Pesquisa Operacional (2017) 37(3): 597-613

Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope
doi: 10.1590/0101-7438.2017.037.03.0597

STOCHASTIC KNAPSACK PROBLEM: APPLICATION TO TRANSPORTATION PROBLEMS

Stefanie Kosuch, Marc Letournel and Abdel Lisser*

Received May 15, 2017 / Accepted November 1, 2017

ABSTRACT. In this paper, we study the stochastic knapsack problem with expectation constraint. We solve the relaxed version of this problem using a stochastic gradient algorithm in order to provide upper bounds for a branch-and-bound framework. Two approaches to estimate the needed gradients are studied, one based on Integration by Parts and one using Finite Differences. The Finite Differences method is a robust and simple approach with efficient results despite the fact that estimated gradients are biased, meanwhile Integration by Parts is based upon more theoretical analysis and permits to enlarge the field of applications. Numerical results on a dataset from the literature as well as a set of randomly generated instances are given.

Keywords: Stochastic knoasack problem, transportation problem, probabilistic constraint, Branch and Bound, Integration by parts.

1 INTRODUCTION

The deterministic knapsack problem is a well known and well studied NP-hard combinatorial optimization problem. It consists in filling a knapsack with items out of a given set such that the weight capacity of the knapsack is respected and the total reward maximized. Several variants of the knapsack problem have been considered for many practical problems amongst all network design problems. In telecommunication network design, the knapsack problem is often used as a subproblem for modelling the capacity of the edges (see Kellerer et al., 2004). It is now generally admitted that the uncertainty is an inherent property in many practical problems. The main reason relies on the fact that the use of average parameters could lead to severe service quality deterioration whereas the use of extreme values could result into costly conservative solutions. In telecommunication networks, the main sources of uncertainty are the traffic demands and the costs. Additional sources could be network and equipments failures. There are two main approaches to deal with uncertainty, namely robust optimization (Ben-Tal & Nemirowski, 2008) or stochastic optimization (Birge & Louveaux, 1997). Demand uncertainty in telecommunication

*Corresponding author.
Laboratoire de Recherche en Informatique (LRI), Université Paris Sud – XI, Bât. 650, 91405 Orsay Cedex, France.
E-mails: stefanie@kosuch.eu; marc.letournel@lri.fr; lisser@lri.fr

network design problems leads to several stochastic models amongst all the stochastic knapsack problem. The latter was used either with chance constraints (Kosuch & Lisser, 2010) or within a two-stage stochastic knapsack problems either for the single formulation of for the multiple knapsack one (Tönissen et al., 2016, 2017). The stochastic knapsack problem with random item sizes has met a large interest in the literature in the last decade. There are at least two ways to deal with a possible overload. When the overload is not considered, Bhalgat & Khanna (2011) proposed to remove the last inserted item. In Chen & Ross (2014), the authors proposed that the knapsack returns zero when it overflows. In the case where the overflow is accepted, the knapsack constraint is replaced by a probabilistic constraint (see Goal & Indyk, 1999; Kosuch & Lisser, 2010).

In stochastic optimization there are three major approaches to deal with stochasticity: *Probabilistic constraint*, *two- or multi-stage settings* as well as *on-line or dynamic modeling*. In the first case, all decisions are made before the random parameters are revealed. The objective function and/or the constraints are thus formulated using expectations and probability measures. In the second case, recourse actions occur when random parameters are known. These recourse actions are formulated as a penalty that has to be paid in the case of violated constraint or are used to correct the decisions made in previous stages. In the last approach, on-line problems are solved dynamically. More precisely, the algorithm implemented to solve the problem is provided with further information during its execution and reacts dependently on this new information as well as previous decisions. The solution of an on-line problem is a decision policy while in the first two cases the current decisions are computed.

In this paper we focus on a particular variant of the single-stage stochastic knapsack problem with random weights: the *expectation constrained* knapsack problem. The paper is organized as follows: In section 2 the mathematical formulation of the problem is presented and discussed. In section 3 we present the stochastic gradient algorithm used to solve the corresponding relaxed problem. Two further methods to estimate the needed gradients are presented. On the opposite to the *Approximation by Convolution method* studied in Kosuch & Lisser (2010) which gives an approximation of the gradient, the method of Integration by Parts used in this article replaces the expected function by another one with exactly the same expectation. In section 4 the convergence of the stochastic gradient algorithm is analyzed from a theoretical as well as numerical points of view and test results concerning the solution of the relaxed problem are presented. In section 6 we use these methods to solve the combinatorial problem described in Cohn & Barnhart (1998) using a branch-and-bound framework presented in Kosuch & Lisser (2010). Numerical results on a set of randomly generated data are given and analyzed.

2 MATHEMATICAL FORMULATIONS

We consider a stochastic knapsack problem of the following form: Given a set of n items, we want to choose these items without knowing the exact value of their weights. Therefore, we handle the weights as random variables and assume that weight χ_i of item i is independently normally distributed with mean $\mu_i > 0$ and standard deviation σ_i. Furthermore, each item has a

fixed reward per weight unit $r_i > 0$. The choice of a reward per weight unit can be motivated by the fact that the value of an item often depends on its weight which we do not know in advance[1]. We denote by χ, μ, σ and r the corresponding n-dimensional vectors. The aim is to maximize the expected total gain $\mathbb{E}[\sum_{i=1}^{n} r_i \chi_i x_i]$. Our knapsack problem has a fixed weight capacity $c > 0$ but due to the stochastic nature of the weights, we need to define correctly what respecting capacity means. In this paper, we solve the following expectation constrained knapsack problem:

Chance Constrained Knapsack Problem ($ECKP$)

$$\max_{x \in \{0,1\}^n} \mathbb{E}\left[\sum_{i=1}^{n} r_i \chi_i x_i\right] \tag{1}$$

$$\text{s.t.} \quad \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))\right] \geq p \tag{2}$$

where $\mathbb{E}[\cdot]$ denotes the expectation, $g(x, \chi) := \sum_{i=1}^{n} \chi_i x_i$ is the total weight of the chosen items, $\mathbb{H}_{\mathbb{R}^+}(\cdot)$ denotes the indicator function of the positive real interval where $\cdot = 1$ if $\cdot \in \mathbb{R}^+$ and 0 otherwise, and $p \in (0.5, 1)$ is the prescribed probability.

