On a Cardinality Constrained Multicriteria Knapsack Problem*

Florian Seipp, Stefan Ruzika, and Luís Paquete

January 25, 2011

Abstract

We consider a variant of a knapsack problem with a fixed cardinality constraint. There are three objective functions to be optimized: one real-valued and two integer-valued objectives. We show that this problem can be solved efficiently by a local search. The algorithm utilizes connectedness of a subset of feasible solutions and has optimal run-time.

1 Introduction

The cardinality constrained multi-dimensional knapsack problem is an extension of the classical binary knapsack problem, for which there is only a fixed number of items which fit into a knapsack with multiple capacity constraints [3, 7].
In this manuscript, we consider the multicriteria version of this problem – the *cardinality constrained multicriteria knapsack problem* – which is obtained by transforming the capacity constraints into minimization criteria. In the case of real-valued objective function coefficients, this problem is NP-hard and it is intractable, that is, the image of the efficient set is of exponential size with respect to the instance size [2]. It is of particular interest to find special cases for an arbitrary fixed cardinality that can be solvable in polynomial time.

More precisely, we study the following variant of a cardinality constrained multicriteria knapsack problem in this manuscript:

*This research was supported by the project "Connectedness and Local Search for Multiobjective Combinatorial Optimization" founded by DAAD - Deutscher Akademischer Austausch Dienst and Conselho de Reitores das Universidades Portuguesas.

$$
\begin{aligned}
\max p(x) &= \sum_{i=1}^{n} p_i x_i \\
\min w^1(x) &= \sum_{i=1}^{n} w_i^1 x_i \\
\min w^2(x) &= \sum_{i=1}^{n} w_i^2 x_i \\
\text{subject to } & \sum_{i=1}^{n} x_i = k \\
& x \in \{0,1\}^n
\end{aligned}
\tag{CCMKP}
$$

where $k \in \mathbb{N}$, $w^1, w^2 \in \{0,1\}^n$ and $p \in \mathbb{R}^n_+$ are parameter values given by the input of the instance. We refer to the set

$$
X \coloneqq \left\{ x \in \{0,1\}^n \colon \sum_{i=1}^{n} x_i = k \right\}
$$

as the set of *feasible solutions* and to the set

$$
Y \coloneqq \left\{ \big(p(x), w^1(x), w^2(x)\big) \in \mathbb{R} \times \mathbb{Z}^2 \colon x \in X \right\}
$$

as the set of *feasible images*.

This problem is referred to as (CCMKP) and can be interpreted as choosing exactly k out of a set $I \coloneqq \{1, \dots, n\}$ of n items. The decision variables are binary, and $x_i = 1$ if item i is packed into the knapsack and 0 otherwise. Each item has assigned an associate (real-valued) positive profit and two kinds of binary weights. Three objective functions have to be considered. The first objective function $p \colon X \to \mathbb{R}_+$ can be interpreted as maximizing the total additive profit of the knapsack. The second and third objectives $w \colon X \to \mathbb{Z}^2$ are to minimize a binary additive weight function.

Since three objectives are to be optimized simultaneously, the notion of optimality must be specified. Following the literature (e.g. [2]), we employ the componentwise ordering to define a preference relation on the set of solutions.

Definition 1. A feasible solution x of (CCMKP) is said to *dominate* another feasible solution $\hat{x}$ if and only if $p(x) \geq p(\hat{x})$, $w^1(x) \leq w^1(\hat{x})$, and $w^2(x) \leq w^2(\hat{x})$ with strict inequality in at least one of these three relations. If there exists no feasible solution dominating x, then x is called *efficient*. The corresponding outcome $\big(p(x), w^1(x), w^2(x)\big) \in \mathbb{R} \times \mathbb{Z}^2$ of an efficient solution x is called *nondominated*. The sets $X_E \subseteq X$ and $Y_N \subseteq Y$ refer to the set of efficient solutions and nondominated points, respectively.

In this manuscript, we are going to show that the set of nondominated points of Problem (CCMKP) can be found efficiently by a local search on the weight space. While the set of efficient solutions of various general multicriteria optimization problems is typically not connected (cf. [4]), this variant of the multicriteria knapsack problem is one of the rare instances possessing the connectedness property. Our algorithm exploits connectedness of a superset of the

efficient solutions of the problem (see [4]), it filters dominated solutions and it is asymptotically optimal.
We remark that – to the best of our knowledge – this is the first local search algorithm to solve a multicriteria combinatorial optimization problem in an exact manner. Our work can be considered an extension of the work of Gorski et al. [5] who suggested a greedy algorithm for an unconstrained version of this problem.

This article is organized as follows. In Section 2, structural properties of the problem (CCMKP) are analyzed. These findings are employed in a solution algorithm which is described in Section 3. The article is concluded in Section 4.

2 Structural Properties

In this section, some properties of the set of feasible and efficient solutions are derived. The following simple observations are frequently used later on.

Observation 2. For the three objective problem (CCMKP), the following statements hold.

1. There are $\binom{n}{k}$ feasible solutions which may all be efficient.
2. Feasible solutions have integer weight values $0 \leq w^1(x), w^2(x) \leq k$.
3. There are at most $(k+1)^2$ nondominated solutions, one for each of the possible weight coordinates.
4. The set of items I can be partitioned into four sets:
   - $\mathcal{O} \coloneqq \{i \in I : w_i^1 = w_i^2 = 0\}$
   - $\mathcal{R} \coloneqq \{i \in I : w_i^1 = 1, w_i^2 = 0\}$
   - $\mathcal{U} \coloneqq \{i \in I : w_i^1 = 0, w_i^2 = 1\}$
   - $\mathcal{D} \coloneqq \{i \in I : w_i^1 = w_i^2 = 1\}$
5. There are at most k elements of each of the sets $\mathcal{O}$, $\mathcal{R}$, $\mathcal{U}$, and $\mathcal{D}$ necessary in order to compute all nondominated solutions (cf. Lemma 3)

Notation. As indicated in the previous observation, the projection of the feasible images Y to the w^1-w^2-plane, i.e.,

$$\{(w^1(x), w^2(x)) \in \mathbb{Z}^2 \colon x \in X\} \subseteq \{0, \ldots, k\} \times \{0, \ldots, k\}$$

plays an important role in the following considerations. We refer to this set as the *set of feasible weights* and to the set $\{0, \ldots, k\} \times \{0, \ldots, k\}$ as the *weight space* or *weight grid*. For some $x \in X$, we refer to the point $(w^1(x), w^2(x))$ as the *weight of the feasible solution* x. Moreover, we refer to a point $(a, b) \in \mathbb{Z}^2$ as a *feasible weight* if it is the weight of some feasible solution $x \in X$.

Note that there might be exponentially many feasible and efficient solutions but only a polynomially bounded number of nondominated outcomes (cf. Example 19). The task of solving (CCMKP) is therefore understood as computing a *minimal complete set of efficient solutions*, i.e., finding all nondominated points $y \in Y_N$ and for each nondominated point $y \in Y_N$ one corresponding efficient solution $x \in X_E$.

Lemma 3. *Let the set of items I be partitioned into the aforementioned sets $\mathcal{O}$, $\mathcal{R}$, $\mathcal{U}$, and $\mathcal{D}$, and let $O \subseteq \mathcal{O}$, $R \subseteq \mathcal{R}$, $U \subseteq \mathcal{U}$, and $D \subseteq \mathcal{D}$ be subsets ordered with respect to non-increasing profit, each only consisting of k most profitable items. Then, for each nondominated point, there is a corresponding solution only consisting of items in O, R, U, and D.*

Lemma 3 suggests to categorize all items with respect to these four sets defined by the two weight objectives and to sort each of these sets with respect to decreasing profit. Our algorithm is going to use these four sets and investigate the projections of feasible outcomes to the weight space w^1-w^2 (see Figure 1).

