# Recoverable Robust Knapsacks: the Discrete Scenario Case

Christina Büsing[1], Arie M. C. A. Koster[2], and Manuel Kutschka[2]

[1] Technische Universität Berlin, Institut für Mathematik, Straße des 17. Juni 136, D-10623 Berlin, Germany, cbuesing@math.tu-berlin.de
[2] RWTH Aachen University, Lehrstuhl II für Mathematik, Wüllnerstr. 5b, D-52062 Aachen, Germany, {koster,kutschka}@math2.rwth-aachen.de

**Abstract.** The knapsack problem is one of the basic problems in combinatorial optimization. In real-world applications it is often part of a more complex problem. Examples are machine capacities in production planning or bandwidth restrictions in telecommunication network design. Due to unpredictable future settings or erroneous data, parameters of such a subproblem are subject to uncertainties.
In high risk situations a robust approach should be chosen to deal with these uncertainties. Unfortunately, classical robust optimization outputs solutions with little profit by prohibiting any adaption of the solution when the actual realization of the uncertain parameters is known. This ignores the fact that in most settings minor changes to a previously determined solution are possible. To overcome these drawbacks we allow a limited recovery of a previously fixed item set as soon as the data are known by deleting at most $k$ items and adding up to $\ell$ new items.
We consider the complexity status of this recoverable robust knapsack problem and extend the classical concept of cover inequalities to obtain stronger polyhedral descriptions. Finally, we present two extensive computational studies to investigate the influence of parameters $k$ and $\ell$ to the objective and evaluate the effectiveness of our new class of valid inequalities.

## 1 Introduction

An instance of the knapsack problem (KP) consists of an item set $N = \{1, \ldots, n\}$ with profits $p_j$ and positive weights $w_j$ for all items $j \in N$, and a capacity $c$. The values $p_j$, $w_j$, and $c$ are taken as integers. The objective of such an instance is to select a subset $X$ of these items such that the total profit of $X$ is maximized and the total weight does not exceed $c$. Despite its simple structure the problem is known to be weakly **NP**-hard [16] but solvable via dynamic programming in pseudo-polynomial time [5]. Different branch-and-cut algorithms are used to solve this problem in practice. A detailed introduction to the knapsack problem and its variations can be found in Martello and Toth [20] and Kellerer et al. [17].

This work was supported by the Federal Ministry of Education and Research (BMBF grant 03MS616A, project *ROBUKOM - Robust Communication Networks*, www.robukom.de), and the DFG Research Center Matheon *Mathematics for key technologies*.

In Yu [23] a robust version of the knapsack problem is defined by introducing uncertainty in the profit values: Let $\mathcal{S}$ be a set of scenarios, each scenario $S$ determining a profit function $p^S : N \to \mathbb{N}$. The robust knapsack problem is to find a set of items maximizing the minimum optimal profit over all scenarios such that the total weight does not exceed the capacity. By a reduction from the set covering problem, Yu showed that for discrete scenarios sets with an unbounded number of scenarios the decision version of the problem is strongly **NP**-complete. As mentioned by Aissi et al. [2] this proof includes the result that the optimization version is not $f$-approximable for any function $f : N \to (1, \infty)$. For a discrete scenario set with a constant number of scenarios the problem is weakly **NP**-complete, solvable in pseudo-polynomial time [14,23] and there exists an FPTAS [2]. Iida [13] provided a computational study on the robust knapsack problem with discrete scenarios to derive upper and lower bounds for the profit. Recently, Klopfenstein and Nace [18] introduced robust knapsacks with uncertainty based on the approach of Bertsimas and Sim [6,7].

Due to the already mentioned drawbacks of robustness, several authors, e.g., Liebchen et al. [19], extended the robust approach by means of recovery. This new concept is a robust version of stochastic programming with recourse and has been successfully applied to several railway applications like platforming or timetabling [10]. This work is part of our investigations to apply recoverable robustness to communication networks.

*Model and Notation.* We introduce for the first time a recoverable robust version of the knapsack problem, in which the weights as well as the profits are subject to uncertainties. Those uncertainties are given via a set of scenarios. As in the robust setting a subset of items is chosen in a first stage such that its first stage weight does not exceed the first stage capacity. Yet, in the second stage, i.e., when a scenario is revealed and the profits and weights are known, $k$ items may be removed from the first stage solution and $\ell$ items may be added. The new subset must satisfy the scenario capacity restriction $c^S$ according to the weights of scenario $S$. The objective is to find a first stage solution with maximum total profit. The total profit is the sum of the first stage profit and the minimum optimum scenario profit. By varying $k$ and $\ell$ the *gain of recovery* can be analyzed. Definitions 1 and 2 give formal descriptions.

**Definition 1 ($(k,\ell)$-Recoverable Robust KP ($(k,\ell)$-rrKP))** *Let $c^0 \in \mathbb{N}$ be the first stage capacity, $N = \{1, \dots, n\}$ be a set of items, $p_j^0$ the first stage profit and $w_j^0$ the first stage weight of each item $j \in N$. Each scenario $S$ of a given set of scenarios $\mathcal{S}$ defines a profit function $p^S : N \to \mathbb{N}$, a weight function $w^S : N \to \mathbb{N}$ and a capacity $c^S \in \mathbb{N}$. Furthermore, the parameter $k \in \mathbb{N}$ limits the number of deletable items from a given set and the parameter $\ell \in \mathbb{N}$ the number of new items contained in a new set. For a given subset $X \subseteq N$ the* recovery set *$\mathcal{X}_X^{(k,\ell)}$ includes all subsets of $N$ which contain at most $\ell$ additional elements and fail to contain at most $k$ elements of $X$, i.e.,*

$$\mathcal{X}_X^{(k,\ell)} = \{X' \subseteq N : |X \backslash X'| \leq k \text{ and } |X' \backslash X| \leq \ell\}.$$