The choice of p is a decision parameter that restricts the percentage of cases where the capacity is exceeded. p is not connected with the amount of overweight. The constraint (2) can be equivalently reformulated as the following chance constraint:

$$P\{g(x, \chi) \leq c\} \geq p \tag{3}$$

Without loss of generality, we assume that $\mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - \chi_i)\right] \geq p$ for all $i \in \{1, \ldots, n\}$: any item that does not satisfy this constraint could be excluded from the beginning. It follows, that the optimal solution vector x^* has at least one non-zero component.

We call J the objective function of the above maximization problem defined by $J(x) = \mathbb{E}[\sum_{i=1}^{n} r_i \chi_i x_i]$. We denote $j(x, \chi) = \sum_{i=1}^{n} r_i \chi_i x_i$.

Furthermore, we refer to the function on the left hand side of the constraint as Θ and to the function inside the expectation of Θ as θ, i.e. $\Theta(x) = \mathbb{E}[\theta(x, \chi)] = \mathbb{E}[\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))]$.

Throughout this paper, we assume that the weights are normally distributed. Normally distributed items have the nice property that their linear combination is still normally distributed. This property ensures easier calculations but is not a foremost condition. In some extent, in the case of normal distributions, it is even possible to compute directly the gradient of the expectation function. Assuming normal distribution is, in a large part, just a practical frame in order to produce numerical results and theoretical formulations. Whenever we use normal distribution properties, we will explain how to overpass this special consideration.

Normally distributed random variables can have negative realizations, and assuming normally distributed weights might thus seem contradictory to the fact that item weights are always strictly

[1] All the methods used and results presented in this article are still valid in the case where the rewards are deterministic or random but independent of the weights. We do not use the fact that the rewards (per item) are normally distributed, therefore any other distribution could be considered.

positive. However, in most real life applications the standard deviation is several times smaller than mean values of the unknown parameters. In this case, the probability of negative weights becomes negligible.

3 PROBLEM SOLVING METHOD

Due to its combinatorial nature, $ECKP$ can be solved using a branch-and-bound framework as presented in Kosuch & Lisser (2010). To obtain upper bounds, the authors propose to solve the corresponding continuous optimization problem. A stochastic gradient algorithm can be used to solve this problem. This method needs to evaluate the gradient of the expected value function, which is an indicator function $\mathbb{H}_{\mathbb{R}^+}(\cdot)$. In Kosuch & Lisser (2010), this computation has been done using Approximation by Convolution. In this paper, we study two different approaches: the first one is a non-biased estimator based on Integration by Parts (called hereafter *IP-method*). The second approach is a Finite Differences estimator (*FD-method*) presented in Andrieu et al. (2007). Like the Approximation by Convolution method, FD-method provides a biased estimator of the gradient.

Instead of replacing $\{0, 1\}^n$ by $[0, 1]^n$ when relaxing $ECKP$, the theoretical analysis will compel us to consider a complementary set of a neighborhood of $0_{[0,1]^n}$. Considering that an empty knapsack is not an optimal solution, it follows that the optimal solution vector of the continuous problem contains at least one component x_κ with $x_\kappa \geq 1/n$. We are thus allowed to replace $[0, 1]^n$ by $\{x \in [0, 1]^n \,|\, ||x||_\infty \geq 1/n\} =: X_{cont}$. Accordingly, we obtain the following feasible set of the relaxed $ECKP$:

$$X_{cont}^{ad} = \{x \in X_{cont} : \Theta(x) \geq p\}$$

3.1 The stochastic gradient type algorithm

To solve the relaxed version of $ECKP$, we use a stochastic gradient type algorithm. Applying the gradient method is promising as the objective function is concave and, in addition, constraint (2) defines a convex feasible set due to the assumption that the weights are independently normally distributed.

We propose to solve $ECKP$ with a Stochastic Arrow-Hurwicz algorithm (Culioli & Cohen, 1995) (hereafter called SAH-algorithm; see Algorithm 3.1) that uses Lagrangian multipliers to deal with the probability constraint.

$\mathcal{L}(x, \lambda) = \mathbb{E}[l(x, \lambda, \chi)]$ denotes the Lagrangian function associated with $ECKP$.

$$\mathcal{L}(x, \lambda) = \mathbb{E}\left[\sum_{i=1}^{n} r_i \chi_i x_i\right] - \lambda \left(p - \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))\right]\right)$$

It follows that

$$(r^k + \lambda^{k-1}\tau^k) = \nabla_x l(x^k, \lambda^k, \chi^k)$$

and

$$(p - \theta(x^{k+1}, \chi^k)) = l'_\lambda(x^{k+1}, \lambda^k, \chi^k)$$

Algorithm 3.1 Stochastic Arrow-Hurwicz Algorithm.

1. Choose $x^0 \in X_{cont}^{ad}$ and $\lambda^0 \in [0, \infty)$ as well as two σ-sequences $(\epsilon^k)_{k\in\mathbb{N}^*}$ and $(\rho^k)_{k\in\mathbb{N}^*}$. Set $k = 1$.
2. Given x^{k-1} and λ^{k-1}, draw χ_k following its normal distribution, compute $r^k = \nabla_x j(x^{k-1}, \chi^k)$, $\tau^k = \nabla_x \theta(x^{k-1}, \chi^k)$ and update x and λ as follows:

$$x^k = x^{k-1} + \epsilon^k(r^k + \lambda^{k-1}\tau^k)$$

$$\lambda^k = \lambda^{k-1} + \rho^k(p - \theta(x^k, \chi^k))$$

3. For all $h = 1, \ldots, n$: If $x_h^k > 1$ set $x_h^k = 1$ and if $x_h^k < 0$ set $x_h^k = 0$.
4. If $x_h^k < 1/n$ for all $h = 1, \ldots, n$, set $x_h^k = 1/n$ for one $h \in \{1, \ldots, n\}$.
5. If $\lambda^k < 0$ set $\lambda^k = 0$.
6. If $k = k_{\max}$: STOP. Else: Set $k = k + 1$. Go to step 0.

where r^k, λ^k and τ^k are defined in Algorithm 3.1. Note that in the deterministic form of the Arrow-Hurwicz algorithm we can use the gradients of the Lagrangian itself. But this function is here an expectation function and its gradient is thus difficult to compute. By drawing independent samples of the random variables at each iteration of the algorithm, the expectation of the gradient is approximated (see Culioli & Cohen, 1995). At each iteration of the algorithm, we need to calculate the gradient of θ at (x^k, χ^k). However, θ is mainly an indicator function and therefore not useful to differentiate. In the following subsection, we present two ways to bypass this disadvantage: either by approximation using Finite Differences or by reformulation of the constraint function Θ using Integration by Parts.

