On Two-Stage Stochastic Knapsack Problems with or without Probability Constraint

Stefanie Kosuch Abdel Lisser

January 21, 2010

Abstract

In this paper we study a particular version of the stochastic knapsack problem with normally distributed weights: the two-stage stochastic knapsack problem. In contrary to the single-stage knapsack problem, items can be added to or removed from the knapsack at the moment the actual weights come to be known (second stage). In addition, a probability constraint is introduced in the first stage in order to restrict the percentage of cases where the items chosen lead to an overload in the second stage. To the best of our knowledge, there is no method known to exactly evaluate the objective function for a given first-stage solution. Therefore, we propose methods to calculate upper and lower bounds. These bounds are used in a branch-and-bound framework in order to search the first-stage solution space. Special interest is given to the case where the items have similar weight means. Numerical results are presented and analyzed.

1 Introduction

The knapsack problem is a widely studied combinatorial optimization problem. Special interest arises from numerous real life applications for example in logistics and scheduling. The basic problem consists in choosing a subset out of a given set of items such that the total weight (or size) of the subset does not exceed a given limit (the capacity of the knapsack) and the total benefit of the subset is maximized (for a survey on (deterministic) knapsack problems see the book by Kellerer et al. [9]).
However, most real life problems are non-deterministic in the sense that some of the parameters are not (exactly) known at the moment when the decision has to be made. For example, the exact capacity available can be unknown due to delays in previous jobs or the rewards of the items might depend on market fluctuations ([3],[8],[17]). In this paper we study the case where the item weights are random. This case entails the problem that we cannot be sure if the items chosen in advance (i.e. before the revealing of the actual weights) meet the knapsack capacity constraint. Depending on the situation given, the resulting stochastic optimization problem can be handled in two different ways: Either as a single-stage problem which means that the final decision has to be made before the random parameters are revealed, or as a two- or multi-stage problem that allows later corrections of the decision made in advance.

If for each of the random variables only a finite number of scenarios have non-zero probability, one can choose the items in a way that the capacity constraint is respected in any of the finitely many possible scenarios. In this single-stage model the stochastic problem is reformulated as a deterministic equivalent problem by introducing a corresponding constraint for every possible scenario. However, this approach is often intractable as the number of constraints grows with the number of scenarios. Another drawback is caused by the fact that we force the chosen items to respect the capacity constraint in any case which might lead to a decision that is far from the optimum in most of the cases.
If an overload is acceptable in some few cases, one can replace the knapsack constraint by a probability constraint that restricts the percentage of cases where the chosen items lead to an overload. The resulting single-stage problem is known as *chance constrained knapsack problem* ([7],[10],[13],[14]).
However, in most applications an overload can or even has to be corrected later or leads at least to a loss of benefit or to additional costs, respectively ([1],[4],[5],[12],[14]). In these cases, a corrective decision has to be made at the moment all or part of the random variables come to be known. If all random variables are revealed at the same instant, we can correct our decision once for all, i.e. we are in the situation of a *two-stage stochastic knapsack problem*. If, in contrary, information is revealed in more than one stage and if we allow corrections to be made on partial information, we are in the case of a *multi-stage* or *on-line stochastic knapsack problem* ([6],[11],[16],[18]).
Stochastic optimization problems with a single stage have the advantage that, given the probability distribution of the random variables, they can generally be reformulated as a deterministic equivalent problem. This reformulation does not always imply the possibility to solve the problem efficiently using a deterministic method, but at least we are able to compute the objective function value to a given decision [14]. A deterministic equivalent reformulation is (generally) also possible for two-stage problems where the second stage only consists in paying a penalty in case of an overload ([5],[1],[14]) or when only finitely many scenarios are considered ([15]). If, in contrary, there are infinitely many scenarios with non-zero probability and if the second-stage decision consists in correcting the first-stage decision by adding and/or removing items, the exact evaluation of the objective function becomes very difficult, or even impossible.
In this paper we study the last case: We assume the item weights to be continuously distributed following a normal distribution. In the second stage, items have to be removed in case of an overload and can be added in case of an underload. Exchanging items in the second stage in order to increase the total benefit is not allowed. We further introduce a probability constraint in the first stage, but all methods presented in this paper are also applicable if the first-stage problem contains no such constraint.
To give an example of an application of our model in logistic optimization consider the following:
Given a logistic company with a certain transport capacity. Its clients are not able to provide exact measurements of the goods they demand to transport, as these might vary around an average value. However, the company has to decide in advance which items to accept. Typically, rejecting items later due to lack of capacity leads to additional costs (e.g. to purchase further capacity) and, probably more important, might give the client reason not to hire the company again. The company is thus obliged to restrict the risk of an overload. In case the capacity is not reached by the total weight of the items chosen in advance, items can be

added in the second stage. However, choosing items at short notice might entail additional costs and therefore lower the reward of the item. For the same reason that the risk of an overload should be restricted, the company might want to avoid removing an item in the second stage in favor of a more cost-effective item.
The first aim of this paper is to propose a method to compute upper bounds on the optimal overall solution value of the studied problem. As there is no method known to exactly evaluate the objective function for a given first-stage solution, our second aim is to give lower bounds that replace this exact evaluation. More precisely, given a first-stage solution, we propose methods to compute a lower bound on the corresponding objective function value. By replacing the exact evaluation of the objective function by the computation of this lower bound, one could use any appropriate framework to search the first-stage solution space for good (or best possible) overall lower bounds.
This paper is organized as follows: in section 2 the studied problem is presented and discussed. Sections 3 and 4 deal with the computation of upper and lower bounds on the overall solution, respectively. Particular lower bounds are given for the case of similar items (subsection 4.4). In section 5 we present a branch-and-bound framework that searches for the best possible overall lower bound that we can obtain by using the proposed bounds. Finally, numerical results are given and discussed in section 6.

2 Mathematical formulation and an application

We consider a stochastic knapsack problem of the following form: Given a knapsack with fixed weight capacity $c > 0$ as well as a set of n items. Each item has a weight that is not known in the first stage but comes to be known before the second-stage decision has to be made. 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 justified by the fact that the value of an item often depends on the item weight which we do not know in advance.
In the first stage, items can be placed in the knapsack. However, if the outcome of the total weight of these items should exceed the capacity, some of these first-stage items have to be removed in the second stage. Therefore, we restrict the percentage of cases where the first-stage items lead to an overload by introducing a probability constraint in the first stage.
At the beginning of the second stage the weights of all items are revealed. In case of an overload, items have to be removed and a penalty d has to be paid for each unit of weight that is unpacked. Our objective is thus to minimize the total penalty. If we decide to remove item i, the corresponding second-stage decision variable y_i^- is set to 1. In this case, all components of y^+ can automatically be set to 0. If the capacity constraint is respected, items can be added if and only if this does not lead to an overload (second-stage items). If we decide to add item i, we set $y_i^+ = 1$. In this second case, all components of y^- can be set to 0. It is easy to see that this model only makes sense if the reward per weight unit $\bar{r}_i$ of an item that is added in the second stage is strictly smaller than r_i. The aim is to maximize the expected total reward of the knapsack:

