

# A linear bound on the number of scalarizations needed to solve discrete tricriteria optimization problems

Kerstin Dächert

Kathrin Klamroth

November 10, 2018

Multi-objective optimization problems are often solved by a sequence of parametric single-objective problems, so-called scalarizations. If the set of nondominated points is finite, the entire nondominated set can be generated in this way. In the bicriteria case it is well known that this can be realized by an adaptive approach which requires the solution of at most  $2|Z_N| - 1$  subproblems, where  $Z_N$  denotes the nondominated set of the underlying problem and a subproblem corresponds to a scalarized problem. For problems with more than two criteria, no methods were known up to now for which the number of subproblems depends linearly on the number of nondominated points. We present a new procedure for finding the entire nondominated set of tricriteria optimization problems for which the number of subproblems to be solved is bounded by  $3|Z_N| - 2$ , hence, depends linearly on the number of nondominated points. The approach includes an iterative update of the search region that, given a (sub-)set of nondominated points, describes the area in which additional nondominated points may be located. If the  $\varepsilon$ -constraint method is chosen as scalarization, the upper bound can be improved to  $2|Z_N| - 1$ .

**Keywords:** Discrete tricriteria optimization; Scalarization; Box algorithm

## 1 Introduction

The determination of the nondominated set is the basis for a multitude of methods in multiple criteria decision making. In multiple objective combinatorial optimization the nondominated set is discrete and, assuming that some natural bounds are given for the objective values, also finite. In this situation, a complete enumeration of all nondominated points can be realized by the successive solution of a series of appropriately formulated scalarized problems which are called *subproblems* in what follows.

Ideally, the total number of subproblems depends linearly on the number of nondominated points. Indeed, in the bicriteria case approaches are known which require the solution of at most  $2|Z_N| - 1$  subproblems, where  $Z_N$  denotes the finite nondominated set of the underlying problem. Thereby,  $|Z_N|$  subproblems are solved to generate all points in  $Z_N$ , and the additional  $|Z_N| - 1$  subproblems are needed to ensure that no further nondominated points exist between the already generated ones (see, e.g., Chalmet et al., 1986; Ralphs et al., 2006). If the  $\varepsilon$ -constraint method is used, the entire nondominated set of a bicriteria problem can be generated within  $|Z_N| + 1$  subproblems (Laumanns et al., 2006). However, up to now, no linear bounds are known for higher dimensional problems. The best known approach has a theoretical bound of  $(|Z_N| + 1)^{m-1}$  subproblems for problems with  $m$  objectives (Laumanns et al., 2006). In this paper, we present a new procedure for finding the entire nondominated set of tricriteria optimization problems for which the number of scalarized subproblems to be solved is bounded by  $3|Z_N| - 2$ . This is achieved by the definition of a new split criterion which allows to exclude redundant parts of the search region. It can then be shown that the number of boxes, into which the search region is decomposed, depends linearly on the number of nondominated points. If the  $\varepsilon$ -constraint method is used, the bound can even be improved to  $2|Z_N| - 1$ .

## 1.1 Terminology and Definitions

We consider multiple criteria optimization problems

$$\min_{x \in X} f(x) = (f_1(x), \dots, f_m(x))^T \quad (1)$$

with  $m \geq 2$  objective functions  $f_i : X \rightarrow \mathbb{R}$ ,  $i = 1, \dots, m$ , and with feasible set  $X \neq \emptyset$ . Throughout this paper we assume that  $X$  is a discrete set. The image set of the feasible set  $X$  in the outcome space is denoted by  $Z := f(X)$  where  $Z$  is a finite set of distinct points in  $\mathbb{R}^m$ .

We use the Pareto concept of optimality: A solution  $\bar{x} \in X$  is called *Pareto optimal* or *efficient* if there does not exist a feasible solution  $x \in X$  such that  $f_i(x) \leq f_i(\bar{x})$  for all  $i = 1, \dots, m$  and  $f_j(x) < f_j(\bar{x})$  for at least one  $j \in \{1, \dots, m\}$ . The corresponding objective vector  $f(x) \in \mathbb{R}^m$  is called *nondominated* in this case. If, on the other hand,  $f(x) \leq f(\bar{x})$  for some feasible  $x \in X$ , i.e.,  $f_i(x) \leq f_i(\bar{x})$  for all  $i = 1, \dots, m$  and  $f_j(x) < f_j(\bar{x})$  for at least one  $j \in \{1, \dots, m\}$ , we say that  $f(x)$  *dominates*  $f(\bar{x})$ , and  $x$  *dominates*  $\bar{x}$ . If strict inequality holds for all  $m$  components, i.e., if  $f_i(x) < f_i(\bar{x})$  for all  $i = 1, \dots, m$ , then  $x$  *strictly dominates*  $\bar{x}$ . If there exists no feasible solution  $x \in X$  that strictly dominates  $\bar{x}$ , then  $\bar{x}$  is called *weakly Pareto optimal* or *weakly efficient*. We denote the set of efficient solutions of (1) by  $X_E$  and refer to it as the *efficient set*.

The image set of the set of efficient solutions is denoted by  $Z_N := f(X_E)$  and is called the *nondominated set* of problem (1). To simplify notation, we will often refer to the points in  $Z$  without relating them back to their preimages in the feasible set. Consequently, we equivalently formulate problem (1) in the outcome space as

$$\min_{z \in Z} z = (z_1, \dots, z_m)^T \quad (2)$$

where  $Z$  is a discrete set of points in  $\mathbb{R}^m$ . For two vectors  $z, \bar{z} \in Z$  we write

$$\begin{aligned} z < \bar{z} & \text{ if } z_i < \bar{z}_i \text{ for all } i = 1, \dots, m, \\ z \leq \bar{z} & \text{ if } z_i \leq \bar{z}_i \text{ for all } i = 1, \dots, m \text{ and } \exists j \in \{1, \dots, m\} : z_j < \bar{z}_j, \text{ and} \\ z \preceq \bar{z} & \text{ if } z_i \leq \bar{z}_i \text{ for all } i = 1, \dots, m. \end{aligned}$$

The symbols  $>$ ,  $\geq$  and  $\succeq$  are used accordingly.

**Definition 1.1** (Nondominance). *A point  $\bar{z} \in Z$  is called nondominated if and only if there exists no point  $z \in Z$  such that  $z \leq \bar{z}$ .*

A lower bound on the nondominated points of (2) is given by the *ideal point* which we denote by  $z^I$ . The  $i$ -th component of the ideal point is defined as the minimum of the  $i$ -th objective, i.e.,  $z_i^I := \min\{z_i : z \in Z\}$  for all  $i = 1, \dots, m$ . A point  $z^U$  that strictly dominates  $z^I$  is called a *utopia point*. Note that in general  $z^I \notin Z$ . On the other hand, the *nadir point*  $z^N$  with components  $z_i^N := \max\{z_i : z \in Z_N\}$  for all  $i = 1, \dots, m$  provides an upper bound on the nondominated set of (2). While it can be easily determined for bi-objective problems, this is in general not the case for higher dimensional problems. However, any upper bound on the nondominated set is sufficient for our purpose. Therefore, we typically use upper bounds on  $Z = f(X)$  which can be determined also in the presence of more than two criteria.

A common technique to solve problems of the form (1) is to iteratively transform the original multiple objective problem into a series of parametric single-objective problems, so called *scalarizations* (see, e.g., Ehrgott, 2005; Miettinen, 1999). A variety of different scalarization methods exists which differ, among other things, with respect to their theoretical properties. Of particular importance is the question whether the solutions generated by a specific method always correspond to nondominated points of (1) and whether all nondominated points of (1) can be generated by appropriately varying the involved parameters. In this article we do not focus on a specific scalarization, but assume to have a scalarization method at hand which possesses these two properties: Every nondominated point can be generated, and the point corresponding to an optimal solution of the scalarization is nondominated.

## 1.2 Literature Review

The idea of solving parametric single-objective optimization problems in order to generate a (complete) set of nondominated points is well known in the literature. Especially for bicriteria problems, the literature is rich. Aneja and Nair (1979) use a parametric weighted sum method in order to find the extreme supported nondominated points of (integer) linear problems. They show that the algorithm performs exactly  $2n - 3$  iterations, where  $n$  denotes the number of extreme nondominated points and the two lexicographic minima are assumed to be known. Chalmet et al. (1986) use an  $\varepsilon$ -constraint method with a weighted sum objective as scalarization for solving bicriteria integer problems. Due to the hybrid scalarization, every nondominated point can be computed. The authors show that a complete representation is obtained by solving  $2|Z_N| + 1$  integer programs including the computation of the lexicographic optima.

Eswaran et al. (1989) employ a weighted Tchebycheff scalarization to determine a complete or incomplete representation of the nondominated set of nonlinear integer bicriteria problems. Solanki (1991) use an augmented weighted Tchebycheff method to generate incomplete representations of mixed integer bicriteria linear programs. Ulungu and Teghem (1995) address bicriteria combinatorial optimization problems. They introduce a two phase procedure, where all extreme supported nondominated points are computed by weighted sum problems. In a second phase, all remaining nondominated points are generated with the help of specific combinatorial methods. In Sayin and Kouvelis (2005), the lexicographic weighted Tchebycheff method and a variant of it serve to solve bicriteria discrete optimization problems. Tchebycheff scalarizations are also used by Ralphs et al. (2006). Their algorithm is shown to find all nondominated points by solving  $2|Z_N| - 1$  subproblems including the generation of the lexicographic optima. The box algorithm of Hamacher et al. (2007) uses lexicographic  $\varepsilon$ -constraint problems. While it is designed for incomplete representations, it can also be used to generate the entire nondominated set.

Also for the discrete multicriteria case, several approaches for finding the entire nondominated set based on parametric algorithms exist. Klein and Hannan (1982) use a kind of  $\varepsilon$ -constraint method to determine the entire nondominated set of linear integer problems. The remaining search region containing possible further nondominated points is described by disjunctive constraints. While Chalmet et al. (1986) mainly address the bicriteria case, they also propose a generalization to the multicriteria case that is based on recursively solving bicriteria problems. Tenfelde-Podehl (2003) presents a generalization of the two phase method to any number of criteria. Sylva and Crema (2004) revisit the idea of Klein and Hannan (1982) and reformulate the disjunctive constraints with the help of binary variables. Laumanns et al. (2006) use lexicographic  $\varepsilon$ -constraint problems and show that at most  $(|Z_N| + 1)^{m-1}$  subproblems need to be solved to generate the entire nondominated set. In the bicriteria case, this yields a total number of only  $|Z_N| + 1$  subproblems, which is smaller than the upper bound of Ralphs et al. (2006) due to the special scalarization employed. The numerical experiments for a knapsack problem with three objectives reveal that the number of subproblems needed is considerably less than  $(|Z_N| + 1)^2$ . The authors state that it is an open question whether the number of subproblems can be bounded linearly in terms of the number of nondominated points for problems with more than two criteria. Laumanns et al. (2005) improve the algorithm of Laumanns et al. (2006). However, no better theoretical bound on the number of subproblems is obtained. Özlen and Azizoglu (2009) use an augmented  $\varepsilon$ -constraint method within a recursive algorithm that is similar to the approach of Chalmet et al. (1986) and demonstrate that  $\mathcal{O}(|Z_N|^{m-1})$  iterations are required in the worst case. Dhaenens et al. (2010) extend the approach of Tenfelde-Podehl (2003) to a three phase procedure. Their numerical experiments show that the determination of the nadir point is very expensive regarding computational time. Przybylski et al. (2010) also propose a two phase method for integer problems with more than two objectives. They also encounter the problem of describing the search region and solve it by saving certain upper bound vectors. Dominated upper bound vectors are filtered out by pair-wise comparisons. Lokman and Köksalan (2013) build on Sylva and Crema (2004) and propose two improved algorithms based

on an augmented  $\epsilon$ -constraint scalarization. While the numerical study of Lokman and Köksalan (2013) suggests a linear bound on the number of subproblems to be solved in the tricriteria case, only an upper bound of  $\mathcal{O}(|Z_N|^2)$  is derived for  $m = 3$ . Ozlen et al. (2014) improve Özlen and Azizoğlu (2009) by saving the right-hand side vectors and the corresponding solutions of the integer problems that have already been solved. Thereby, a huge saving of computational time is achieved. Kirlik and Sayın (2014) improve the method of Laumanns et al. (2006) by changing the order in which the subproblems are solved. While the numerical results are very competitive, no better theoretical bound on the number of subproblems is proven.

### 1.3 Goals and Outline

We present an algorithm that generates the entire nondominated set of a discrete tricriteria optimization problem by solving at most  $3|Z_N| - 2$  subproblems, if  $|Z_N| \geq 3$  and if bounds on the set of feasible outcomes are given. Thereby, to the best of our knowledge, a linear bound with respect to the number of nondominated points is given for the first time for tricriteria problems. Our algorithm does not depend on a specific scalarization but can be used with any scalarization method that is suited for discrete and, in general, non-convex problems. Our method is also applicable if a subset of nondominated points is already known and the search region potentially containing further nondominated points shall be generated.

The remainder of this paper is organized as follows: In Section 2 we present a decomposition of the search region based on nondominance and develop a first generic box algorithm. We show that this generic algorithm may produce redundant boxes which makes the algorithm inefficient. Under the technical assumption that all nondominated points differ pairwise in every component, we show in Section 3 how to construct a decomposition in the tricriteria case that only contains non-redundant boxes. The number of boxes is proven to be at most  $3|Z_N| - 2$  for  $|Z_N| \geq 3$ . Finally, we show that the algorithm can also be applied if the nondominated points are in arbitrary position, i.e., every pair of points may have up to  $m - 2$  equal components. The upper bound  $3|Z_N| - 2$  is also valid in this general case. In Section 4 we demonstrate that the upper bound can be improved to  $2|Z_N| - 1$  when an  $\epsilon$ -constraint method is used as scalarization. In Section 5 we present numerical results.

## 2 Split of the search region for multicriteria problems

Let  $B_0$  denote an initial search region of the form

$$B_0 := \{z \in \mathbb{R}^m : l_j \leq z_j < u_j, j = 1, \dots, m\}$$

with  $l, u \in \mathbb{R}^m, l \leq u$ . As lower and upper bound of  $B_0$  we choose a global lower and upper bound on the nondominated set, for example,  $l := z^I$  and  $u := z^M$ , where  $z^I$  is the ideal point and  $z_j^M := \max\{z_j : z \in Z\} + \delta$  for all  $j = 1, \dots, m$  with  $\delta > 0$  is an upper bound on the set  $Z$ . If no special scalarization method is employed, the iterative reduction of the search region can solely be based on nondominance. Thereby, every

generated nondominated point allows to restrict the search region, as, by Definition 1.1, for any  $z^* \in Z_N$ , the two sets

$$S_1(z^*) := \{z \in B_0 : z \leq z^*\} \quad \text{and} \quad S_2(z^*) := \{z \in B_0 : z \geq z^*\}$$

do not contain any nondominated points besides  $z^*$ , i.e.,  $S_1(z^*) \cap Z_N = S_2(z^*) \cap Z_N = \{z^*\}$ . Moreover,  $S_1(z^*) \cap Z = \{z^*\}$ , thus,  $S_1(z^*) \setminus \{z^*\}$  contains no feasible points.