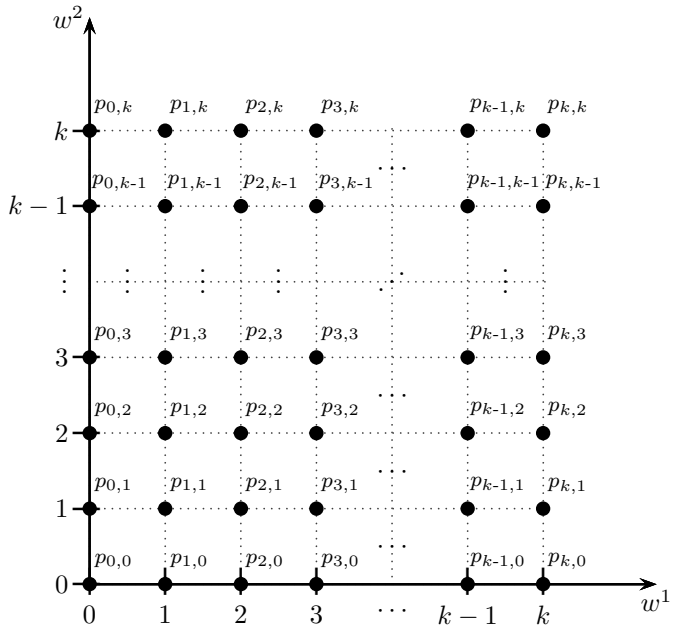

Figure 1: Grid of feasible solutions in the weight space w^1-w^2.

Figure 1 depicts all possible w^1-w^2-weights. For each feasible weight, the solution yielding the largest profit is a potentially efficient solution.

Definition 4. Let $p_{i,j}$ denote the largest possible profit of all feasible solutions with weight (i, j), i.e.,

$$p_{i,j} \coloneqq \max_{\substack{x \in X, \\ w^1(x) = i, \\ w^2(x) = j}} p(x).$$

Then, a corresponding solution $x \in X$ attaining $p_{i,j}$ is called *viable*. The set $X_V \subseteq X$ refers to the set of all viable solutions.

Since the inclusion chain $X_E \subseteq X_V \subseteq X$ holds, it is sufficient to compute X_V (more precisely, a complete set of viable solutions) and then apply some dominance check on the set X_V filtering a minimal complete set of efficient solutions.

In the following we describe an algorithm that explores some structural results of the efficient set, which is going to be proved in this section:

- starting in the origin, the algorithm visits every point in the w_1-w_2-space and computes a viable solution if one exists (the treatment of infeasible solutions is described in Section 3.2)
- the algorithm keeps only track of viable solutions having the maximal number of items from the set D referred to as *diagonal elements* or *diagonal items*
- when following this aforesaid convention it holds that any solution uses at least as many diagonal elements as the solution below or to the left of it on the weight grid
- any solution uses at most one additional diagonal element compared to the neighbors to the left and below on the weight grid

Observation 5. Note that the weight of a solution is completely determined by the composition of its k items. Let $x \in X$ with $w(x) = (a, b)$ and denote by o, r, u, and d the number of items of type O, R, U, and D, respectively, i.e.,

$$o \coloneqq |\{i \in I : x_i = 1\} \cap O| \qquad r \coloneqq |\{i \in I : x_i = 1\} \cap R|$$
$$u \coloneqq |\{i \in I : x_i = 1\} \cap U| \qquad d \coloneqq |\{i \in I : x_i = 1\} \cap D|.$$

Then it holds

$$\begin{aligned} k &= o + r + u + d \\ a &= r + d \\ b &= u + d. \end{aligned}$$

Obviously, taking the first items out of the four sorted lists yields a solution with largest possible profit for a given item composition and therefore a natural candidate for a viable solution.

Moreover, when looking for a solution with weight $w(x) = (a, b)$, fixing the number d of diagonal elements predetermines the numbers of items to be chosen from the three remaining sets.

Observation 6. Let (a, b) be weight point, $a, b \in \{0, \ldots, k\}$, and $d \in \{0, \ldots, k\}$ be a number of items that need to be chosen from D. Since only elements of D and R influence the first weight component, we need to choose exactly $a - d$ elements from R in order to obtain the desired w_1-weight. With a similar argument, the number of elements chosen from U has to be $b - d$. The remaining $k - (d + a - d + b - d)$ items are chosen from O.
In case that any of the numbers d, $a - d$, $b - d$, or $k - a - b + d$ are negative or exceed the cardinality of the sets D, R, U, and O, respectively, there is no feasible solution for weight (a, b) containing exactly d items of weight $(1, 1)$.

Definition 7 (Neighborhood)**.** Two feasible solutions $x, z \in X$ are called *neighbors* if they have $k - 1$ items in common. The process of replacing an item by another in a feasible solution is referred to as *(neighborhood) swap*.

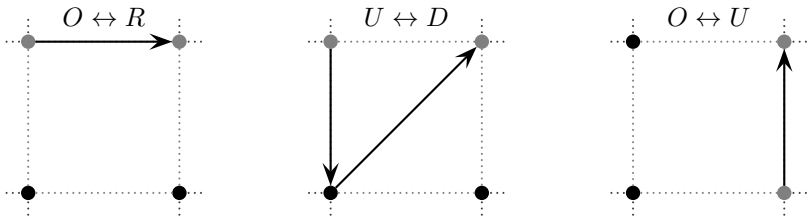

Figure 2: Visualization of important neighborhood swaps in the weight space.

Figure 2 visualizes three of the 16 possible neighborhood swaps in the weight space. These three swaps are going to be the only ones of importance in our neighborhood search algorithm.

The following two theorems and their proofs give an insight into the relation between viable solutions and neighborhood swaps.

Theorem 8. *Let x_1^* be a viable solution for weight (a, b) and let $(a + \ell, b)$, $\ell \in \{1, \ldots, k-a\}$, be another weight of a feasible solution x_2. Then the following holds:*

(i) There is a viable solution x_2^* for the weight $(a + \ell, b)$ which uses at least as many diagonal elements as x_1^*.

(ii) All intermediate weights $(a + j, b)$ with $j \in \{1, \ldots, \ell\}$ are feasible.

Proof. For (i):
Suppose all viable solutions for the weight $(a+\ell, b)$ use less diagonal items than x_1^*. Let x_2^* be the solution attaining profit $p_{a+\ell,b}$ which uses the most diagonal items, and, for $i = 1, 2$, let d_i, r_i, u_i, and o_i be the number of items of the sets D, R, U, and O included in x_i^*, respectively. The idea of the proof is to derive a contradiction by showing that replacing an R-item and a U-item in x_2^* by a D-item and an O-item leads to a feasible solution with the same weight and a larger profit value.
Since x_1^* and x_2^* are feasible solutions for the weights (a, b) and $(a + \ell, b)$, respectively, it holds:

$$\begin{aligned} k &= d_i + r_i + u_i + o_i, && i = 1, 2 \\ b &= d_i + u_i, && i = 1, 2 \\ a &= d_1 + r_1, \\ a + \ell &= d_2 + r_2. \end{aligned}$$

Using the assumption $\delta \coloneqq d_1 - d_2 > 0$, it follows:

$$\begin{aligned} u_2 &= u_1 + \delta > u_1, \\ r_2 &= r_1 + \delta + \ell > r_1, \\ o_2 &= k - (d_1 - \delta) - (r_1 + \delta + \ell) - (u_1 + \delta) \\ &= o_1 - \ell - \delta < o_1. \end{aligned}$$

Hence, we can replace a D-item and an O-item in x_1^* by an R-item and a U-item, obtaining a feasible solution $\hat{x}_1$ with weight (a, b):

$$\begin{aligned} w(\hat{x}_1) &= w(x_1^*) - (1,1) - (0,0) + (1,0) + (0,1) \\ &= w(x_1^*) = (a, b) \end{aligned}$$