The Two-Stage Stochastic Knapsack Problem with Probability Constraint ($TSKP$)

$$
\begin{aligned}
(TSKP) \quad \max_{x\in\{0,1\}^n} \quad & \mathbb{E}\left[\sum_{i=1}^{n} r_i\chi_i x_i\right] + \mathbb{E}\left[\mathcal{Q}(x,\chi)\right] \\
\text{s.t.} \quad & \mathbb{P}\{\sum_{i=1}^{n}\chi_i x_i \le c\} \ge p && \text{(1a)}\\
& \mathcal{Q}(x,\chi) = \max_{y^+,y^-\in\{0,1\}^n} \sum_{i=1}^{n}\overline{r}_i\chi_i y_i^+ - d\sum_{i=1}^{n}\chi_i y_i^-, && \text{(1b)}\\
& \text{s.t. } y_k^+ \le 1 - x_k,\ k=1,\dots,n. && \text{(1c)}\\
& \quad y_k^- \le x_k,\ k=1,\dots,n. && \text{(1d)}\\
& \quad y_k^+ \le \mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n}\chi_i x_i)\cdot y_k^+,\ k=1,\dots,n. && \text{(1e)}\\
& \quad y_k^- \le \mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n}\chi_i x_i - c)\cdot y_k^-,\ k=1,\dots,n. && \text{(1f)}\\
& \quad \sum_{i=1}^{n}(x_i + y_i^+ - y_i^-)\chi_i \le c, && \text{(1g)}
\end{aligned}
$$

where $\mathbb{P}\{A\}$ denotes the probability of an event A, $\mathbb{E}[\cdot]$ the expectation, $\mathbb{1}_{\mathbb{R}^+}$ denotes the indicator function of the positive real interval, $\overline{r}_i < r_i$ for all $i = 1,\dots,n$, $d \ge \max_{i\in\{1,\dots,n\}} r_i$ and $p \in (0.5, 1]$ is the prescribed probability.
Constraints (1e) and (1f) assure that items can only be added in the second stage in case the first-stage items do not lead to an overload, and they can only be removed in case of an overload. Therefore, the above setting does not allow an exchange of items in the second stage. These $2n$ non-linear constraints could be replaced by the $n^2 - n$ following linear constraints:

$$y_i^+ \le (1 - y_j^-) \quad \forall\, i,j \in \{1,\dots,n\}, i \ne j$$

However, we chose the non-linear problem formulation (1) in order to be able to rewrite the second stage problem as a linear combination of two independent problems, one treating the second stage decision variables y_i^+, the other the decision variables y_i^- (see subsection 2.1, Property 3).
Throughout, we assume that the optimal first-stage solution is non-zero, i.e. that at least one item is put in the knapsack in the first stage.

2.1 Properties of the $TSKP$

The following properties are mainly used in section 4 to calculate lower bounds on the objective function value for a given first-stage solution.

Property 1: The two-stage problem (1) is a *relatively complete recourse* problem, i.e. for every first-stage decision there exists a feasible second-stage decision.

Property 2: Given a first-stage decision and a realization of χ, solving the second-stage problem means solving a deterministic knapsack problem:
In case of an underload, the capacity of the "second-stage knapsack" equals the amount of capacity that is still unused after the first stage. The set of items that one can choose from contains all items that have not been added in the first stage.
In case of an overload, the second-stage knapsack is the initial knapsack, but in contrary to the first stage we can only choose from those items that had been previously added.
It is thus clear how to solve the second-stage problem once the first-stage decision is fixed and the actual item weights are revealed. However, for instance no method to compute the expected second-stage solution value for a given first-stage decision is known, mainly due to the assumption that the item weights follow continuous distributions. This implies, that for a given first-stage solution, we are not able to compute the corresponding objective function value, either. Instead, one can compute a lower bound (see section 4), for example in order to get an idea of the quality of the chosen decision and/or to replace the exact evaluation of the objective function value in an appropriate framework in order to search for good overall lower bounds (see section 5).

Property 3: The random variable $\mathcal{Q}(x,\chi)$ can be written as the linear combination of two random variables $\mathcal{Q}_1(x,\chi)$ and $\mathcal{Q}_2(x,\chi)$:

$$\mathcal{Q}_1(x,\chi) = \max_{y^+ \in \{0,1\}^n} \sum_{i=1}^{n} \overline{r}_i y_i^+ \chi_i \tag{2a}$$

$$\text{s.t.} \quad y_k^+ \leq 1 - x_k, \ k = 1, \ldots, n. \tag{2b}$$

$$y_k^+ \leq \mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} x_i \chi_i) \cdot y_k^+, \ k = 1, \ldots, n. \tag{2c}$$

$$\mathbb{1}_{\mathbb{R}^+}(c - \sum x_i \chi_i) \sum_{i=1}^{n} (x_i + y_i^+) \chi_i \leq c. \tag{2d}$$

and

$$\mathcal{Q}_2(x,\chi) = \min_{y^- \in \{0,1\}^n} \sum_{i=1}^{n} y_i^- \chi_i \tag{3a}$$

$$\text{s.t.} \quad y_k^- \leq x_k, \ k = 1, \ldots, n. \tag{3b}$$

$$y_k^- \leq \mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n} x_i \chi_i - c) \cdot y_k^-, \ k = 1, \ldots, n. \tag{3c}$$

$$\mathbb{1}_{\mathbb{R}^+}(\sum x_i \chi_i - c) \sum_{i=1}^{n} (x_i - y_i^-) \chi_i \leq c. \tag{3d}$$

It follows $\mathbb{E}[\mathcal{Q}(x,\chi)] = \mathbb{E}[\mathcal{Q}_1(x,\chi)] - d \cdot \mathbb{E}[\mathcal{Q}_2(x,\chi)]$.

Property 4: Let $\tilde{x}$ be a given feasible first-stage solution. If $\tilde{x}$ leads to an underload, i.e. if for the realization $\hat{\chi}$ of χ we have $\sum_{i=1}^{n} \hat{\chi}_i \tilde{x}_i > c$, it follows $\mathcal{Q}_2(\tilde{x},\hat{\chi}) = 0$. On the contrary, in the case of an overload, we have $\mathcal{Q}_1(\tilde{x},\hat{\chi}) = 0$.

3 Computing upper bounds on the optimal solution of the TSKP

Given a feasible first-stage solution vector $\tilde{x}$, the expectation of the second-stage solution can be bounded (from above) by

$$\mathbb{E}[\mathcal{Q}(\tilde{x},\chi)] \leq \overline{r}_{max} \cdot \mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i\chi_i]^+\right] - d \cdot \mathbb{E}\left[[\sum_{i=1}^{n} \tilde{x}_i\chi_i - c]^+\right]$$

where $[x]^+ := \max(0,x) = x \cdot \mathbb{1}_{\mathbb{R}^+}(x)$ $(x \in \mathbb{R})$ and $\overline{r}_{max} = \max\{\overline{r}_i | i \in \{1,\ldots,n\}\}$. The right hand side of the inequality is in fact the expectation of the optimal second-stage solution value in case of continuous second-stage variables and a uniform second-stage reward $\overline{r}_{max}$. More precisely, if y_i^+ can take any value between 0 and 1, we could fill the knapsack up to the capacity in case of an underload and the expectation of $\mathcal{Q}_1(\tilde{x},\chi)$ would be at most $\overline{r}_{max} \cdot \mathbb{E}\,[[c - \sum_{i=1}^{n} \tilde{x}_i\chi_i]^+]$. And if $y_i^- \in [0,1]$, we would have to pay exactly d times the overload as additional costs in case the capacity is not respected, i.e. $\mathbb{E}[\mathcal{Q}_2(\tilde{x},\chi)] = d \cdot \mathbb{E}\,[[\sum_{i=1}^{n} \tilde{x}_i\chi_i - c]^+]$. An upper bound on the optimal solution of $TSKP$ (1) is thus given by the optimal solution of the following simple recourse problem:

$$(SRKP) \quad \max_{x \in \{0,1\}^n \setminus \{0\}} \quad \mathbb{E}\left[\sum_{i=1}^{n} r_i x_i \chi_i\right] + \overline{r}_{max} \cdot \mathbb{E}\left[[c - \sum_{i=1}^{n} x_i\chi_i]^+\right] - d \cdot \mathbb{E}\left[[\sum_{i=1}^{n} x_i\chi_i - c]^+\right]$$

$$\text{s.t.} \quad \mathbb{P}\{\sum_{i=1}^{n} x_i\chi_i \leq c\} \geq p \tag{4a}$$

We solve the $SRKP$ approximately by the method proposed by Kosuch and Lisser in [14]: Their basic idea is to solve the corresponding Lagrangian relaxation using the Stochastic Arrow-Hurwicz algorithm. It is also shown how to approximate the gradient of the function $\mathbb{E}\,[[\sum_{i=1}^{n} x_i\chi_i - c]^+]$ using Approximation by Convolution.
In order to handle the probability constraint using a Lagrange multiplier, let us rewrite the constraint as follows:

$$p - \mathbb{P}\{\sum_{i=1}^{n} x_i\chi_i \leq c\} \leq 0$$

Unfortunately, the left hand side of this inequality is generally not convex. However, this can be achieved by further reformulating the constraint (see e.g. [2]):

$$p - \mathbb{P}\{\sum_{i=1}^{n} x_i\chi_i \leq c\} \leq 0 \iff \sum_{i=1}^{n} x_i\chi_i + \Phi^{-1}(p)\|\Sigma^{1/2}x\| - c \leq 0 \tag{5}$$

Here Φ denotes the cumulative distribution function of the standard normal distribution and Σ the diagonal covariance matrix of the random vector χ. As $p > 0.5$ and therefore $\Phi^{-1}(p) > 0$, the obtained inequality is a Second Order Cone constraint with a convex left hand side function whose gradient on $\{0,1\}^n \setminus \{0\}$ can easily be computed. We thus get a well defined Lagrangian problem which can be solved using a stochastic gradient algorithm as described in [14].