3.1.1 Computation of the gradient of θ

The first method consists in approximating the h^{th} component of the gradient of θ by the corresponding difference ratio

$$\frac{\theta(x + \delta\nu^h, \chi) - \theta(x - \delta\nu^h, \chi)}{2\delta}$$

where $\delta > 0$ and $\nu^h \in \{0, 1\}^n$ such that $\nu_h^h = 1$ and $\nu_i^h = 0$ for $i \neq h$. This leads to the following approximation of the h^{th} component of the gradient of θ:

$$(\nabla_x(\theta_\delta)(x, \chi))_h = \frac{\mathbb{H}_{\mathbb{R}^+}(c - g(x + \delta\nu^h, \chi)) - \mathbb{H}_{\mathbb{R}^+}(c - g(x - \delta\nu^h, \chi))}{2\delta}$$

The second method (IP-method) involves Integration by Parts to reformulate $\mathbb{E}[\theta(x, \chi)]$ and to obtain a function in expectation $\mathbb{E}[\widetilde{\theta}(x, \chi)]$ s.t. $\mathbb{E}[\widetilde{\theta}(x, \chi)] = \mathbb{E}[\theta(x, \chi)] = \Theta(x)$. $\widetilde{\theta}$ is the function obtained by Integration by parts, it is differentiable and the idea is to use the gradient of $\widetilde{\theta}$ in SAH-algorithm.

Andrieu et al. (Andrieu et al., 2007) presented how to compute the gradient of an indicator function in expectation using Integration by Parts (see Theorem 5.5 in Andrieu, 2004). We state and

proof their theorem for the case of $ECKP$. The variables and functions used in this proposition are defined in section 2:

Proposition 1. *Let $\mathbb{Y}_{\mathbb{R}^+}(\cdot)$ be a primitive of $\mathbb{H}_{\mathbb{R}^+}(\cdot)$. Let $x \in X^{ad}_{cont}$ and let $\kappa = \kappa(x) \in \{1, \ldots, n\}$ be defined such that $x_\kappa = ||x||_\infty \geq 1/n$. Then, using Integration by Parts, we get*

$$\Theta(x) = \mathbb{E}\left[\mathbb{Y}_{\mathbb{R}^+}(c - g(x, \chi))M_\kappa(x, \chi)\right]$$

where

$$M_\kappa(x, \chi) = -\frac{(\chi_\kappa - \mu_\kappa)}{\sigma_\kappa^2}\frac{1}{x_\kappa}$$

It follows

$$\nabla_x \Theta(x) \quad = \quad \mathbb{E}\left[-\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))M_\kappa(x, \chi)\chi + \mathbb{Y}_{\mathbb{R}^+}(c - g(x, \chi))\nabla_x M_\kappa(x, \chi)\right]$$

Proof. Let φ denote the density function of the random vector $\chi = (\chi_1, \ldots, \chi_n)$ and define

$$u'_{\chi_\kappa}(x, \chi) := -\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))x_\kappa \qquad \text{and}$$
$$v(x, \chi) := -\frac{\varphi(\chi)}{x_\kappa}$$

It follows

$$\Theta(x) = \int_{-\infty}^{\infty} \mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))\varphi(\chi)\ \mathrm{d}\chi = \int_{-\infty}^{\infty} u'_{\chi_\kappa}(x, \chi)v(x, \chi)\ \mathrm{d}\chi$$

Integration by Parts over χ_κ leads to

$$\Theta(x) = \left[u(x, \chi)v(x, \chi)\right]_{-\infty}^{\infty} - \int_{-\infty}^{\infty} u(x, \chi)v'_{\chi_\kappa}(x, \chi)\ \mathrm{d}\chi$$
$$= -\int_{-\infty}^{\infty} u(x, \chi)v'_{\chi_\kappa}(x, \chi)\ \mathrm{d}\chi = -\int_{-\infty}^{\infty} \mathbb{Y}_{\mathbb{R}^+}(c - g(x, \chi))v'_{\chi_\kappa}(x, \chi)\ \mathrm{d}\chi$$

In our case the random variables are independently distributed. With φ^i being the density function of χ_i we get

$$\varphi'_{\chi_\kappa}(\chi) = \prod_{i \neq \kappa} \varphi^i(\chi_i) \cdot (\varphi^\kappa)'_{\chi_\kappa}(\chi_\kappa) = \prod_{i \neq \kappa} \varphi^i(\chi_i) \cdot \left(-\frac{(\chi_\kappa - \mu_\kappa)}{\sigma_\kappa^2}\varphi^\kappa(\chi_\kappa)\right)$$
$$= -\frac{(\chi_\kappa - \mu_\kappa)}{\sigma_\kappa^2}\varphi(\chi)$$

It follows

$$v'_{\chi_\kappa}(x,\chi) = \frac{\partial}{\partial \chi_\kappa}\left(-\frac{\varphi(\chi)}{x_\kappa}\right) = \frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\varphi(\chi)$$

and therefore

$$\begin{aligned}\Theta(x) &= -\int_{-\infty}^{\infty} \mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\varphi(\chi)\,\mathrm{d}\chi \\ &= \mathbb{E}\Big[-\mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\Big]\end{aligned}$$