Since we assumed x_1^* to be viable, we have

$$\begin{aligned} p_{a,b} = p(x_1^*) &= \sum_{i=1}^{d_1} p(D_i) + \sum_{i=1}^{r_1} p(R_i) + \sum_{i=1}^{u_1} p(U_i) + \sum_{i=1}^{o_1} p(O_i) \\ \geq p(\hat{x}_1) &= \sum_{i=1}^{d_1-1} p(D_i) + \sum_{i=1}^{r_1+1} p(R_i) + \sum_{i=1}^{u_1+1} p(U_i) + \sum_{i=1}^{o_1-1} p(O_i) \\ \Leftrightarrow p(D_{d_1}) + p(O_{o_1}) &\geq p(R_{r_1+1}) + p(U_{u_1+1}) \end{aligned} \tag{1}$$

Using the above argument in the opposite direction we can also obtain a feasible solution by replacing an R-item and a U-item in x_2^* by a D-item and an O-item. Again, the resulting solution, say $\hat{x}_2$, has the same weight as x_2^*. Recall that by assumption x_2^* uses the most D-items among all solutions with profit $p_{a+\ell,b}$. Hence, it is

$$\begin{aligned} p_{a+\ell,b} = p(x_2^*) &= \sum_{i=1}^{d_2} p(D_i) + \sum_{i=1}^{r_2} p(R_i) + \sum_{i=1}^{u_2} p(U_i) + \sum_{i=1}^{o_2} p(O_i) \\ > p(\hat{x}_2) &= \sum_{i=1}^{d_2+1} p(D_i) + \sum_{i=1}^{r_2-1} p(R_i) + \sum_{i=1}^{u_2-1} p(U_i) + \sum_{i=1}^{o_2+1} p(O_i) \\ \Leftrightarrow p(R_{r_2}) + p(U_{u_2}) &> p(D_{d_2+1}) + p(O_{o_2+1}). \end{aligned} \tag{2}$$

On the other hand,

$$\begin{aligned} p(D_{d_2+1}) + p(O_{o_2+1}) &\geq p(D_{d_1}) + p(O_{o_1}) \\ &\geq p(R_{r_1+1}) + p(U_{u_1+1}) \\ &\geq p(R_{r_2}) + p(U_{u_2}), \end{aligned}$$

where the second inequality is obtained from (1), and the first and last inequality are a consequence of the ordering of the item lists, using

$$d_2 + 1 \leq d_1, \quad o_2 + 1 \leq o_1, \quad r_1 + 1 \leq r_2, \quad u_1 + 1 \leq u_2.$$

This is a contradiction to Equation (2), hence x_2^* uses at least d_1 diagonal items.

For (ii):
From part (i) we know the feasible solutions x_1^* for weight (a, b) and x_2^* for

$(a+\ell, b)$, with

$$\begin{aligned} k &= d_i + r_i + u_i + o_i, && i = 1,2 \\ b &= d_i + u_i, && i = 1,2 \\ a &= d_1 + r_1, \\ a + \ell &= d_2 + r_2, \end{aligned}$$

$$\begin{aligned} d_2 &= d_1 + \delta, \quad \delta \in \mathbb{N}, \\ u_2 &= u_1 - \delta, \\ r_2 &= r_1 - \delta + \ell, \\ o_2 &= o_1 - \ell + \delta. \end{aligned}$$

Now, choose $j \in \{1, \ldots, \ell\}$ arbitrarily.
Case 1: $j \leq \delta$.
Then, define a solution by choosing $d_1 + j$ items from D, $u_1 - j$ items from U, r_1 items from R, and o_1 items from O. This obviously describes a feasible solution, say $\bar{x}$, since $d_1 \leq d_1 + j \leq d_2$, $u_2 \leq u_1 - j \leq u_1$ and $k = d_1 + j + u_1 - j + r_1 + o_1$. It holds

$$\begin{aligned} w(\bar{x}) &= (d_1 + j)\,(1,1) + (u_1 - j)\,(0,1) + r_1(1,0) + o_1(0,0) \\ &= (d_1 + j + r_1, d_1 + j + u_1 - j) \\ &= (a + j, b)\,. \end{aligned}$$

Case 2: $j > \delta$.
Observe that in this case also $\ell > \delta$. We define a solution by choosing d_2 items from D, u_2 items from U, $r_1 - \delta + j$ items from R and $o_1 - j + \delta$ items from O. Again, this describes a feasible solution $\bar{x}$, since $r_1 \leq r_1 - \delta + j \leq r_2$, $o_2 \leq o_1 - j + \delta \leq o_1$ and $d_2 + u_2 + r_1 - \delta + j + o_1 - j + \delta = d_2 + u_2 + r_2 + o_2 = k$. The weight of $\bar{x}$ is

$$\begin{aligned} w(\bar{x}) &= (d_2 + r_1 - \delta + j, d_2 + u_2) \\ &= (d_1 + r_1 + j, d_2 + u_2) \\ &= (a + j, b). \end{aligned}$$

□

Theorem 9. *Let x_1^* be the viable solution which uses the most D-items for $w(x_1^*) = (a, b)$. If $(a+1, b)$ is a feasible weight, then none of its viable solutions uses more than one additional D-item compared to x_1^*.*

Proof. Let x_2^* be the viable solution that uses the most D-items for $(a + 1, b)$. We denote by r_i, u_i, d_i, and o_i the number of R-, U-, D-, and O-items used in x_i^*, $i = 1, 2$.
Due to Theorem 8 part(i) it holds that $d_2 \geq d_1$. Now suppose that $d_2 > d_1 + 1$. Then the following equations hold:

$$\begin{aligned} k &= d_i + r_i + u_i + o_i, && i = 1,2 \\ b &= d_i + u_i, && i = 1,2 \\ a &= d_1 + r_1, \\ a + 1 &= d_2 + r_2. \end{aligned}$$

Plugging in the assumption $\delta \coloneqq d_2 - d_1 > 1$, we get

$$\begin{aligned} u_1 &= u_2 + \delta > u_2 + 1, \\ r_1 &= r_2 + \delta - 1 > r_2, \\ o_1 &= k - (r_2 + \delta - 1) - (u_2 + \delta) - (d_2 - \delta) \\ &= o_2 - \delta + 1 < o_2. \end{aligned}$$

Hence, we can obtain a feasible solution $\hat{x}_2$ of weight $(a+1, b)$ by replacing a D-item and an O-item in x_2^* by an R-item and a U-item. Since x_2^* has the highest possible profit for weight $(a+1, b)$ it holds

$$p(D_{d_2}) + p(O_{o_2}) \geq p(R_{r_2+1}) + p(U_{u_2+1}). \tag{3}$$

On the other hand, we also obtain a feasible solution $\hat{x}_1$ with weight (a, b) by replacing an R-item and a U-item in x_1^* with a D-item and an O-item. By the ordering according to profits in the lists and Equation (3) we get

$$\begin{aligned} p(D_{d_1+1}) + p(O_{o_1+1}) &\geq p(D_{d_2}) + p(O_{o_2}) \\ &\geq p(R_{r_2+1}) + p(U_{u_2+1}) \\ &\geq p(R_{r_1}) + p(U_{u_1}) \end{aligned}$$

This is a contradiction to the assumption of x_1^* being the viable solution for (a, b) that uses the most D-items. □

In cases in which several neighbor swaps lead to the same profit, a tie-breaking rule is necessary to decide which one to take. Theorem 8 justifies the storage of a single item (in the process of a neighborhood search) by preferring swaps with diagonal items in the case of ties. In addition, Example 10 demonstrates that we may not obtain correct profit values or miss feasible points if we decide otherwise.

Example 10. Consider the instance given by $k = 3$ with items already partitioned and sorted into sets $O = [10, 0, 0]$, $R = [6, 5, 4]$, $U = [4, 3, 2]$, and $D = [10, 8, 6]$.