4 Computing lower bounds on the optimal solution of the TSKP

In this section we propose lower bounds on the overall solution value for a given fixed first-stage decision $\tilde{x}$. More precisely, we bound $\mathbb{E}[Q_1(\tilde{x},\chi)]$ (from below) as well as $\mathbb{E}[Q_2(\tilde{x},\chi)]$ (from above) in order to get a lower bound on the expected second-stage solution $\mathbb{E}[Q(\tilde{x},\chi)]$. Let in the following S denote the index set of those items that have been chosen in the first stage and $\overline{S}$ its complement (i.e. $S = \{i \in \{1,\dots n\} | \tilde{x}_i = 1\}$ and $\overline{S} = \{i \in \{1,\dots n\} | \tilde{x}_i = 0\}$) and define $\overline{r}_{min} = \min\{\overline{r}_i | i \in \{1,\dots,n\}\}$.

4.1 Lower bound on the expectation of $\mathcal{Q}_1$

We first show the following:

Proposition 4.1.

$$\mathbb{E}[\mathcal{Q}_1(\tilde{x},\chi)] > \overline{r}_{min} \cdot \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i\chi_i]^+ \right] - \widetilde{p} \cdot \mathbb{E}\left[\max_{i \in \overline{S}} \chi_i \right] \right)$$

where $\widetilde{p} = \mathbb{P}\{\sum_{i=1}^{n} \tilde{x}_i\chi_i \leq c\}$ *denotes the probability of an underload.*

Proof. Let us rewrite the optimal solution of the (partial) second-stage problem $\mathcal{Q}_1(\tilde{x},\hat{\chi})$ for a realization $\hat{\chi}$ of χ as

$$\mathcal{Q}_1(\tilde{x},\hat{\chi}) = \sum_{i=1}^{n} \overline{r}_i (y^+)_i^* \hat{\chi}_i \geq \overline{r}_{min} \cdot \sum_{i=1}^{n} (y^+)_i^* \hat{\chi}_i \tag{6}$$

$$= \overline{r}_{min} \cdot \left((c - \sum_{i=1}^{n} \tilde{x}_i\hat{\chi}_i) + \left(\sum_{i=1}^{n} (y^+)_i^* \hat{\chi}_i - (c - \sum_{i=1}^{n} \tilde{x}_i\hat{\chi}_i) \right) \right) \tag{7}$$

where $(y^+)^* = (y^+)^*(\tilde{x},\hat{\chi})$ is the (unknown) optimal second-stage solution vector that depends on $\tilde{x}$ as well as $\hat{\chi}$.
In case of an underload, $(c - \sum_{i=1}^{n} \tilde{x}_i\hat{\chi}_i) - \sum_{i=1}^{n} (y^+)_i^* \hat{\chi}_i = c - \sum_{i=1}^{n} (\tilde{x}_i + (y^+)_i^*)\hat{\chi}_i$ is the

difference between the capacity and the final weight of the knapsack. Note that this quantity figures negatively in (7). This amount can be bounded independently of the second-stage solution $(y^+)^*$, as in the worst case we have to keep a weight of $\max_{i\in\overline{S}} \hat{\chi}_i - \epsilon$ of the knapsack unused (where $\epsilon > 0$):

$$(c - \sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i) - \sum_{i=1}^{n} (y^+)_i^* \hat{\chi}_i < \max_{i\in\overline{S}} \hat{\chi}_i \tag{8}$$

It follows

$$\mathcal{Q}_1(\tilde{x}, \hat{\chi}) > \overline{r}_{min} \left((c - \sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i) - \max_{i\in\overline{S}} \hat{\chi}_i \right) \tag{9}$$

Whenever we are in the case on an overload, $\mathcal{Q}_1$ is zero. We thus get:

$$\begin{aligned}
\mathbb{E}[\mathcal{Q}_1(\tilde{x}, \chi)] &> \overline{r}_{min} \left(\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \left((c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) - \max_{i\in\overline{S}} \chi_i \right) \right] \right) \\
&= \overline{r}_{min} \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+ \right] - \mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \cdot \max_{i\in\overline{S}} \chi_i \right] \right)
\end{aligned}$$

In the case of independently normally distributed items, $\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+\right]$ has a deterministic equivalent formulation that is a linear combination of the density and cumulative distribution function of the standard normal distribution. It can therefore be computed exactly (see e.g. [5] or [14]). So let us see how to further bound $\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \cdot \max_{i\in\overline{S}} \chi_i\right]$:

Claim 4.2.

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \cdot \max_{i\in\overline{S}} \chi_i \right] \leq \widetilde{p} \cdot \mathbb{E}[\max_{i\in\overline{S}} \chi_i]$$

Proof of the claim: It is easy to see that

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \cdot \max_{i\in\overline{S}} \chi_i \right] + \mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n} \tilde{x}_i \chi_i - c) \cdot \max_{i\in\overline{S}} \chi_i \right] = \mathbb{E}\left[\max_{i\in\overline{S}} \chi_i \right]$$

so either

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \cdot \max_{i\in\overline{S}} \chi_i \right] \leq \widetilde{p} \cdot \mathbb{E}[\max_{i\in\overline{S}} \chi_i]$$

or

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n} \tilde{x}_i \chi_i - c) \cdot \max_{i\in\overline{S}} \chi_i \right] \leq (1 - \widetilde{p}) \cdot \mathbb{E}[\max_{i\in\overline{S}} \chi_i]$$

and if one of the two inequalities is satisfied with equality, the second would be satisfied with equality, too. We further have

$$\mathbb{E}[\max_{i\in\overline{S}}\chi_i|\sum_{i=1}^{n}\tilde{x}_i\chi_i\leq c]\leq\mathbb{E}[\max_{i\in\overline{S}}\chi_i|\sum_{i=1}^{n}\tilde{x}_i\chi_i\geq c]$$

which excludes the case that

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n}\tilde{x}_i\chi_i-c)\cdot\max_{i\in\overline{S}}\chi_i\right]<(1-\widetilde{p})\cdot\mathbb{E}[\max_{i\in\overline{S}}\chi_i]\quad\text{ and}$$

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c-\sum_{i=1}^{n}\tilde{x}_i\chi_i)\cdot\max_{i\in\overline{S}}\chi_i\right]>\widetilde{p}\cdot\mathbb{E}[\max_{i\in\overline{S}}\chi_i]$$

Thus,

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c-\sum_{i=1}^{n}\tilde{x}_i\chi_i)\cdot\max_{i\in\overline{S}}\chi_i\right]\leq\widetilde{p}\cdot\mathbb{E}[\max_{i\in\overline{S}}\chi_i]$$

q.e.d.