□

If $\nabla_x \Theta(x) = \mathbb{E}\Big[\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\right)\Big]$ we get the result. Thus, it remains to proof the following claim:

Claim 3.1.

$$\nabla_x \Theta(x) = \mathbb{E}\Big[\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\right)\Big]$$

Proof of the claim. This is a classical result under the assumption of uniform bounding of the gradient function inside the integral. In our case, we can easily show that this bounding is obtained according to the assumption that $x_\kappa \geq 1/n$. First of all let us observe that we can choose $\mathbb{Y}_{\mathbb{R}^+}(x) = \mathbb{H}_{\mathbb{R}^+}(x) \cdot x = [x]^+$. Therefore, the κ-th component of

$$\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\right)$$

can be reformulated as follows:

$$\begin{aligned}&\left(\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\right)\right)_\kappa \\ &= \mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\chi_\kappa + [c - g(x,\chi)]^+\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa^2 \sigma_\kappa^2} \\ &= \mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))(\chi_\kappa - \mu_\kappa)\left(\frac{\chi_\kappa}{x_\kappa \sigma_\kappa^2} + \frac{(c - g(x,\chi))}{x_\kappa^2 \sigma_\kappa^2}\right)\end{aligned}$$

For $(\chi_\kappa - \mu_\kappa) > 0$ and as $[(c - g(x,\chi)) \geq 0 \Leftrightarrow \mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi)) = 1]$ it follows

$$\begin{aligned}0 &\leq (\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\right))_\kappa \\ &\leq (\chi_\kappa - \mu_\kappa)\left(n \cdot \frac{\chi_\kappa}{\sigma_\kappa^2} + n^2 \cdot \frac{(c - g(x,\chi))}{\sigma_\kappa^2}\right)\end{aligned}$$

If $(\chi_\kappa - \mu_\kappa) < 0$, the bounds are inversed.

For the other components of the gradient, we just have $\mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\chi_h$ and the same limitations holds for the κ-th component. Note that $\tilde{\Theta}$ is differentiable except in 0. □

4 CONVERGENCE OF THE STOCHASTIC ARROW-HURWICZ ALGORITHM

4.1 Theoretical convergence

Culioli & Cohen (1995) showed how to obtain the convergence of SAH-algorithm in the case where the objective function J is convex despite the fact that j is not (for a global survey see Carpentier, 2010). More precisely, they adapt the classical SAH-algorithm in the case where the constraint is given in expectation by considering a subgradient both in dual variables x and λ instead of only in λ. The drawback is to check technical assumptions on both gradients that have to be linearly bounded. The way we transform the expected function in the constraint permits to check correctly all the assumptions with some limitations explained below.

Theorem 1 (Culioli, Cohen (1995)). *Suppose the following assumptions to be satisfied:*

h1: $\theta(\cdot, \chi)$ *is differentiable with gradient uniformly bounded with respect to* χ.

h2: The associated Lagrangian admits a saddle point $\widetilde{x}$ *and* J *is strictly convex.*

h3: $\forall \chi \in \mathbb{R}^n$, $\theta(\cdot, \chi)$ *is locally Lipschitz continuous.*

h4: There exists $c_1, c_2 > 0$ *such that*
$\forall\, \chi \in \mathbb{R}^n, \forall\, x, y \in X_{cont}, ||\theta(x, \chi) - \theta(y, \chi)|| \leq c_1||x - y|| + c_2$.

h5: $\Theta(x)$ *is Lipschitz continuous and concave.*

h6: There exists $\alpha, \beta > 0$ *such that*
$\forall\, \chi \in \mathbb{R}^n, \forall\, x \in X_{cont}\ ||\nabla_x j(x, \chi)|| \leq \alpha||x - \widetilde{x}|| + \beta$.

h7: ϵ^k / ρ^k *is monotone non-increasing.*

h8: There exists $\gamma, \delta > 0$ *such that*
$\forall\, x \in X_{cont}\ \mathbb{E}[\theta(x, \chi) - \Theta(x)]^2 < \gamma||x - \widetilde{x}||^2 + \delta$.

Then, the sequence (x^k, λ^k) *is bounded and* x^k *converges weakly towards* $\widetilde{x}$.

We are now going to check all the assumptions on our sample set and situation, we focus on the IP-method:

h1 As presented in subsection 3.1.1, it is possible to replace $\theta(x, \chi)$ by a differentiable function $\widetilde{\theta}(x, \chi)$ such that $\mathbb{E}[\widetilde{\theta}(x, \chi)] = \mathbb{E}[\theta(x, \chi)]$ (IP-method).

Using the IP-method we obtain the following gradient:

$$\begin{aligned}\nabla_x \widetilde{\theta}(x, \chi) = \quad & \mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi)) \tfrac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2} \chi \\ & + [c - g(x, \chi)]^+ \tfrac{(\chi_\kappa - \mu_\kappa)}{x_\kappa^2 \sigma_\kappa^2} \nu^\kappa\end{aligned}$$

The fact that $\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi)) = 1$ only for $c - g(x, \chi) \geq 0$ limits χ to a compact domain and h1 is checked.

h2 We suppose that the Lagrangian function admits a saddle point. Nonetheless, we observe that strict convexity of J is a sufficient condition to get the stability of the Lagrangian function. In our case, the function J is linear.

h3 For all $\chi \in \mathbb{R}^n$, $\widetilde{\theta}(\cdot, \chi)$ is locally Lipschitz continuous according to the expression of $\nabla_x \widetilde{\theta}(x, \chi)$ in all points $x \in X_{cont}$ since $x_\kappa \geq \frac{1}{n}$.

h4 Choose $c_2 = 1$. It follows $\forall\, \chi \in \mathbb{R}^n, \forall\, x, y \in X_{cont}\ ||\theta(x, \chi) - \theta(y, \chi)|| \leq c_2$.

h5 As we assume the item weights to be normally distributed, $\Theta(x) = \mathbb{P}\{g(x, \chi) \leq c\}$ is Lipschitz continuous: Let F be the cumulative distribution function of the standard normal distribution. Then we have

$$\Theta(x) = F\left(\frac{c - \sum_{i=1}^{n} \mu_i x_i}{\sqrt{\sum_{i=1}^{n} \sigma_i^2 x_i^2}}\right)$$

It is easy to see that Θ is continuous on X_{cont}. In addition, we have $0 \leq \Theta(x) \leq 1$ for all $x \in \mathbb{R}^n$ and as X_{cont} is a compact set, it follows that Θ is Lipschitz continuous on X_{cont}.