The solutions on the axes of the w^1-w^2-plane are unique. Together with their profit values, they are shown in Figure 3. Note that the solution in the origin is composed of O-items only while the item composition of all other solutions can be obtained by swaps as indicated in the picture.

For the point with weight coordinates $(1, 1)$ there are two possible compositions of viable solutions with profit 20, and they both can be obtained by item exchanges with neighboring solutions: a swap with a U-item in $(1, 0)$ and a swap with a D-item in $(0, 0)$ (cf. Figure 4). The first alternative (and the resulting consequences) is depicted in the left column of Figure 4, while the second is shown in the right column.

The decision for the one or the other composition does not influence the profit of the neighboring grid points (2,1) and (1,2) (although e.g. moving to the right from (1,1) to (2,1) swaps different O- and R-items in the two cases) as illustrated in Figure 5.

However, the profit of the point (2,2) is affected, and this nondominated point could be mistaken for a dominated point if taking the first variant (see Figure 6). By choosing the first variant for (1,1), a feasible solution for (2,2) is not

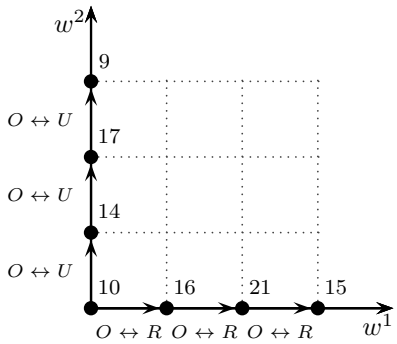

Figure 3: Unique feasible weights on the axes in the weight space.

considered: the one consisting of two D-items and one O-item. This solution is also viable.

Another undesired phenomenon occurs subsequently concerning the point (3,3) (see Figure 6): if the first variant is chosen, this point is considered infeasible as none of the potential predecessors contains any O-items.

$\triangleleft$

Due to symmetry of the two weight functions, Theorem 8 and 9 hold analogously for comparing two solutions located on a vertical line instead of a horizontal line in the grid of weights. They are the foundation of an exact neighborhood search algorithm as they suggest to only keep track of viable solutions having the most diagonal items. Suppose that we have already collected all such viable solutions located on the horizontal line b in the weight grid up to solution x_1^* for point (a, b). Then we know that a viable solution of interest x_2^* for $(a + 1, b)$ uses either the same number of diagonal elements or one additional than x_1^*. In both cases x_2^* can be obtained from x_1^* by a single neighborhood swap.

3 An Algorithm for the CCMKP

The idea of our algorithm is to use the structural results in order to explore the grid of possible weights row by row. Starting from the origin, we detect the first, i.e., the leftmost feasible point in a row, store a corresponding viable solution containing as many diagonal items as possible, and then iteratively move to the right on the weight grid until we reach the last feasible point in the current row.

We have to distinguish two major situations in the following. In the simple case, all of the $(k + 1)^2$ weights are feasible, hence we need only worry about how to obtain the desired viable solutions. In the more challenging situation, there exist some weights without corresponding feasible solution. Since our algorithm might have to visit these points on the way to viable solutions, this can cause difficulties. A decision for a neighborhood swap simply based on the higher profit can result in the algorithm missing a nondominated solution. We are going to come up with a modification ensuring that all feasible solutions are found and allowing us to treat both cases equally.

Lemma 11. *All $(k + 1)^2$ possible weights are feasible if and only if the item set I contains at least k items of each weight type.*

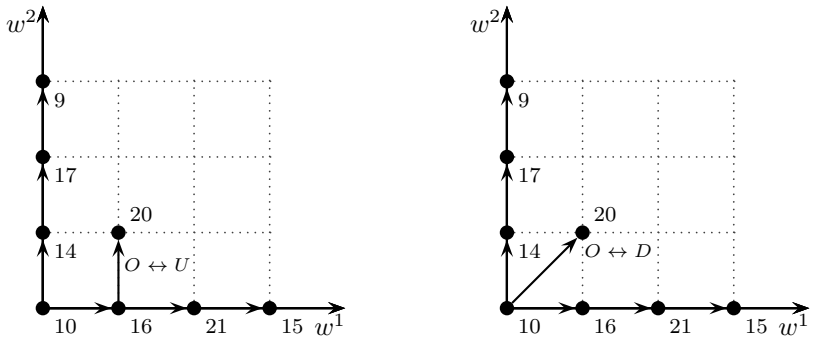

Figure 4: Two different swaps generating the nondominated point $(1, 1)$.

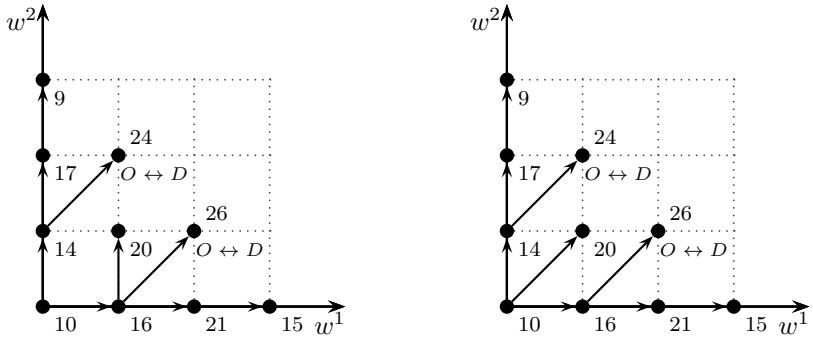

Figure 5: No influence on the neighbors to the right and above (1,1).

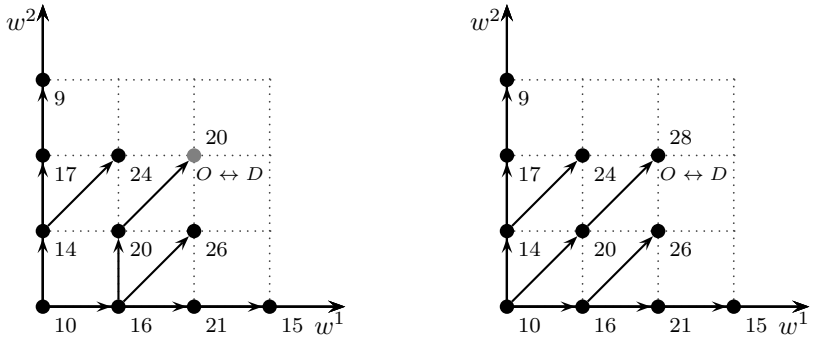

Figure 6: Two different profits for the point $(2, 2)$.

Proof. For sufficiency, let us first assume that after the initialization it is $|O| = |R| = |U| = |D| = k$. Consider an arbitrary grid point $(a, b) \in \{0, \dots, k\}^2$. Since the weight functions are interchangeable, we may assume without loss of generality $a \geq b$. Then, choosing $a - b$ items from R, b items from D and $k - a$ items from O results in a feasible solution x with weight

$$w(x) = (a - b)(1, 0) + b(1, 1) = (a, b).$$

For the converse, assume that there are less than k items of weight $(i, j) \in \{0, 1\}^2$ contained in I. Then, there exists no feasible solution corresponding to the grid point $k(i, j) = (k \cdot i, k \cdot j)$ as choosing k items of weight (i, j) would be the only option (cf. Observation 5). □

Example 12. Consider the instance given by $k = 3$ and item sets $O = [2]$, $R = [6, 5, 4]$, $U = [3]$ and $D = [10]$. Because there are less than $k = 3$ items in O, U, and D, not all weights are feasible and the leftmost feasible weight in a row has to be detected in order to apply a neighborhood search. In particular, the origin $(0, 0)$ is an infeasible grid point, as there is only one item of weight type O, hence two more items have to be packed to obtain a feasible knapsack.

Figure 7 shows an illustration of this example in the weight space. The gray dots depict infeasible weights, as the corresponding solutions contain less than 3 items, e.g. the solution for the origin only contains a single O-item. Observe that the infeasible knapsack only consisting of one O-item and one diagonal item has a profit of 12 which is larger than the profit of the actual viable solution for grid point $(1, 1)$.