This concludes the proof. □

Unfortunately, (even) in the case of independently normally distributed random variables, $\mathbb{E}[\max_{i\in\overline{S}}\chi_i]$ is not easily computable. We thus bound it by discretization of the probability space of the random variable $\max_{i\in\overline{S}}\chi_i$:
Let us define a new random variable $\chi_{max}^{\overline{S}}:=\max_{i\in\overline{S}}\chi_i$. Let Φ_i be the cumulative distribution function of χ_i, and $\Phi_{max}^{\overline{S}}$ the cumulative distribution function of $\chi_{max}^{\overline{S}}$. Then one can easily show that $\Phi_{max}^{\overline{S}}=\prod_{i\in\overline{S}}\Phi_i$.
Furthermore, if there exists a $\beta<\infty$ such that $\mathbb{P}\{\chi_{max}^{\overline{S}}\in(-\infty,\beta]\}=1$, we can bound $\mathbb{E}\left[\chi_{max}^{\overline{S}}\right]$ by splitting the interval $(-\infty,\beta]$ in K disjoint intervals (K scenarios) $(\alpha_k,\beta_k]$, $k=1,\ldots,K$, and it follows

$$\mathbb{E}\left[\chi_{max}^{\overline{S}}\right]\leq\sum_{k=1}^{K}\beta_k\mathbb{P}\{\max_{i\in\overline{S}}\chi_i\in(\alpha_k,\beta_k]\}=\sum_{k=1}^{K}\beta_k(\Phi_{max}^{\overline{S}}(\beta_k)-\Phi_{max}^{\overline{S}}(\alpha_k))$$

Unfortunately, in the case of normally distributed weights, (theoretically) no such β exists. However, as for every ϵ there exists a β such that $\mathbb{P}\{\exists i\in\overline{S}|\chi_i\geq\beta\}\leq\epsilon$, we can approximate the upper bound on $\mathbb{E}\left[\chi_{max}^{\overline{S}}\right]$ by defining β in such a way that $\mathbb{P}\{\exists i\in\overline{S}|\chi_i\geq\beta\}$ is (sufficiently) small. From a numerical point of view, ϵ can even be chosen to be zero as we generally deal with distributions that have small variances (compared to the mean) and whose density functions go to zero very fast. So we finally get:

Corollary 4.3. *Let α_k, β_k $(k = 1, \ldots, K)$ be defined as above. Then*

$$\mathbb{E}[\mathcal{Q}_1(\tilde{x}, \chi)] > \overline{r}_{min} \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+ \right] - \widetilde{p} \cdot \sum_{k=1}^{K} \beta_k (\Phi^{\overline{S}}_{max}(\beta_k) - \Phi^{\overline{S}}_{max}(\alpha_k)) \right)$$

where $\widetilde{p} = \mathbb{P}\{\sum_{i=1}^{n} \tilde{x}_i \chi_i \leq c\}$ denotes the probability of an underload.

□

Remark: As the item weights have been assumed to be independently normally distributed, $X := \sum_{i=1}^{n} \tilde{x}_i \chi_i$ is normally distributed with mean $\sum_{i=1}^{n} \tilde{x}_i \mu_i$ and standard deviation $\sqrt{\sum_{i=1}^{n} \tilde{x}_i^2 \sigma_i^2}$. We are thus able to compute $\widetilde{p} = \mathbb{P}\{X \leq c\}$ as well as $\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+\right] = \mathbb{E}\left[[c - X]^+\right]$.

4.2 Upper bound on the expectation of $\mathcal{Q}_2$

Similar to the approach presented in the previous subsection, we first bound $\mathbb{E}[Q_2(\tilde{x}, \chi)]$ depending on the expected maximum first-stage item weight $\mathbb{E}[\max_{i \in S} \chi_i]$. However, in the case of an overload no claim such as claim (4.2) can be proved.

Proposition 4.4.

$$\mathbb{E}[\mathcal{Q}_2(\tilde{x}, \chi)] < \mathbb{E}\left[[\sum_{i=1}^{n} \tilde{x}_i \chi_i - c]^+ \right] + \mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n} \tilde{x}_i \chi_i - c) \cdot \max_{i \in S} \chi_i \right]$$

Proof. The proof is similar to the proof of proposition 4.1: Let $(y^-)^* = (y^-)^*(\tilde{x}, \hat{\chi})$ be the corresponding optimal second-stage solution. It follows

$$\begin{aligned} \mathcal{Q}_2(\tilde{x}, \hat{\chi}) &= \sum_{i=1}^{n} \hat{\chi}_i (y^-)^*_i \\ &= (\sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i - c) + \left(\sum_{i=1}^{n} (y^-)^*_i \hat{\chi}_i - (\sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i - c) \right) \end{aligned}$$

$\sum_{i=1}^{n} (y^-)^*_i \hat{\chi}_i - (\sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i - c)$ is the amount of weight that we remove from the knapsack in addition to the overweight. This amount can be once more bounded independently of the second-stage solution $(y^-)^*$, as in the worst case the knapsack weight might fall $\max_{i \in S} \hat{\chi}_i - \epsilon$ under the capacity due to the removal of items in the second stage (where $\epsilon > 0$). Thus

$$\sum_{i=1}^{n} (y^-)^*_i \hat{\chi}_i - (\sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i - c) < \max_{i \in S} \hat{\chi}_i \tag{10}$$

and it follows

$$\mathcal{Q}_2(\tilde{x}, \hat{\chi}) < (\sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i - c) + \max_{i \in S} \hat{\chi}_i \tag{11}$$

Whenever we are in the case on an underload, $\mathcal{Q}_2$ is zero:

$$\mathbb{E}[\mathcal{Q}_2(\tilde{x},\chi)] < \mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n}\tilde{x}_i\chi_i - c)\left((\sum_{i=1}^{n}\tilde{x}_i\chi_i - c) + \max_{i\in S}\chi_i\right)\right]$$

□

We finally bound $\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n}\tilde{x}_i\chi_i - c)\cdot\max_{i\in S}\chi_i\right]$ by discretization in the same way as $\mathbb{E}\left[\max_{i\in\overline{S}}\chi_i\right]$ has been bounded in subsection 4.1: Under the condition that $\sum_{i\in S}\chi_i \geq c$, we have $\max_{i\in S}\chi_i \geq c/|S|$. So let α_k^c, β_k^c $(k = 1,\ldots,K^c)$ such that $\cup_{k=1}^{K^c}(\alpha_k^c,\beta_k^c] = (\frac{c}{|S|},\beta]$ (if $\frac{c}{|S|} \geq \beta$ we define $K^c = 0$). It follows

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n}\tilde{x}_i\chi_i - c)\cdot\max_{i\in S}\chi_i\right] \leq \sum_{k=1}^{K^c}\beta_k^c(\Phi_{max}^S(\beta_k^c) - \Phi_{max}^S(\alpha_k^c))$$

which leads to the following corollary:

Corollary 4.5. *Let α_k^c, β_k^c $(k = 1,\ldots,K^c)$ be defined as above. Then*

$$\mathbb{E}[\mathcal{Q}_2(\tilde{x},\chi)] < \mathbb{E}\left[[\sum_{i=1}^{n}\tilde{x}_i\chi_i - c]^+\right] + \sum_{k=1}^{K^c}\beta_k^c(\Phi_{max}^S(\beta_k^c) - \Phi_{max}^S(\alpha_k^c))$$

□

4.3 Lower bound on the overall solution

Combining the results from the previous subsections (corollaries 4.3 and 4.5) and as

$$\mathbb{E}\left[[c - \sum_{i=1}^{n}x_i\chi_i]^+\right] = \mathbb{E}\left[[\sum_{i=1}^{n}x_i\chi_i - c]^+\right] - \mathbb{E}\left[\sum_{i=1}^{n}x_i\chi_i - c\right]$$

we get the following lower bound on the overall solution for a given feasible first-stage vector $\tilde{x}$:

Proposition 4.6. *Let α_k,β_k, $(k = 1,\ldots,K)$, α_k^c,β_k^c $(k = 1,\ldots,K^c)$ be defined as in subsections 4.1 and 4.2. Let $\tilde{x}$ be a feasible first-stage solution. Then the following lower bound on the corresponding overall solution holds:*

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i=1}^{n}r_i\chi_i\tilde{x}_i\right] + \mathbb{E}\left[\mathcal{Q}(\tilde{x},\chi)\right] > \\
&\sum_{i=1}^{n}(r_i - \overline{r}_{min})\mu_i\tilde{x}_i + (\overline{r}_{min} - d)\cdot\mathbb{E}\left[[\sum_{i=1}^{n}\tilde{x}_i\chi_i - c]^+\right] \\
&+\overline{r}_{min}\cdot\left(c - \widetilde{p}\cdot\sum_{k=1}^{K}\beta_k(\Phi_{max}^{\overline{S}}(\beta_k) - \Phi_{max}^{\overline{S}}(\alpha_k))\right) - d\cdot\sum_{k=1}^{K^c}\beta_k^c(\Phi_{max}^S(\beta_k^c) - \Phi_{max}^S(\alpha_k^c))
\end{aligned}$$

where $\widetilde{p} = \mathbb{P}\{\sum_{i=1}^{n}\tilde{x}_i\chi_i \leq c\}$ denotes the probability of an underload.