*With $\mathcal{X}^S$ for $S \in \mathcal{S}$ we denote all subsets of $X' \subseteq N$ that satisfy the* scenario weight constraint $\sum_{i \in X'} w_i^S \leq c^S$. *A* feasible first stage solution *of the $(k,\ell)$-recoverable robust knapsack problem is a subset $X \subseteq N$ which satisfies the* first stage weight constraint $\sum_{i \in X} w_i^0 \leq c^0$ *and $\mathcal{X}_X^{(k,\ell)} \cap \mathcal{X}^S \neq \emptyset$ for all $S \in \mathcal{S}$. An* optimal *solution $X^* \subseteq N$ of the $(k,\ell)$-recoverable robust knapsack problem is a feasible solution with maximum total profit. The* total profit $p(X)$ *of a feasible solution $X$ is defined as*

$$p(X) = \sum\nolimits_{i \in X} p_i^0 + \min\nolimits_{S \in \mathcal{S}} \max\nolimits_{X^S \in \mathcal{X}_X^{(k,\ell)} \cap \mathcal{X}^S} \sum\nolimits_{i \in X^S} p_i^S.$$

**Definition 2** *Given $k$ and $\ell$, the* gain of recovery *of a $(k,\ell)$-Recoverable Robust Knapsack instance is the percentual increase of the optimal solution value of $(k,\ell)$-rrKP instance w.r.t. the optimal solution value of $(0,0)$-rrKP instance.*

In a discrete scenario set $\mathcal{S}_D$ every scenario is explicitly given with its weight function $w^S : N \to \mathbb{N}$, its profit function $p^S : N \to \mathbb{N}$ and its capacity $c^S$. The following integer program models the $(k,\ell)$-recoverable robust knapsack problem

$$\begin{aligned}
\max \quad & \sum\nolimits_{i \in N} p_i^0 x_i + \omega & & & & (1)\\
s.t. \quad & \sum\nolimits_{i \in N} w_i^0 x_i & \leq & c^0 & & (2)\\
& \sum\nolimits_{i \in N} w_i^S x_i^S & \leq & c^S & \forall S \in \mathcal{S}_D & (3)\\
& x_i^S - x_i - y_i^S & \leq & 0 & \forall S \in \mathcal{S}_D,\ i \in N & (4)\\
& x_i - x_i^S - z_i^S & \leq & 0 & \forall S \in \mathcal{S}_D,\ i \in N & (5)\\
& \sum\nolimits_{i \in N} y_i^S & \leq & \ell & \forall S \in \mathcal{S}_D & (6)\\
& \sum\nolimits_{i \in N} z_i^S & \leq & k & \forall S \in \mathcal{S}_D & (7)\\
& \omega - \sum\nolimits_{i \in N} p_i^S x_i^S & \leq & 0 & \forall S \in \mathcal{S}_D & (8)\\
& \omega \in \mathbb{Q}_+,\ x_i, x_i^S, y_i^S, z_i^S \in & & \{0,1\} & \forall S \in \mathcal{S}_D, i \in N & (9)
\end{aligned}$$

The variable vector $x$ represents the first stage solution, $x^S$ the solution taken in scenario $S \in \mathcal{S}_D$, $y_i^S$ determines if item $i$ is added in $S$ and $z_i^S$ if item $i$ is removed from $x$ in $S$. Inequalities (4)-(7) guarantee that $x^S$ is a feasible recovery for the first stage solution $x$ and (2) and (3) that the weight constraints are obeyed. The last inequality (8) in combination with the objective function models the total profit

$$\max \sum\nolimits_{i \in N} p_i^0 x_i + \min\nolimits_{S \in \mathcal{S}_D} \max \sum\nolimits_{i \in N} p_i^S x_i^S.$$

The minimum scenario profit, i.e., $\min_{S \in \mathcal{S}_D} \max \sum_{i \in N} p_i^S x_i^S$, is captured in the variable $\omega$. Clearly, the size of the program depends on the number of scenarios.

Through out the paper we will use the following notation: Given a weight function $f : N \to \mathbb{N}$, an item set $X \subseteq N$ and an integer $r$, we define $f(X) := \sum_{i \in X} f(i)$, the weight of the $r$ heaviest items $f(\max, X, r) := \max_{K \subseteq X\ |K| \leq r} f(K)$ and the weight of the $r$ lightest items $f(\min, X, r) := \min_{K \subseteq X,\ |K| \geq r} f(K)$.

*Results.* The contribution is twofold. On the one hand, we introduce and analyze the *gain of recovery* for the (robust) knapsack problem. Experiments show that an additional gain of up to 45% compared to the profit of a robust solution can be achieved by allowing the removal and addition of items. On the other hand, we investigate the computational complexity of the problem in theory and practice. We report on the hardness and (in)approximability of the problem. To enhance the solving process, we study the polytope described by the first stage variables. We show that (i) a generalization of the well-known cover inequalities provides an alternative formulation, (ii) these inequalities can be strengthened in a novel way (compared to the classical knapsack problem), and (iii) the integrated separation of the strengthened inequalities is significantly better than the separate-and-extend procedure.

*Outline.* In Sect. 2, we analyze the complexity status of the $(k, \ell)$-rrKP. Next, in Sect. 3, we generalize the classical concept of (extended) covers, whereas its separation is the topic of Sect. 4. The experimental analysis can be found in Sect. 5. We close with conclusions in Sect. 6.

## 2 Complexity of the $(k, \ell)$-rrKP

We start with an analysis of the complexity status of the $(k, \ell)$-rrKP for discrete scenarios and thus consider its canonical decision version. If such an instance $I$ is a yes-instance, there exist a set $X \subseteq N$ and sets $X^S \subseteq N$ for all $S \in \mathcal{S}_D$, such that $X$ is a feasible first stage solution, $X^S \in \mathcal{X}_X^{(k,\ell)} \cap \mathcal{X}^S$ and $p(X) = \sum_{i \in X} p_i^0 + \min_{S \in \mathcal{S}} \sum_{j \in X^S} p_j^S \geq K$. Since the feasibility of $X$ and $X^S \in \mathcal{X}_X^{(k,\ell)} \cap \mathcal{X}^S$ can be tested in polynomial time, the decision version of the $(k, \ell)$-rrKP problem with discrete scenarios is in **NP**. Yet, there will be no polynomial certificate to test for a given subset $X \subseteq N$ without any further information whether $X$ can be extended to a feasible solution with a total profit $p(X) \geq K$, unless $\mathbf{P} = \mathbf{NP}$. This results from a reduction from the knapsack problem. However, in the special case that $k$ and $\ell$ are constant, the recovery set $\mathcal{X}_X^{(k,\ell)}$ contains a constant number of solutions for any $X \subseteq N$. Then the total profit can be computed by enumeration.