In the following, we decompose a given initial search region  $B_0$  iteratively into subsets  $B \subset B_0$  of the same form, i.e., into sets  $B := \{z \in \mathbb{R}^m : l_j \leq z_j < u'_j, j = 1, \dots, m\}$  with  $u' \in \mathbb{R}^m, l \leq u' \leq u$ . As the initial search region that potentially contains nondominated points of (1) as well as each subset  $B$  as defined above describe rectangular subsets of  $\mathbb{R}^m$  with sides parallel to the coordinate axes, we call these sets *boxes* in the following. The search region is always represented as the union of certain boxes  $B$ . With the generation of every new nondominated point we replace some of the boxes of the current search region by appropriate new boxes such that the whole search region is covered. This property is called correctness in the following.

**Definition 2.1** (Correct decomposition). *Let  $B_0$  denote the starting box, let  $\mathcal{B}_s$  denote the set of boxes at the beginning of iteration  $s \geq 1$ , where  $\mathcal{B}_1 := \{B_0\}$ , and let  $z^p \in Z_N, p = 1, \dots, s-1$ , be already determined nondominated points. We call  $\mathcal{B}_s$  correct with respect to  $z^1, \dots, z^{s-1}$ , if*

$$B_0 \setminus \left( \bigcup_{B \in \mathcal{B}_s} B \right) = \bigcup_{p=1, \dots, s-1} S_2(z^p) \quad (3)$$

holds, where  $S_2(z^p) := \{z \in B_0 : z \geq z^p\}$  denotes that subset of the box  $B_0$  that is dominated by the point  $z^p \in Z_N, p = 1, \dots, s-1$ .

Any split presented in the following maintains a correct decomposition of the search region at any time. Under this basic condition, we try to generate as few boxes as possible, as for every generated box a scalarized subproblem needs to be solved. Our aim is to keep the number of subproblems low. The simplest split decomposes a box  $B$  which contains a new outcome  $z^* \in (B \cap Z_N)$  into  $m$  subboxes.

**Definition 2.2** (Full  $m$ -split). *Let a nondominated point  $z^* \in (B \cap Z_N)$  be given. We call the replacement of  $B$  by the  $m$  sets*

$$B_i := \{z \in B : z_i < z_i^*\} \quad \forall i = 1, \dots, m \quad (4)$$

a full  $m$ -split of  $B$ .

Note that similar decomposition approaches are proposed, e.g., in Tenfelde-Podehl (2003), Dhaenens et al. (2010) and Przybylski et al. (2010). Recursively applying the full  $m$ -split to every box which contains the current nondominated point yields a correct decomposition, as the following lemma shows.

**Lemma 2.3** (Correctness of the full  $m$ -split). *Let  $\mathcal{B}_s, s \geq 1$ , with  $\mathcal{B}_1 := \{B_0\}$  be a correct decomposition with respect to the nondominated points  $z^1, \dots, z^{s-1}$ , and let  $z^s \in Z_N$ . If a full  $m$ -split is applied to all boxes  $B \in \mathcal{B}_s$  with  $z^s \in B$ , then the resulting decomposition is correct.*

*Proof.* By induction on  $s$ .

$s = 1$  : Let  $\mathcal{B}_1 := \{B_0\}$ ,  $z^1 \in Z_N$ . Then, by definition of the full  $m$ -split,  $B_0$  is replaced by  $m$  boxes. It holds that

$$B_0 \setminus \left( \bigcup_{B \in \mathcal{B}_2} B \right) = B_0 \setminus \left( \bigcup_{i=1, \dots, m} \{z \in B_0 : z_i < z_i^1\} \right) = S_2(z^1),$$

thus,  $\mathcal{B}_2$  is correct.

$s \rightarrow s+1$  : Let  $\mathcal{B}_s$  be correct and let  $z^s \in Z_N$ . Let  $\bar{\mathcal{B}}_s \subset \mathcal{B}_s$  denote the set of all boxes  $B \in \mathcal{B}_s$  for which  $z^s \in B$  holds. Let  $I$  be the index set of these boxes and let  $Q := |\bar{\mathcal{B}}_s|$ . Now, let a full  $m$ -split with respect to  $z^s$  be applied to all  $B \in \bar{\mathcal{B}}_s$ , i.e., each of the boxes  $B^{I(q)}$ ,  $q = 1, \dots, Q$ , is replaced by  $m$  new boxes  $B_1^{I(q)}, \dots, B_m^{I(q)}$ ,  $q = 1, \dots, Q$  and

$$\bigcup_{\substack{i=1, \dots, m \\ q=1, \dots, Q}} B_i^{I(q)} = \bigcup_{B \in \bar{\mathcal{B}}_s} B \setminus S_2(z^s)$$

holds. Then

$$\begin{aligned} B_0 \setminus \left( \bigcup_{B \in \mathcal{B}_{s+1}} B \right) &= B_0 \setminus \left( \left( \bigcup_{B \in \mathcal{B}_s \setminus \bar{\mathcal{B}}_s} B \right) \cup \left( \bigcup_{\substack{i=1, \dots, m \\ q=1, \dots, Q}} B_i^{I(q)} \right) \right) \\ &= B_0 \setminus \left( \left( \bigcup_{B \in \mathcal{B}_s \setminus \bar{\mathcal{B}}_s} B \right) \cup \left( \bigcup_{B \in \bar{\mathcal{B}}_s} B \setminus S_2(z^s) \right) \right) = B_0 \setminus \left( \left( \bigcup_{B \in \mathcal{B}_s} B \setminus S_2(z^s) \right) \right) \\ &= \left( B_0 \setminus \left( \bigcup_{B \in \mathcal{B}_s} B \right) \right) \cup S_2(z^s) = \bigcup_{p=1, \dots, s} S_2(z^p). \end{aligned}$$

□

□

Note that all new boxes  $B \in \mathcal{B}_{s+1}$ ,  $s \geq 2$ , obtained from boxes in  $\bar{\mathcal{B}}_s$ , are defined as sets with open upper boundary, as we need to exclude  $z^s$  from the search region in order to prevent it from further generation. In practical applications, it will often be useful to replace the boxes by closed subsets and exclude  $z^s$  by using, for example, appropriate scalarization approaches.

Also note that we describe the boxes by their upper bound  $u$  only and that the lower bound of all boxes is kept constant. This means that our decomposition contains the union of the sets  $S_1(z^p) \setminus \{z^p\}$ ,  $1 \leq p \leq s$ , for all nondominated points  $z^p \in Z_N$  which have already been generated by the algorithm, even if these sets do not contain any feasible points. However, the splitting operation is simplified by including these sets, since a box is never split into more than  $m$  new boxes in this case.

---

**Algorithm 1** Algorithm with full  $m$ -split

---

**Input:** Image of the feasible set  $Z \subset \mathbb{R}^m$ , implicitly given by some problem formulation

```

1:  $Z_N := \emptyset$ ;  $\delta > 0$ ;
2: INITSTARTINGBOX( $Z, \delta$ );
3:  $s := 1$ ; // Initialize starting box
4: while  $\mathcal{B}_s \neq \emptyset$  do
5:   Choose  $B \in \mathcal{B}_s$ ;
6:    $z^s := \text{opt}(Z, u(B))$ ; // Solve subproblem
7:   if  $z^s = \emptyset$  then // Subproblem infeasible
8:      $\mathcal{B}_{s+1} := \mathcal{B}_s \setminus \{B\}$ ; // Remove (empty) box
9:   else
10:     $Z_N := Z_N \cup \{z^s\}$ ; // Save nondominated point
11:     $\mathcal{B}_{s+1} := \mathcal{B}_s$ ; // Copy set of current boxes
12:    GENERATENEWBOXES( $\mathcal{B}_s, z^s, z^I, \mathcal{B}_{s+1}$ );
13:   end if
14:    $s := s + 1$ ;
15: end while
Output: Set of nondominated points  $Z_N$ 

16: procedure INITSTARTINGBOX( $Z, \delta$ )
17:   for  $j = 1$  to  $m$  do // Compute bounds on  $Z$ 
18:      $z_j^I := \min\{z_j : z \in Z\}$ ;
19:      $z_j^M := \max\{z_j : z \in Z\} + \delta$ ;
20:      $u_j(B_0) := z_j^M$ ;
21:   end for
22:    $\mathcal{B}_1 := \{B_0\}$ ; // Initialize set of boxes
23:   return  $\mathcal{B}_1$ 
24: end procedure

25: procedure GENERATENEWBOXES( $\mathcal{B}_s, z^s, z^I, \mathcal{B}_{s+1}$ )
26:   for all  $\hat{B} \in \mathcal{B}_s$  do
27:     if  $z^s < u(\hat{B})$  then // Point is contained in box
28:       for  $i = 1$  to  $m$  do // Apply full  $m$ -split to  $\hat{B}$ 
29:         if  $z_i^s > z_i^I$  then
30:            $u(\hat{B}_i) := u(\hat{B})$ ; // Create a copy of  $\hat{B}$ 
31:            $u_i(\hat{B}_i) := z_i^s$ ; // Update upper bound
32:            $\mathcal{B}_{s+1} := \mathcal{B}_{s+1} \cup \{\hat{B}_i\}$ ; // Append new box
33:         end if
34:       end for
35:        $\mathcal{B}_{s+1} := \mathcal{B}_{s+1} \setminus \{\hat{B}\}$ ; // Remove box
36:     end if
37:   end for
38:   return  $\mathcal{B}_{s+1}$ 
39: end procedure

```

---

## 2.1 A generic algorithm based on the full $m$ -split

Algorithm 1 shows a basic algorithm using the full  $m$ -split. Due to Lemma 2.3, the algorithm is correct as it does not exclude regions from the search region which might contain further nondominated points. A problem formulation is given as input, which is denoted by  $Z$ . Note that this does not mean that the set of feasible outcomes is known explicitly, but it is to be understood as a substitute for the objective functions and the constraints. As long as  $\mathcal{B}_s$  contains unexplored boxes, a box  $B$  is selected according to some specified rule. As we are interested in generating the entire nondominated set, no special rule is employed in the following, and we may, for example, always take the first box in the list  $\mathcal{B}_s$ . The upper bounds of the chosen box  $B$  are used to determine the parameters of the selected scalarization. Thereby, the scalarization method can be chosen freely, as long as it is guaranteed that the method finds a nondominated point in  $B$  whenever there exists one. For example, the augmented weighted Tchebycheff scalarization is an appropriate method, see, e.g., Dächert et al. (2012) and Dächert (2014) for an adaptive parameter choice in the bicriteria and multicriteria case, respectively. The result of the subproblem is either a nondominated point  $z^s$  in the considered box  $B$  or the detection of infeasibility, which corresponds to the situation that  $B$  does not contain further nondominated points. In the latter case,  $B$  is removed from the list  $\mathcal{B}_s$  and the iteration is finished. Otherwise,  $z^s$  is saved and all boxes  $\hat{B} \in \mathcal{B}_s$  are identified that contain  $z^s$ . All these boxes are split with respect to all  $i \in \{1, \dots, m\}$  for which  $z_i^s > z_i^I$  holds and are replaced by the new boxes. The algorithm iterates until all boxes have been explored. Then the entire nondominated set has been detected.

## 2.2 Bicriteria case

For  $m = 2$ , Algorithm 1 is not only correct but also efficient, in the sense that the number of subproblems that need to be solved depends linearly on the number of nondominated points. As the decomposition does not contain redundant boxes, an upper bound on the number of boxes can easily be derived, which can be seen as follows. Let  $B_0$  denote the starting box and let  $z^1 \in B_0 \cap Z_N$  be the first generated point. Consider the two new boxes  $B_1, B_2$  replacing  $B_0$  in the first iteration. It holds that

$$\begin{aligned} B_1 \cap Z &= \{z \in B_0 : z_1 < z_1^1\} \cap Z = (\{z \in B_0 : z_1 < z_1^1\} \cap Z) \setminus S_1(z^1) \\ &= \{z \in B_0 : z_1 < z_1^1, z_2 > z_2^1\} \cap Z \end{aligned}$$

and, analogously,

$$B_2 \cap Z = \{z \in B_0 : z_2 < z_2^1\} \cap Z = \{z \in B_0 : z_2 < z_2^1, z_1 > z_1^1\} \cap Z,$$

thus,  $(B_1 \cap Z) \cap (B_2 \cap Z) = \emptyset$ . Therefore, the second generated point  $z^2 \in Z_N$  is contained in exactly one of the two boxes  $B_1, B_2$ . This box is again split into two new boxes whose intersections with  $Z$  are disjoint among themselves as well as from

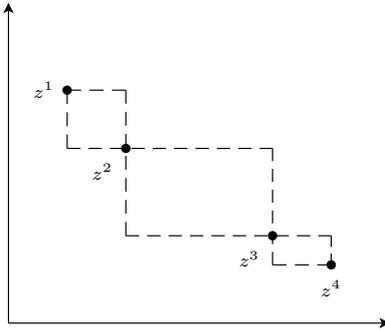


Figure 1: Decomposition of the search region for  $m = 2$

the box (intersected with  $Z$ ) which has not been changed in the current iteration. Repeating this argument, we see that for  $m = 2$ , no redundancy occurs. Therefore, we can easily indicate the running time of Algorithm 1 in the bicriteria case based on the knowledge that a new nondominated point lies in exactly one box. In the initialization phase,  $z^I$  and  $z^M$  are computed in order to define  $B_0$ . In every iteration, either a (new) nondominated point is generated or a box is discarded from the search region. For every new nondominated point  $z^s > z^I$ , two new boxes replace the currently investigated box, and for each of the two lexicographic optimal points (defining the ideal point) the current box is replaced by one new box. So, the total number of iterations is  $2|Z_N| - 1$ , cf. Chalmet et al. (1986) and Ralphs et al. (2006). In Figure 1, we illustrate the search region after having generated and inserted four nondominated points  $z^1, z^2, z^3$  and  $z^4$ . If we assume that these points build the entire nondominated set, Algorithm 1 terminates after seven iterations.

### 2.3 Multicriteria case ( $m \geq 3$ )

The application of the full  $m$ -split in the case of three criteria is illustrated in Figure 2. Different from the bicriteria case, a nondominated point may lie in the intersection of multiple boxes for  $m \geq 3$ . If we perform the full  $m$ -split in every box which contains the current nondominated point, we typically create nested and, thus, redundant subboxes. This is illustrated in the following example.