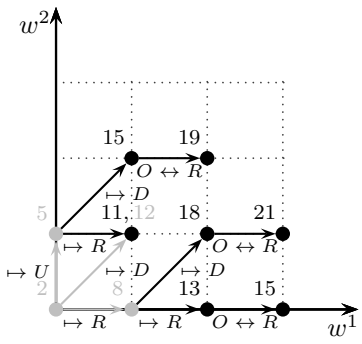

Figure 7: Existence of infeasible solutions.

◁

In the following, we are going to distinguish two cases: the case of all weights being feasible and the case in which some points in the weight grid are not feasible.

3.1 Special Case: All Weights are Feasible

In this special case there are a few conditions simplifying matters. First, it is straightforward to obtain the first (leftmost) feasible point in each row, since all points $(0, i)$, $i \in \{0, \dots, k\}$, are feasible with a unique corresponding item

composition. A similar condition holds for the last (rightmost) feasible point (k, i) in row i.

Lemma 13. *Let all weights be feasible, i.e., $|O| = |U| = |R| = |D| = k$. Then, the viable solutions for the points $(0, i)$ and (k, i) are unique with respect to the ordering of the item lists. For the point $(0, i)$, they consist of the first i items in U and the first $k - i$ items in O. For the point (k, i) they consist of the first i items in D and the first $k - i$ items in R.*

Proof. Obviously, a feasible solution x with weight $w(x) = (0, i)$ cannot contain any items of type D or R because of the first weight component. Hence, in order to obtain the weight $w^2(x) = i$, the number of U-items must be i. Since x is feasible, the remaining $k - i$ items have to be chosen from O.

Now consider a feasible solution y with weight $w(y) = (k, i)$. Since $w^1(y) = k$, all of the k items have to be of type R or D. Because R-items do not influence the second weight component, i of these items have to be diagonal items. □

Lemma 13 can be combined with the results of Theorem 8 and Theorem 9. Since the corresponding viable solutions of the first and the last grid point in line i are unique, these solutions trivially are the ones containing the most diagonal items. Hence, we can start with this first solution in line i, iteratively find a viable solution containing the most diagonal items for the right neighbor on the weight grid, and eventually end up in the last solution in line i.

Theorem 14 (Generating neighbor solutions)**.** *Let x_1^* be a viable solution with the most possible diagonal items for grid point (a, b). A viable solution x_2^* containing as many diagonal items as possible for grid point $(a+1, b)$ can be generated from x_1^* by either swapping the least profitable U-item of the solution with the most profitable diagonal item not yet chosen, or (in case that this swap is impossible or inferior) by swapping the least profitable O-item with the most profitable R-item not yet contained in the solution.*

Proof. Let x_2^* be this viable solution that uses the most D-items for $(a + 1, b)$ and denote by r_i, u_i, d_i, and o_i the number of R-, U-, D-, and O-items used in x_i^*, $i = 1, 2$. We have the following dependencies:

$$\begin{aligned} a + 1 &= d_2 + r_2 = d_1 + r_1 + 1 \\ b &= d_2 + u_2 = d_1 + u_1 \\ k &= d_2 + u_2 + r_2 + o_2 \\ &= d_1 + u_1 + r_1 + o_1 \end{aligned}$$

By Theorems 8 and 9 we know that the viable solution for $(a + 1, b)$ which contains the most diagonal items has either the same number of D-items than x_1^*, or one more.

If x_2^* contains more D-items than x_1^*, i.e. $d_2 = d_1 + 1$, then this implies $r_1 = r_2$, $u_1 = u_2 + 1$ and $o_1 = o_2$. In other words, x_2^* can be generated from x_1^* by a neighborhood swap of a U-item with a diagonal item.

In the second possible case, x_2^* and x_1^* contain the same number of D-items, i.e., $d_2 = d_1$. Then it holds $r_2 = r_1 + 1$, $u_2 = u_1$, and $o_1 = o_2 + 1$. Therefore, we can obtain x_2^* from x_1^* by swapping an O-item with an R-item.

By assumption we know that grid point $(a+1, b)$ is feasible, hence we only have to check and compare these two neighborhood swaps. Since we are looking for the viable solution with the most diagonal items we prefer the $U \leftrightarrow D$-swap unless it is not possible, i.e., if there are no further D-items to pack or no U-items to replace in the current solution, or in case the $O \leftrightarrow R$-swap results in a larger profit. □

It should be pointed out that Theorem 14 justifies a neighborhood check in constant time since only two possibilities have to be compared.

3.2 The General Case

In the general case we cannot guarantee that all weights are feasible, i.e., for some grid points in the weight space there might not exist a corresponding feasible solution. For example if the list O contains less than k items, there is no feasible solution for the origin of the weight space. In order to handle this situation we introduce so called *dummy items* that allow us - by keeping track of some additional information - to treat the general case like the case of feasible weights only.

Notation (Dummy Items)**.** A *dummy item* is an artificial item used to fill up the four item lists in order for all of them to contain k elements. The profit of a dummy item is set to -1 in order to distinguish it from a regular item with non-negative profit.
Note that a solution containing dummy items is infeasible. As a canonical extension, the profit $\bar{p}(x)$ of a solution x containing dummy items is still defined as the sum of profits of its items. However, this value $\bar{p}(x)$ may now be negative.

Having filled up the item lists with dummy items, we can apply the previously discussed strategy of searching the neighborhood by lines. However, we have to make sure that we do not mistake a solution including dummy items for feasible or miss a feasible weight point because we preferred an infeasible solution at some step.

The following lemma generalizes Lemma 13 in the presence of infeasible weights.

Lemma 15. *All solutions corresponding to the first (leftmost) feasible weight in a row of the weight grid have the same item composition. Hence, there is a unique corresponding viable solution for this weight with respect to the order in the item lists.*

Proof. Suppose it is not unique. Then by Observation 5 there must be two feasible solutions which contain a different number of diagonal items. Comparing the item composition of these two solutions shows the existence of a feasible solution with a smaller w^1-weight, which is a contradiction. □

Lemma 16. *A viable solution can be detected for each feasible point by minimizing the number of dummy items contained in a solution while performing the neighborhood search for the case of feasible solutions only (cf. Theorem 14).*

Proof. By minimizing the number of dummy items contained in a knapsack, the unique leftmost feasible solution on a row mentioned in Lemma 15 can be detected. Theorem 8 states that all grid points between the first and the

last feasible solution on a line are also feasible. Hence, the general neighborhood search can be performed to collect all viable solutions without the use of dummy items. □

3.3 Local Search Algorithm

Algorithm 2 introduces the pseudo-code of our approach. For each possible point $(i, j) \in \{0, \ldots, k\}^2$ in the weight space the algorithm stores the corresponding solution $x_{i,j}$ (which may be infeasible) with the largest possible profit, the corresponding profit $\bar{p}_{i,j} = \bar{p}(x_{i,j})$, and the number of dummy items. Note that for a given point (i, j), the solution $x_{i,j}$ is feasible if and only if it does not contain any dummy items.

A solution can be stored in a compact way by using a vector $v = (o, r, u, d, dum)$ where o, r, u, d are the number of items chosen from sets O, R, U and D, respectively, and dum is the number of dummy items. In this case a neighbor solution can be generated very efficiently, e.g. a swap of $O \leftrightarrow R$ corresponds to decreasing the first component by one and increasing the second component by one in v. Recall the meaning of a swap: one item is replaced by a substitute item. If the profit of the item to be replaced is negative, then this item is a dummy item and the fifth component of v decreases by one. If the profit of the substitute item is negative then the fifth component of v increases by one. The profit can also be updated in constant time by adding the difference of profits between those two items.