Since the knapsack problem is a special case of the $(k, \ell)$-rrKP, the $(k, \ell)$-rrKP remains at least weakly **NP**-complete for one scenario. Next, we report on two further complexity results (proofs can be found in Büsing et al. [8]).

**Theorem 3** *The $(k, \ell)$-rrKP is strongly* **NP***-complete for unbounded sets of discrete scenarios even if either $p^0 = 0$ or $p^S = 0$ for all $S \in \mathcal{S}_D$ holds.*

Note, a lower bound on the approximation ratio of $(\ell + 1)/\ell$ can be obtained which implies that any $(k, 0)$-rrKP is inapproximable, unless $\mathbf{P} = \mathbf{NP}$. Further, the $(k, \ell)$-rrKP can be solved in pseudo-polynomial time for a bounded number of scenarios by extending the dynamic programming approach of Yu [23]. See [8].

## 3 Extended Cover-Inequalities

As pointed out by Crowder et al. [9] one motivation for studying the polytope of the knapsack problem is that valid inequalities can be used as cutting planes for general 0-1 linear integer programs (ILP). The idea is to consider each individual constraint of a 0-1 ILP as a 0-1 knapsack constraint. Several classes of inequalities are known for the knapsack polytope, such as the lifted cover inequalities of Balas [3] and Wolsey [22], or the weight inequalities of Weismantel [21]. For the class of $(k,\ell)$-rrKP problems we will focus on extended cover inequalities which have been shown to be quite efficient in solving knapsack instances by Kaparis and Letchford [15].

For a given $(k,\ell)$-rrKP instance the $(k,\ell)$-*recoverable robust knapsack polytope* $\mathcal{K}_D(k)$ of a discrete scenario set $\mathcal{S}_D$ is the convex hull over all valid first stage solutions, i.e.,

$$\mathcal{K}_D(k) := \operatorname{conv}\left\{x \in \{0,1\}^n : \sum_{i\in N} w_i^0 x_i \le c^0 \text{and } \min_{\substack{T\subseteq N\\ |T|\le k}} \sum_{i\in N\setminus T} w_i^S x_i \le c^S \ \forall S \in \mathcal{S}\right\}.$$

Note that $\ell$ does not play a role in the feasibility of a first stage solution. The polytope $\mathcal{K}_D(k)$ has full dimension if and only if $w_i^0 \le c^0$ for all $i \in N$ (and $w_i^S \le c^S$ for all $S \in \mathcal{S}$ if $k = 0$). We will now extend the well-known concepts of covers to define feasible inequalities for the $\mathcal{K}_D(k)$ polytope. In contrast to the deterministic case, such a cover already takes the recovery action into account.

**Definition 4** *A set $C \subseteq N$ is called an* rrKP cover *if either $w^0(C) \ge c^0 + 1$ or there exists $S \in \mathcal{S}_D$ with $w^S(C) - w^S(\max, C, k) \ge c^S + 1$. An rrKP cover $C$ is* minimal *if it satisfies $w^0(C) - w^0(\min, C, 1) \le c^0$ and $w^S(C) - w^S(\max, C, k) - w^S(\min, C, 1) \le c^S$ for all $S \in \mathcal{S}_D$. An rrKP cover defines the following* cover inequality

$$x(C) \le |C| - 1. \tag{10}$$

**Theorem 5** *Let $x$ be a 0-1 point. Then $x \in \mathcal{K}_D(k)$ if and only if $x$ satisfies all minimal cover inequalities.*

*Proof.* Let $T_x := \{i \in N \mid x_i = 1\}$ be the support of $x$, which satisfies all minimal cover inequalities. Let us further assume that $x$ is infeasible. Then $T_x$ is an rrKP cover by definition. In a last step we modify $T_x$ to become a minimal rrKP cover. For this reason we repeatedly remove an item $i \in T_x$ with $i = \arg\min_{i\in T_x} w_i^0$ if $w^0(T_x) \ge c^0 + 1$ or $i = \arg\min_{i\in T_x} w_i^S$ if $w^S(T_x) - w^S(\max, T_x, k) \ge c^S + 1$, until the remaining set $T_x'$ is a minimal rrKP cover. Thus, $x$ violates the (minimum) cover inequality defined by $T_x'$.

Conversely, a feasible point not satisfying all minimum cover inequalities either violates the first stage capacity constrained or does not contain a feasible recovery for one scenario. This is a contradiction to its feasibility. □

The concept of covers can be extended to strengthen the cover inequalities in a similar fashion as for the knapsack polytope. A canonical way to define an extension is by adding all items whose weight is greater than or equal to the highest not recovered item in an rrKP cover $C$. More formally, let $C$ be an rrKP cover and $S$ a scenario such that the scenario weight inequality is violated by $C$. A *canonical extension* $\overline{E}^S(C)$ is given by

$$\overline{E}^S(C) = \left\{ i \in N : w_i^S \geq w^S(\max, C, k+1) - w^S(\max, C, k) \right\} \cup C. \tag{11}$$

Yet, it even suffices for an item to be added to $C$ if its weights exceeds (i) the residual capacity according to the weights of the first $|C| - k - 1$ lowest-weight-items and (ii) the weight of the $k+2$ highest-weight-item in $C$.

**Definition 6** *Let $C$ be an rrKP cover for scenario $S$. An* extension $E^S(C)$ *of $C$ according to $S$ is defined by*

$$E^S(C) = C \cup \left\{ i \in N : \begin{array}{l} w_i^S \geq c^S - w^S(C) + w^S(\max, C, k+1) + 1, \\ w_i^S \geq w^S(\max, C, k+2) - w^S(\max, C, k+1) \end{array} \right\}$$

*and determines the* rrKP extended cover inequality

$$x(E^S(C)) \leq |C| - 1 \tag{12}$$

RrKP extended cover inequalities are valid for $\mathcal{K}_D(k)$, as it can be shown by applying carefully the corresponding definitions.