**Example 2.4.** *Let  $m = 3$  and let the initial search region be given by*

$$B_0 := \{z \in Z : 0 \leq z_i \leq 5 \forall i = 1, 2, 3\}.$$

*Assume that the first nondominated point that is generated is  $z^1 = (2, 2, 2)^\top$ . Performing a full 3-split in  $B_0$  with respect to  $z^1$  replaces the search region  $B_0$  by the three sets*

$$B_{1,i} := \{z \in B_0 : z_i < 2\}, \quad i = 1, 2, 3.$$

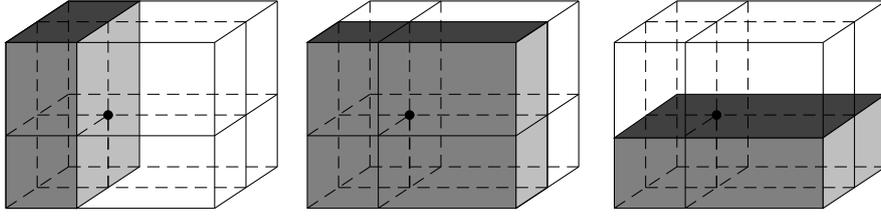


Figure 2: Boxes  $B_i, i = 1, 2, 3$ , in  $\mathbb{R}^3$  obtained by a full 3-split of the initial search region; the nondominated point with respect to which the split is performed is represented by a dot

Let  $z^2 = (1, 1, 4)^\top$  be the next nondominated point that is generated. It holds that  $z^2 \in B_{11}$  as well as  $z^2 \in B_{12}$ , but  $z^2 \notin B_{13}$ . Performing a full 3-split in  $B_{11}$  yields

$$\begin{aligned} B_{21} &:= \{z \in B_0 : z_1 < 1\}, \\ B_{22} &:= \{z \in B_0 : z_1 < 2, z_2 < 1\}, \\ B_{23} &:= \{z \in B_0 : z_1 < 2, z_3 < 4\}. \end{aligned}$$

Performing a full 3-split in  $B_{12}$  yields

$$\begin{aligned} B'_{21} &:= \{z \in B_0 : z_1 < 1, z_2 < 2\}, \\ B'_{22} &:= \{z \in B_0 : z_2 < 1\}, \\ B'_{23} &:= \{z \in B_0 : z_2 < 2, z_3 < 4\}. \end{aligned}$$

It holds that  $B'_{21} \subset B_{21}$  and  $B_{22} \subset B'_{22}$ , thus, the boxes  $B_{22}$  and  $B'_{21}$  are redundant in the decomposition of  $B_0$ .

If redundant boxes are kept in the decomposition, this typically increases the running time of the algorithm, as additional, unnecessary subproblems are solved. Depending on the given problem, this may be time-consuming. Thus, redundant boxes should be detected and removed immediately. In the following, we analyze under which conditions redundant boxes occur. We first define our notion of non-redundancy.

**Definition 2.5** (Non-redundant decomposition). *Let  $B_0$  denote the starting box and let  $\mathcal{B}_s$  be a correct decomposition at the beginning of iteration  $s \geq 1$ . We call  $\mathcal{B}_s$  (and every  $B \in \mathcal{B}_s$ ) non-redundant, if for every pair of boxes  $B, \tilde{B} \in \mathcal{B}_s, B \neq \tilde{B}$ , it holds:*

$$\exists i \in \{1, \dots, m\} : u_i(B) < u_i(\tilde{B}) \text{ and } \exists j \in \{1, \dots, m\} : u_j(B) > u_j(\tilde{B}).$$

*In case that  $u(B) \leq u(\tilde{B})$  we say that box  $\tilde{B}$  dominates  $B$ .*

Note that the definition of a dominated box is somehow opposite to the definition of a dominated point. While  $u \in \mathbb{R}^m$  is dominated by  $u' \in \mathbb{R}^m$  if  $u \geq u'$ , box  $B$  is dominated by  $B'$  if  $u(B) \leq u(B')$ .

For simplicity, we make a technical assumption concerning the values of the nondominated points that will be removed later. Moreover, we define our general setting.

**Assumption 2.6.** *Let the following hold:*

1. *For all nondominated points  $z^p \in Z_N, p = 1, \dots, s$ , generated up to iteration  $s \geq 1$ , it holds that  $z_j^p \neq z_j^q$  for all  $j = 1, \dots, m$  and  $1 \leq q < p$ .*
2. *The starting box  $B_0$  is non-empty, and  $\mathcal{B}_1 := \{B_0\}$  denotes the initial decomposition of the search region.*
3. *For every  $1 \leq p \leq s$ ,  $\mathcal{B}_p$  is a correct, non-redundant decomposition of the search region. By  $\overline{\mathcal{B}}_p := \{B \in \mathcal{B}_p : z^p \in B\}$  we denote the subset of boxes in iteration  $p$  containing  $z^p$ .*

**Lemma 2.7** (Generation of redundant boxes). *Let Assumption 2.6 be satisfied. If we apply a full  $m$ -split to every box  $B \in \overline{\mathcal{B}}_s$ , then redundancy can only occur among the ‘descendants’ of two different boxes which have been split with respect to the same component in this iteration.*

*Proof.* We first show that no redundancy occurs between two boxes if at least one of the boxes has not been changed in the current iteration. Therefore, consider two arbitrary boxes  $B, \tilde{B} \in \mathcal{B}_s$  where  $\tilde{B} \in \mathcal{B}_s \setminus \overline{\mathcal{B}}_s$ :

1. If  $B \in \mathcal{B}_s \setminus \overline{\mathcal{B}}_s$ , both boxes remain unchanged in the current iteration and, thus, due to Assumption 2.6 (3) are non-redundant.
2. If  $B \in \overline{\mathcal{B}}_s$ , none of the boxes obtained from a split in  $B$  can dominate  $\tilde{B}$ , as  $B$  does not dominate  $\tilde{B}$  and the upper bound of  $B$  is only decreased by the split. Conversely,  $\tilde{B}$  cannot dominate any of the boxes obtained from a split in  $B$ , since  $\tilde{B} \in \mathcal{B}_s \setminus \overline{\mathcal{B}}_s$  implies that  $u_j(\tilde{B}) \leq z_j^s$  for at least one  $j \in \{1, \dots, m\}$ . As  $z^s < u(B)$ , for every  $B_i, i = 1, \dots, m$ , resulting from a split of  $B$  it holds that  $z_i^s = u_i(B_i)$  and  $z_j^s < u_j(B_i)$  for all  $j \neq i$ . Thus,  $\tilde{B}$  dominates  $B_i$  if and only if  $z_i^s = u_i(\tilde{B})$  holds. This, however, is excluded by Assumption 2.6 (1).

Therefore, redundancy can only occur among newly generated boxes. Consider two boxes  $B_i \neq \hat{B}_j$  obtained from  $B, \hat{B} \in \overline{\mathcal{B}}_s$  (the case  $B = \hat{B}$  is included) that are split with respect to components  $i \neq j$ . Then  $u_i(B_i) = z_i^s < u_i(\hat{B}_j)$  and  $u_j(B_i) > z_j^s = u_j(\hat{B}_j)$ , thus, none of the boxes can dominate the other one. It follows that redundancy can only occur among the descendants of two different boxes split with respect to the same component.  $\square$   $\square$

**Corollary 2.8.** *Let Assumption 2.6 hold. If only one box is split in some iteration, then all  $m$  resulting subboxes are non-redundant. In particular, the boxes obtained in the first iteration are always non-redundant.*

**Corollary 2.9.** *Let Assumption 2.6 hold. Let two boxes  $B, \hat{B} \in \overline{\mathcal{B}}_s$  be split with respect to the same component  $i = 1, \dots, m$ . Then the resulting boxes  $B_i, \hat{B}_i$  are non-redundant if and only if there exists an index  $p \neq i$  such that  $u_p(B_i) < u_p(\hat{B}_i)$  and there exists an index  $q \neq i$  such that  $u_q(B_i) > u_q(\hat{B}_i)$ .*

These observations allow to detect redundant boxes by checking specific boxes of the current decomposition. Translating this to an algorithm for arbitrary  $m \geq 3$ , we apply, in every iteration, a full  $m$ -split to every box containing the current nondominated point. If more than one box is split in one iteration, we compare the upper bounds of those new boxes that were generated with respect to the same component pairwise to detect redundancy. The respective boxes are then removed from the decomposition. A similar approach is followed in Przybylski et al. (2010) where a new upper bound is compared to all other upper bounds excluding the one from which it has been derived. In doing so, ‘dominated’ upper bounds are filtered out directly.

A corresponding algorithm can be improved further if redundant boxes are identified without comparing their upper bounds to some or all other boxes. In the next section we develop an explicit criterion for tricriteria problems that indicates already before the split is performed whether the resulting box is redundant or not, and, thus, allows to maintain only non-redundant boxes in the decomposition. We prove that the number of non-redundant boxes or, equivalently, the number of subproblems to be solved in the course of an algorithm based on such an improved split operation depends linearly on the number of nondominated points.

### 3 A split criterion to avoid redundant boxes for $m = 3$

According to Definition 2.5, a non-redundant box can be characterized as follows. A box is non-redundant if and only if it contains a non-empty subset which is not part of any other box of the decomposition. These subsets are studied in the following.

**Definition 3.1** (Individual subsets). *Let  $\mathcal{B}_s, s \geq 1$  be a non-redundant decomposition. For every  $B \in \mathcal{B}_s$ , the set*

$$V(B) := B \setminus \left( \bigcup_{\tilde{B} \in \mathcal{B}_s \setminus \{B\}} \tilde{B} \right) \quad (5)$$

*is called individual subset of  $B$ .*

Obviously, for every  $B \in \mathcal{B}_s, s \geq 1$ , it holds that  $V(B) \subseteq B$  and  $V(B) \cap V(\tilde{B}) = \emptyset$  for every  $\tilde{B} \in \mathcal{B}_s, \tilde{B} \neq B$ . Figure 3 shows the individual subsets of the three boxes  $B_i, i = 1, 2, 3$ , in  $\mathbb{R}^3$ , obtained by a full 3-split of the initial box, which are depicted in Figure 2.

Now, maintaining only non-redundant boxes in the decomposition of the search region is equivalent to maintaining boxes with non-empty individual subsets. An explicit split criterion should indicate already before performing the split whether a given box will have a non-empty individual subset after having performed the split. To this end, we have to describe the individual subsets explicitly. For  $m = 3$ , we observe that the individual subset of a box is bounded by the neighbors of that box. After defining the neighbor of a box with respect to a certain component, we show its existence and indicate the respective neighboring boxes by a constructive proof.

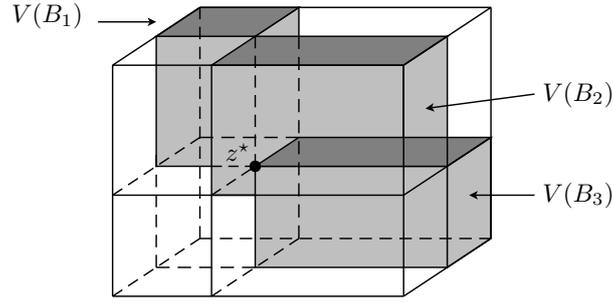


Figure 3: Individual subsets  $V(B_i)$ ,  $i = 1, 2, 3$ , in  $\mathbb{R}^3$  obtained by a full 3-split of the initial search region with respect to  $z^* \in Z_N$

**Definition 3.2** (Neighbor of a box). Let  $\mathcal{B}_s, s \geq 1$  be a non-redundant decomposition of the search region, and let  $\underline{u}_i := \min\{u_i(B) : B \in \mathcal{B}_s\}$ . Let any  $\bar{B} \in \mathcal{B}_s$  be given. For every  $i \in \{1, 2, 3\}$ , for which  $u_i(\bar{B}) > \underline{u}_i$ , we call a box  $\hat{B} \in \mathcal{B}_s \setminus \{\bar{B}\}$  that satisfies

$$u_i(\hat{B}) < u_i(\bar{B}) \quad (6)$$

$$u_j(\hat{B}) > u_j(\bar{B}) \quad \text{for some } j \neq i \quad (7)$$

$$u_k(\hat{B}) \geq u_k(\bar{B}) \quad \text{for } k \neq i, j \quad (8)$$

and

$$u_i(\hat{B}) = \max\{u_i(B) : B \in \mathcal{B}_s \setminus \{\bar{B}\}, u_i(B) < u_i(\bar{B})\} \quad (9)$$

the neighbor of  $\bar{B}$  with respect to  $i$  at the beginning of iteration  $s$ , denoted by  $B_i^s(\bar{B})$ .

**Example 3.3.** Consider Figure 2, which depicts the three boxes that are obtained in the first iteration. At the beginning of iteration  $s = 2$ , it holds that  $B_1^2(B_2) = B_1$ , since  $B_1$  is the unique box satisfying (6)–(9) for  $\bar{B} := B_2$ . Analogously,  $B_3^2(B_2) = B_3$  holds. A neighbor  $B_2^2(B_2)$  is not defined as  $u_2(B_2) = \underline{u}_2$ .

The following lemma shows that, under appropriate assumptions, for every box  $\bar{B} \in \mathcal{B}_s$  and every component  $i \in \{1, 2, 3\}$  for which  $u_i(\bar{B}) > \underline{u}_i$  holds there exists a unique neighbor  $B_i^s(\bar{B})$  satisfying (6)–(9) of Definition 3.2. These neighbors  $B_i^s(\bar{B}), i \in \{1, 2, 3\}$ , which will be indicated with the help of a constructive proof will turn out to be the boxes that define the individual subset of  $\bar{B}$ .

**Assumption 3.4.** Let the following hold:

1. For all nondominated points  $z^p \in Z_N, p = 1, \dots, s$ , generated up to iteration  $s \geq 1$ , it holds that  $z_j^p \neq z_j^q$  for all  $j \in \{1, 2, 3\}$  and  $1 \leq q < p$ .
2. The starting box  $B_0$  is non-empty, and  $\mathcal{B}_1 := \{B_0\}$  denotes the initial decomposition of the search region.

3. For every iteration  $1 \leq p \leq s$ , the set  $\mathcal{B}_{p+1}$  is obtained from  $\mathcal{B}_p$  by applying a full 3-split to every  $B \in \overline{\mathcal{B}}_p$ , where  $\overline{\mathcal{B}}_p := \{B \in \mathcal{B}_p : z^p \in B\}$ . All redundant boxes are removed from  $\mathcal{B}_{p+1}$  at the end of the respective iteration  $p$ .

Note that Assumption 3.4 substantiates Assumption 2.6 by specifying that the correct, non-redundant decompositions are obtained by iterative full 3-splits and that redundant boxes are removed.

As the proof of the following lemma is rather technical, we illustrate it with the help of two examples. In both examples,  $u(B_0) := (5, 5, 5)^\top$  is assumed, and  $z^1 := (2, 2, 2)^\top$  is inserted as a first nondominated point. In the first example, depicted in Figure 4,  $z^2 := (3, 1, 4)^\top$  is inserted as a second nondominated point. In the second example, depicted in Figure 5,  $z^2 := (1, 1, 4)^\top$  represents the second nondominated point.