□

4.4 Lower bound on the expectation of $\mathcal{Q}$ in the case of similar items

In order to tighten the lower bounds, we make the assumption that the item weights are similar in the following sense:

The probability of an item to have twice the size of another item is zero. (∗)

In the following, we refer to this assumption as assumption (∗).
In the case of normally distributed items assumption (∗) is theoretically never true. However, in many practical problems the probability that the weight of an item varies by 100% is practically zero. Hence, whenever we treat items whose means are similar, assumption (∗) is practically true.

Lower bound on the expectation of $\mathcal{Q}_1$
In the previous sections we bounded the optimal second-stage solution value depending on the expectation of the maximum item weight among all the items that had not been added in the first stage. In the case of similar items, this lower bound can be improved due to the following lemma:

Lemma 4.7. *Let $\overline{r}_i = \overline{r}_j$ for all $i, j \in \{1, \dots, n\}$. Then, under assumption (∗), it follows*

$$(c - \sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i) - \sum_{i=1}^{n} (y^+)_i^* \hat{\chi}_i < \min_{i \in \overline{S}} \hat{\chi}_i$$

Proof. If the first-stage items lead to an overload, the left hand side of the inequality is negative and the lemma is true for positive item weights. So let us suppose that we are in the case of an underload. Let T be the set of indexes such that $i \in T$ if and only if $\tilde{x}_i = (y^+)_i^* = 0$.
It is clear that $(c - \sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i) - \sum_{i=1}^{n} (y^+)_i^* \hat{\chi}_i < \min_{i \in T} \hat{\chi}_i$ as otherwise we could have added another item in the second stage. It follows that if (at least) one of the items having minimum weight of all the items that have not been added in the first stage has not been added in the second stage, either, we are done, as in this case $\min_{i \in T} \hat{\chi}_i = \min_{i \in \overline{S}} \hat{\chi}_i$. So let $\min_{i \in \overline{S}} \hat{\chi}_i < \min_{i \in T} \hat{\chi}_i$ and define $j := \arg\min_{i \in \overline{S}} \hat{\chi}_i$ as well as $j' := \arg\min_{i \in T} \hat{\chi}_i$. It follows that $(c - \sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i) - \sum_{i=1}^{n} (y^+)_i^* \hat{\chi}_i + \hat{\chi}_j < \hat{\chi}_{j'}$ as otherwise we could have added item j' instead of item j which would have increased the optimal objective function (here we use the assumption that $\overline{r}_i = \overline{r}_j$ for all $i, j \in \{1, \dots, n\}$). We consequently get

$$(c - \sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i) - \sum_{i=1}^{n} (y^+)_i^* \hat{\chi}_i < \hat{\chi}_{j'} - \min_{i \in \overline{S}} \hat{\chi}_i \leq 2 \min_{i \in \overline{S}} \hat{\chi}_i - \min_{i \in \overline{S}} \hat{\chi}_i = \min_{i \in \overline{S}} \hat{\chi}_i$$

□

Using this lemma we can prove the following lower bound on $\mathbb{E}[\mathcal{Q}_1(\tilde{x}, \chi)]$ in case of similar items:

Proposition 4.8. *Under assumption* $(*)$*, the following holds:*

$$\mathbb{E}[\mathcal{Q}_1(\tilde{x}, \chi)] > \overline{r}_{min} \cdot \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+ \right] - \widetilde{p} \cdot \frac{\sum_{i \in \overline{S}} \mu_i}{|\overline{S}|} \right)$$

where $\widetilde{p}$ *denotes the probability of an underload.*

Proof. First of all we replace the second stage rewards $\overline{r}_i$ $(i = 1, \ldots, n)$ by $\overline{r}_{min}$. This replacement allows us to apply lemma 4.7 and we get:

$$\begin{aligned}
\mathbb{E}[\mathcal{Q}_1(\tilde{x}, \chi)] &> \overline{r}_{min} \cdot \left(\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \left((c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) - \min_{i \in \overline{S}} \chi_i \right) \right] \right) \\
&= \overline{r}_{min} \cdot \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+ \right] - \mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \cdot \min_{i \in \overline{S}} \chi_i \right] \right)
\end{aligned}$$

The following claim is similar to claim 4.2 (and so is its proof):

Claim 4.9. *Under assumption* $(*)$*, the following holds:*

$$\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(c - \sum_{i=1}^{n} \tilde{x}_i \chi_i) \cdot \min_{i \in \overline{S}} \chi_i \right] \leq \widetilde{p} \cdot \mathbb{E}\left[\min_{i \in \overline{S}} \chi_i \right]$$

where $\widetilde{p}$ *denotes the probability of an underload.*

□

It follows:

$$\mathbb{E}[\mathcal{Q}_1(\tilde{x}, \chi)] > \overline{r}_{min} \cdot \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+ \right] - \widetilde{p} \cdot \mathbb{E}\left[\min_{i \in \overline{S}} \chi_i \right] \right)$$

For any set $\overline{S} \subseteq \{1, \ldots, n\}$, $\mathbb{E}\left[\min_{i \in \overline{S}} \chi_i\right]$ can easily be bounded, as the weight of the lightest item among all items in $\overline{S}$ is at most $1/|\overline{S}|$ of the total weight of the items in $\overline{S}$:

$$\min_{i \in \overline{S}} \chi_i \leq \frac{\sum_{i \in \overline{S}} \chi_i}{|\overline{S}|} \tag{12}$$

It follows

$$\mathbb{E}[\min_{i \in \overline{S}} \chi_i] \leq \frac{\mathbb{E}[\sum_{i \in \overline{S}} \chi_i]}{|\overline{S}|} = \frac{\sum_{i \in \overline{S}} \mu_i}{|\overline{S}|}$$

which ends the proof. □

The previous proposition can be generalized as follows:

Proposition 4.10. *If the probability for an item to have k times the size of another item is zero ($k \geq 2$), it follows*

$$\mathbb{E}[\mathcal{Q}_1(\tilde{x}, \chi)] > \overline{r}_{min} \cdot \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+ \right] - \widetilde{p} \cdot (k-1) \cdot \frac{\sum_{i \in \overline{S}} \mu_i}{|\overline{S}|} \right)$$

where $\widetilde{p}$ denotes the probability of an underload.

□

Upper bound on the expectation of $\mathcal{Q}_2$
The following lemma is the counterpart of lemma 4.7 in the underload case. The proof is similar:

Lemma 4.11. *Under assumption $(*)$, the following holds:*

$$\sum_{i=1}^{n} (y^-)_i^* \hat{\chi}_i - (\sum_{i=1}^{n} \tilde{x}_i \hat{\chi}_i - c) < \min_{i \in S} \hat{\chi}_i$$

□

Proposition 4.12. *Under assumption $(*)$, the following holds:*

$$\mathbb{E}[\mathcal{Q}_2(\tilde{x}, \chi)] < (1 + \frac{1}{|S|}) \mathbb{E}\left[[\sum_{i=1}^{n} \tilde{x}_i \chi_i - c]^+ \right] + \frac{c \cdot (1 - \tilde{p})}{|S|}$$

where $\widetilde{p}$ denotes the probability of an underload.