For the deterministic knapsack problem the set of inequalities obtained by extending minimal covers is independent of the extension method: Let $C$ be a minimal cover and $j_{\max} = \arg\max_{j \in C} w(C)$ and $i_{\min} = \arg\min_{i \in E(C) \setminus \overline{E}(C)} w(i)$. Hence, $C' = C \cup \{i_{\min}\} \setminus \{j_{\max}\}$ is a minimal cover with $|C'| = |C|$ and $\overline{E}(C') = \overline{E}(C)$. Yet, for the recoverable robust knapsack problem this is not the case:

**Example 7** *Consider an instance with 6 items, two scenarios and $k = 1$. Scenario $S_a$ sets weights $\{2, 2, 3, 4, 8, 9\}$ and scenario $S_b$ sets weights $\{10, 1, 5, 2, 1, 0\}$ to the items $1, \ldots, 6$. The scenario capacity is 6 for $S_a$ and 3 for $S_b$ respectively. Then the set $\overline{C} = \{1, 2, 4, 5\}$ is a minimal cover. If we extend $\overline{C}$ by the second method according to scenario $S_a$, $E^{S_a}(\overline{C}) = \{1, 2, 3, 4, 5, 6\}$ with $w^{S_a}(3) < w^{S_a}(4)$. But the set $C' = \{1, 2, 3, 5\}$ is not a minimal cover, since $w^{S_b}(C') - w^{S_b}(\max, C', 1) - w^{S_b}(\min, C', 1) > 3$. Also no other minimal cover induces a canonical extended cover inequality as strong as the one from $E^{S_a}(\overline{C})$.*

## 4 Separation algorithms

To experimentally evaluate the cover inequalities we study the separation problem for rrKP extended covers. The separation problem is to find an inequality,

that is violated by a given non-integer point, or prove that non exists. In our case, we are interested in finding a violated rrKP extended cover inequality for a given fractional solution $x^* \in [0,1]^n$. Since the definition of a cover is based on a single scenario or the first stage weight set, we can consider each scenario $S \in \mathcal{S}_D$ and the first stage separately. For simplicity we drop the $S$ superscript and assume $k = 0$ in the case of the first stage weight constraint.

Our approach uses integer programming [11] and is based on the following observation: In order to find a violated extended cover inequality, it suffices to determine the items which are part of the cover and not recovered. We call those items the *core of the cover*. All other items exceeding the maximum item weight in the core of the cover, can be added to form, possibly, an extended rrKP cover. This is the case, if more than $k$ items are added to the core. Although those extra items are fixed as soon as the core of a cover is known, we introduce an integer program to determine both sets for two reasons: (i) the number of additional items is crucial for the detection of an rrKP cover, and (ii) even for deterministic knapsack instances a fractional point $x^*$ may not violate any cover inequality but an extended cover inequality.

In oder to model the separation problem we introduce two different binary variables $y_i$ and $z_i$ for each item $i \in N$. The variable $y_i$ determines whether item $i$ is part of the core of the cover. A cover has to satisfy $\sum_{i \in N} w_i y_i \geq c + 1$. An item is recovered or in the extension ($z_i = 1$), if its weight exceeds the weight of every item in the core. Note, that if an item with a sufficiently large weight is added to the recovery or the extension, all items with larger weights may also be added to the extension as they are exchangeable with the first one. To efficiently implement this condition, we group the items according to their weights: Let $0 \leq w_{i_1} < w_{i_2} < \ldots < w_{i_\theta} \leq c$ be an ordering of all different item weights occurring in the scenario and $T := \{1, \ldots, \theta\}$. We define $N(t) := \{j \in N : w_j = w_{i_t}\}$ for all $t \in T$. Then $z_i \leq z_j$ is valid for all $i \in N(t)$, $j \in N(t+1)$, $t = 1, \ldots, \theta - 1$. Hence, we obtain the following ILP

$$\max \quad \sum\nolimits_{j \in N} (x_j^* - 1) y_j + \sum\nolimits_{j \in N} x_j^* z_j - k \tag{13}$$

$$s.t. \quad \sum\nolimits_{j \in N} w_j y_j \geq c + 1 \tag{14}$$

$$y_j + z_j \leq 1 \qquad \forall j \in N \tag{15}$$

$$z_i - z_j \leq 0 \qquad \forall i \in N(t), j \in N(t+1), t \in T \tag{16}$$

$$y_j, z_j \in \{0,1\} \qquad \forall j \in N. \tag{17}$$

An optimal solution defines a violated rrKP extended cover inequality, if and only if the objective value is greater than $-1$. Note that in this case more than $k$ items are added to the core of the cover.

To speed up the solving of (13)-(17), several preprocessing rules can be applied to reduce the number of variables and constraints. In addition, it suffices to find a (non-optimal) feasible solution with objective value $> -1$. Found (extended) covers can be extended further greedily.

Based on this integer formulation and dynamic programming we finally introduce a pseudo-polynomial algorithm to solve the separation problem for the extended cover inequalities. We define $U = \cup_{i=1}^{n}\{w_i\}$ and $D = \sum_{i=1}^{n} w_i$. For all $t = 1, \ldots, n$, $d = 0, \ldots, D$ and $\omega \in U$ we solve the problem to find a part of the core of a cover and added items within the set $\{1, \ldots, t\}$ such that the violation of $x^*$ is maximized, the weight of the core equals $d$, their maximum weight is below $\omega$ and the weight of all items added in the extension are greater or equal than $\omega$. More formally we consider the following function

$$
\begin{aligned}
f_\omega(t,d) = \max \sum_{i=1}^{t}(x_i^* - 1)y_i + \sum_{i=1}^{t} x_i^* z_i & \\
\sum_{i=1}^{t} w_i y_i = d & \\
y_i + z_i \leq 1 & \qquad \forall i = 1, \ldots, t \\
z_i = 0 & \qquad \forall i : w_i < \omega \\
y_i = 0 & \qquad \forall i : w_i > \omega \\
y_i, z_i \in \{0,1\} & \qquad \forall i = 1, \ldots, t.
\end{aligned}
$$

The optimal solution to the separation problem is given by

$$\max_{\substack{\omega \in U \\ d \geq c+1}} f_\omega(n,d),$$

The function $f_\omega(1,d)$ can easily be solved via three case distinctions: (Case 1) if $w_1 = d$ and $\omega > d$, then $f_\omega(1,d) = (x_1^* - 1)$; (Case 2) if $d = 0$ and $w_1 \geq \omega$, then $f_\omega(1,d) = x_1^*$; (Case 3) otherwise, $f_\omega(1,d) = -\infty$.