**Lemma 3.5.** *Let Assumption 3.4 be satisfied. Then, for every  $s \geq 2$ , every  $\bar{B} \in \mathcal{B}_s$  and every  $i \in \{1, 2, 3\}$ , for which  $u_i(\bar{B}) > \underline{u}_i := \min\{u_i(B) : B \in \mathcal{B}_s\}$  holds, there exists a unique neighbor  $B_i^s(\bar{B}) \in \mathcal{B}_s$  satisfying (6)–(9). Particularly,  $u_k(B_i^s(\bar{B})) = u_k(\bar{B})$  holds, i.e.,  $B_i^s(\bar{B})$  satisfies*

$$u_i(B_i^s(\bar{B})) < u_i(\bar{B}) \tag{10}$$

$$u_j(B_i^s(\bar{B})) > u_j(\bar{B}) \quad \text{for some } j \neq i \tag{11}$$

$$u_k(B_i^s(\bar{B})) = u_k(\bar{B}) \quad \text{for } k \neq i, j, \tag{12}$$

and

$$u_i(B_i^s(\bar{B})) = \max\{u_i(B) : B \in \mathcal{B}_s, u_i(B) < u_i(\bar{B})\}. \tag{13}$$

If  $u_i(\bar{B}) = \underline{u}_i$ , then we set  $B_i^s(\bar{B}) := \emptyset$ .

*Proof.* By induction on  $s$ .

$s = 2$ :  $\mathcal{B}_1 = \{B_0\} = \overline{\mathcal{B}}_1$ , as  $z^1 < u(B_0) = z^M$ . The starting box is split into three subboxes  $\hat{B}_i := \{z \in B_0 : z_i < z_i^1\}, i \in \{1, 2, 3\}$ . Due to Lemma 2.7, the new boxes are non-redundant, thus,  $\mathcal{B}_2 = \{\hat{B}_1, \hat{B}_2, \hat{B}_3\}$ . Consider  $\hat{B}_i$  for fixed  $i \in \{1, 2, 3\}$ : As  $u_i(\hat{B}_i) = z_i^1 < u_i(\hat{B}_j)$  for every  $j \neq i$ , by definition,  $B_i^2(\hat{B}_i) = \emptyset$ . As  $u_j(\hat{B}_i) > z_j^1 = \min\{u_j(B) : B \in \mathcal{B}_2\}$ , for every  $j \neq i$ , a unique neighbor  $B_j^2(\hat{B}_i)$  exists and, obviously,  $B_j^2(\hat{B}_i) = \hat{B}_j$  holds.

$s \rightarrow s + 1$ : We assume that unique neighbors satisfying (6)–(9) exist for all  $B \in \mathcal{B}_s$  and that, additionally, (12) holds. We insert  $z^s \in Z_N$ . Due to the correctness of the full  $m$ -split, there exists at least one box which is split, i.e.,  $|\overline{\mathcal{B}}_s| \geq 1$ .

Case 1:  $|\overline{\mathcal{B}}_s| = 1$ , i.e., only one box is split. Let  $\hat{B}$  be this box and let  $\hat{B}_i, i \in \{1, 2, 3\}$ , be the subboxes resulting from the split. Due to Lemma 2.7, the new boxes are non-redundant, thus,  $\mathcal{B}_{s+1} = (\mathcal{B}_s \setminus \{\hat{B}\}) \cup \{\hat{B}_1, \hat{B}_2, \hat{B}_3\}$ . The corresponding box in Figure 4 (a) is  $\hat{B} = B_{12}$ , which is replaced by  $\hat{B}_1 = B_{21}, \hat{B}_2 = B_{22}, \hat{B}_3 = B_{23}$ , see Figure 4 (b).

Consider an arbitrary box  $\hat{B}_i, i \in \{1, 2, 3\}$ . Then the following holds:

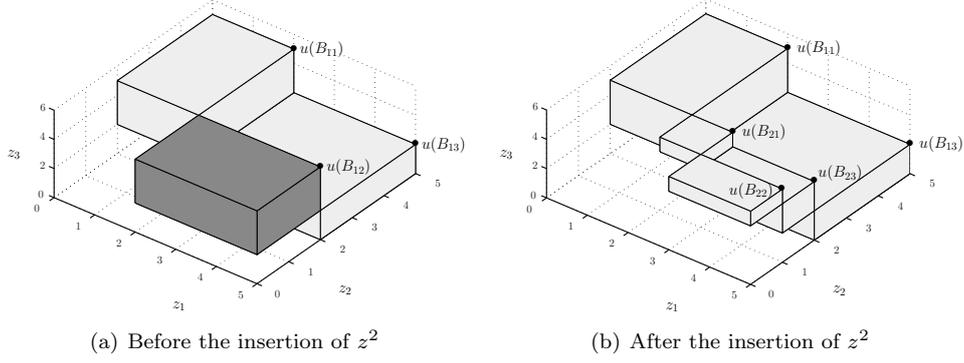


Figure 4: Visualization of the upper bound vectors  $u(B)$  in case 1 of the proof of Lemma 3.5:  $z^2 = (3, 1, 4)^\top$  lies only in box  $B_{12}$  with  $u(B_{12}) = (5, 2, 5)^\top$ , i.e.,  $|\bar{\mathcal{B}}_2| = 1$ . For a better illustration, the individual subsets  $V(B)$  of all boxes are depicted.

- (i)  $B_i^{s+1}(\hat{B}_i) = B_i^s(\hat{B}_i)$ :

$\hat{B}_i$  is the only new box  $B$  with  $u_i(B) = z_i^s$  and there is no other new box  $B$  satisfying  $u_i(B) < z_i^s$ . Hence,  $B_i^{s+1}(\hat{B}_i) \notin \{\hat{B}_1, \hat{B}_2, \hat{B}_3\}$ . If  $B_i^s(\hat{B}_i) = \emptyset$  then  $B_i^{s+1}(\hat{B}_i) = \emptyset$ . Otherwise, i.e., if  $B_i^s(\hat{B}_i)$  exists,  $u_i(B_i^s(\hat{B}_i)) < u_i(\hat{B}_i)$ ,  $u_j(B_i^s(\hat{B}_i)) > u_j(\hat{B}_i)$  for some  $j \neq i$  and  $u_k(B_i^s(\hat{B}_i)) = u_k(\hat{B}_i)$  for  $k \neq i, j$  hold due to the induction hypothesis. Now  $u_i(B_i^s(\hat{B}_i)) \leq z_i^s$  must be satisfied, as otherwise  $B_i^s(\hat{B}_i) \in \bar{\mathcal{B}}_s$  would hold, a contradiction to the assumption that  $\bar{\mathcal{B}}_s = \{\hat{B}_i\}$ . Moreover,  $u_i(B_i^s(\hat{B}_i)) = z_i^s$  is excluded due to Assumption 3.4 (1). Thus, (10)-(13) holds for  $B_i^{s+1}(\hat{B}_i) = B_i^s(\hat{B}_i)$ . The uniqueness of  $B_i^s(\hat{B}_i)$  follows from the induction hypothesis.

In Figure 4 (b), an example of this case is given by  $B_1^3(B_{21}) = B_1^2(B_{12}) = B_{11}$ .

- (ii)  $B_j^{s+1}(\hat{B}_i) = \hat{B}_j$  for all  $j \neq i$ :

$\hat{B}_j$  is the only new box  $B$  with  $u_j(B) = z_j^s$  and there is no other new box  $B$  satisfying  $u_j(B) < z_j^s$ . Furthermore,  $\hat{B}_j$  satisfies (10)-(12), as  $u_j(\hat{B}_j) < u_j(\hat{B}_i)$ ,  $u_i(\hat{B}_j) > u_i(\hat{B}_i)$  and  $u_k(\hat{B}_j) = u_k(\hat{B}_i)$  for  $k \neq i, j$  hold. Moreover,  $u_j(\hat{B}_j)$  is maximal, as  $u_j(\hat{B}_j) = z_j^s \geq u_j(B_j^s(\hat{B}_i))$  if  $B_j^s(\hat{B}_i) \neq \emptyset$  and  $u_j(B_j^s(\hat{B}_i))$  maximal due to the induction hypothesis. As  $z_j^s = u_j(B_j^s(\hat{B}_i))$  is excluded due to Assumption 3.4 (1), the uniqueness of  $B_j^{s+1}(\hat{B}_i)$  follows.

In Figure 4 (b), examples are given by  $B_2^3(B_{21}) = B_{22}$  and  $B_3^3(B_{21}) = B_{23}$ .

Now consider an arbitrary box  $B \neq \hat{B}$ . Then the following holds:

- (iii) If  $B_i^s(B) \neq \hat{B}_i$  for some  $i \in \{1, 2, 3\}$ , then  $B_i^{s+1}(B)$  remains unchanged:

Assume that  $B_i^{s+1}(B)$  changes due to the split of  $\hat{B}_i$ . Then the only candidate

for  $B_i^{s+1}(B)$  is  $\hat{B}_i$  and only in case that  $u_i(\hat{B}_i) < u_i(B) \leq u_i(\hat{B})$ , as otherwise  $B_i^s(B) = \hat{B}$  would have been valid. Now suppose that  $B_i^{s+1}(B) = \hat{B}_i$ . As  $u_j(\hat{B}_i) = u_j(\hat{B})$  for all  $j \neq i$  and, by definition of  $B_i^s(B)$ ,  $u_j(\hat{B}_i) \geq u_j(B)$  for all  $j \neq i$ , we have that  $u_j(\hat{B}) \geq u_j(B)$  for all  $j \neq i$  and hence  $u_l(\hat{B}) \geq u_l(B)$  for all  $l \in \{1, 2, 3\}$ , a contradiction to  $\mathcal{B}_s$  being non-redundant. Thus,  $B_i^{s+1}(B)$  remains unchanged.

In the example depicted in Figure 4 (b), let  $B = B_{13}$ . As  $B_1^2(B_{13}) = B_{11} \neq B_{12}$ , the neighbor remains unchanged, thus,  $B_1^3(B_{13}) = B_{11}$ .

- (iv) If  $B_i^s(B) = \hat{B}$  for some  $i = 1, \dots, m$ , then  $B_i^{s+1}(B) = \hat{B}_j$  with  $j$  being the unique index for which  $u_j(\hat{B}) > u_j(B)$  holds:

By the induction hypothesis,  $u_i(\hat{B}) < u_i(B)$ ,  $u_j(\hat{B}) > u_j(B)$  for some  $j \neq i$  and  $u_k(\hat{B}) = u_k(B)$  for  $k \neq i, j$ . As  $z_i^s = u_i(\hat{B}_i) < u_i(\hat{B})$  and  $u_l(\hat{B}_i) = u_l(\hat{B})$  for all  $l \neq i$ ,  $\hat{B}_i$  is a candidate for  $B_i^{s+1}(B)$ . As  $B \notin \bar{\mathcal{B}}_s$  and  $z_i^s < u_l(B)$  for all  $l \neq j$  it follows that  $z_j^s \geq u_j(B)$ , and, due to Assumption 3.4 (1),  $z_j^s > u_j(B)$ . Thus,  $u_i(\hat{B}_j) = u_i(\hat{B}) < u_i(B)$ ,  $u_j(\hat{B}_j) = z_j^s > u_j(B)$  and  $u_k(\hat{B}_j) = u_k(\hat{B}) = u_k(B)$  hold. Therefore,  $\hat{B}_j$  is the unique other candidate for  $B_i^{s+1}(B)$  besides  $\hat{B}_i$ . As  $u_i(\hat{B}_i) < u_i(\hat{B}_j) = u_i(\hat{B})$ ,  $\hat{B}_j$  is the unique neighbor  $B_i^{s+1}(B)$  after the split.

In the example depicted in Figure 4 (b),  $B_2^2(B_{13}) = B_{12} = \hat{B}$  and  $B_2^3(B_{13}) = B_{23}$ . Box  $B_{22}$  is the unique other candidate for  $B_2^3(B_{13})$ , but as  $u_2(B_{22}) < u_2(B_{23})$  it holds that  $B_2^3(B_{13}) = B_{23}$ .

Case 2:  $|\bar{\mathcal{B}}_s| > 1$ .

By definition of  $\bar{\mathcal{B}}_s$ , it holds that  $z_i^s < u_i(B)$  for all  $i \in \{1, 2, 3\}$  and  $B \in \bar{\mathcal{B}}_s$ , thus,  $z_i^s < \min\{u_i(B) : B \in \bar{\mathcal{B}}_s\}$  for all  $i \in \{1, 2, 3\}$ . According to Lemma 2.7 and Corollary 2.9, redundancy occurs only for boxes  $\hat{B}, \tilde{B} \in \bar{\mathcal{B}}_s$  which are split with respect to the same component  $i \in \{1, 2, 3\}$  (i.e.,  $u_i(\hat{B}_i) = u_i(\tilde{B}_i) = z_i^s$ ) and for which  $u_l(\hat{B}_i) \geq u_l(\tilde{B}_i)$  or  $u_l(\hat{B}_i) \leq u_l(\tilde{B}_i)$  holds for all  $l \neq i$ . By assumption, those boxes are removed, i.e.,  $\mathcal{B}_{s+1}$  contains only non-redundant boxes. We illustrate this case by the example depicted in Figure 5.

Let  $\bar{\mathcal{B}}_s = \{\hat{B}^1, \dots, \hat{B}^P\}$  with  $P \in \mathbb{N}, P \geq 2$ . The corresponding boxes in Figure 5 (a) are  $\hat{B}^1 = B_{11}$  and  $\hat{B}^2 = B_{12}$ . For every  $i \in \{1, 2, 3\}$ , let  $I_i \subseteq \{1, \dots, P\}$  be the index set of the boxes from  $\bar{\mathcal{B}}_s$  whose split with respect to  $i$  yields a non-redundant box. Note that  $I_i \neq \emptyset$  for every  $i \in \{1, 2, 3\}$ , which can be seen as follows: Consider an arbitrary box  $B \in \bar{\mathcal{B}}_s$ . Applying the full 3-split to  $B$  results in three new boxes. Now any of the resulting boxes is removed if and only if there exists another box that dominates it. According to Lemma 2.7 the dominating box must have been created by a split with respect to the same component as the dominated box. Therefore,  $I_i \neq \emptyset$  for every  $i \in \{1, 2, 3\}$ . We set  $Q_i := |I_i| \geq 1$  for every  $i \in \{1, 2, 3\}$ . Furthermore, let  $\bar{u}_i := \max\{u_i(B), B \in \bar{\mathcal{B}}_s\}$  for all  $i \in \{1, 2, 3\}$  in the following, which is well defined as  $\bar{\mathcal{B}}_s \neq \emptyset$ .

In the example depicted in Figure 5 (b),  $Q_1 = Q_2 = 1$  and  $Q_3 = 2$ . Moreover, from Figure 5 (a) we see that  $\bar{u} = (5, 5, 5)^\top$ .

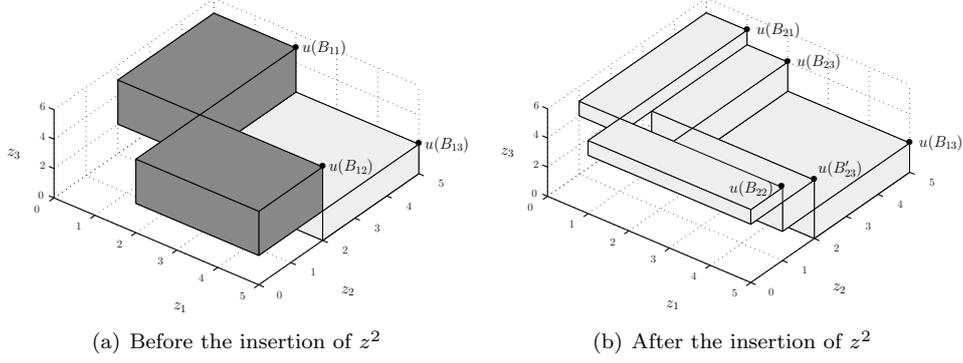


Figure 5: Visualization of the upper bound vectors  $u(B)$  in case 2 of the proof of Lemma 3.5:  $z^2 = (1, 1, 4)^\top$  lies in the two boxes  $B_{11}$  with  $u(B_{11}) = (2, 5, 5)^\top$  and  $B_{12}$  with  $u(B_{12}) = (5, 2, 5)^\top$ , i.e.,  $|\bar{\mathcal{B}}_2| = 2$ . For a better illustration, the individual subsets  $V(B)$  of all boxes are depicted.

