44%
8	61440	0.95	648	98.95%
8	65536	0.95	486	99.26%

Table 4: Efficiency of Bundle Preprocessing Approach: Discrete Probability Distribution

		Reduction Percentage			
Problem Instances		D: Equal Probabilities		D: General Probabilities	
n	$\|S\|$	Criterion 1	Criterion 2	Criterion 1	Criterion 2
10	500	89.82%	62.24%	87.19%	64.10%
20	500	98.26%	68.21%	98.53%	68.75%
50	500	$> 99.90\%$	$> 99.91\%$	98.59%	$> 99.94\%$
10	1000	91.96%	48.33%	94.30%	58.53%
20	1000	99.83%	77.79%	99.86%	73.52%
50	1000	$> 99.65\%$	$> 99.68\%$	$> 99.66\%$	$> 99.67\%$

Table 5: Reduction in CPU Times by "Strengthening Valid Inequalities" ($p = 0.9$)

Problem Instances		Equal Probabilities		General Probabilities	
n	$\|S\|$	Criterion 1	Criterion 2	Criterion 1	Criterion 2
10	500	99.86%	99.79%	99.94%	99.73%
20	500	99.99%	99.41%	99.98%	99.55%
50	500	99.90%	99.97%	99.90%	99.94%
10	1000	99.93%	99.98%	99.95%	99.99%
20	1000	99.89%	99.90%	99.90%	99.90%
50	1000	99.65%	99.68%	99.66%	99.67%

Table 6: Reduction in CPU Times by "Fixing Strategy" ($p = 0.9$)

Recall that the use of the fixing strategy makes the solution method heuristic. The applicability of the heuristic depends, apart from the computational efficiency, on the solution quality (effectiveness). In order to evaluate the effectiveness, we calculate the optimality gap, which requires the knowledge of the optimal objective function value. However, as mentioned before, solving the MKP formulation directly using a standard MIP solver is hard for large problem instances. We derive a "stronger formulation of MKP" using a modeling technique that Luedtke et al. (2010) employed for probabilistically constrained linear programming problems. In order to evaluate the computational efficiency and effectiveness of the heuristic algorithm, we solve the reformulated problem for each problem instance, compare the CPU times, and analyze the optimality gap associated with the proposed heuristic. We note that the approach of solving the stronger formulation of MKP to obtain an optimal solution is referred as the "stronger formulation method".

Problem Instances			Average CPU			Average Number of Iterations	
D	n	$\|S\|$	Criterion 2	Criterion 1	Stronger Form. Meth.	Criterion 2	Criterion 1
Equal	10	500	0.44	0.50	0.48	10.2	8.5
	20	500	1.16	1.15	0.94	7.2	6.2
	50	500	3.36	3.38	2.43	4.3	4.0
	10	1000	1.04	2.05	1.78	10.5	9.2
	20	1000	3.37	3.86	3.63	8.00	7.00
	50	1000	11.12	12.07	9.00	4.0	4.0
	10	2000	2.64	29.24	7.07	10.3	9.5
	20	2000	11.49	124.32	13.92	8.0	7.0
	10	3000	3.75	138.29	15.67	10.2	9.3
	10	5000	8.05	1264.19	78.17	10.5	10.2
General	10	500	0.47	0.54	0.50	10.3	9.0
	20	500	1.20	1.24	0.94	7.3	6.3
	50	500	3.54	3.51	2.44	6.7	6.0
	10	1000	1.17	1.99	1.76	10.7	9.0
	20	1000	3.40	3.59	3.58	8.0	7.0
	50	1000	11.27	11.93	8.90	6.7	5.7
	10	2000	2.56	16.66	7.10	10.8	9.8
	20	2000	11.86	55.86	13.97	8.3	7.0
	10	3000	3.97	170.30	15.68	10.2	9.5
	10	5000	7.25	1166.92	78.16	10.2	9.8

Table 7: Efficiency of Heuristic Algorithm ($p = 0.9$).

Effectiveness: The optimality gap values are all zero.