For all other combinations of $t \geq 2$, $d = 0, \ldots, D$ and $\omega \in U$, the general recursive formula referring to the three cases above holds

$$
\begin{aligned}
f_\omega(t,d) = \max\{ & f_\omega(t-1,d), \\
& f_\omega(t-1, d-w_t) + (x_t^* - 1) && \text{if } w_t \leq \omega, \\
& f_\omega(t-1,d) + x_t^* && \text{if } w_t \geq \omega\}.
\end{aligned}
$$

In other words we decide, whether $t$ is not in the core and not added to the extension, is part of the core of the cover or is added to the extension taking into account the weight bound imposed by $\omega$. Obviously we can construct via an optimal solution of $f_\omega(t-1,d)$, $f_\omega(t-1,d-w_t)$, and $f_\omega(t-1,d)$ three feasible solutions for $f_\omega(t,d)$. Note that the recursion formula is valid, since otherwise we obtain a contradiction to the optimality of $f_\omega(1,d')$. The run-time of this approach is in $\mathcal{O}(D \cdot n^2)$, since $|U| \leq n$.

## 5 Computational Experiments

In this section we present computational results on the rrKP. First, we investigate the *gain of recovery*, i. e., the relative increase in the objective value of an optimal solution obtained by allowing recovery compared to the optimal value without recovery. Second, we study computationally the first closure of the relaxed recoverable robust knapsack problem with respect to the class of rrKP extended cover inequalities and different parameter settings.

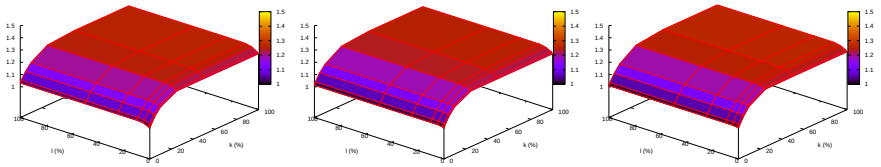
*Problem Instances.* The considered rrKP instances are slight modifications of multi-dimensional knapsack instances taken from the ORLIB [4] created by Chu and Beasley. The ORLIB provides instances with three different knapsack tightness ratios, where from we selected all instances with a medium tightness ratio of 0.5 for all combinations of $n$ and $m$, where $n \in \{100, 250, 500\}$ denotes the number of items and $m \in \{5, 10, 30\}$ the number of constraints. This yields 90 instances. For each rrKP instance, the first knapsack is treated as first stage constraint, and each remaining knapsack as individual discrete scenario. For each item, the profit of the corresponding multi-dimensional knapsack instance is scaled by 0.7 and used as first stage profit. The scenario profits are also determined by scaling these values but the scaling factor is uniformly random generated in $[0.2, 0.4]$. In order to analyze the impact of the recovery parameters $k$ and $\ell$, we chose them as fraction of the number of items, e. g., $k = 0.25$ means that 25% of the total set of items may be removed in each scenario. All combinations of $k, \ell \in \{0\%, 1\%, 5\%, 10\%, 25\%, 50\%, 100\%\}$ are tested.

*Setting.* We implemented the ILP formulation (1)-(9) of the recoverable robust knapsack problem in C++ using SCIP 1.2.0 [1] as branch-and-cut framework and IBM ILOG CPLEX 12.1 as underlying LP solver. The discussed valid inequalities are separated using the callback functionality. In our first study concerning the gain of recovery the separator is irrelevant.

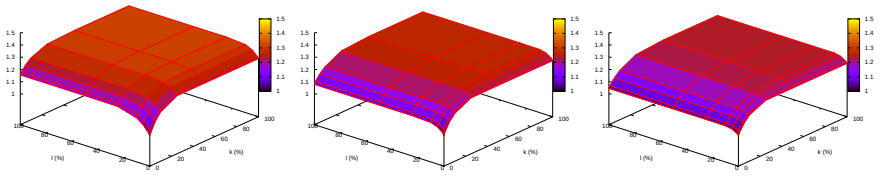
The computations were carried out on a Linux machine with 2.93 GHz Intel Xeon W3540 CPU and 12 GB RAM. A time limit of 1 hour was set for solving each problem instance. All other solver settings were left at their defaults.

### 5.1 Gain of recovery

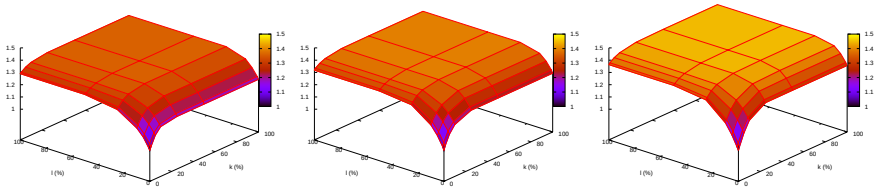
In our first computational study we investigate the gain of recovery. Therefore we limit the allowed recovery by the parameters $k$ and $\ell$ where $k$ ($\ell$) determines how many items chosen (not chosen) in the first stage may be removed (added) in the scenario. The setting $k = \ell = 0$ is the standard robust setting which is equivalent to the classical multi-dimensional knapsack problem. For given values of $k$ and $\ell$ we compare the objective value of the optimal solution (resp. best



(a) 4 scenarios, 100 items (b) 4 scenarios, 250 items (c) 4 scenarios, 500 items



(d) 9 scenarios, 100 items (e) 9 scenarios, 250 items (f) 9 scenarios, 500 items



(g) 29 scenarios, 100 items (h) 29 scenarios, 250 items (i) 29 scenarios, 500 items

**Fig. 1.** Gain of recovery for selected values of $k$ and $\ell$. Averages are shown for #scenarios=4, 9, 29 and #items=100, 250, 500. All values are normalized to $k = \ell = 0$.

known solution within the time limit as the achieved optimality gaps are very small) with the $k = \ell = 0$ setting to evaluate the gain of recovery.