Consider now  $i$  arbitrary but fixed. Let  $\hat{B}^{I_i(1)}, \dots, \hat{B}^{I_i(Q_i)}$  denote the boxes whose split with respect to component  $i$  yields a non-redundant box. As  $u_i(\hat{B}_i^{I_i(q)}) = z_i^s$  holds for all  $\hat{B}_i^{I_i(q)} \in \mathcal{B}_{s+1}$ ,  $q = 1, \dots, Q_i$ , as  $m = 3$  and as we assume non-redundancy, we can order the boxes with respect to their upper bounds increasingly by some component  $j \neq i$  and decreasingly by component  $k \neq i, j$ , i.e.,

$$z_j^s < u_j(\hat{B}_i^{I_i(1)}) < u_j(\hat{B}_i^{I_i(2)}) < \dots < u_j(\hat{B}_i^{I_i(Q_i)}), \quad (14)$$

$$u_k(\hat{B}_i^{I_i(1)}) > u_k(\hat{B}_i^{I_i(2)}) > \dots > u_k(\hat{B}_i^{I_i(Q_i)}) > z_k^s. \quad (15)$$

Thereby,  $u_j(\hat{B}_i^{I_i(Q_i)}) = u_j(\hat{B}_i^{I_i(1)}) = \bar{u}_j$  holds, since in the other case, i.e., if there was some  $\tilde{B} \in \bar{\mathcal{B}}_s$  with  $u_j(\tilde{B}) > u_j(\hat{B}_i^{I_i(Q_i)})$ , either  $\tilde{B}$  would have been the last box in (14) with index  $I_i(Q_i)$  or  $\hat{B}_i^{I_i(Q_i)}$  would have been dominated by  $\tilde{B}$ , both in contradiction to the construction. Analogously,  $u_k(\hat{B}_i^{I_i(1)}) = u_k(\hat{B}_i^{I_i(Q_i)}) = \bar{u}_k$  must hold.

In the example depicted in Figure 5 (a), consider  $i = 3$  and, without loss of generality, let  $\hat{B}^{I_3(1)} = B_{11}$  and  $\hat{B}^{I_3(2)} = \hat{B}^{I_3(Q_3)} = B_{12}$ . The upper bounds  $u(B_{11}) = (2, 5, 5)^\top$  and  $u(B_{12}) = (5, 2, 5)^\top$  can be ordered increasingly with respect to component  $j = 1$  and decreasingly with respect to component  $k = 2$ . It holds that  $u_1(\hat{B}_3^{I_3(Q_3)}) = 5 = \bar{u}_1$  and  $u_2(\hat{B}_3^{I_3(1)}) = 5 = \bar{u}_2$ .

If  $u_i(\hat{B}_i^{I_i(1)}) = \max\{u_i(B) : B \in \mathcal{B}_s, u_k(B) = \bar{u}_k\} =: \bar{u}_{i,k}$  holds, then the split of  $\hat{B}_i^{I_i(1)}$  with respect to  $j$  generates a non-redundant box, too, and, depending on the chosen enumeration,  $I_i(1)$  either equals  $I_j(1)$  or  $I_j(Q_j)$ . Without loss of generality, we can set  $I_i(1) = I_j(1)$ . Otherwise, i.e., if  $u_i(\hat{B}_i^{I_i(1)}) < \bar{u}_{i,k}$  holds, then  $\hat{B}_i^{I_i(1)}$  is dominated by a unique box  $\tilde{B} \in \bar{\mathcal{B}}_s$  with  $u_k(\tilde{B}) = \bar{u}_k$  and  $u_i(\tilde{B}) = \bar{u}_{i,k}$ . Then  $\tilde{B} = \hat{B}_i^{I_i(1)}$  holds.

Analogously, if  $u_i(\hat{B}^{I_i(Q_i)}) = \max\{u_i(B) : B \in \mathcal{B}_s, u_j(B) = \bar{u}_j\} =: \bar{u}_{i,j}$  holds, then the split of  $\hat{B}^{I_i(Q_i)}$  with respect to  $k$  generates a non-redundant box, too, and, without loss of generality, we can identify  $I_i(Q_i) = I_k(Q_k)$ . Otherwise, i.e., if  $u_i(\hat{B}^{I_i(Q_i)}) < \bar{u}_{i,j}$  holds, then  $\hat{B}_k^{I_i(Q_i)}$  is dominated by a unique box  $\tilde{B} \in \bar{\mathcal{B}}_s$  with  $u_j(\tilde{B}) = \bar{u}_j$  and  $u_i(\tilde{B}) = \bar{u}_{i,j}$ . Then  $\tilde{B} = \hat{B}^{I_k(Q_k)}$  holds.

Note that if  $Q_i = 1$ , then  $\hat{B}^{I_i(1)} = \hat{B}^{I_i(Q_i)} =: \hat{B}$  and  $u_j(\hat{B}) = \bar{u}_j$  as well as  $u_k(\hat{B}) = \bar{u}_k$  hold. In this case,  $u_i(\hat{B}) < \bar{u}_i$  must be satisfied, as otherwise  $\hat{B}$  would dominate any other box in  $\bar{\mathcal{B}}_s$ , a contradiction to  $|\bar{\mathcal{B}}_s| > 1$  and  $\mathcal{B}_s$  being non-redundant.

In the example depicted in Figure 5, consider  $i = 3$  and  $k = 2$ . It holds that  $u_3(\hat{B}^{I_3(1)}) = 5 = \max\{u_3(B) : B \in \mathcal{B}_s, u_2(B) = 5\}$ . The split of  $\hat{B}^{I_3(1)}$  with respect to  $j = 1$  generates the non-redundant box  $B_{21}$ . If we consider  $i = 1$ , then  $u_1(\hat{B}^{I_1(1)}) = u_1(B_{11}) = 2 < 5 = \max\{u_1(B) : B \in \mathcal{B}_s, u_3(B) = 5\}$ , hence, the split of  $B_{11}$  with respect to component  $j = 2$  must be redundant, and, indeed, the resulting box is dominated by  $B_{22}$ .

Analogously to Case 1, we will now indicate the neighbor boxes explicitly. Therefore, consider  $\hat{B}_i^{I_i(q)} \in \mathcal{B}_{s+1}$  for fixed  $i \in \{1, 2, 3\}, q \in \{1, \dots, Q_i\}$ . It holds that

(i)  $B_i^{s+1}(\hat{B}_i^{I_i(q)}) = B_i^s(\hat{B}_i^{I_i(q)})$ :

Assume  $B_i^s(\hat{B}_i^{I_i(q)}) \in \bar{\mathcal{B}}_s$ . By definition of  $B_i^s$ ,  $u_i(B_i^s(\hat{B}_i^{I_i(q)})) < u_i(\hat{B}_i^{I_i(q)})$  and  $u_l(B_i^s(\hat{B}_i^{I_i(q)})) \geq u_l(\hat{B}_i^{I_i(q)})$  for all  $l \neq i$  hold. But then, by an  $i$ -split of  $\hat{B}_i^{I_i(q)}$  and  $B_i^s(\hat{B}_i^{I_i(q)})$ , the box  $\hat{B}_i^{I_i(q)}$  would be redundant. Therefore,  $B_i^s(\hat{B}_i^{I_i(q)}) \notin \bar{\mathcal{B}}_s$  must hold. Analogously to Case 1(i), we obtain  $B_i^{s+1}(\hat{B}_i^{I_i(q)}) = B_i^s(\hat{B}_i^{I_i(q)})$ .

In the example depicted in Figure 5, it holds that  $B_3^3(B_{23}) = B_3^2(B_{11}) = B_{13}$  and  $B_3^3(B'_{23}) = B_3^2(B_{12}) = B_{13}$ .

(ii) Determination of  $B_j^{s+1}(\hat{B}_i^{I_i(q)})$  and  $B_k^{s+1}(\hat{B}_i^{I_i(q)})$  for  $j, k \neq i$ :

Consider all  $\hat{B}_i^{I_i(q)} \in \mathcal{B}_{s+1}, q = 1, \dots, Q_i$ , ordered as in (14) and (15): It holds that

$$B_j^{s+1}(\hat{B}_i^{I_i(q)}) = \hat{B}_i^{I_i(q-1)} \quad \text{for all } q = 2, \dots, Q_i,$$

as for all other boxes  $\hat{B}_i^{I_i(p)}, p \neq q - 1$ , either  $u_j(\hat{B}_i^{I_i(p)}) < u_j(\hat{B}_i^{I_i(q-1)})$  or  $u_j(\hat{B}_i^{I_i(p)}) > u_j(\hat{B}_i^{I_i(q)})$  holds. Moreover, all new boxes split with respect to  $j$  have component  $u_j$  smaller than  $u_j(\hat{B}_i^{I_i(q-1)})$  and all new boxes split with respect to  $k$  have component  $u_k$  smaller than  $u_k(\hat{B}_i^{I_i(q)})$ , and, thus, do not satisfy (12). For all boxes  $B \notin \bar{\mathcal{B}}_s$ , it holds that  $u_l(B) < \min\{u_l(B) : B \in \bar{\mathcal{B}}_s\}$  for some  $l$ , so either  $u_j(B) < u_j(\hat{B}_i^{I_i(q-1)})$  or (12) is not satisfied.

Next, we determine  $B_j^{s+1}(\hat{B}_i^{I_i(1)})$ : As  $u_j(\hat{B}_i^{I_i(q)}) > u_j(\hat{B}_i^{I_i(1)})$  for all  $q = 2, \dots, Q_i$ , no box split with respect to  $i$  can be the neighbor  $B_j^{s+1}(\hat{B}_i^{I_i(1)})$ . Furthermore, as  $u_k(\hat{B}_i^{I_i(1)}) > z_k^s$ ,  $B_j^{s+1}(\hat{B}_i^{I_i(1)})$  can not be found among the new boxes split with respect to  $k$ . Therefore,  $B_j^{s+1}(\hat{B}_i^{I_i(1)})$  can only be found among the boxes

split with respect to component  $j$ . Now, as shown above,  $u_k(\hat{B}^{I_i(1)}) = \bar{u}_k$  holds, which implies that  $u_k(B_j^{s+1}(\hat{B}_i^{I_i(1)})) = \bar{u}_k$  must be satisfied. Therefore, the unique candidate for  $B_j^{s+1}(\hat{B}_i^{I_i(1)})$  is  $\hat{B}_j^{I_j(1)}$ , which, as explained above, either equals the box obtained from  $\hat{B}_i^{I_i(1)}$  by a split with respect to  $j$  or the unique box dominating it.

Analogously, it can be shown that

$$B_k^{s+1}(\hat{B}_i^{I_i(q)}) = \hat{B}_i^{I_i(q+1)} \quad \text{for all } q = 1, \dots, Q_i - 1,$$

and

$$B_k^{s+1}(\hat{B}_i^{I_i(Q_i)}) = \hat{B}_k^{I_k(Q_k)},$$

where  $\hat{B}_k^{I_k(Q_k)}$  either equals the box obtained from  $\hat{B}_i^{I_i(Q_i)}$  by a split with respect to  $k$  (then  $I_i(Q_i) = I_k(Q_k)$ ) or the unique box dominating it.

In the example depicted in Figure 5,  $B_1^3(B'_{23}) = B_{23}$  and  $B_1^3(B_{23}) = B_{21}$  hold.

Finally, for all  $B \notin \bar{\mathcal{B}}_s$  we obtain the following results which are equivalent to Case 1:

(iii) If  $B_i^s(B) \notin \bar{\mathcal{B}}_s$  for some  $i \in \{1, 2, 3\}$ , then  $B_i^{s+1}(B)$  remains unchanged.

As in the example depicted in Figure 5 box  $B_{13}$  is the unique box which is not split and all of its neighbors are split, this case does not occur.

(iv) If  $B_i^s(B) =: \hat{B} \in \bar{\mathcal{B}}_s$  for some  $i \in \{1, 2, 3\}$ , then, following the same argumentation as in Case 1(iv),  $z_j^s > u_j(B)$  for one unique index  $j \neq i$  and, thus, the correct candidate for  $B_i^{s+1}(B)$  would be  $\hat{B}_j$ . It remains to show that  $\hat{B}_j$  exists and that  $u_i(\hat{B}_j) = \max\{u_i(\tilde{B}) : \tilde{B} \in \mathcal{B}_{s+1}, u_i(\tilde{B}) < u_i(B)\}$ .

Assume that  $\hat{B}_j$  does not exist, i.e., it is redundant in  $\mathcal{B}_{s+1}$ . Then there exists  $\bar{B} \in \bar{\mathcal{B}}_s$  with  $u_i(\bar{B}) \geq u_i(\hat{B})$  and  $u_k(\bar{B}) \geq u_k(\hat{B})$ . As  $\bar{B}, \hat{B} \in \mathcal{B}_s$  and  $\mathcal{B}_s$ , by induction, is non-redundant,  $u_j(\bar{B}) < u_j(\hat{B})$  must hold. As  $\bar{B} \in \bar{\mathcal{B}}_s$  it follows that  $z_j^s < u_j(\bar{B})$ , so  $u_j(B) < z_j^s < u_j(\bar{B})$  and  $u_k(B) = u_k(\hat{B}) \leq u_k(\bar{B})$  hold. If  $u_i(\bar{B}) \geq u_i(B)$ ,  $B$  would have been redundant in  $\mathcal{B}_s$ . Thus,  $u_i(\bar{B}) < u_i(B)$  must hold. However, as  $B_i^s(B) = \hat{B}$ , the induction hypothesis then implies that  $u_i(\bar{B}) < u_i(\hat{B})$ , a contradiction to the assumption on  $\bar{B}$ . Thus,  $\hat{B}_j$  is non-redundant and  $u_i(\hat{B}_j) = u_i(\hat{B}) = \max\{u_i(\tilde{B}) : \tilde{B} \in \mathcal{B}_{s+1}, u_i(\tilde{B}) < u_i(B)\}$  holds.

In the example depicted in Figure 5, consider  $B_1^2(B_{13}) = B_{11}$ . Since  $j = 3$  is the unique index  $\neq 1$  such that  $z_j^2 > u_j(B_{13})$ , it holds that  $B_1^3(B_{13}) = B_{23}$ , i.e., the new neighbor is the box which results from  $B_{11}$  by a split with respect to  $j$ .

□

□

In the following corollary we summarize the properties of the neighbors of all new boxes obtained in the constructive proof of Lemma 3.5.