We don't really need normal distribution to establish this assumption, as we replaced θ by $\widetilde{\theta}$, we get that :

$$\forall \chi, \forall (x, y) \in X_{cont}\ ||\widetilde{\theta}(x, \chi) - \widetilde{\theta}(y, \chi)|| \leq \tau ||x - y||$$

where τ bounds $||\nabla_x \widetilde{\theta}||$ given in *h1* on X_{cont}. Then, we integrate the inequality, and we get the bounding on Θ.

The question of concavity of Θ is more complex and depends on the sample values of the objects. There are two different questions to solve: is the constraint concave on the hypercube, and if not, is the constraint concave on a partial subset of the hypercube? The answer to the first question is definitely no. For further details concerning chance constraints, we refer to Prekopa (1995). The second question depends on initial parameters: in a first approach we begin to observe that in the case of a mono dimensional vector x, Θ is concave for $x < \widetilde{x}$ where $\widetilde{x}$ depends on the initial conditions.

In case of a higher dimensional problem, the situation becomes more complex. Despite there exists a domain for x near to 0 where Θ is concave (see Fig. 1 for the value of Θ observed for a two dimensional vector), the boundaries of this domain still depend on the characteristics of the sample set.

A practical check combined with a theoretical analysis of the concavity is then possible. We begin to compute the eigenvalues of the Hessian matrix $H(\widetilde{x})$ at a point $\tilde{x}$ close to 0 of the set X_{cont} and we find them all negative. According to the fact that the determinant of the Hessian matrix is a continuous function of x, it follows that there exists a topologically connected component $C \subseteq X_{cont}$ of $\widetilde{x}$ such that $\det H(x) \neq 0$ for all $x \in C$ and $\widetilde{x} \in C$. We consider the boundary $\overline{C} - C$ of this set and we set $p = \min\{\Theta(x) : x \in \overline{C} - C\}$.

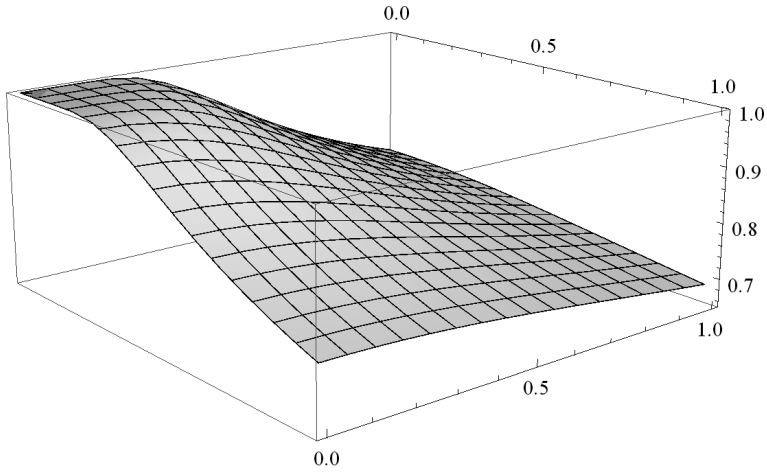

Figure 1 – Constraint function Θ in 2-dimensional case.

To conclude on the above observations, we formulate the following proposition:

Proposition 2. *Let $\chi_1, \ldots, \chi_n$ be a fixed set of independently normally distributed random variables and let c be a fixed capacity such that for the vector x with all components near to 0, the Hessian matrix has all eigenvalues strictly negative. Then, there exists $p = p(\chi_1, \ldots, \chi_n, c)$ such that $\Theta(x)$ is concave on the set $\{x \in X_{cont} | \Theta(x) \geq p\}$.*

In other words, we establish that for a given instance there exists p such that the corresponding Lagrangian relaxation is concave on the feasible set.

h6 j is linear in each component of x. h6 is thus satisfied.

h7 Condition h7 is satisfied for all sequences of type $\frac{const}{k}$ with $const > 0$.

h8 We have

$$\theta(x, \chi) \leq 1 \quad \text{and} \quad \Theta(x) = P\{g(x, \chi) \leq c\} \geq 0$$
$$\Rightarrow \theta(x, \chi) - \Theta(x) \leq 1 \Rightarrow \mathbb{E}[\theta(x, \chi) - \Theta(x)]^2 \leq 1$$

It follows that condition h8 is satisfied for all $\delta > 1$.

4.2 Implementation of the SAH algorithm

4.2.1 Convergence of the Stochastic Arrow-Hurwicz Algorithm involving FD-method

A stopping criterion with SAH algorithm cannot be expressed with an observation of a decreasing speed of variation for the objective value. The classical approach is to set a maximum number of iterations. We fixed the number of iterations to 500 for FD-method according to several tests

where only slight variations can be observed after around 300 iterations (for numerical results see subsection 5 and Table 2).

Table 1 – Numerical results for the continuous $ECKP$.

	Arrow-Hurwicz & FD-method			Arrow-Hurwicz & IP-method			SOCP	
n	Optimum	CPU-time (msec)	Gap	Optimum	CPU-time (msec)	Gap	Optimum	CPU-time (msec)
C./B.	4695.525	3	0.02%	4667.790	30	0.75%	4696.413	4
15	4954.314	5	0.01%	4910.434	30	0.89%	4954.804	4
20	6712.219	8	0.03%	6653.941	40	0.89%	6713.987	6
30	10308.293	14	0.02%	10174.878	53	1.31%	10310.45	18
50	16992.444	28	0.01%	16743.260	92	1.47%	16993.514	65
75	25565.525	52	0.01%	25155.023	138	1.63%	25569.379	213
100	33902.119	85	0%	33208.278	181	2.05%	33903.672	503
150	50677.120	174	0%	49609.044	271	2.10%	50678.312	1802
250	85186.119	438	0%	83785.657	431	1.64%	**	**
500	170226.568	1638		167342.988	868		**	**
1000				334984.436	1720		**	**
5000				1676670.246	8640		**	**
20000				6731686.102	36031		**	**

*Exceeding of the available memory space.