Figure 1 visualizes the results of our study for selected values of $k$ and $\ell$. Only averages of the normalized values are shown for groups of 10 instances with the same number of items $n$ and the same number of scenarios $m$. All values are normalized to the corresponding $k = \ell = 0$ setting. Details can be found in Table 1.

Fixing the number of items we observe a rise in the gain of recovery when the number of scenarios increases (e. g., compare Fig. 1(a), 1(d), and 1(g)). This can be explained as the non-recovery solution of these instances is more conservative due to the higher number of scenarios. Therefore recovery allows a larger gain. Considering 4 (9, 29) scenarios an additional gain of 26% (31%, 45%) compared to the $k = \ell = 0$ setting can be achieved. Unfortunately, fixing the number of scenarios and varying the number of items does not give us a clear correlation.

| #scenarios | | 4 | | | 9 | | | 29 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | #items | 100 | 250 | 500 | 100 | 250 | 500 | 100 | 250 | 500 |
| $k$ | $\ell$ | | | | | | | | | |
| 0.00 | 0.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| | 0.01 | 1.00 | 1.00 | 1.00 | 1.01 | 1.01 | 1.01 | 1.02 | 1.02 | 1.02 |
| | 0.05 | 1.02 | 1.01 | 1.01 | 1.06 | 1.05 | 1.03 | 1.09 | 1.10 | 1.11 |
| | 0.10 | 1.03 | 1.01 | 1.01 | 1.10 | 1.07 | 1.04 | 1.15 | 1.17 | 1.19 |
| | 0.25 | 1.03 | 1.01 | 1.01 | 1.15 | 1.07 | 1.04 | 1.25 | 1.28 | 1.32 |
| | 0.50 | 1.03 | 1.01 | 1.01 | 1.15 | 1.07 | 1.04 | 1.28 | 1.31 | 1.36 |
| | 1.00 | 1.03 | 1.01 | 1.01 | 1.15 | 1.07 | 1.04 | 1.29 | 1.31 | 1.36 |
| 0.01 | 0.00 | 1.02 | 1.02 | 1.02 | 1.02 | 1.02 | 1.02 | 1.02 | 1.03 | 1.02 |
| | 0.01 | 1.02 | 1.03 | 1.02 | 1.04 | 1.04 | 1.03 | 1.04 | 1.05 | 1.05 |
| | 0.05 | 1.04 | 1.03 | 1.02 | 1.09 | 1.07 | 1.05 | 1.11 | 1.13 | 1.13 |
| | 0.10 | 1.04 | 1.03 | 1.02 | 1.12 | 1.09 | 1.06 | 1.17 | 1.19 | 1.21 |
| | 0.25 | 1.04 | 1.03 | 1.02 | 1.17 | 1.09 | 1.06 | 1.26 | 1.30 | 1.34 |
| | 0.50 | 1.04 | 1.03 | 1.02 | 1.17 | 1.09 | 1.06 | 1.29 | 1.32 | 1.37 |
| | 1.00 | 1.04 | 1.03 | 1.02 | 1.17 | 1.09 | 1.06 | 1.29 | 1.33 | 1.37 |
| 0.05 | 0.00 | 1.08 | 1.08 | 1.08 | 1.09 | 1.09 | 1.07 | 1.10 | 1.11 | 1.12 |
| | 0.01 | 1.09 | 1.09 | 1.08 | 1.11 | 1.10 | 1.08 | 1.12 | 1.13 | 1.14 |
| | 0.05 | 1.10 | 1.09 | 1.08 | 1.15 | 1.12 | 1.10 | 1.18 | 1.19 | 1.21 |
| | 0.10 | 1.10 | 1.09 | 1.08 | 1.18 | 1.13 | 1.10 | 1.23 | 1.25 | 1.28 |
| | 0.25 | 1.10 | 1.09 | 1.08 | 1.21 | 1.14 | 1.10 | 1.30 | 1.34 | 1.38 |
| | 0.50 | 1.10 | 1.09 | 1.08 | 1.21 | 1.14 | 1.10 | 1.32 | 1.35 | 1.40 |
| | 1.00 | 1.10 | 1.09 | 1.08 | 1.21 | 1.14 | 1.10 | 1.32 | 1.35 | 1.40 |
| 0.10 | 0.00 | 1.13 | 1.13 | 1.14 | 1.15 | 1.14 | 1.12 | 1.15 | 1.17 | 1.20 |
| | 0.01 | 1.14 | 1.14 | 1.14 | 1.16 | 1.15 | 1.13 | 1.17 | 1.19 | 1.22 |
| | 0.05 | 1.15 | 1.14 | 1.14 | 1.20 | 1.17 | 1.14 | 1.22 | 1.25 | 1.28 |
| | 0.10 | 1.15 | 1.14 | 1.14 | 1.23 | 1.18 | 1.15 | 1.27 | 1.29 | 1.33 |
| | 0.25 | 1.15 | 1.14 | 1.14 | 1.25 | 1.18 | 1.15 | 1.33 | 1.37 | 1.41 |
| | 0.50 | 1.15 | 1.14 | 1.14 | 1.25 | 1.18 | 1.15 | 1.34 | 1.37 | 1.42 |
| | 1.00 | 1.15 | 1.14 | 1.14 | 1.25 | 1.18 | 1.15 | 1.34 | 1.37 | 1.42 |
| 0.25 | 0.00 | 1.20 | 1.20 | 1.21 | 1.22 | 1.20 | 1.18 | 1.17 | 1.22 | 1.29 |
| | 0.01 | 1.21 | 1.21 | 1.22 | 1.23 | 1.22 | 1.19 | 1.19 | 1.25 | 1.31 |
| | 0.05 | 1.23 | 1.22 | 1.23 | 1.27 | 1.24 | 1.21 | 1.25 | 1.30 | 1.36 |
| | 0.10 | 1.23 | 1.22 | 1.23 | 1.29 | 1.24 | 1.21 | 1.31 | 1.35 | 1.41 |
| | 0.25 | 1.23 | 1.22 | 1.23 | 1.30 | 1.25 | 1.21 | 1.36 | 1.39 | 1.44 |
| | 0.50 | 1.23 | 1.22 | 1.23 | 1.30 | 1.25 | 1.21 | 1.36 | 1.39 | 1.44 |
| | 1.00 | 1.23 | 1.22 | 1.23 | 1.30 | 1.25 | 1.21 | 1.36 | 1.39 | 1.44 |
| 0.50 | 0.00 | 1.20 | 1.20 | 1.22 | 1.22 | 1.20 | 1.18 | 1.17 | 1.22 | 1.29 |
| | 0.01 | 1.21 | 1.21 | 1.23 | 1.24 | 1.22 | 1.19 | 1.19 | 1.25 | 1.31 |
| | 0.05 | 1.24 | 1.23 | 1.25 | 1.27 | 1.25 | 1.22 | 1.25 | 1.30 | 1.37 |
| | 0.10 | 1.25 | 1.24 | 1.26 | 1.30 | 1.26 | 1.23 | 1.31 | 1.35 | 1.41 |
| | 0.25 | 1.25 | 1.25 | 1.26 | 1.31 | 1.26 | 1.23 | 1.36 | 1.39 | 1.45 |
| | 0.50 | 1.25 | 1.25 | 1.26 | 1.31 | 1.26 | 1.23 | 1.36 | 1.39 | 1.45 |
| | 1.00 | 1.25 | 1.25 | 1.26 | 1.31 | 1.26 | 1.23 | 1.36 | 1.39 | 1.45 |
| 1.00 | 0.00 | 1.20 | 1.20 | 1.22 | 1.22 | 1.20 | 1.18 | 1.17 | 1.22 | 1.29 |
| | 0.01 | 1.21 | 1.21 | 1.23 | 1.23 | 1.22 | 1.19 | 1.19 | 1.25 | 1.31 |
| | 0.05 | 1.24 | 1.23 | 1.25 | 1.27 | 1.25 | 1.22 | 1.25 | 1.31 | 1.37 |
| | 0.10 | 1.25 | 1.24 | 1.26 | 1.30 | 1.26 | 1.23 | 1.31 | 1.35 | 1.41 |
| | 0.25 | 1.25 | 1.25 | 1.26 | 1.31 | 1.26 | 1.23 | 1.36 | 1.39 | 1.45 |
| | 0.50 | 1.25 | 1.25 | 1.26 | 1.31 | 1.26 | 1.23 | 1.36 | 1.39 | 1.45 |
| | 1.00 | 1.25 | 1.25 | 1.26 | 1.31 | 1.26 | 1.23 | 1.36 | 1.39 | 1.45 |