Proof. Due to lemma 4.11 we have

$$\mathbb{E}[\mathcal{Q}_2(\tilde{x}, \chi)] < \mathbb{E}\left[[\sum_{i=1}^{n} \tilde{x}_i \chi_i - c]^+ \right] + \mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+} (\sum_{i=1}^{n} \tilde{x}_i \chi_i - c) \cdot \min_{i \in S} \chi_i \right]$$

Using inequality (12) we further have

$$\begin{aligned} \mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+} (\sum_{i=1}^{n} \tilde{x}_i \chi_i - c) \cdot \min_{i \in S} \chi_i \right] &\leq \mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+} (\sum_{i=1}^{n} \tilde{x}_i \chi_i - c) \cdot \frac{\sum_{i=1}^{n} \tilde{x}_i \chi_i}{|S|} \right] \\ &= \frac{1}{|S|} \left(\mathbb{E}\left[[\sum_{i=1}^{n} \tilde{x}_i \chi_i - c]^+ \right] + c \cdot (1 - \tilde{p}) \right) \end{aligned}$$

□

Proposition 4.12 can be generalized as follows:

Proposition 4.13. *If the probability for an item to have k times the size of another item is zero (for $k \geq 2$), it follows*

$$\mathbb{E}[\mathcal{Q}_2(\tilde{x}, \chi)] < (1 + \frac{k-1}{|S|})\mathbb{E}\left[[\sum_{i=1}^{n} \tilde{x}_i \chi_i - c]^+\right] + \frac{c \cdot (1-\tilde{p}) \cdot (k-1)}{|S|}$$

where $\widetilde{p}$ denotes the probability of an underload.

□

Lower bound on the expectation of $\mathcal{Q}$ in case of similar items
Combining the previous results from this subsection, we get the following lower bounds on the expectation of the optimal second-stage solution $Q(x, \chi)$:
From Propositions 4.8 and 4.12 it follows:

Proposition 4.14. *Let $\tilde{x}$ be a feasible first-stage solution. Under assumption $(*)$, the following upper bound on the corresponding overall solution holds*

$$\mathbb{E}\left[\sum_{i=1}^{n} r_i \chi_i \tilde{x}_i\right] + \mathbb{E}\left[\mathcal{Q}(\tilde{x}, \chi)\right] \geq \sum_{i=1}^{n} r_i \mu_i \tilde{x}_i + \overline{r}_{min} \cdot \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+\right] - \widetilde{p} \cdot \frac{\sum_{i \in \overline{S}} \mu_i}{|\overline{S}|}\right)$$
$$- d \cdot \left(\left(1 + \frac{1}{|S|}\right)\mathbb{E}\left[[\sum_{i=1}^{n} \tilde{x}_i \chi_i - c]^+\right] + \frac{c \cdot (1-\tilde{p})}{|S|}\right)$$

where $\widetilde{p}$ denotes the probability of an underload.

□

From Propositions 4.10 and 4.13 we get:

Proposition 4.15. *Let $\tilde{x}$ be a feasible first-stage solution. If the probability for an item to have k times the size of another item is zero (for $k \geq 2$), the following upper bound on the corresponding overall solution holds:*

$$\mathbb{E}\left[\sum_{i=1}^{n} r_i \chi_i \tilde{x}_i\right] + \mathbb{E}\left[\mathcal{Q}(\tilde{x}, \chi)\right] \geq \sum_{i=1}^{n} r_i \mu_i \tilde{x}_i + \overline{r}_{min} \cdot \left(\mathbb{E}\left[[c - \sum_{i=1}^{n} \tilde{x}_i \chi_i]^+\right] - \widetilde{p} \cdot (k-1) \cdot \frac{\sum_{i \in \overline{S}} \mu_i}{|\overline{S}|}\right)$$
$$- d \cdot \left(\left(1 + \frac{k-1}{|S|}\right)\mathbb{E}\left[[\sum_{i=1}^{n} \tilde{x}_i \chi_i - c]^+\right] + \frac{c \cdot (1-\tilde{p}) \cdot (k-1)}{|S|}\right)$$

where $\widetilde{p}$ denotes the probability of an underload.

□

5 Branch-and-bound algorithm

In the previous section we proposed methods to calculate a lower bound on the objective function value for a given first-stage solution $\tilde{x}$. However, for a bad choice of $\tilde{x}$ this lower bound might be far from the optimal overall solution.
In order to search the first-stage solution space for a best possible lower bound on the overall problem (1), we chose a branch-and-bound framework involving the bounds proposed above. Concerning the computation of the upper bounds that serve to evaluate subtrees, we apply the following policy:
In [5], Cohn and Barnhart solve the $SRKP$ (4) without the probability constraint (4a), i.e. the basic *Simple Recourse Knapsack Problem (basic SRKP)* (or *Static Stochastic Knapsack Problem*). They propose very simple and fast computable upper bounds for the basic $SRKP$ that serve them to prune non-valuable subtrees in a branch-and-bound framework. We use their upper bounds for the same purpose as relaxing the probability constraint gives us an upper bound on the $SRKP$ and thus also on the initial problem. If their upper bounds are not tight enough to prune the subtree that we are currently evaluating, we solve the $SRKP$ as proposed in section 3.
As described in [5] and [14] the first step of the branch-and-bound algorithm is to rank the items following their value of r_i/σ_i^2 which defines the binary search tree. The full framework is presented in Algorithm 5.1.

6 Numerical Results

The aim of this section is twofold: First of all we want to give the reader an idea of the relative gap between the proposed upper bound and the best possible lower bound that one can get by using the lower bounds proposed in section 4. The latter is obtained by applying the branch-and-bound algorithm given in the previous section. The second aim is to analyze the quality of the branch-and-bound algorithm itself, i.e. the CPU-time it needs to find the best possible lower bound and the number of upper bounds it has to calculate. Obviously, these subjects are linked as the better the quality of the upper and lower bounds, the more subtrees are pruned during the branch-and-bound algorithm and the less upper bounds have to be computed.
We tested our method on 50 randomly generated instances for each case and each dimension. In the general case the weight means were generated uniformly between 100 and 1000. For the case of similar items, we chose a normal distribution with mean 225 and variance 25 to generate the weight means (see [5]). We tested our method on the problem variant with probability constraint in the first stage and fixed the prescribed probability p to 95%. The tests were carried out on an Intel PC with 2MB RAM and we allowed a maximum average CPU-time of $1h$. In the following tables, * indicates that an average CPU-time of more than $1h$ would have been needed to treat all instances of this dimension.
The following analysis of the numerical tests is divided in three parts: In the first part, we study the case where, in the second stage, items can be added only. When solving the instances with similar items we compare the general lower bounds with the lower bounds proposed particularly for this case. In the second part, we study the problem variant where items can be rejected only. The last part contains the numerical results for the problem

Branch-and-Bound Algorithm

1. Rank the items by their value of r_i/σ_i^2. This ranking defines the binary tree with the highest ranked item at the root.
2. Set $INF \leftarrow 0$. Define L as the set of first-stage decisions to explore. Set $L \leftarrow \emptyset$.
3. Plunge the tree to find the first first-stage decision x: Beginning at the root of the tree, add the current item if and only if the lower bound on the overall objective function value increases and the chance constraint is still satisfied after adding the item. Calculate a lower bound $LB(x)$ on the first-stage objective function. Set $INF \leftarrow LB(x)$. Add the found first-stage decision x to L. Set the associated upper bound $UB(x)$ to infinity.
4. **If** $L = \emptyset$, go to step 8.
   Else select $x \in L$ such that $x = \arg\max_{x \in L} LB(x)$. Go to step 5.
5. **If** $UB(x) > INF$, go to step 6.
   Else remove x from L. Go to step 4.
6. **If** there is no accepted item left in the selected first-stage decision x such that the subtree defined by rejecting this item has not already been plunged or rejected, remove x from L. Go back to step 4.
   Else, following our ranking, choose the first accepted item such that the subtree defined by rejecting this item has not already been plunged or rejected. Calculate an upper bound UB on this subtree. Go to step 7.
7. **If** $UB \leq INF$, reject the subtree, go to step 6.
   Else plunge the subtree as described in step 3 to find a new first-stage decision $\widetilde{x}$. Calculate $LB(\widetilde{x})$ and set $UB(\widetilde{x}) \leftarrow UB$. Add $\widetilde{x}$ together with the values $UB(\widetilde{x})$ and $LB(\widetilde{x})$ to L. If $LB(\widetilde{x}) > INF$, set $INF \leftarrow LB(\widetilde{x})$.
   Go to step 4.
8. The current value INF is the best lower bound possible for problem (1) that can be obtained by the proposed method(s).