4.2.2 Enhancements in problem formulation when using the Stochastic Arrow-Hurwicz Algorithm involving IP-method

Some improvements in our implementations worth being notified: During our first numerical tests, we remarked that SAH-algorithm involving the IP-method was inefficient and even did not converge in some cases (see Fig. 2). Then we analyzed and modified the implementation of the IP-method. The first enhancement deals with the formulation of $ECKP$. The expectation constraint states

$$\mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))\right] \geq p \Leftrightarrow p - \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))\right] \leq 0$$

We thus get the Lagrangian

$$\mathcal{L}(x, \lambda) = \mathbb{E}\left[\sum_{i=1}^{n} r_i \chi_i x_i\right] - \lambda\left(p - \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))\right]\right)$$

Using the IP-method, we rewrite this Lagrangian as follows:

$$\mathcal{L}(x, \lambda) = \mathbb{E}\left[\sum_{i=1}^{n} r_i \chi_i x_i\right] - \lambda\left(p + \mathbb{E}[\mathbb{Y}_{\mathbb{R}^+}(c - g(x, \chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}]\right) \tag{4}$$

Let us denote $\widetilde{l}$ the function inside the expectation of the Lagrangian (4), i.e.

$$\widetilde{l}(x, \lambda, \chi) = \sum_{i=1}^{n} r_i \chi_i x_i - \lambda \left(p + \mathbb{Y}_{\mathbb{R}^+}(c - g(x, \chi)) \frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2} \right)$$

It follows

$$\left(\nabla_x \widetilde{l}(x, \lambda, \chi)\right)_h = r_h \chi_h + \lambda \left(\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi)) \frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2} \left(\chi_h + \frac{(c - g(x, \chi))}{x_\kappa} \nu_h^\kappa \right) \right)$$

Remark that the term that multiplies λ is zero whenever the capacity constraint *is not* satisfied. In these cases *all* the components of the current x^k are incremented (as all the components of $(r_1 \chi_1, \ldots, r_n \chi_n)^T$ are positive) although at least one component should be decremented in order to better fit the capacity.

The constraint in expectation can be equivalently reformulated as

$$\mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(g(x, \chi) - c)\right] \leq 1 - p \Leftrightarrow \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(g(x, \chi) - c)\right] - (1 - p) \leq 0$$

In this case the h^{th} component of the gradient of $\widetilde{l}$ is

$$\left(\nabla_x \widetilde{l}(x, \lambda, \chi)\right)_h = r_h \chi_h - \lambda \left(\mathbb{H}_{\mathbb{R}^+}(g(x, \chi) - c) \frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2} \left(\chi_h - \frac{(g(x, \chi) - c)}{x_\kappa} \nu_h^\kappa \right) \right)$$

Here the term that multiplies λ is zero whenever the capacity constraint *is* satisfied. In this case the components of x^k are incremented by the components of the positive vector $(r_1 \chi_1^k, \ldots, r_n \chi_n^k)^T$ (multiplied by the corresponding factor σ^k). When the capacity constraint is not satisfied the term with coefficient λ is subtracted from this vector in order to correct x^k. This term is positive whenever $(\chi_\kappa^k - \mu_\kappa) > 0$ and the Lagrange multiplier λ is thus playing its assigned role of a penalty factor in case the constraint is violated.

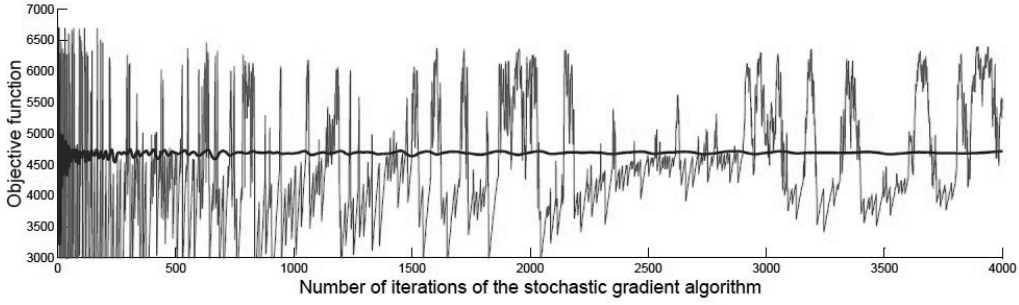

Figure 2 – Initial divergence for the Stochastic Arrow-Hurwicz algorithm solving the continuous $ECKP$: FD-method (bold curve) versus initial IP-method.

A second improvement can be obtained by a specific choice of the component to integrate by parts: The h^{th} component of the gradient of the Lagrangian function obtained by the IP-method after reformulation:

$$\left(\nabla_x \widetilde{l}(x, \lambda, \chi)\right)_h = r_h \chi_h - \lambda \left(\mathbb{H}_{\mathbb{R}^+}(g(x, \chi) - c) \frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2} \left(\chi_h - \frac{(g(x, \chi) - c)}{x_\kappa} \nu_h^\kappa \right) \right)$$

In case of an overload, we expect this gradient to be negative for some indexes h in order to decrease the total weight of the knapsack. However, for $h \neq \kappa$ and $\mathbb{H}_{\mathbb{R}^+}(g(x, \chi) - c) = 1$, the term that multiplies λ is positive if and only if $(\chi_\kappa - \mu_\kappa) > 0$, which is not always the case. When this event does not occur, the gradient is strictly positive and *all* components of x with index different from κ are incremented despite the overload. Due to this observation we propose the following improved choice of the index κ: Instead of just choosing κ in the k^{th} iteration such that $x_\kappa^k = ||x^k||_\infty$ (see Proposition 1), we choose κ as follows:

$$\kappa = \arg \max_{i=1,\ldots,n} \left\{ x_i^k | x_i^k \geq 1/n \quad \text{and} \quad (\chi_i^k - \mu_i) > 0 \right\}$$

However, if $\{x_i^k | x_i^k \geq 1/n \text{ and } (\chi_i^k - \mu_i) > 0\} = \emptyset$, we choose κ as before, i.e. such that $x_\kappa^k = ||x^k||_\infty \geq 1/n$.