Additionally, for each grid point (i, j) we compute a value $val_{i,j}$ which is used for an on-the-fly dominance check between the viable solutions collected. The value $val_{i,j}$ is defined as the maximal profit of all feasible solutions $x \in X$ having weight $w^1(x) \leq i$ and $w^2(x) \leq j$. There are two important cases to consider for the computation of $val_{i,j}$:

i) If $x_{i,j}$ is feasible and $\bar{p}_{i,j}$ is larger than both $val_{i,j-1}$ and $val_{i-1,j}$, then $x_{i,j}$ itself attains the largest profit of any feasible solution with weight $(a, b) \leqq (i, j)$. Hence, $val_{i,j} = \bar{p}_{i,j}$ and moreover $x_{i,j}$ is efficient.

ii) Otherwise, $x_{i,j}$ is infeasible or dominated by some solution. Therefore $\bar{p}_{i,j}$ does not need to be considered as $val_{i,j}$ is the larger of the values $val_{i,j-1}$ and $val_{i-1,j}$.

Remark 17. Identifying a viable solution with its item composition is computationally beneficial. However, there may be several different solutions being composed of the same number of items of the four types and having the same optimal profit for this composition, namely in cases in which the order in the lists is not unique. The solution corresponding to a vector v is only unique up to permutation of items having the same weight and profit. ◁

As motivated by Lemma 3 and Lemma 11, a preprocessing of the input data is necessary to guarantee an efficient and correct operation of Algorithm 2. A pseudo-code is given in Algorithm 1.

This preprocessing step can be realized in running time $\mathcal{O}(n \log n)$ in a naïve way. The time can be improved to $\mathcal{O}(n + k \log k)$ if we do not sort the four lists right away, but instead identify the k items that are actually needed. For this

purpose, the k-th profitable item of each set has to be extracted (note that this so-called *k-th order statistics* can be found in $\mathcal{O}(n)$, see [1]), and in a second run all items with a larger profit can be collected, and the k items in each list have to be sorted.

Algorithm 1 Preprocessing of the data

1. according to weights partition the items into lists O, R, U, D for weights $(0,0), (1,0), (0,1), (1,1)$, respectively
2. **for all** item lists **do**
3. **if** the list contains less than k entries **then**
4. add dummy items until it contains k items
5. **else**
6. identify k most profitable items
7. sort these items w.r.t. non-increasing profit
8. **end if**
9. **end for**

Theorem 18. *Algorithm 2 computes a viable solution for each feasible weight and identifies a minimal complete set of efficient solutions, and its running time is $\mathcal{O}(n + k^2)$.*

Proof. The correctness is an immediate corollary of Lemma 15 and 16, and the discussion preceding Remark 17 in Section 3.3.

It has already been argued that the preprocessing can be implemented in $\mathcal{O}(n + k \log k)$. The first for-loop in line 2 of Algorithm 2 can be realized in $\mathcal{O}(k)$. Even if the initialization of the grid points in line 6 is done explicitly, it is in $\mathcal{O}(k)$. The operations – this includes finding the best neighbor and the dominance check – in the third for-loop in line 11 can be done in constant time. The running time of the algorithm follows due to the observation that the two nested for-loops run in $\mathcal{O}(k^2)$. □

The running time of this algorithm can be considered asymptotically optimal since any exact algorithm has to explore all n items and there are instances with $(k+1)^2$ different nondominated outcomes and exponentially many efficient solutions.

Example 19. Consider the instance given by k items of each of the four weight types, in which all the O-items have a profit of 0, all the R-items and all the U-items yield a profit of 1, and all the D-items have profit 2. The input is therefore specified by

$$\begin{aligned} O &= [0, 0, \ldots, 0], \\ R &= [1, 1, \ldots, 1], \\ U &= [1, 1, \ldots, 1], \\ D &= [2, 2, \ldots, 2], \end{aligned}$$

where all the lists have a length of k.

Algorithm 2 Cardinality constrained knapsack by rows

Input: Set I of n items, each having a profit $p \in \mathbb{R}^+$ and two weights $w^1, w^2 \in \{0,1\}$, cardinality constraint k.
Output: Minimal complete set L of efficient solutions.

1. call Algorithm 1 for preprocessing of the data
2. **for all** $i \in \{0,\dots,k\}$ **do**
3. set $val_{i,-1} \coloneqq -\infty$ and $val_{-1,i} \coloneqq -\infty$ // grid points outside the feasible weight range obtain an artificial value that can be accessed
4. **end for**
5. **for all** $j \in \{0,\dots,k\}$ **do**
6. initialize row j in the grid point $(0,j)$ with the solution $x_{0,j}$ consisting of j U-items and $k-j$ O-items, i.e.:
   – set $\bar{p}_{0,j} \coloneqq \bar{p}(x_{0,j})$
   – store the number of dummy items contained in $x_{0,j}$
7. set $val_{0,j} \coloneqq \begin{cases} \max(val_{0,j-1},\ \bar{p}_{0,j}), & \text{if } x_{0,j} \text{ is feasible} \\ val_{0,j-1}, & \text{otherwise} \end{cases}$
8. **if** $x_{0,j}$ is feasible **and** $\bar{p}_{0,j} > val_{0,j-1}$ **then**
9. $L = L \cup \{x_{0,j}\}$
10. **end if**
11. **for all** $i \in \{0,\dots,k-1\}$ **do**
12. determine the possible neighborhood swaps $\mathcal{S} \subseteq \{O \leftrightarrow R, U \leftrightarrow D\}$ to reach point $(i+1,j)$ from $x_{i,j}$
13. generate the solution(s) obtained from $x_{i,j}$ by the swaps in $\mathcal{S}$
14. **if** both solutions have equally many dummies **then**
15. **if** both solutions have equal profit **then**
16. choose as $x_{i+1,j}$ the solution containing more diagonal items
17. **else**
18. choose as $x_{i+1,j}$ the solution with the larger profit
19. **end if**
20. **else**
21. choose as $x_{i+1,j}$ the solution using the fewest dummy items
22. **end if**
23. set $\bar{p}_{i+1,j} \coloneqq \bar{p}(x_{i+1,j})$
24. set $val_{i+1,j} \coloneqq \begin{cases} \max(val_{i+1,j-1},\ val_{i,j},\ \bar{p}_{i+1,j}), & \text{if } x_{i+1,j} \text{ feasible} \\ \max(val_{i+1,j-1},\ val_{i,j}), & \text{otherwise} \end{cases}$
25. **if** $x_{i+1,j}$ is feasible **and** $\bar{p}_{i+1,j} > \max(val_{i,j}, val_{i+1,j-1})$ **then**
26. $L = L \cup \{x_{i+1,j}\}$
27. **end if**
28. **end for**
29. **end for**

Let $x \in X$ be an arbitrary feasible solution, and denote by o, r, u, and d the number of items of type O, R, U, and D contained in x, respectively. Then we have

$$\begin{aligned} w^1(x) &= r + d \\ w^2(x) &= u + d \\ p(x) &= r + u + 2d = w^1(x) + w^2(x). \end{aligned} \tag{4}$$

It is easy to see that x is an efficient solution. Let $z \in X$ be a solution with $w^1(z) \leq w^1(x)$, $w^2(z) \leq w^2(x)$ and $p(z) \geq p(x)$. Then by Equation (4) it also holds $p(z) = w^1(z) + w^2(z) \leq w^1(x) + w^2(x) = p(x)$, and we can conclude $\big(p(z), w^1(z), w^2(z)\big) = \big(p(x), w^1(x), w^2(x)\big)$.