Algorithm 5.1:

setting where items can be added in case of an underload and have to be removed in case of an overload.
The following tables present for each set of instances the minimal gap (between upper and best lower bound found; second column) and the maximal gap (third column) recorded as well as the average gap (fourth column) and the standard deviation of the gaps (fifth column). Moreover, the tables contain the average number of nodes considered during the branch-and-bound algorithm (i.e. the number of computed upper bounds per run; sixth column) as well as the average CPU-time needed by the branch-and-bound algorithm (seventh column). For a study of the stochastic gradient algorithm used to solve the continuous problem (4) see [14].

	General items					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	3.18%	31.09%	11.41%	4.98%	0.1	29
20	2.43%	21.51%	9.35%	4.94%	0.16	38
30	2.12%	14.87%	7.02%	3.48%	0.40	70
50	1.48%	16.50%	6.82%	3.36%	55.00	6693
75	1.08%	11.02%	6.49%	2.14%	12.36	970
100	2.07%	14.29%	6.69%	2.41%	59.00	3501
150	2.30%	10.99%	6.61%	2.06%	928.02	36807
250	*	*	*	*	*	*

Table 1: Items can only be added in the second stage

6.1 Items can only be added in the second stage

Table 1 shows the results of our method for non-similar items. First of all we observe that up to 30 items the gaps improve visibly while their diminution is less obvious for a greater number of items.
Furthermore, we notice that the number of nodes considered during the branch-and-bound algorithm depends not only on n, but also on the hardness of each particular instance: while for instances with 75 and 100 items the branch-and-bound algorithm considers on average 970 and 3501 nodes, respectively, it computes on average 6693 upper bounds to solve the instances of size 50 due to one single, hard instance (230370 nodes).
When applying the general lower bounds to instances of similar items (Table 2) we observe much smaller relative gaps and standard deviations compared to the general instances. In addition, the minimal and the average gap are continuously decreasing for increasing n. As a result of the tighter bounds, the branch-and-bound algorithm needs to compute less upper bounds. As the computation of upper bounds using a stochastic gradient algorithm is the most time consuming part of Algorithm 5.1 (see [14]), the corresponding CPU-times are, as well, significantly smaller. Given a maximum average CPU-time of $1h$, we are thus able to treat instances of more than 2000 items, contrary to 150 items in the general case.
When comparing the performance of the algorithm that uses the general method to compute lower bounds with that proposed for the special case of similar items, we notice that the latter performs only slightly better due to unsignificantly better lower bounds.

6.2 Items can only be rejected in the second stage

When solving instances with non-similar weight means (see Table 3), we first of all remark the high average relative gap between upper and lower bound for small size items. We assume that the dimension of the gap is essentially due to the lower bound whose quality depends mainly on the estimation of $\mathbb{E}\left[\mathbb{1}_{\mathbb{R}^+}(\sum_{i=1}^{n} \tilde{x}_i\chi_i - c) \cdot \max_{i\in S} \chi_i\right]$ (see subsection 4.2). However, contrary to the case where items can only be added, the average gap is continuously

shrinking with increasing n and for medium size instances it is even smaller than in the former case. We are thus able to solve instances with up to 150 items, as well.

When applying the general method to compute lower bounds on instances with similar items (Table 4), the gaps are less important for instances of small size. However, unlike all the cases studied before, the average gap as well as the minimum gap seem to increase with n (at least within the dimensions tested).

Contrary to the case where items can only be added in the second stage, we observe an important improvement of the gap when replacing the general method to compute lower bounds by the particular method for similar items. In terms of computing time, we are now able to solve problems with up to 1000 items (and more) in less than $1h$ (on average).

However, a drawback of our approach is revealed: To compute upper bounds, we use a stochastic gradient method that only computes approximate solutions of the relaxed version of our problem. More precisely, the value obtained might be slightly smaller than the optimum of the relaxation. Consequently, in case of large size instances with a very small integrality gap, we cannot be sure if the upper bound computed is in fact an upper bound on our problem. If in addition the best lower bound obtained is close to the optimal solution, we might have the strange situation that the computed upper bound is smaller than the computed lower bound, which was the case for some of the instances with 500 and 1000 items (Table 4).

6.3 Items can be added or rejected in the second stage

Concerning the evaluation of the expected second-stage solution, this case is the most difficult among the three problem variants. Therefore, it is not surprising that the gaps in the case where items can be added or rejected in the second stage are slightly higher than in the previous cases (see Tables 5 and 6).

Concerning the instances of similar items, we observe once more a significant improvement of the results when replacing the general method by the particular one and the results are of nearly the same quality as in the case where items can be added only. In fact, the latter is what one could expect due to the probability constraint, as in only 5% of the cases items have to be rejected in the second stage.

6.4 Recapitulation of the numerical results

To summarize the above results, we assume that whenever items can be rejected in the second stage, the lower bounds obtained by the general method are of a rather poor quality. Furthermore, we notice that in this same case the particular method for similar items significantly improves the gap between upper and lower bound, the number of considered nodes and the CPU-time compared to the general method. If, in addition, we allow items to be added in case of an underload, we are able to treat instances of more than 2000 items in an average CPU-time of less than $30min$.

In contrary, in the case where items can only be added in the second stage the general method performs on instances with similar items nearly as good as the particular one. We deduce that claim 4.2 and claim 4.9 are important factors in the computation of the general lower bounds.

7 Conclusion

In this paper we study a two-stage knapsack problem with random weights. We introduce a probability constraint in the first stage in order to restrict the probability of an overload, but all methods and bounds proposed are still valid if the problem contains no such constraint. As the weights are assumed to be normally distributed, no method to exactly evaluate the expectation of the second-stage solution for a given first stage decision is known. Instead, we propose to compute lower bounds on this expectation and we use a branch-and-bound framework in order to find first-stage solutions that provide best possible lower bounds. For comparison, upper bounds are computed by solving a continuous version of the problem using a stochastic gradient algorithm. Special interest is given to the case where the items have similar weight means.

Numerical tests have shown that our method provides bounds with small relative gaps for medium size instances with similar items and even near-optimal lower bounds for large size instances with similar items. However, for non-similar items the gaps are rather high, especially when we allow items to be rejected in the second stage. Further work will therefore primarily consist in improving the quality of the general bounds.

Furthermore, when searching for good lower bounds, one could replace the exhaustive branch-and-bound framework by a heuristic. Especially in cases where the best lower bound that we can obtain is close to the computed upper bound (and thus to the optimal solution), it might be promising to just search a small subspace of the first-stage solution space.

In order to evaluate the quality of the upper and lower bounds, an idea could be to apply a Sample Average Approach method to the initial problem, i.e. to approximate the optimal solution by discretization of the continuous sample space and solving the corresponding deterministic equivalent linear problem. However, this approach is only practicable if already a relatively small sample is sufficient to obtain good approximations.

References

[1] Semra Ağralıand Joseph Geunes. A class of stochastic knapsack problems with poisson resource requirements. http://plaza.ufl.edu/sagrali/research_files/Poisson_KP_ORL_Submission.pdf (Accessed 24 October 2008), 2008.

[2] Stephen Boyd, Herve Lebret, Miguel Soma Lobo, and Lieven Vandenberghe. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[3] Robert L. Carraway, Robert L. Schmidt, and Lawrence R. Weatherford. An algorithm for maximizing target achievement in the stochastic knapsack problem with normal returns. *Naval research logistics*, 40(2):161–173, 1993.