Table 7 presents the CPU times and the number of iterations averaged over 6 problem instances from Family 1 and shows that the heuristic algorithm based on "Criterion 2" terminates very fast. Moreover, using any of the criterion ("Criterion 1" or "Criterion 2"), the heuristic provides the optimal solution of MKP (i.e., zero optimality gap) for each problem instance. This indicates that, for the Family 1 problem instances, the optimal solution obtained with the fixing strategy defines an exact pLEP.

Tables 8 and 9 focus on the solution quality of the heuristic, and show that, for very fine and stratified sampled probability distributions (Table 8) as well as for the discrete distributions with very large support (Table 9), the heuristic is very fast to find solutions with very small optimality gaps. This indicates that the quasi pLEP defined by the solution, which is obtained by the heuristic, is very close being an exact pLEP. Thus, the requirements imposed by the quasi pLEP are only marginally more demanding than those defined by the exact pLEP and can be used to derive an inner approximation for problem (1)-(4), that is essentially of the "quality" (i.e., as tight) as the inner approximation obtained from the exact pLEP."

We would like to remark that we solve the stronger formulation of MKP after applying the bundle preprocessing technique. The number of scenarios is extremely large for some instances, and, without preprocessing, we were not able to solve the "stronger formulation of MKP".

The numerical results show that the selection criterion used while constructing the sets $T_0^{(t)}$ and $T_1^{(t)}$ has a significant role in the computational performance of the outer approximation algorithm. Even if both

criteria perform similarly in terms of the number of iterations required for the algorithm to terminate, it turns out that for the generated problem instances the random criterion, which we refer to as "Criterion 2", provide results with smaller CPU times in general. Thus, there is a room for further improvements by coming up with alternate criteria. One may also consider different ways of setting the value of Q (see Section 4.1) for further computational improvements.

Problem Instances				Average Optimality Gap		Average CPU		Average Number of Iterations	
Skewness	n	$\|S\|$	p	Criterion 1	Criterion 2	Criterion 2	Criterion 1	Criterion 2	Criterion 1
`Symmetric`	5	1000	0.9	0.76%	0.00%	0.16	0.2	20.2	20.6
	5	1000	0.8	0.00%	0.00%	5.82	6.54	20.8	20.8
	5	1000	0.7	0.00%	0.00%	0.43	0.43	20.8	20.8
	10	2000	0.9	0.00%	0.00%	0.28	0.36	9.2	10.0
	10	2000	0.8	1.12%	1.73%	69.5	191.21	17.2	17.6
`Left-skewed`	5	1000	0.9	0.00%	0.00%	0.06	0.05	13.4	14.6
	5	1000	0.8	0.00%	1.09%	1.18	1.31	20.4	20.6
	5	1000	0.7	1.71%	0.46%	6.21	6.06	20.0	19.75
	10	2000	0.9	0.00%	0.00%	0.01	0.01	3.0	2.5
	10	2000	0.8	0.00%	0.00%	0.61	0.6	12.8	12.8
`Right-skewed`	5	1000	0.9	0.00%	0.00%	0.02	0.02	6.6	6.0
	5	1000	0.8	0.62%	1.58%	0.29	1.4	16.4	17.4
	5	1000	0.7	0.52%	0.78%	2.22	3.06	18.4	17.2
	10	2000	0.9	0.35%	1.34%	17.33	39.43	18.6	17.8
	10	2000	0.8	2.08%	2.30%	58.15	212.69	19.0	18.6

Table 8: Effectiveness of Heuristic Algorithm: Stratified Sampling Instances (D: Equal Probabilities)

Problem Instances		Average Optimality Gap		Average CPU		Average Number of Iterations	
n	$\|S\|$	Criterion 1	Criterion 2	Criterion 2	Criterion 1	Criterion 2	Criterion 1
6	9375	0.14%	0.45%	3.42	1.67	19	19
6	12500	0.05%	0.59%	3.36	3.3	19	19
6	15625	0.67%	1.28%	5.49	6.52	19	19
8	36864	0.41%	0.67%	3.02	4.26	16	17
8	61440	0.00%	0.47%	2.92	4.04	16	16
8	65536	0.73%	0.94%	5.18	7.67	16	16