With this modification we were able to significantly improve the convergence of SAH-algorithm involving the IP-method (see Fig. 3) and could reduce the maximum number of iterations to 1000 with effective results.

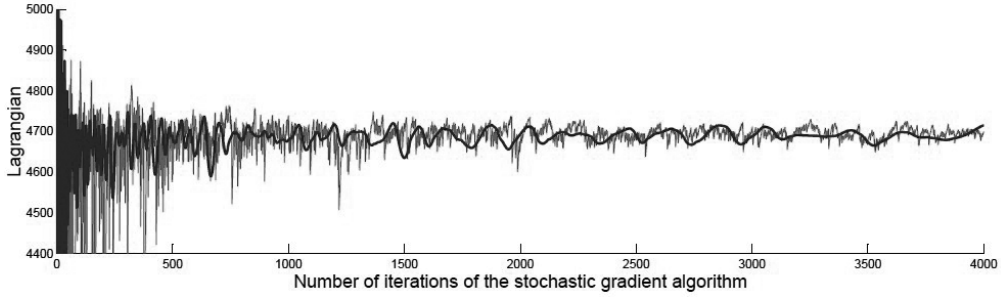

Figure 3 – Results after modifications on IP method: FD-method (gray curve) versus IP-method (bold curve).

5 NUMERICAL RESULTS

All numerical tests have been carried out on an Intel PC with $2GB$ RAM. SAH-algorithm algorithms as well as the branch-and-bound framework have been implemented in C language. The random numbers needed for the computations of the stochastic gradient algorithm were generated in advance using the gsl C-library and stored in a file.

We tested SAH-algorithm on the instance used in Cohn & Barnhart (1998) called hereafter C./B. as well as on 50 randomly generated instances for each dimension. The data given in the tables

2 and 3 are average values over these instances. As noticed at the end of section 2, the choice of normally distributed weights implies that theoretically weights can take negative values. The test instances were generated in such a way that the variance/mean ratio is below $1/4$ (see Cohn & Barnhart, 1998 or Kosuch & Lisser, 2010 for details). This implies a very low probability for the realization of negative weights: Although a high number of scenarios were generated (either 500 or 1000 for each run of SAH-algorithm), we got no negative weight realization.

In Table 2, the numerical results of SAH-algorithm involving FD- or IP-method are compared with those using a *Second Order Cone Programming*-solver (*SOCP*): As mentioned, $ECKP$ can be equivalently reformulated as a chance constrained knapsack problem that, in turn, can be reformulated as a deterministic equivalent $SOCP$-problem (for details see Boyd et al., 1998 or Kosuch & Lisser, 2010). To solve the $SOCP$-problem we used the $SOCP$-solver by MOSEK in C-programming language that applies an interior point method (MOSEK, 2009). The gaps given in tables 2 and 3 are the relative gaps between the MOSEK solution and the approximate solution obtained when using the stochastic gradient algorithm.

Table 2 – Numerical results for the continuous $ECKP$.

	Arrow-Hurwicz & FD-method		Arrow-Hurwicz & IP-method		MOSEK
n	CPU (msec)	Gap	CPU (msec)	Gap	CPU (msec)
C./B.	1	0.03%	2	0.03%	25
15	1	0.02%	2	0.02%	22
20	1	0.02%	3	0.02%	22
30	1	0.02%	4	0.02%	22
50	3	0.01%	6	0.02%	24
75	4	0.02%	9	0.02%	26
100	5	0.01%	12	0.01%	28
150	8	0.01%	17	0.01%	31
250	13	0.01%	28	0.01%	37
500	25	0.01%	56	0.01%	52
1000	51	0.01%	111	0.01%	89

First of all, we remark that the optimal solution values of SAH-algorithm involving FD-method are comparable to those produced by the IP-method. Some fluctuations occur and can be interpreted in terms of choice of implementation, like the choice of the two σ-sequences for SAH-algorithm. Choosing the right parametrization for these sequences has an important influence on the convergence of the algorithm and the best found solution.

In terms of running time, both methods outperform $SOCP$-algorithm for small and medium size instances. For higher dimensional problems, this is still true for FD-method. However, SAH-algorithm involving the IP-method needs approximately twice the time than when using FD-

method. This is of course due to the total number of iterations that we fixed at 500 when using FD-method, while we need 1000 iterations with the IP-method in order to obtain equally good solutions. For higher dimensional instances Mosek interior point method needs therefore less CPU-time than the IP-method.

6 SOLVING THE (COMBINATORIAL) ECKP – NUMERICAL RESULTS

The combinatorial problem has been solved using a branch-and-bound algorithm as described in Cohn & Barnhart (1998) and Kosuch & Lisser (2010). The algorithm has been tested on the instances previously used for the tests of SAH-algorithm.

We stored the random numbers needed for the test runs of SAH in a batch file. As the total number of runs during the branch-and-bound algorithm is unknown and the number of random numbers needed for all those runs is generally very high, we only stored random numbers for a limited number of runs. Before running SAH, we chose randomly one of the instances of random numbers. Remark that, as the runs of SAH are independent, one stored instance of random numbers would theoretically be sufficient.

The results given in Table 3 indicate that the branch-and-bound algorithm that uses $SOCP$-program to obtain upper bounds, considers more nodes than when using SAH-algorithm. This is not due to a better choice of the upper bounds in the latter method as in both algorithms the upper bounds are supposed to be the same (i.e. the optima of the corresponding relaxed problems). However, as the best solution found by the approximate SAH-algorithm might be slightly smaller than the solution value of the relaxed problem, more branches are pruned than with the primal-dual $SOCP$-algorithm. Note that this could theoretically also cause the pruning of a subtree that contains the optimal solution.

Table 3 – Numerical results for the (combinatorial) $ECKP$.