[4] Joăo Claro and Jorge Pinho de Sousa. A multiobjective metaheuristic for a mean-risk static stochastic knapsack problem. www.springerlink.com/index/b668736740218j72.pdf (Accessed 24 October 2008), 2008.

[5] Amy Cohn and Cynthia Barnhart. 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)*, 1998.

[6] Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. In *Proceedings 45th Annual IEEE 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 208–217, 2004.

[7] A. Goel and P. Indyk. Stochastic load balancing and related problems. In *40th Annual Symposium on Foundations of Computer Science*, pages 579 – 586, 1999.

[8] Mordechai I. Henig. Risk criteria in a stochastic knapsack problem. *Operations Research*, 38(5):820–825, 1990.

[9] Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack Problems*. Springer-Verlag (Berlin, Heidelberg), 2004.

[10] Jon Kleinberg, Yuval Rabani, and Éva Tardos. Allocating bandwidth for bursty connections. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 664 – 673, 1997.

[11] Anton J. Kleywegt and Jason D. Papastavrou. The dynamic and stochastic knapsack problem with random sized weights. *Operations Research*, 49:26–41, 2001.

[12] Anton J. Kleywegt, Alexander Shapiro, and Tito Homem de mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12:479–502, 2001.

[13] Olivier Klopfenstein and Dritan Nace. A robust approach to the chance-constrained knapsack problem. http://www.optimization-online.org/DB_HTML/2006/03/1341.html (Accessed 24 October 2008), 2006.

[14] Stefanie Kosuch and Abdel Lisser. Upper bounds for the 0-1 stochastic knapsack problem and a b&b algorithm. *Annals of Operations Research (Online First)*, 2009. http://dx.doi.org/10.1007/s10479-009-0577-5.

[15] Abdel Lisser, Rafael Lopez, and Xu Hu. Stochastic quadratic knapsack with recourse. working paper (Laboratoire de recherche en informatique, Université Paris Sud, Orsay, France), 2009.

[16] Alberto Marchetti-Spaccamela and Carlo Vercellis. Stochastic on-line knapsack problems. *Mathematical Programming*, 68:73–104, 1995.

[17] David P. Morton and R. Kevin Wood. *Advances in Computational and Stochastic Optimization, Logic Programming and Heuristic Search*, chapter On a stochastic knapsack problem and generalizations, pages 149–168. Kluwer Academic Publishers (Norwell, MA, USA), 1997.

[18] K. W. Ross and D. H. K. Tsang. The stochastic knapsack problem. *IEEE Transactions on Communications*, 37(7):740–747, 1989.

	Similar items, general lower bounds					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	2.64%	18.86%	7.11%	3.38%	0.06	14
20	1.87%	12.26%	5.46%	2.41%	0.08	17
30	1.58%	6.29%	3.05%	0.96%	0.16	25
50	0.92%	8.37%	2.09%	1.15%	0.44	47
75	0.72%	1.85%	1.28%	0.34%	0.96	73
100	0.57%	5.17%	1.07%	0.85%	1.70	97
150	0.41%	0.93%	0.65%	0.17%	4.40	170
250	0.29%	0.63%	0.43%	0.10%	14.28	335
500	0.17%	0.33%	0.25%	0.05%	68.80	809
1000	0.08%	0.19%	0.14%	0.02%	308.44	1776
2000	0.06%	0.11%	0.09%	0.01%	1927.06	5586
3000	*	*	*	*	*	*
	Similar items, particular lower bounds					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	2.39%	18.49%	6.67%	3.38%	0.06	13
20	1.73%	11.95%	5.22%	2.38%	0.06	15
30	1.42%	6.05%	2.86%	0.95%	0.14	23
50	0.83%	8.23%	1.96%	1.15%	0.42	45
75	0.60%	1.77%	1.20%	0.34%	0.88	67
100	0.50%	5.10%	1.01%	0.84%	1.62	93
150	0.37%	0.89%	0.60%	0.17%	3.96	154
250	0.26%	0.55%	0.43%	0.06%	14.24	314
500	0.16%	0.32%	0.23%	0.04%	62.86	747
1000	0.09%	0.17%	0.13%	0.02%	276.78	1633
2000	0.06%	0.11%	0.08%	0.01%	1534.18	4477
3000	*	*	*	*	*	*

Table 2: Items can only be added in the second stage (similar items)

	General items					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	17.77%	42.44%	27.79%	5.63%	0.30	93
20	14.18%	43.36%	23.67%	7.31%	0.84	215
30	10.15%	36.81%	21.57%	7.52%	10.44	1887
50	5.85%	23.93%	9.61%	3.95%	47.76	5408
75	4.10%	13.56%	5.97%	1.86%	142.98	11223
100	3.17%	6.53%	4.19%	0.72%	291.10	17244
150	2.13%	3.31%	2.69%	0.30%	2273.92	90672
250	*	*	*	*	*	*

Table 3: Items can only be rejected in the second stage

	Similar items, general lower bounds					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	8.32%	25.71%	14.84%	3.65%	0.08	27
20	6.71%	24.26%	14.42%	3.87%	0.30	77
30	9.18%	22.16%	14.99%	2.60%	2.76	495
50	10.85%	21.33%	14.85%	2.16%	69.98	7863
75	13.03%	21.22%	15.74%	2.12%	3325.820	249275
100	*	*	*	*	*	*
	Similar items, particular lower bounds					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	0.53%	8.14%	3.82%	2.32%	0.02	7
20	0.29%	7.26%	2.61%	1.78%	0.06	12
30	0.30%	3.72%	1.72%	1.03%	0.12	19
50	0.22%	2.23%	1.02%	0.62%	0.34	36
75	0.16%	1.41%	0.69%	0.38%	0.72	53
100	0.10%	1.03%	0.40%	0.27%	1.16	66
150	0.05%	0.60%	0.26%	0.18%	2.62	101
250	0.01%	0.35%	0.13%	0.11%	7.22	170
500	*-0.03%*	*0.17%*	*0.04%*	*0.05%*	*25.16*	*300*
1000	*-0.06%*	*0.08%*	*0.00%*	*0.03%*	*98.68*	*590*

Table 4: Items can only be rejected in the second stage (similar items)

	General items					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	19.67%	45.12%	28.63%	5.33%	0.72	223
20	17.66%	42.91%	27.15%	2.57%	5.86	1441
30	12.08%	32.08%	23.43%	5.43%	478.62	75234
50	*	*	*	*	*	*

Table 5: Items can be added or rejected in the second stage

	Similar items, general lower bounds					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	9.58%	25.05%	15.36%	3.16%	0.14	43
20	9.38%	22.38%	14.55%	2.68%	0.52	133
30	9.32%	19.23%	13.62%	1.90%	3.48	621
50	9.64%	18.10%	13.13%	1.78%	113.26	12791
75	*	*	*	*	*	*
	Similar items, particular lower bounds					
n	**Minimal Gap**	**Maximal Gap**	**Average Gap**	**Standard Deviat.**	**CPU-time (sec)**	**# considered nodes**
15	2.92%	18.80%	6.87%	3.33%	0.04	13
20	1.90%	11.95%	5.30%	2.34%	0.06	16
30	1.59%	6.05%	2.88%	0.94%	0.14	24
50	1.06%	8.23%	1.99%	1.15%	0.42	45
75	0.78%	1.76%	1.23%	0.31%	0.90	69
100	0.62%	5.24%	1.04%	0.85%	1.62	98
150	0.45%	0.96%	0.63%	0.15%	4.32	170
250	0.30%	0.61%	0.42%	0.09%	13.68	324
500	0.19%	0.34%	0.24%	0.04%	63.48	755
1000	0.11%	0.17%	0.14%	0.02%	303.40	1797
2000	0.06%	0.11%	0.09%	0.01%	1739.30	5104
3000	*	*	*	*	*	*

Table 6: Items can be added or rejected in the second stage (similar items)