On the other hand, due to Lemma 11 we know that all of the possible $(k+1)^2$ weights are feasible. Thus, all the $\binom{n}{k}$ feasible solutions of the given instance are efficient and correspond to $(k+1)^2$ different nondominated points. Figure 7 visualizes the weight space for the case $k = 5$ and $n = 20$. ◁

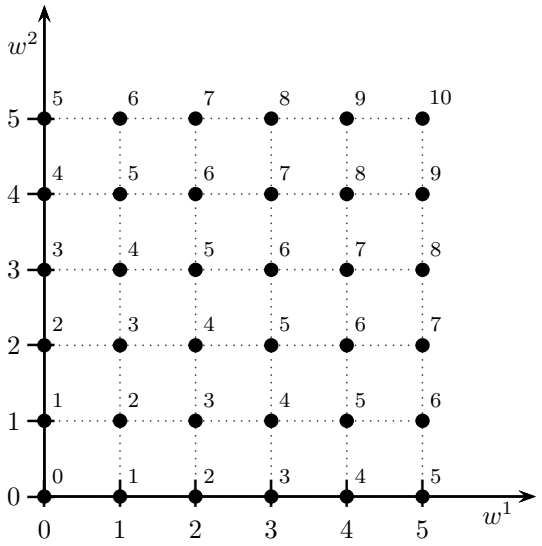

Figure 8: Example of the generic instance for $k = 5$ and $n = 20$.

3.4 More on Structural Properties

In the following we provide some further insight in the structure of viable and efficient solutions.

Given two viable or efficient solutions, we are going to prove that it is possible to find a sequence of swaps transforming one solution into the other by visiting only viable or efficient solutions in intermediate steps.

Definition 20. A subset $X' \subseteq X$ of feasible solutions is called *connected* if for all pairs $u, v \in X'$ there is a sequence $(u = x_1, \ldots, x_k = v)$ such that $x_i \in X'$ for all $i = 1, \ldots, k$ and x_i is neighbor of x_{i+1} for all $i = 1, \ldots, k-1$.

Theorem 21. *All viable solutions are connected with respect to the 1-swap neighborhood.*

Proof. At first, observe that viable solutions which are composed of the same number of items of the four types are connected. Therefore, we only need to show the connectedness of viable solutions having different item compositions.

Obviously, all the solutions collected by Algorithm 2 in a row of the weight grid are connected. By Theorem 8 and Lemma 16 we derive that the set of viable solutions that the algorithm collects in a row of the weight space are connected.

Recall that due to Lemma 15 the leftmost viable solution of a row has a unique item composition. In the next step we are going to show that each viable solution is connected to this unique leftmost viable solution in its row by a sequence of viable solutions. Let x_1 be a viable solution with $w(x_1) = (a, b)$ which is not visited by the algorithm. It suffices to show that there is a neighbor viable solution of x_1 with weight $(a-1, b)$. We know that there exists a solution x_2 found by the algorithm, with $w(x_2) = w(x_1)$ and $p(x_2) = p(x_1) = p_{a,b}$. Since the solutions have a different item composition, and by construction of the algorithm, it follows that x_1 contains less diagonal items than x_2. Denote by d_i, r_i, u_i, and o_i the number of items of the sets D, R, U, and O included in x_i, $i \in \{1, 2\}$, respectively. Comparing the solutions we obtain the following relations:

$$\begin{aligned} \delta &:= d_2 - d_1 \geq 1 \\ u_2 &= u_1 - \delta \\ r_2 &= r_1 - \delta \\ o_2 &= o_1 + \delta. \end{aligned}$$

Since both x_1 and x_2 are viable, it holds

$$p(D_{d_2}) + p(O_{o_2}) \geq p(R_{r_2+1}) + p(U_{u_2+1}) \tag{5}$$
$$p(R_{r_1}) + p(U_{u_1}) \geq p(D_{d_1+1}) + p(O_{o_1+1}). \tag{6}$$

Combining Equations (5) and (6) with the monotonicity of the profit in the ordered item lists, we derive

$$p(D_{d_2}) + p(O_{o_2}) = p(R_{r_2+1}) + p(U_{u_2+1}).$$

Now, consider the viable solution y for weight $(a-1, b)$ that was collected by the algorithm right before x_2. Suppose, that x_2 was derived from y by a swap $U \leftrightarrow D$. Then, the profit of solution y is

$$\begin{aligned} p(y) &= p(x_2) + p(U_{u_2+1}) - p(D_{d_2}) \\ &= p(x_1) + p(O_{o_2}) - p(R_{r_2+1}) \\ &\leq p(x_1) + p(O_{o_1+1}) - p(R_{r_1}), \end{aligned} \tag{7}$$

where the last inequality again follows from monotonicity of the profit. Observe that the right hand side of (7) is the profit of a feasible solution z emerging from x_1 when performing a swap $R \leftrightarrow O$. Since the weight of z is $w(z) = (a, b) - (1, 0) = w(y)$, we can conclude that z is also a viable solution for $(a-1, b)$.

Suppose, that x_2 was derived from y by a swap $O \leftrightarrow R$. In this case it holds that

$$\begin{aligned} p(y) &= p(x_2) + p(O_{o_2+1}) - p(R_{r_2}) \\ &= p(x_1) + p(O_{o_2+1}) - p(R_{r_2}) \\ &\leq p(x_1) + p(O_{o_1+1}) - p(R_{r_1}), \end{aligned}$$

and again we conclude the existence of a viable solution z, with $w(z) = (a-1, b)$, which is neighbor of x_1.

It remains to prove that the set of leftmost viable solutions of rows of the weight grid is connected. Let $x \in X$, with $w(x) = (a, b)$, be a viable solution for the unique item composition of the leftmost feasible weight in row b. Consider the problem of finding the smallest feasible w_1-weight in row i,

$$\begin{aligned} (P(i)) \qquad & \min\ r + d \\ & \text{s.t. } u + d = i && (8) \\ & \qquad r + o = k - i && (9) \\ & \qquad o \in \{0, \ldots, |O|\} \\ & \qquad r \in \{0, \ldots, |R|\} \\ & \qquad u \in \{0, \ldots, |U|\} \\ & \qquad d \in \{0, \ldots, |D|\}, \end{aligned}$$

where o, r, u, and d are variables representing the number of items to be chosen from sets O, R, U and D, respectively. By assumption, the optimal objective value for $P(b)$ is a. Let (o^*, r^*, u^*, d^*) denote the optimal solution to $P(b)$ corresponding to x.

Suppose that there exist feasible weights in row $b+1$. Then, $P(b+1)$ has feasible solutions, so by (8) we can choose at least $b+1$ items from sets U and D. In particular, we can increase at least one of the values u^* or d^* in x and remain feasible. With a similar argument for Equation (9), it follows that r^* or o^* can be decreased by one without exceeding the feasible limits. Hence, there exists some feasible solution for $P(b+1)$ with objective value $a-1$, a or $a+1$, which is a neighbor of x. Denote by y the feasible solution having the smallest objective value among these neighbors of x. Then due to optimality of x for $P(b)$, no solution of $P(b+1)$ can have an objective value smaller than $w_1(y)$. This means that y has the unique item composition of the leftmost viable solution in row $b+1$. □

Theorem 22. *The image of a viable solution in the objective space is located on the boundary of the convex hull of all feasible images, i.e.,*

$$\left\{ \left(p(x), w^1(x), w^2(x)\right) \in Y : x \in X_V \right\} =: Y_V \subseteq bd\left(conv(Y)\right).$$

Proof. Suppose there exists a viable point $y = (p_{a,b}, a, b) \in Y_V$ which is located in the interior of the convex hull of all feasible images. Then there is a convex combination of some feasible points $z^1, \ldots, z^q \in Y$, $q \geq 2$, having the same weight, and a larger profit than y:

$$z := \sum_{i=1}^{q} \lambda_i z^i = (p, a, b),\ p > p_{a,b},\ \sum_{i=1}^{q} \lambda_i = 1,\ \lambda_i > 0. \tag{10}$$

Without loss of generality we can assume $z^i \in Y_V$, for $i = 1, \ldots, q$, since otherwise we could replace z^i by a viable point having the same weight. We denote by $x^i \in X_V$ the corresponding viable solution to $z^i \in Y_V$, $i = 1, \ldots, q$.