Table 9: Effectiveness of Heuristic Algorithm: Discrete Probability Distribution ($p = 0.9$)

7. Concluding Remarks Probabilistically constrained problems, in which the random variables are finitely distributed, are generally not convex and thus very challenging to solve. The methods based on the p-efficiency concept have been widely used to solve such problems and typically use an enumeration algorithm to find the p-efficient points. In this study, we propose an alternative approach to the existing enumeration algorithms. First, we formulate a MIP problem whose optimal solution defines a pLEP and whose feasible solutions with small optimality gaps define quasi pLEPs. Quasi pLEPs impose conditions that are sufficient for the satisfaction of probabilistic constraint (2) and that are only marginally more demanding than those defined by pLEPs. Second, we propose a mathematical programming framework to generate exact and quasi

pLEPs. The proposed framework constructs a sequence of increasingly tighter outer approximation problems. The exact solution method uses a bundle preprocessing technique and strengthening valid inequalities. The heuristic method combines the above two techniques with a fixing strategy. We perform an extensive computational study considering different types of probability distributions with finite support (discrete distribution with finite support, sampled distributions obtained with random and stratified sampling). The numerical results attest to the effectiveness and computational efficiency of each of the individual algorithmic techniques as well as of the overall outer approximation framework.

Finally, we note the following additional contributions of this study. First, the outer approximation method can be used to derive, not only one, but a subset of pLEPs. Second, the development of powerful methods for the application of the p-efficiency concept to random variables described by a finite set of scenarios is novel. Third, the bundle preprocessing method is very powerful, since *(i)* it reduces tremendously the number of scenarios, thereby allowing for the solution of very complex problems that could not be solved otherwise, and *(ii)* it is general enough to be incorporated in other existing solution methods for probabilistically constrained problems.

Acknowledgment. The first author is supported by Grant # W911NF-09-1-0497 from the Army Research Office.

References

[1] Avital, I. Chance-Constrained Missile-Procurement and Deployment Models for Naval Surface Warfare. Naval Postgraduate School Dissertation; Monterrey, CA; 2005.

[2] Beraldi, P, Ruszczyński, A. The Probabilistic Set Covering Problem. Operations Research 2002; 50; 956-967.

[3] Calafiore G.C, Campi, M.C. Uncertain Convex Programs: Randomized Solutions and Confidence Levels. Mathematical Programming 2005; 102; 25-46.

[4] Charnes A, Cooper, W.W, Symonds, G.H. Cost Horizons and Certainty Equivalents: An Approach to Stochastic Programming of Heating Oil. Management Science 1958; 4; 235-263.

[5] Dentcheva D. Optimization Models with Probabilistic Constraints. In: Calafiore G, Dabbene, F (Eds), Probabilistic and Randomized Methods for Design under Uncertainty. Springer-Verlag: London; 2006.

[6] Dentcheva D, Lai, B, Ruszczyński, A. Dual Methods for Probabilistic Optimization Problems. Mathematical Methods of Operations Research 2004; 60; 331-346.

[7] Dentcheva D, Prékopa, A, Ruszczyński, A. Concavity and Efficient Points of Discrete Distributions in Probabilistic Programming. Mathematical Programming 2001; 47 (3); 1997-2009.

[8] Dentcheva, D, Prékopa, A, Ruszczyński, A. Bounds for Probabilistic Integer Programming Problems. Discrete Applied Mathematics 2002; 124; 55-65.

[9] Duran M.A, Grossmann, I.E. An Outer-Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programs. Mathematical Programming 1986; 36 (3); 307-339.

[10] Fletcher R, Leyffer, S. Solving Mixed Integer Nonlinear Programs by Outer Approximation. Mathematical Programming 1994; 66; 327-349.