**Corollary 3.6.** *Let Assumption 3.4 be satisfied. For every  $i \in \{1, 2, 3\}$ , let  $I_i \subseteq \{1, \dots, P\}$ ,  $I_i \neq \emptyset$ ,  $P \in \mathbb{N}$ ,  $|I_i| = Q_i$ , be the index set of the boxes of  $\bar{\mathcal{B}}_s$  whose split with respect to  $i \in \{1, 2, 3\}$  yields a non-redundant box. Then for all new boxes  $\hat{B}_i^{I_i(q)}$ ,  $q = 1, \dots, Q_i$ , it holds that*

$$B_i^{s+1}(\hat{B}_i^{I_i(q)}) = B_i^s(\hat{B}_i^{I_i(q)}) \quad \forall q = 1, \dots, Q_i, \quad (16)$$

$$B_j^{s+1}(\hat{B}_i^{I_i(q)}) = \begin{cases} \hat{B}_j^{I_j(1)} & q = 1, \\ \hat{B}_i^{I_i(q-1)} & \forall q = 2, \dots, Q_i, \end{cases} \quad (17)$$

$$B_k^{s+1}(\hat{B}_i^{I_i(q)}) = \begin{cases} \hat{B}_i^{I_i(q+1)} & \forall q = 1, \dots, Q_i - 1, \\ \hat{B}_k^{I_k(Q_k)} & q = Q_i, \end{cases} \quad (18)$$

where the indices  $j$  and  $k$  are chosen as in (14) and (15), i.e., the boxes  $\hat{B}_i^{I_i(q)}$ ,  $q = 1, \dots, Q_i$ , are ordered with respect to their upper bounds increasingly by component  $j \neq i$  and decreasingly by component  $k \neq i, j$ . Moreover,  $I_j(1)$  and  $I_k(Q_k)$  are chosen such that  $\hat{B}_j^{I_j(1)}$  and  $\hat{B}_k^{I_k(Q_k)}$  either equal  $\hat{B}_j^{I_i(1)}$  and  $\hat{B}_k^{I_i(Q_i)}$ , respectively, or the unique box dominating it.

Using Lemma 3.5 we can derive an explicit formulation of the individual subsets  $V(B)$  for  $m = 3$ :

**Lemma 3.7.** *Let Assumption 3.4 hold. Then, for  $m = 3$ , the individual subsets  $V(B)$ ,  $B \in \mathcal{B}_s$ , which are introduced in Definition 3.1, can be represented as*

$$V(B) = \{z \in B_0 : v(B) \leq z < u(B)\}$$

with

$$v_i(B) := \begin{cases} u_i(B_i^s(B)), & \text{if } B_i^s(B) \neq \emptyset, \\ z_i^I, & \text{otherwise} \end{cases}, \quad i \in \{1, 2, 3\}. \quad (19)$$

*Proof.* For  $\bar{B} \in \mathcal{B}_s$ ,  $s \geq 1$ , by definition,

$$V(\bar{B}) := \bar{B} \setminus \left( \bigcup_{\tilde{B} \in \mathcal{B}_s \setminus \{\bar{B}\}} \tilde{B} \right)$$

holds. We consider the sets  $\mathcal{B}_{s,i} := \{B \in \mathcal{B}_s : u_i(B) < u_i(\bar{B})\}$  for  $i = 1, 2, 3$ . For fixed  $i \in \{1, 2, 3\}$ , the following two cases can occur: If  $\mathcal{B}_{s,i} \neq \emptyset$ , then, as shown in Lemma 3.5,  $B_i^s(\bar{B}) \neq \emptyset$  and  $B_i^s(\bar{B}) \in \mathcal{B}_{s,i}$ , where  $u_i(B_i^s(\bar{B})) = \max\{u_i(B) : B \in \mathcal{B}_{s,i}\}$ . Furthermore, as  $u_l(B_i^s(\bar{B})) \geq u_l(\bar{B})$  for all  $l \neq i$ ,

$$\bar{B} \setminus \left( \bigcup_{\tilde{B} \in \mathcal{B}_{s,i}} \tilde{B} \right) = \{z \in \bar{B} : z_i \geq u_i(B_i^s(\bar{B}))\}.$$

Otherwise, i.e., if  $\mathcal{B}_{s,i} = \emptyset$ , then, obviously,  $\bar{B} \setminus \left( \bigcup_{\tilde{B} \in \mathcal{B}_{s,i}} \tilde{B} \right) = \bar{B}$ . So, in both cases, it holds that

$$\bar{B} \setminus \left( \bigcup_{\tilde{B} \in \mathcal{B}_{s,i}} \tilde{B} \right) = \{z \in \bar{B} : z_i \geq v_i(\bar{B})\}$$

with

$$v_i(\bar{B}) := \begin{cases} u_i(B_i^s(\bar{B})), & \text{if } B_i^s(\bar{B}) \neq \emptyset, \\ z_i^I, & \text{otherwise.} \end{cases}$$

As every box  $B \in \mathcal{B}_s \setminus \{\bar{B}\}$  belongs, due to the assumption of non-redundancy, at least to one set  $\mathcal{B}_{s,i}, i \in \{1, 2, 3\}$ , there does not exist any other box which can reduce  $V(\bar{B})$  further. Thus, we obtain the desired representation.  $\square$   $\square$

Lemma 3.7 shows that for  $m = 3$  the individual subset of a box can be represented as a box itself. As the upper bound of  $V(B)$  and  $B$  are the same,  $V(B)$  can be described by its lower bound  $v(B) \in \mathbb{R}^m$  only. Next we show, using Corollary 3.6, how the lower bounds  $v(B)$  can be updated in an iterative algorithm.

**Lemma 3.8.** *Let Assumption 3.4 be satisfied. We use the notation of Corollary 3.6. Let  $\hat{B}_i^{I_i(q)}, q = 1, \dots, Q_i$ , be the non-redundant boxes obtained from  $\hat{B}^{I_i(q)} \in \bar{\mathcal{B}}_s$  by a split with respect to  $i \in \{1, 2, 3\}$ . Then the lower bound vectors  $v(B) \in \mathbb{R}^m$  of these new boxes in  $\mathcal{B}_{s+1}$  are determined by*

$$\begin{aligned} v_i(\hat{B}_i^{I_i(q)}) &= v_i(\hat{B}^{I_i(q)}) \quad \forall q = 1, \dots, Q_i, \\ v_j(\hat{B}_i^{I_i(q)}) &= \begin{cases} u_j(\hat{B}_j^{I_j(1)}) = z_j^s & q = 1, \\ u_j(\hat{B}_i^{I_i(q-1)}) & \forall q = 2, \dots, Q_i, \end{cases} \\ v_k(\hat{B}_i^{I_i(q)}) &= \begin{cases} u_k(\hat{B}_i^{I_i(q+1)}) & \forall q = 1, \dots, Q_i - 1, \\ u_k(\hat{B}_k^{I_k(Q_k)}) = z_k^s & q = Q_i. \end{cases} \end{aligned}$$

All individual subsets  $V(B)$  of all  $B \notin \bar{\mathcal{B}}_s$  remain unchanged.

*Proof.* The update of  $v(\hat{B}_i^{I_i(q)})$  of all new boxes  $\hat{B}_i^{I_i(q)}, q = 1, \dots, Q_i$ , for some fixed  $i \in \{1, 2, 3\}$  is derived directly from Corollary 3.6. The individual subsets of all boxes which are not split in the current iteration do not change, as, according to the proof of Lemma 3.5, either  $B_i^{s+1}(B)$  remains unchanged (Case (iii)) or  $B_i^{s+1}(B) = \hat{B}_j$  (Case (iv)), i.e.,  $u_i(B)$  remains unchanged.  $\square$   $\square$

Recall that we want to split a box  $B \in \bar{\mathcal{B}}_s$  with respect to a component  $i \in \{1, 2, 3\}$  if and only if the individual subset  $V(B_i)$  of the resulting box  $B_i$  is non-empty, which is equivalent to  $B_i$  being non-redundant. With the vector  $v(B) \in \mathbb{R}^m$  at hand, this can be easily checked, as the following lemma shows.

**Lemma 3.9.** *Let Assumption 3.4 hold up to iteration  $s - 1$  for  $s \geq 2$ , i.e., let  $\mathcal{B}_s$  be a correct, non-redundant decomposition of the search region obtained by iterative 3-splits. Let  $z^s \in Z_N$  satisfy Assumption 3.4 (1), and let  $B_i$  be the box obtained from*

$B \in \overline{\mathcal{B}}_s$  by a split with respect to component  $i \in \{1, 2, 3\}$ . Then  $B_i$  is non-redundant if and only if  $z_i^s > v_i(B)$  holds.

*Proof.* Consider a fixed  $i \in \{1, 2, 3\}$ .

“ $\Rightarrow$ ”: Let  $B_i$  be non-redundant and assume that  $z_i^s < v_i(B)$  holds. (The case  $z_i^s = v_i(B)$  does not occur due to Assumption 3.4 (1).) Then  $v_i(B) > z_i^s$ , and, thus,  $v_i(B) = u_i(B_i^s)$  with  $B_i^s(B) \neq \emptyset$ . As  $u_l(B_i^s(B)) \geq u_l(B)$  for all  $l \neq i$ ,  $z^s \in B_i^s(B)$  must hold. But then,  $B_i$  would be redundant as it would be dominated by the box obtained from  $B_i^s(B)$  by a split with respect to  $i$ , a contradiction to the assumption of non-redundancy.

“ $\Leftarrow$ ”: Let  $z_i^s > v_i(B)$ . A split of  $B$  with respect to  $i$  yields  $B_i = \{z \in B : z_i < z_i^s\}$ . Assume that there exists  $\tilde{B}_i \neq B_i$  which dominates  $B_i$ . As  $\mathcal{B}_s$  is non-redundant and due to Lemma 2.7,  $\tilde{B}_i$  must result from a split with respect to  $i$  from some box  $\tilde{B} \in \overline{\mathcal{B}}_s$ , i.e.,  $z^s \in \tilde{B}$  must hold. As  $B$  and  $\tilde{B}$  are split with respect to  $i$ ,  $u_i(B_i) = u_i(\tilde{B}_i) = z_i^s$  holds, and, due to the assumption that  $\tilde{B}_i$  dominates  $B_i$ ,  $u_l(\tilde{B}) \geq u_l(B)$  for all  $l \neq i$ . Now  $B, \tilde{B} \in \mathcal{B}_s$  and  $\mathcal{B}_s$  being non-redundant imply that  $u_i(\tilde{B}) < u_i(B)$ . This in turn means that  $v_i(B) \geq u_i(\tilde{B})$ . But then  $z_i^s > u_i(\tilde{B})$ , a contradiction to  $z^s \in \tilde{B}$ . It follows that  $B_i$  is non-redundant.  $\square$

Lemma 3.9 provides a tool for defining a split operation for tricriteria problems which generates all boxes that are necessary for maintaining the correctness of a decomposition, but avoids the generation of redundant boxes. We call the split based on the individual subsets  $V(B)$  a  $v$ -split in the following.

**Definition 3.10** ( $v$ -split). *Let Assumption 3.4 hold up to iteration  $s - 1$  for  $s \geq 2$ , i.e., let  $\mathcal{B}_s$  be a correct, non-redundant decomposition of the search region obtained by iterative 3-splits, and let  $z^s \in Z_N$ . We call the split of a box  $B \in \overline{\mathcal{B}}_s$  with respect to components  $i \in \{1, 2, 3\}$ , for which*

$$z_i^s \geq v_i(B) \tag{20}$$

*holds, a  $v$ -split of  $B$ .*

Note that equality in (20) does not occur due to Assumption 3.4 (1). However, as Assumption 3.4 (1) will be removed in Section 3.3, we present the  $v$ -split already at this point in this general form.

**Lemma 3.11.** *Let Assumption 3.4 (1),(2) hold. Then the iterative application of a  $v$ -split to every  $B \in \overline{\mathcal{B}}_s$  in every iteration  $s \geq 1$  yields a correct, non-redundant decomposition.*

*Proof.* Due to Assumption 3.4 (1),  $z_i^s \geq v_i(B)$  is equivalent to  $z_i^s > v_i(B)$ . According to Lemma 3.9, the  $v$ -split avoids exactly the generation of redundant boxes and, therefore, yields a correct, non-redundant decomposition.  $\square$   $\square$

### 3.1 An algorithm for tricriteria problems based on the $v$ -split

Algorithm 2 implements the  $v$ -split. As in Algorithm 1, an initial box  $B_0$  is computed, which is represented by its upper bound  $u(B_0)$ . Additionally, for  $B_0$  as well as for all other boxes  $B$  which are generated in the course of the algorithm, the lower bound of the individual subset  $v(B)$  is saved. Analogously to Algorithm 1, as long as the decomposition contains unexplored boxes, a box is selected and a subproblem is solved. If the problem is infeasible, the selected box is deleted from the list of unexplored boxes. Otherwise, the nondominated point  $z^s$  is saved and all boxes are determined that contain  $z^s$ . Now, different from Algorithm 1,  $z^s$  is compared componentwise to  $v(B)$  for every  $B \in \overline{B}_s$ . A split with respect to component  $i$  is performed if and only if  $z_i^s \geq v_i(B)$  and  $z_i^s > z_i^I$  hold. If  $v_i(B) > z_i^s$  for all  $i \in \{1, 2, 3\}$ , then  $B$  is deleted. Finally, the vectors  $v$  of all new boxes are updated according to Lemma 3.8 and a new iteration starts.

Note that according to the proof of Lemma 3.5 we can order all newly generated, non-redundant boxes resulting from a split with respect to component  $i$  such that their upper bound values  $u$  are increasing in one component  $j \neq i$  and decreasing in the remaining component  $k \neq i, j$ . Hence, in Line 46 in the procedure UPDATEINDIVIDUALSUBSETS, strict inequalities hold between the components of each pair of upper bounds. However, in order to make the algorithm also applicable when Assumption 3.4 (1) is removed, see Section 3.3 below, we formulate Algorithm 2 already in a general form. Therefore, in Line 46, the strict inequalities are replaced by inequalities and in Lines 47 to 49 the case that the upper bound vectors of two boxes are equal is handled. Note that this case does not occur under Assumption 3.4 (1).

According to Lemma 3.11, Algorithm 2 maintains a correct, non-redundant decomposition in each iteration.