Recall the problem of finding the largest possible profit of feasible solutions with a given weight (a, b):

$$
\begin{aligned}
p_{a,b} \coloneqq \max & \sum_{i=1}^{n} p_i x_i \\
\text{subject to} \quad & \sum_{i=1}^{n} x_i = k && (11)\\
& \sum_{i=1}^{n} w_i^1 x_i = a && (12)\\
& \sum_{i=1}^{n} w_i^2 x_i = b && (13)\\
& x \in \{0,1\}^n.
\end{aligned} \tag{$\mathrm{IP}_{a,b}$}
$$

The polyhedron corresponding to Equations (11)-(13) is given by

$$
\begin{pmatrix}
\underbrace{\begin{matrix}1 & 1 & & 1\\ 1 & 1 & \cdots & 1\\ 0 & 0 & & 0\end{matrix}}_{R\text{-items}} &
\underbrace{\begin{matrix}1 & 1 & & 1\\ 1 & 1 & \cdots & 1\\ 1 & 1 & & 1\end{matrix}}_{D\text{-items}} &
\underbrace{\begin{matrix}1 & 1 & & 1\\ 0 & 0 & \cdots & 0\\ 1 & 1 & & 1\end{matrix}}_{U\text{-items}} &
\underbrace{\begin{matrix}1 & 1 & & 1\\ 0 & 0 & \cdots & 0\\ 0 & 0 & & 0\end{matrix}}_{O\text{-items}}
\end{pmatrix} x = \begin{pmatrix} k \\ a \\ b \end{pmatrix}, \tag{14}
$$

where the columns of the defining matrix A are arranged according to the four weight types as indicated. Observe that A has the consecutive ones property and thus is totally unimodular. By polyhedral theory of single criterion problems it follows that the polyhedron is integral. In particular, $p_{a,b}$ can be computed with the linear program relaxation of ($\mathrm{IP}_{a,b}$), which always possesses an integral optimal solution:

$$
\begin{aligned}
p_{a,b} = \max & \sum_{i=1}^{n} p_i x_i \\
\text{subject to} \quad & \sum_{i=1}^{n} x_i = k \\
& \sum_{i=1}^{n} w_i^1 x_i = a \\
& \sum_{i=1}^{n} w_i^2 x_i = b \\
& 0 \le x_i \le 1, \qquad i = 1, \ldots n.
\end{aligned} \tag{$\mathrm{LP}_{a,b}$}
$$

Now consider the solution

$$
x \coloneqq \sum_{i=1}^{q} \lambda_i x^i,
$$

which is the convex combination of the viable solutions corresponding to the

viable points z^i from Equation (10). We verify that x is feasible for ($\mathrm{LP}_{a,b}$):

$$\begin{aligned}\sum_{j=1}^{n} x_j &= \sum_{j=1}^{n}\sum_{i=1}^{q} \lambda_i x_j^i \\ &= \sum_{i=1}^{q} \lambda_i \sum_{j=1}^{n} x_j^i \\ &= \sum_{i=1}^{q} \lambda_i k = k, \end{aligned} \tag{15}$$

where Equation (15) is due to the feasibility of x^i for (CCMKP). Furthermore, it is

$$\begin{aligned}\sum_{j=1}^{n} w_j^1 x_j &= \sum_{j=1}^{n} w_j^1 \sum_{i=1}^{q} \lambda_i x_j^i \\ &= \sum_{i=1}^{q} \lambda_i \sum_{j=1}^{n} w_j^1 x_j^i \\ &= \sum_{i=1}^{q} \lambda_i w^1(x^i) = a, \end{aligned} \tag{16}$$

where Equation (16) is a consequence of the construction of z in (10). Similarly it holds that

$$\sum_{j=1}^{n} w_j^2 x_j = b. \tag{17}$$

Since x_j^i is a binary variable for $i = 1, \ldots, q$ and $j = 1, \ldots, n$, it follows

$$0 \leq x_j = \sum_{i=1}^{q} \lambda_i x_j^i \leq 1, \qquad j = 1, \ldots, n. \tag{18}$$

So Equations (15)-(18) show feasibility of x for ($\mathrm{LP}_{a,b}$). But the profit of x is

$$\begin{aligned}\sum_{j=1}^{n} p_j x_j &= \sum_{j=1}^{n} p_j \sum_{i=1}^{q} \lambda_i x_j^i \\ &= \sum_{i=1}^{q} \lambda_i \sum_{j=1}^{n} p_j x_j^i \\ &= \sum_{i=1}^{q} \lambda_i p(x^i) \\ &= p > p_{a,b}. \end{aligned}$$

This yields a contradiction to the definition of $p_{a,b}$. □

Theorem 22 in particular holds for efficient solutions. We obtain the following corollary:

Corollary 23. *All efficient solutions $x \in X_E$ are supported.*

Conversely, any optimal solution of a weighted sum problem is efficient. It is a well known fact that the set of supported solutions is connected with respect to the neighborhood of basic feasible solutions of the LP relaxation (e.g. [6]). But this neighborhood coincides with our 1-swap neighborhood, as shown by Gorski, Klamroth and Ruzika [4]. Hence, we derive

Corollary 24. *The set of efficient solutions X_E is connected with respect to the 1-swap neighborhood.*

4 Conclusion

In this manuscript, a combinatorial optimization problem is considered. From a set of items, exactly k many have to be chosen such that three objective functions are optimized. This selection process can be considered as filling a knapsack with a fixed cardinality. Each item has one non-negative profit and two binary weights. The filling of the knapsack should maximize (additive) profit while minimizing (additive) weights simultaneously. This optimization problem is understood as computing all nondominated (with respect to the componentwise ordering) points and for each of them one feasible solution.

An exact neighborhood search algorithm solving this problem is proposed. This algorithm relies on categorizing items into groups depending on their weights. Two solutions are adjacent (i.e. they are neighbors) if they differ by an exchange of two items. Starting in the origin, the integral weight grid in $\mathbb{R}^2$ is explored by moving to subsequent neighbors and storing exactly one solution for each possible grid point. The set of solutions generated in the algorithm, called *viable solutions*, is a non-trivial superset of the set of efficient solutions and a dominance check filters dominated solutions immediately. As a side-effect, correctness of the algorithm also implies connectedness of the set of viable solutions, i.e., given any viable solution, it is possible to attain any other viable solution by item swaps while generating viable solutions only. To the best of our knowledge, this is the first exact local search algorithm for a multicriteria optimization problem which relies on some concept of adjacency of a superset of the set efficient solutions. The algorithm runs in $O(n + k^2)$ time which can be considered optimal.

The problem at hand can also be solved by some naïve dynamic programming approach. A numerical comparison of this naïve approach with the algorithm proposed could demonstrate the effectiveness of the method proposed in this manuscript. A generalization of the algorithm to more than two or, even better, an arbitrary number of binary weight functions is certainly desirable but probably difficult to develop due to the exponentially increasing number of possible item swaps.

References

[1] M. Blum, R. Floyd, V. Pratt, R. Rivest, and R. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7:448–461, 1973.

[2] M. Ehrgott. *Multicriteria optimization (2nd ed.)*. Springer Verlag, Berlin, 2005.

[3] A. Fréville. The multidimensional 0-1 knapsack problem: An overview. *European Journal of Operational Research*, 155(1):1–21, 2004.

[4] J. Gorski, K. Klamroth, and S. Ruzika. Connectedness of efficient solutions in multiple objective combinatorial optimization. Report in Wirtschaftsmathematik (WIMA Report) 102/2006, University of Kaiserslautern, Department of Mathematics, 2006.

[5] J. Gorski and L. Paquete. On a particular case of the multi-criteria unconstrained optimization problem. *Electronic Notes in Discrete Mathematics*, Jan. 2010.

[6] H. Isermann. The enumeration of the set of all efficient solutions for a linear multiple objective program. *Operational Research Quarterly*, 28(3):711–725, 1977.

[7] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer Verlag, 2004.