**Table 1.** Gain of recovery for selected values of $k$ and $\ell$. Averages are shown for #scenarios=4, 9, 29 and #items=100, 250, 500. All values are normalized to $k = \ell = 0$.

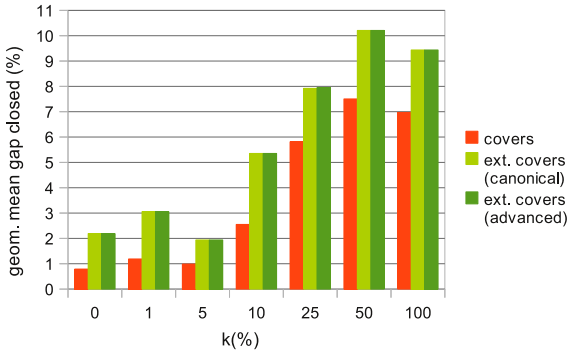



**Fig. 2.** Integrality Gap closed (%) by separating violated valid inequalities for rrKP

For the 100 (250, 500) item instances we can achieve an additional gain up to 36% (39%, 45%). Next, we observe that increasing the parameter $k$, i. e., allowing more items to be removed in each scenario, clearly leads to an increase in the gain of recovery. This is plausible as large items violating a scenario knapsack constraint may more likely be removed. Finally, we note that the impact of increasing $\ell$ grows with the number of scenarios.

In summary, allowing recovery yields a gain of up to 45%. Even a more restricted setting as $k = \ell = 0.1$ gives a gain of recovery in the range from 15% to 33%.

### 5.2 First rrKP extended cover closure

In the following we present the results of our second experiment where we investigate the effectiveness of rrKP extended cover inequalities. Therefore, we implemented the ILP formulation (13)-(17) of the corresponding separation problem to separate violated rrKP extended cover inequalities exactly (but still with an overall 1 hour time limit per instance). Whenever our separator is called the first stage knapsack is checked. If no violated extended cover is found, all scenarios are tested beginning with the last scenario which provided a violated cut, until a violation is determined. This inequality is then added to the LP and the separation round is aborted. Only the root node of the recoverable robust knapsack problem is solved in this study. We consider four settings which differ by the integer program solved (i.e., whether the canonical extension is integrated or not) and how the solution is strengthened in a post processing step (i.e., items with $x_j^* = 0$ are not necessarily selected in the separation ILP): (i) standard ILP solver, (ii) cover inequalities with canonical greedy extension, (iii) extended cover inequalities with canonical greedy extension, and (iv) extended cover inequalities with advanced greedy extension.

In Figure 2 the average gap closed is shown for selected values of $k$ (violated rrKP extended cover inequalities do not depend on $\ell$) and settings (ii) to (iv) w. r. t. setting (i). Details of the reported results can be found in [8].