**Example 3.12** (Application of Algorithm 2). *Consider again the tricriteria problem of Example 2.4 with initial search region  $B_0 := \{z \in Z : 0 \leq z_i \leq 5 \forall i = 1, 2, 3\}$  and  $V(B_0) = B_0$ , thus,  $v(B_0) = (0, 0, 0)^\top$ . Consider  $z^1 = (2, 2, 2)^\top$ . The  $v$ -split applied to the initial box equals a full 3-split and, thus, results in*

$$B_{1,i} := \{z \in B_0 : z_i < 2\}, \quad i = 1, 2, 3.$$

The corresponding individual subsets are

$$V(B_{1,i}) := \{z \in B_{1,i} : z_j \geq 2 \forall j \neq i\}, \quad i = 1, 2, 3,$$

thus,  $v(B_{11}) = (0, 2, 2)^\top$ ,  $v(B_{12}) = (2, 0, 2)^\top$  and  $v(B_{13}) = (2, 2, 0)^\top$ . Let  $z^2 = (1, 1, 4)^\top$ . It holds that  $z^2 \in B_{11}$  as well as  $z^2 \in B_{12}$ , but  $z^2 \notin B_{13}$ . Consider first the  $v$ -split in  $B_{11}$ : As  $z_1^2 \geq v_1(B_{11})$ ,  $z_2^2 \not\geq v_2(B_{11})$  and  $z_3^2 \geq v_3(B_{11})$ ,  $B_{11}$  is split with respect to the first and third component into

$$B_{21} := \{z \in B_{11} : z_1 < 1\} \quad \text{and} \quad B_{23} := \{z \in B_{11} : z_3 < 4\}$$

and  $\mathcal{S}_1 = \{B_{21}\}$ ,  $\mathcal{S}_2 = \emptyset$  and  $\mathcal{S}_3 = \{B_{23}\}$ . Applying the  $v$ -split to  $B_{12}$  results in a split with respect to the second and third component into

$$B_{22} := \{z \in B_{12} : z_2 < 1\} \quad \text{and} \quad B'_{23} := \{z \in B_{12} : z_3 < 4\}$$

---

**Algorithm 2** Algorithm with  $v$ -split for  $m = 3$ 

---

**Input:** Image of the feasible set  $Z \subset \mathbb{R}^m$ , implicitly given by some problem formulation

- 1:  $Z_N := \emptyset$ ;  $\delta > 0$ ;
- 2: INITSTARTINGBOXVSPLIT( $Z, \delta$ ); // Initialize starting box
- 3:  $s := 1$ ;
- 4: **while**  $\mathcal{B}_s \neq \emptyset$  **do**
- 5:   Choose  $\bar{B} \in \mathcal{B}_s$ ; // Select a box from the decomposition
- 6:    $z^s := \text{opt}(Z, u(\bar{B}))$ ; // Solve subproblem
- 7:   **if**  $z^s = \emptyset$  **then** // Subproblem infeasible
- 8:      $\mathcal{B}_{s+1} := \mathcal{B}_s \setminus \{\bar{B}\}$ ; // Remove (empty) box
- 9:   **else**
- 10:      $Z_N := Z_N \cup \{z^s\}$ ; // Save nondominated point
- 11:      $\mathcal{B}_{s+1} := \mathcal{B}_s$ ; // Copy set of current boxes
- 12:     GENERATENEWBOXESVSPLIT( $\mathcal{B}_s, z^s, z^I, \mathcal{B}_{s+1}$ );
- 13:     UPDATEINDIVIDUALSUBSETS( $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{B}_{s+1}$ );
- 14:   **end if**
- 15:    $s := s + 1$ ;
- 16: **end while**
- 17: **return** Set of nondominated points  $Z_N$

---

and  $\mathcal{S}_1 = \{B_{21}\}$ ,  $\mathcal{S}_2 = \{B_{22}\}$  and  $\mathcal{S}_3 = \{B_{23}, B'_{23}\}$ . Note that the redundant boxes which were obtained with the full 3-split in Example 2.4 are not generated by the  $v$ -split.

Finally, the individual subsets of the new boxes of each set  $\mathcal{S}_i, i \in \{1, 2, 3\}$ , are updated: Box  $B_{21}$  is the only box generated for  $i = 1$ , box  $B_{22}$  the only one for  $i = 2$ . Therefore,  $v(B_{21}) = (v_1(B_{11}), z_2^2, z_3^2)^\top = (0, 1, 4)^\top$  and  $v(B_{22}) = (z_1^2, v_2(B_{12}), z_3^2)^\top = (1, 0, 4)^\top$ . Boxes  $B_{23}$  and  $B'_{23}$  are both generated by a split with respect to the third component  $i = 3$ . We can order the upper bounds of the boxes  $u(B_{23}) = (2, 5, 4)^\top$  and  $u(B'_{23}) = (5, 2, 4)^\top$  increasingly with respect to component  $j = 1$  and, at the same time, decreasingly with respect to  $k = 2$ , thus,  $B_3^{I(1)} := B_{23}$  and  $B_3^{I(2)} := B'_{23}$  and then set

$$\begin{aligned} v_1(B_{23}) &= z_1^2 = 1, & v_2(B'_{23}) &= z_2^2 = 1, \\ v_2(B_{23}) &= u_2(B'_{23}) = 2, & v_1(B'_{23}) &= u_1(B_{23}) = 2. \end{aligned}$$

The third component is not changed, so  $v(B_{23}) = (1, 2, 2)^\top$  and  $v(B'_{23}) = (2, 1, 2)^\top$ .

### 3.2 A linear bound on the number of subproblems for $m = 3$

In the following, we will bound the number of boxes generated in the course of Algorithm 2. If a box  $B \in \mathcal{B}_s$  contains the current point  $z^s$ , i.e., if  $B \in \bar{\mathcal{B}}_s$ , then we can make the following assertion concerning the neighbors of  $B$ :

**Lemma 3.13.** *Let  $s \geq 1$  be an iteration of Algorithm 2 in which a nondominated point is generated, i.e.,  $z^s \neq \emptyset$ . Consider any  $B \in \bar{\mathcal{B}}_s$ . We denote by  $J_B \subseteq \{1, 2, 3\}$  the*

---

```

18: procedure INITSTARTINGBOXVSPLIT( $Z, \delta$ )
19:   for  $j = 1$  to 3 do                                     // Compute bounds on  $Z$ 
20:      $z_j^I := \min\{z_j : z \in Z\}$ ;
21:      $z_j^M := \max\{z_j : z \in Z\} + \delta$ ;
22:      $v_j(B_0) := z_j^I$ ;  $u_j(B_0) := z_j^M$ ;
23:   end for
24:    $\mathcal{B}_1 := \{B_0\}$ ;                                       // Initialize set of boxes
25:   return  $\mathcal{B}_1$ 
26: end procedure

27: procedure GENERATENEWBOXESVSPLIT( $\mathcal{B}_s, z^s, z^I, \mathcal{B}_{s+1}$ )
28:    $\mathcal{S}_i := \emptyset, i = 1, 2, 3$ ;
29:   for all  $B \in \mathcal{B}_s$  do
30:     if  $z^s < u(B)$  then                                   // Point is contained in box
31:       for  $i = 1$  to 3 do                                   // Apply  $v$ -split
32:         if  $z_i^s \geq v_i(B)$  and  $z_i^s > z_i^I$  then
33:            $u(B_i) := u(B)$ ;  $v(B_i) := v(B)$ ;               // Create a copy of  $B$ 
34:            $u_i(B_i) := z_i^s$ ;                               // Update upper bound
35:            $\mathcal{S}_i := \mathcal{S}_i \cup \{B_i\}$ ;                   // Save new box in respective set  $\mathcal{S}_i$ 
36:         end if
37:       end for
38:        $\mathcal{B}_{s+1} := \mathcal{B}_{s+1} \setminus \{B\}$ ;             // Remove  $B$ 
39:     end if
40:   end for
41:   return  $\mathcal{B}_{s+1}, \mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ ;
42: end procedure

43: procedure UPDATEINDIVIDUALSUBSETS( $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{B}_{s+1}$ )
44:   for  $i = 1$  to 3 do
45:      $Q := |\mathcal{S}_i|$ ;
46:     Sort all boxes  $B_i^{I_i(q)}, q = 1, \dots, Q_i$ , in  $\mathcal{S}_i$  such that for  $j, k \neq i$ 
47:        $u_j(B_i^{I_i(1)}) \leq u_j(B_i^{I_i(2)}) \leq \dots \leq u_j(B_i^{I_i(Q_i)})$  and
48:        $u_k(B_i^{I_i(1)}) \geq u_k(B_i^{I_i(2)}) \geq \dots \geq u_k(B_i^{I_i(Q_i)})$ ;
49:     if  $u(B_i^{I_i(q)}) = u(B_i^{I_i(q+1)})$  for some  $q = 1, \dots, Q_i - 1$  then
50:       (re)sort  $B_i^{I_i(q)}$  and  $B_i^{I_i(q+1)}$  such that
51:        $v_j(B_i^{I_i(q)}) \leq v_j(B_i^{I_i(q+1)})$  and  $v_k(B_i^{I_i(q)}) \geq v_k(B_i^{I_i(q+1)})$ ;
52:     end if
53:     Set  $v_j(B_i^{I_i(1)}) := z_j^s$ ;  $v_k(B_i^{I_i(Q_i)}) := z_k^s$ ; // Update  $v$ 
54:     for  $q = 2$  to  $Q_i$  do
55:        $v_j(B_i^{I_i(q)}) := u_j(B_i^{I_i(q-1)})$ ;  $v_k(B_i^{I_i(q-1)}) := u_k(B_i^{I_i(q)})$ ;
56:     end for
57:      $\mathcal{B}_{s+1} := \mathcal{B}_{s+1} \cup \mathcal{S}_i$ ;                       // Append new boxes
58:   end for
59:   return  $\mathcal{B}_{s+1}$ 
60: end procedure

```

---

index set of all components with respect to which  $B$  is split, and by  $\bar{J}_B := \{1, 2, 3\} \setminus J_B$  the complement of  $J_B$ . Then the following holds:

1. If  $\bar{J}_B \neq \emptyset$ , then for every  $j \in \bar{J}_B$ , the neighbor  $B_j^s(B)$  exists and contains  $z^s$ , i.e.,  $B_j^s(B) \neq \emptyset$  and  $B_j^s(B) \in \bar{\mathcal{B}}_s$  holds for every  $j \in \bar{J}_B$ .
2. If  $\bar{J}_B = \emptyset$ , then  $\bar{\mathcal{B}}_s = \{B\}$  holds.

*Proof.* Let  $B \in \bar{\mathcal{B}}_s$ . By definition of the  $v$ -split, it holds that  $z^s < u(B)$ ,  $z_j^s \geq v_j(B)$  for every  $j \in J_B$  and  $z_j^s < v_j(B)$  for every  $j \in \bar{J}_B$ . Thus,  $v_j(B) > z_j^s$  holds for every  $j \in \bar{J}_B$ . This, however, implies that  $B_j^s(B) \neq \emptyset$  for every  $j \in \bar{J}_B$ , see the update of  $v$  in (19).

First, let  $\bar{J}_B \neq \emptyset$ . Then, for fixed  $j \in \bar{J}_B$  and according to Definition 3.2,  $u_j(B_j^s(B)) \leq u_j(B)$  and  $u_l(B_j^s(B)) \geq u_l(B)$  for all  $l \neq j$ . As  $u_j(B_j^s(B)) = v_j(B) > z_j^s$ ,  $B_j^s(B) \in \bar{\mathcal{B}}_s$  holds.

Now, consider the case  $\bar{J}_B = \emptyset$ . Then, due to Lemma 3.5, every box  $\tilde{B} \in \mathcal{B}_s \setminus \{B\}$  has upper bound  $u_l(\tilde{B}) \leq v_l(B)$  for at least one  $l \in \{1, 2, 3\}$ . This implies that  $z^s \notin \tilde{B}$  for any  $\tilde{B} \in \mathcal{B}_s \setminus \{B\}$ , thus,  $\bar{\mathcal{B}}_s = \{B\}$ .  $\square$

With the help of Lemma 3.13, we can bound the number of new boxes that are generated in each iteration of Algorithm 2.

**Lemma 3.14.** *In every iteration  $s \geq 1$  of Algorithm 2 in which a nondominated point  $z^s$  is found, i.e.,  $z^s \neq \emptyset$ , the number of boxes in the decomposition increases by at most two.*

*Proof.* If there exists a box  $B \in \bar{\mathcal{B}}_s$  which is split with respect to all three components, then, using Lemma 3.13,  $\bar{\mathcal{B}}_s = \{B\}$  holds, thus,  $|\bar{\mathcal{B}}_s| = 1$ . In this case, the box  $B$  is removed and replaced by three new boxes in the decomposition, and, thus, the number of boxes in the decomposition increases by two.

It follows that if  $|\bar{\mathcal{B}}_s| > 1$ , then every  $B \in \bar{\mathcal{B}}_s$  is split with respect to at most two components. Let  $|\bar{\mathcal{B}}_s| > 1$  and let  $B \in \bar{\mathcal{B}}_s$  be split with respect to two components  $i, j \in \{1, 2, 3\}$ ,  $j \neq i$ . Then, for all other boxes  $\tilde{B} \in \mathcal{B}_s \setminus \{B\}$  it holds that  $u_l(\tilde{B}) \leq v_l(B)$  for some  $l \in \{1, 2, 3\}$ . If  $l = i$ , then  $u_i(\tilde{B}) \leq v_i(B) \leq z_i^s$ , thus, the box is not split with respect to  $i$ . Analogously, if  $l = j$ , then  $u_j(\tilde{B}) \leq v_j(B) \leq z_j^s$ , thus, the box is not split with respect to  $j$ . If  $l = k$  (with  $k \neq i, j$ ), then, for any  $\tilde{B}$  satisfying  $u_k(\tilde{B}) \leq v_k(B)$  it holds that  $v_i(\tilde{B}) \geq u_i(B)$  or  $v_j(\tilde{B}) \geq u_j(B)$ , thus,  $\tilde{B}$  can not be split with respect to both components  $i$  and  $j$ .

Therefore, if two boxes are split with respect to two components, these components must differ in one component. This implies that in one iteration, at most three boxes are split with respect to two components. Any other boxes in  $\bar{\mathcal{B}}_s$  are split with respect to at most one component.

In case that three boxes are split with respect to two components, six new boxes would replace three old ones, thus, the number of boxes would increase by three. So it remains to show that in this case, at least one box  $B$  in  $\bar{\mathcal{B}}_s$  is removed without being split, i.e.,  $v(B) > z^s$  holds for at least one  $B \in \bar{\mathcal{B}}_s$ . In other words, we have to prove

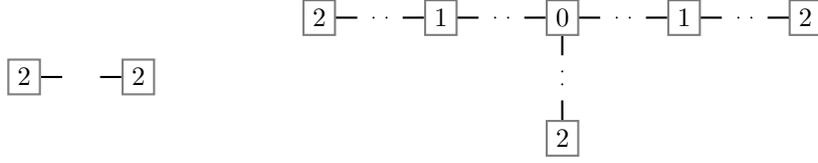


Figure 6: Possible neighborhood structures of boxes in  $\bar{\mathcal{B}}_s$ . Left figure:  $\bar{\mathcal{B}}_s$  contains two '2-boxes'; right figure:  $\bar{\mathcal{B}}_s$  contains three '2-boxes'.

the existence of a '0-box', i.e., a box, which is contained in  $\bar{\mathcal{B}}_s$ , but is not split with respect to any component.