[11] Fourer R, Gay, D.M, Kernighan, B.W. AMPL: A Modeling Language for Mathematical Programming. Second Edition. Duxbury Press Brooks Cole Publishing Co; 2003.

[12] Hammer P.L, Johnson, E.L, Peled, U.N. Facets of Regular 0-1 Polytopes. Mathematical Programming 1975; 8; 179-206.

[13] Henrion R. 2004. Introduction to Chance-Constrained Programming. Tutorial Paper for the Stochastic Programming Community Home Page. `http://stoprog.org/index.html?SPIntro/intro2ccp.html`.

[14] ILOG. 2008. ILOG AMPL CPLEX System Version 11.0 User's Guide. ILOG CPLEX Division. `http://www.ilog.com`.

[15] Kress M, Penn, M, Polukarov, M. The Minmax Multidimensional Knapsack Problem with Application to a Chance-Constrained Problem. Naval Research Logistics 2007; 54; 656-666.

[16] Kücükyavuz S. 2009. On Mixing Sets Arising in Probabilistic Programming. Working Paper: `http://www.optimization-online.org/DB_HTML/2009/03/2255.html`.

[17] Lejeune M.A. Preprocessing Techniques and Column Generation Algorithms for p-Efficiency. Journal of Operational Research Society 2008; 59; 1239-1252.

[18] Lejeune M.A. Linear Reformulation of Probabilistically Constrained Optimization Problems Using Combinatorial Patterns. International Colloquium on Stochastic Modeling and Optimization, Dedicated to the 80th birthday of Professor András Prékopa. Piscataway, NJ; 2009.

[19] Lejeune M.A, Ruszczyński, A. An Efficient Trajectory Method for Probabilistic Inventory-Production-Distribution Problems. Operations Research 2007; 55 (2); 378-394.

[20] Luedtke J, Ahmed, S, Nemhauser, G. An Integer Programming Approach for Linear Programs with Probabilistic Constraints. Mathematical Programming 2010; 122 (2); 247–272.

[21] Nemirovski A, Shapiro, A. Convex Approximations of Chance Constrained Programs. SIAM Journal on Optimization 2006; 17; 969–996.

[22] Prékopa A. On Probabilistic Constrained Programming. In: Proceedings of the Princeton Symposium on Mathematical Programming. Princeton University Press: Princeton, NJ; 1970; p. 113–138.

[23] Prékopa A. Contributions to the Theory of Stochastic Programming. Mathematical Programming 1973; 4;202–221.

[24] Prékopa A. Dual Method for a One-Stage Stochastic Programming Problem with Random RHS Obeying a Discrete Probability Distribution, Zeithschrift für Operations Research 1990; 34; 441–461.

[25] Prékopa A. Stochastic Programming. Kluwer Academic: Dordrecht, Boston; 1995.

[26] Prékopa A. Probabilistic Programming. In: Ruszczynski A, Shapiro A (Eds), Stochastic Programming, Handbooks in Operations Research and Management Science, vol.10. Elsevier: Amsterdam; 2003; p. 267–351.

[27] Prékopa A, Vizvari, B, Badics, T. Programming Under Probabilistic Constraint with Discrete Random Variable. In: Giannessi F., Komlósi, S., Rapcsák, T. (Eds), New Trends in Mathematical Programming. Boston, MA, USA; 1998.

[28] Rockafellar R.T, Wets, R.J-B. Scenarios and Policy Aggregation in Optimization Under Uncertainty. Mathematics of Operations Research 1991; 16 (1); 119-147.

[29] Ruszczyński A. Probabilistic Programming with Discrete Distributions and Precedence Constrained Knapsack Polyhedra. Mathematical Programming 2002; 93; 195-215.

[30] Saxena A, Goyal, V, Lejeune, M.A. MIP Reformulations of the Probabilistic Set Covering Problem. Mathematical Programming 2010; 121 (1); 1-31.

[31] Sen S. Relaxations for Probabilistically Constrained Programs with Discrete Random Variables. Operations Research Letters 1992; 11; 81-86.