The results for setting (ii) are in line with those for the classical knapsack problem [12]. We observe the separation of violated extended cover inequalities always closes the integrality gap more than covers only. The strengthening of the canonical extension (11) (setting (ii)) to the advanced extension given in Definition 6 (setting (iv)) does not have a strong impact on the gap closed, e. g., 8.0% (iv) compared to 7.9% (iii) on average for $k = 25\%$.

Focusing on the results for the canonical extension (setting (ii)), we observe that the geometric mean of the integrality gap closed lies in the range from 1.5% ($k = 5\%$) to 10.3% ($k = 50\%$). Considering all instances with 4 (9, 29) scenarios it ranges from 0.1% (0.1%, 0.8%) to 8.6% (10.2%, 23.9%). This suggests that an increase in the number of scenarios may result in a larger gap closed. Unfortunately, there is no clear dependency. Considering all instances with 100 (250, 500) items the integrality gap closed ranges from 0.8% (1.3%, 0.1%) to 13.3% (16.0%, 23.9%). Again, there is no clear trend when fixing $k$ or $m$.

In summary, this study shows that by adding violated rrKP extended cover inequalities the integrality gap is always lowered. The best achievement has been a gap closed by 23.9%. In setting (ii) (setting (iii)) the gap could be closed by more than 5% in 35% (55%) of all instances. In addition, the overall computation time spent for one setting of this study was less than half an hour. Hence, the separation of violated rrKP extended cover cuts has a high potential to tighten the linear relaxation of the recoverable robust knapsack problem and to speed-up the solving process significantly.

## 6 Conclusion

In this paper, we considered the recoverable robust knapsack problem with discrete scenarios. For a fixed number of discrete scenarios $(k, \ell)$-rrKP is weakly **NP**-complete. If the number of discrete scenarios is part of the input, the problem is strongly **NP**-complete. We introduced the class of rrKP cover inequalities generalizing its well-known counterpart for the knapsack problem. In addition to a canonical extension we presented a stronger extension exploiting the scenario-based structure of the problem. This second extension is still stronger if restricted to the robust knapsack polytope.

Computational experiments on several thousand individual test runs have shown that a gain of up to 45% in the objective can be achieved by recovery. Further, computations are sped up by separating either canonical extended cover inequalities or strengthened canonical extended cover inequalities.

In future work, alternative separation algorithms (e. g. heuristics) should be considered. The strength of the valid inequalities should be polyhedrally investigated as well.

## References


1. T. Achterberg. SCIP: solving constraint integer programs. *Math. Prog. Comp.*, 1(1):1–42, 2009.
2. H. Aissi, C. Bazgan, and D. Vanderpooten. Min-max and min-max regret versions of some combinatorial optimization problems: a survey. 2007. `http://hal.archives-ouvertes.fr/docs/00/15/86/52/PDF/AN7LAMSADE_1-32.pdf`.
3. E. Balas. Facets of the knapsack polytope. *Math. Prog.*, 8:146–164, 1975.
4. J.E. Beasley. OR-Library: distributing test problems by electronic mail. *J. of the OR Society*, 41:1069–1072, 1990. `http://people.brunel.ac.uk/~mastjjb/jeb/`.
5. R. Bellman. Notes on the theory of dynamic programming iv - maximization over discrete sets. *Naval Research Logistics Quarterly*, 3:67–70, 1956.
6. D. Bertimas and M. Sim. Robust discrete optimization and network flows. *Math. Prog., Series B*, 98:49–71, 2003.
7. D. Bertimas and M. Sim. The Price of Robustness. *Oper. Res.*, 52:35–53, 2004.
8. C. Büsing and A.M.C.A. Koster and M. Kutschka. Recoverable Robust Knapsacks: the Discrete Scenario Case. Technical Report 018-2010, TU Berlin, 2010. `http://www.math.tu-berlin.de/coga/publications/techreports/2010/`.
9. H. Crowder, E. Johnson, and M. Padberg. Large-scale zero-one linear programming problems. *Oper. Res.*, 31:803–834, 1983.
10. S. Cicerone, G. D'Angelo, G. Di Stefano, D. Frigioni, A. Navarra, M. Schachtebeck, and A. Schöbel. Recoverable robustness in shunting and timetabling. *Robust and Online Large-Scale Optimization*, vol. 5868 of *LNCS*, 28–60. Springer, Berlin, 2009.
11. M. Fischetti, A. Lodi, and D. Salvagnin. Just MIP it! *Annals of Information Systems 10*, 39–70, 2009.
12. V. Gabrel and M. Minoux. A scheme for exact separation of extended cover inequalities and application to multidimensional knapsack problems. *OR Letters*, 30:252–264, 2002.
13. H. Iida. A note on the max-min 0-1 knapsack problem. *J. of Comb. Opt.*, 3:89–94, 1999.
14. R. Kalai and D. Vanderpooten. Lexicographic $\alpha$-robust knapsack problems: complexity results. *IEEE International Conference on Services Systems and Service Managment*, 1103 – 1107, 2006.
15. K. Kaparis and A. Letchford. Separation algorithms for 0-1 knapsack polytopes. *Math. Prog.*, 124(1-2):69–91, 2010.
16. R. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, 85–103, 1972.
17. H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems.* Springer, 2004.
18. O. Klopfenstein and D. Nace. Valid inequalities for a robust knapsack polyhedron - Application to the robust bandwidth packing problem. *Networks*, to appear (2010).
19. C. Liebchen, M. E. Lübbecke, R. H. Möhring, and S. Stiller. The concept of recoverable robustness, linear programming recovery, and railway applications. *Robust and Online Large-Scale Optimization*, vol. 5868 of *LNCS*, 1–27. Springer, 2009.
20. S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations.* Wiley, 1990.
21. R. Weismantel. On the 0-1 knapsack polytope. *Math. Prog.*, 77:49–68, 1997.
22. L. Wolsey. Faces for a linear inequality in 0-1 variables. *Math. Prog.*, 8:165–178, 1975.
23. G. Yu. On the max-min 0-1 knapsack problem with robust optimization applications. *Oper. Res.*, 44:407–415, 1996.