	Arrow-Hurwicz & FD-method				Arrow-Hurwicz & IP-method				SOCP		
n	Optimum	Number of nodes	CPU-time B-and-B	Gap	Optimum	Number of nodes	CPU-time B-and-B	Gap	Optimum	Number of nodes	CPU-time B-and-B
C./B.	4595	122	0.09	0%	4595	122	0.21	0%	4595	122	0.406
15	4840	31	0.02	0%	4840	34	0.06	0%	4840	34	0.082
20	6634	69	0.07	0%	6634	63	0.13	0%	6634	66	0.236
30	10272	271	0.35	0%	10272	436	1.22	0%	10272	350	1.801
50	16975	3341	6.43	0%	16975	7051	31.34	0%	16975	7406	70.914
75	25548	6187	18.43	0%	25548	23911	161.47	0%	25548	62175	1535.520
100	33895	12093	45.87	–	33894	98479	1049.18	–	*	*	*
150	50672	37076	244.11	–	*	*	*	*	*	*	*
250	85189	65890	623.53	–	*	*	*	*	*	*	*
500	*	*	*	*	*	*	*	*	*	*	*

*CPU-time exceeds 1h.

Similarly, SAH-algorithm involving the IP-method considers an average number of nodes greater than when using FD-method. This implies that the solutions of the relaxed subproblems produced by FD-method are not as good as those obtained when using IP-method. As both methods perform equally on the relaxed overall problems (see subsection 5), we conclude that

FD-method is less robust: Instead of choosing particular σ-sequences for each instance or even each subproblem that has to be solved during the branch-and-bound algorithm, we fixed one parametrization for each dimension. However, the subproblems solved during the branch-and-bound algorithm are mostly lower dimensional problems. Moreover, SAH-algorithm using the fixed σ-sequences is less performant on these subproblems. This seems to be especially the case when using FD-method.

If we only allow an average computing time of $1h$, $SOCP$-algorithm can only be used up to a dimension of 50. On the contrary, when using FD-method and allowing 500 iterations in SAH-algorithm, we are able to solve problems up to a dimension of 250 within an average CPU-time of $1h$.

7 CONCLUSION

In this paper, we study and analyse two methods for computing the gradient of the Stochastic Knapsack Problem, where the expectation constraint is considered. More than just considering efficiency of compared methods, it is interesting to analyze what can be concluded when using such methods. In one case, FD-method is very simple, robust, efficient and fast, but presents some disadvantages, such as computing an approximated value of the gradient of the constraint. On the other case, IP-method introduces no approximation in the computation, but is more complex to handle correctly. Nonetheless, this second method opens more opportunities to investigate. First of all, handling an approaching value of the gradient seems not to be a source of disorder here but remains a first order approximation. Secondly, we keep in mind that when using integration by parts, we always have to choose one part to integrate, and one part to derivate in a product, but the way we choose these parts is not fixed. This allows for instance to check conditions of boundaries of $\tilde{\Theta}$. In the future, we will examine ways of choosing the different parts of the Integration by Parts, and also will extend our approach to other stochastic combinatorial problems.

REFERENCES

[1] ANDRIEU L. 2004. Optimisation sous contrainte en probabilité. Ecole Nationale des Ponts et Chaussées.

[2] ANDRIEU L, COHEN G & VÁZQUEZ-ABAD F. 2007. Stochastic programming with probability constraints. `http://fr.arxiv.org/abs/0708.0281` (Accessed 24 October 2008).

[3] BEN-TAL A & NEMIROWSKI A. 2008. Selected topics in robust convex optimization. *Math. Programming*, **112**(1): 125–158.

[4] BHALGAT A, GOEL A & KHANNA S. 2011. Improved approximation results for stochastic knapsack problems. In: *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, page 1647–1665.

[5] BIRGE JR & LOUVEAUX F. 1997. *Introduction to stochastic programming*. Springer-Verlag, New York.

[6] BOYD S, LEBRET H, LOBO MS & VANDENBERGHE L. 1998. Applications of second-order cone programming. *Linear Algebra and its Applications*, **284**: 193–228.

[7] CARPENTIER 2010. Méthodes numériques en optimisation stochastique. Ecole Nationale Supérieure des Techniques Avancées. `http://www.ensta.fr/$\sim$pcarpent/MNOS/` (Accessed march 2010).

[8] CHEN K & ROSS S. 2014. An adaptive stochastic knapsack problem. *European Journal of Operations Research*, **239**(3): 625–635.

[9] COHN A & BARNHART C. 1998. The stochastic knapsack problem with random weights: A heuristic approach to robust transportation planning. In *Proceedings of the Triennial Symposium on Transportation Analysis (TRISTAN III)*.

[10] CULIOLI JC & COHEN G. 1995. Optimisation stochastique sous contraintes en espérance. *Comptes rendus de l'Académie des sciences, Paris, Série I*, **320**(6): 753–758.

[11] GOAL A & INDYK P. 1999. Stochastic load balancing and related problems. In: *Proceedings of Foundations of Computer Science*, pages 579–586.

[12] KELLERER H, PFERSCHY U & PISINGER D. 2004. *Knapsack problems*. Springer, Berlin.

[13] KOSUCH S & LISSER A. 2010. Upper bounds for the 0-1 stochastic knapsack problem and a b&b algorithm. *Annals of Operations Research*, **176**(1): 77–93.

[14] MOSEK. 1998-2009. *The MOSEK optimization tools manual. Version 6.0 (Revision 53).* `http://www.mosek.com/fileadmin/products/6\_0/tools/doc/pdf/tools.pdf`.

[15] PREKOPA A. 1995. *Stochastic Programming*. Kluwer Academic Publishers (Dordrecht, Boston).

[16] TÖNISSEN D, BOUMAN P, VEN DER AKKER J & HOOGEVEEN J. 2016. Decomposition approaches for recoverable robust optimization problems. *European Journal of Operations Research*, **251**(3): 739–750.

[17] TÖNISSEN D, VEN DER AKKER J & HOOGEVEEN J. 2017. column generation strategies and decomposition appraoches for the two-stage stochastic knapsack problem. *Computers and Operations Research*, **83**: 125–139.