To this end, we assume to the contrary that  $\bar{\mathcal{B}}_s$  contains three boxes which are split with respect to two components ('2-boxes'), respectively, but that no '0-box' exists. From Lemma 3.13 we see that a '2-box' has exactly one neighbor in  $\bar{\mathcal{B}}_s$ , as  $\bar{J}_B$  contains exactly one index. A '1-box' has exactly two neighbors in  $\bar{\mathcal{B}}_s$ , while all three neighbors are contained in  $\bar{\mathcal{B}}_s$  in case of a '0-box'. Now, starting from a '2-box', one uniquely defined neighbor of it must be in  $\bar{\mathcal{B}}_s$ . If that box is also a '2-box' (see Figure 6 on the left), then no neighbor of the latter box is in  $\bar{\mathcal{B}}_s$ . The third '2-box' would require a neighbor in  $\bar{\mathcal{B}}_s$ , but only '1-boxes' are available, which require a second neighbor in turn. Thus, a fourth '2-box' would be needed, which, however, does not exist. Therefore, the three '2-boxes' must all be connected by one structure of neighbors. But this implies that there exists exactly one '0-box' connecting the three branches emerging from each '2-box' (see Figure 6 on the right).  $\square$   $\square$

**Theorem 3.15.** *Let a problem with a finite set of nondominated points be given. After having computed the starting box based on the ideal point and a global upper bound on  $Z$ , Algorithm 2 requires the solution of at most  $3|Z_N| - 2$  subproblems in order to generate the entire nondominated set  $Z_N$ .*

*Proof.* In every iteration of Algorithm 2, one subproblem is solved. Thus, the number of subproblems to be solved equals the number of iterations. When a nondominated point is generated, the number of boxes increases by at most two according to Lemma 3.14. As every nondominated point is generated exactly once, and since every empty box is investigated exactly once in order to verify that no further nondominated points are contained, at most  $3|Z_N|$  boxes are explored in the course of the algorithm. Together with the initial box, at most  $3|Z_N| + 1$  boxes are explored, which corresponds to the number of subproblems to be solved. As we additionally assume that the ideal point is given, we can reduce this bound further: In every iteration in which the current nondominated point equals the ideal point in at least one component, one box per component equal to the ideal point can be directly discarded. For each component  $i \in \{1, 2, 3\}$ , there must exist at least one nondominated point whose  $i$ -th component equals  $z_i^I$ . Therefore, the total number of subproblems to be solved is at most  $3|Z_N| - 2$ .  $\square$   $\square$

### 3.3 Applying the $v$ -split to arbitrary nondominated sets for tricriteria problems

For the construction of the  $v$ -split we assumed that no pair of nondominated points has an identical value in at least one component, i.e., that all values are pairwise different (Assumption 3.4 (1)). Under this assumption, the individual subsets of all (non-redundant) boxes are boxes themselves, which is the basis for the  $v$ -split criterion. In practice, Assumption 3.4 (1) may be violated since arbitrary nondominated points may coincide in up to  $m - 2$  components, i.e., in one component for  $m = 3$ . In this case, additional redundant boxes may occur as the following example shows.

**Example 3.16.** Let  $z^1 = (3, 1, 4)^\top$ ,  $z^2 = (3, 2, 1)^\top$  and let the initial search region be given as  $B_0 := \{z \in Z : 0 \leq z_i \leq 5 \forall i = 1, 2, 3\}$ . If we insert  $z^1$  into  $B_0$ , we obtain the three subboxes  $B_{1,i} := \{z \in B_0 : z_i < z_i^1\}$ ,  $i = 1, 2, 3$ , with respective upper bounds

$$u(B_{11}) = (3, 5, 5)^\top, u(B_{12}) = (5, 1, 5)^\top, u(B_{13}) = (5, 5, 4)^\top.$$

The second point  $z^2 = (3, 2, 1)^\top$  is only contained in  $B_{13}$ . Thus,  $B_{13}$  is replaced by the three subboxes  $B_{2,i} := \{z \in B_{13} : z_i < z_i^2\}$ ,  $i = 1, 2, 3$ , with respective upper bounds

$$u(B_{21}) = (3, 5, 4)^\top, u(B_{22}) = (5, 2, 4)^\top, u(B_{23}) = (5, 5, 1)^\top.$$

It holds that  $B_{21} \subseteq B_{11}$ .

Note that under Assumption 3.4 (1) no redundancy appears if  $|\overline{\mathcal{B}}_s| = 1$ , which is, as shown in Example 3.16, no longer true for arbitrary nondominated points. If the redundant box  $B_{21}$  is removed from the decomposition, i.e., if we set  $\mathcal{B}_3 := \{B_{11}, B_{12}, B_{22}, B_{23}\}$ , then, however, the individual subset  $V(B_{11})$  does not have the structure of a box anymore, as

$$\begin{aligned} V(B_{11}) &:= B_{11} \setminus \left( \bigcup_{\tilde{B} \in \mathcal{B}_3 \setminus \{B_{11}\}} \tilde{B} \right) \\ &= \{z \in B_{11} : z \geq (0, 2, 1)^\top\} \cup \{z \in B_{11} : z \geq (0, 1, 4)^\top\}. \end{aligned}$$

Nevertheless, the box format of the individual subsets can be preserved if we maintain the redundant box  $B_{21}$  in the decomposition. Then,  $V(B_{11}) = \{z \in B_{11} : z \geq v(B_{11})\}$  with  $v(B_{11}) = (0, 1, 4)^\top$  holds. Also the individual subset of  $B_{21}$  has box format with  $v(B_{21}) = (3, 2, 1)^\top$ . However, as  $B_{21} \subseteq B_{11}$ , it holds that  $V(B_{21}) = \emptyset$ .

Despite some individual subsets being empty, the  $v$ -split can be applied regularly in the following iterations since the lower bound  $v(B)$  is compared to the current nondominated point  $z^s$  component-wise. Thus, it is irrelevant for the  $v$ -split whether  $V(B)$  for some  $B \in \mathcal{B}_s$  is empty or not. Clearly, in Example 3.16, box  $B_{21}$  cannot be split with respect to the first component, but it can be split with respect to the second and the third component like a ‘regular’ non-redundant box.

In order to distinguish the redundant boxes that appear in the case that a point equals a previously generated point in one component from the actual redundant boxes,

we call the former boxes *quasi non-redundant* boxes. Evidently, Lemma 3.5 does not hold if quasi non-redundant boxes are part of the decomposition. However, the neighborhood structure which was obtained under Assumption 3.4 (1) can be preserved in the presence of quasi non-redundant boxes if we use a recursive update of the neighbors that is analogous to the non-redundant case, i.e., that uses the same sorting of the boxes. Then,  $B_i^s(B)$  can be set as derived in Corollary 3.6. This in turn means that the  $v$ -split as well as Algorithm 2 do not need to be changed, but can be applied also when Assumption 3.4 (1) is removed. Thus, Theorem 3.15 which shows that the number of subproblems is bounded by  $3|Z_N| - 2$  holds independently of Assumption 3.4 (1). Finally, we revisit and extend the previous example in order to illustrate how Algorithm 2 is applied in the presence of quasi non-redundant boxes.

**Example 3.17.** *As in Example 3.16, let  $\mathcal{B}_1 := \{B_0\}$  with*

$$B_0 := \{z \in Z : 0 \leq z_i \leq 5 \forall i = 1, 2, 3\}$$

*be a given initial decomposition, and let  $z^1 = (3, 1, 4)^\top$  and  $z^2 = (3, 2, 1)^\top$  be two non-dominated points. Then, the decomposition of the search region (including the quasi non-redundant box  $B_{21}$ ) at the beginning of the third iteration is  $\mathcal{B}_3 := \{B_{11}, B_{12}, B_{21}, B_{22}, B_{23}\}$  with*

$$\begin{aligned} u(B_{11}) &= (3, 5, 5)^\top, & v(B_{11}) &= (0, 1, 4)^\top, \\ u(B_{12}) &= (5, 1, 5)^\top, & v(B_{12}) &= (3, 0, 4)^\top, \\ u(B_{21}) &= (3, 5, 4)^\top, & v(B_{21}) &= (3, 2, 1)^\top, \\ u(B_{22}) &= (5, 2, 4)^\top, & v(B_{22}) &= (3, 1, 1)^\top, \\ u(B_{23}) &= (5, 5, 1)^\top, & v(B_{23}) &= (3, 2, 0)^\top. \end{aligned}$$

*The corresponding individual subsets are depicted in Figure 7 (a). Note that the empty individual subset of the quasi non-redundant box  $B_{21}$  is illustrated as the two-dimensional face*

$$\{z \in B_0 : v(B_{21}) \leq z \leq u(B_{21})\}.$$

*Let now as third nondominated point  $z^3 = (2, 2, 2)^\top$  be given. As  $z^3$  is contained in  $B_{11}$  and  $B_{21}$ , we consider  $v(B_{11}) = (0, 1, 4)^\top$  and  $v(B_{21}) = (3, 2, 1)^\top$  for the  $v$ -split. Comparing  $z^3$  with these two vectors reveals that  $B_{11}$  is split with respect to the first and the second component, and  $B_{21}$  is split with respect to the second and the third component, which yields*

$$u(B_{31}) = (2, 5, 5)^\top, u(B_{32}) = (3, 2, 5)^\top, u(B'_{32}) = (3, 2, 4)^\top, u(B_{33}) = (3, 5, 2)^\top.$$

*Hence,  $\mathcal{B}_4 := \{B_{12}, B_{22}, B_{23}, B_{31}, B_{32}, B'_{32}, B_{33}\}$ . As  $B_{31}$  is the only box obtained by a split with respect to the first component, we obtain  $v(B_{31}) = (0, 2, 2)^\top$ . Analogously,  $v(B_{33}) = (2, 2, 1)^\top$ . For the update of  $v(B_{32})$  and  $v(B'_{32})$ , the upper bound vectors  $u(B_{32})$  and  $u(B'_{32})$  are ordered increasingly with respect to one component  $j \neq 2$  and decreasingly with respect to the remaining component  $k \neq j, k \neq 2$ , e.g.,  $j = 1$  and*

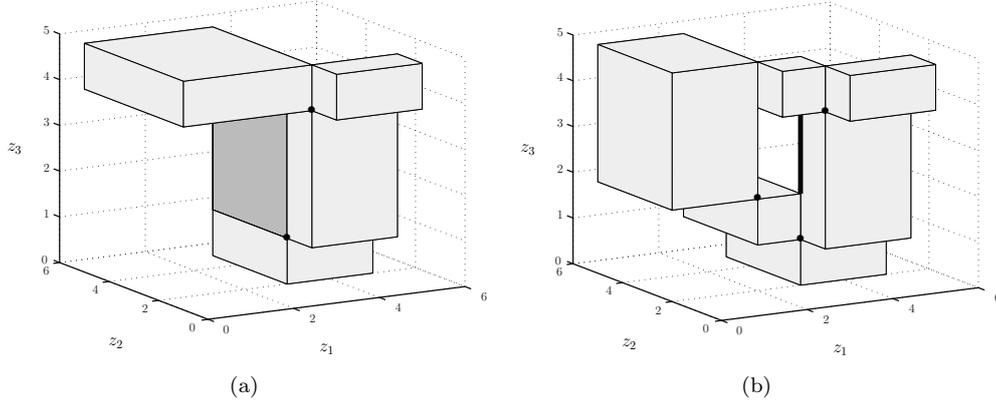


Figure 7: Illustration of the sets  $V(B)$  in Example 3.17; In (a) the individual subset of the occurring quasi non-redundant box (which is actually empty) is represented as a slightly darker, two-dimensional face, in (b) as a black one-dimensional face, i.e., a line.

$k = 3$ . As  $u_1(B_{32}) = u_1(B'_{32})$  and  $u_3(B_{32}) > u_3(B'_{32})$ , we can order the boxes strictly decreasingly with respect to  $k = 3$ . Therefore,

$$v(B_{32}) = (2, 1, 4)^\top \quad \text{and} \quad v(B'_{32}) = (3, 2, 2)^\top$$

is obtained. Note that  $v_1(B'_{32}) = u_1(B'_{32})$  and  $v_2(B'_{32}) = u_2(B'_{32})$ . The quasi non-redundant box  $B'_{32}$  is completely contained in  $B_{22}$  and  $B_{32}$ , as  $(3, 2, 4)^\top \leq (5, 2, 4)^\top$  and  $(3, 2, 4)^\top \leq (3, 2, 5)^\top$ , respectively.

The individual subsets of all  $B \in \mathcal{B}_4$  are depicted in Figure 7 (b). As the individual subset  $V(B'_{32})$  is empty, we depict the set

$$\{z \in B_0 : v(B'_{32}) \leq z \leq u(B'_{32})\} = \{z \in B_0 : (3, 2, 2)^\top \leq z \leq (3, 2, 4)^\top\}$$

instead, which describes a one-dimensional face. It is represented as a black line in Figure 7 (b).

Figure 8 shows an example with 68 nondominated points for which Assumption 3.4 (1) does not hold. After having determined the initial search box,  $3|Z_N| - 2 = 202$  sub-problems are solved until the termination criterion of Algorithm 2 is reached, i.e., the upper bound derived in Theorem 3.15 is sharp and holds also when quasi non-redundant boxes occur.

## 4 Using the $\varepsilon$ -constraint method as scalarization

Algorithm 2 presented in Section 3 is formulated independently of a specific scalarization. In every iteration, only points that are dominated by the current nondominated

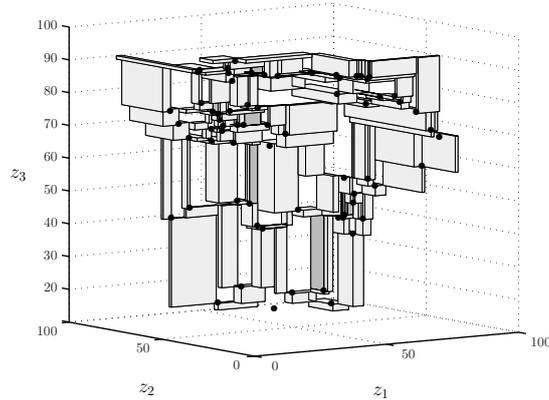


Figure 8: Individual subsets of all boxes of the final decomposition for an example with 68 nondominated points

point are eliminated. In this section we show that we can reduce the search region and, thereby, the number of subproblems further if we use the  $\varepsilon$ -constraint method. The specific properties of the  $\varepsilon$ -constraint method are also used, e.g., in Laumanns et al. (2006), Lokman and Köksalan (2013), Ozlen et al. (2014) and Kirlik and Sayın (2014). However, as to the best of our knowledge we present the first algorithm for tricriteria problems whose number of subproblems is proven to depend linearly on the number of nondominated points, the bound which is derived in this section is new as well.

The reduction of the search region stems from the following property of the  $\varepsilon$ -constraint method, which holds for any number of criteria. First, recall that for every  $z^* \in Z_N$ , by definition of nondominance, we can exclude the two sets

$$S_1(z^*) := \{z \in B : z \leq z^*\} \quad \text{and} \quad S_2(z^*) := \{z \in B : z \geq z^*\}$$

from every box  $B$  that contains  $z^*$ . If the point  $z^*$  has been obtained as an optimal solution of an  $\varepsilon$ -constraint problem of the form

$$\begin{aligned} \min \quad & z_1 \\ \text{s.t.} \quad & z_i < u_i(\bar{B}) \quad \forall i = 2, \dots, m, \end{aligned} \tag{21}$$

where  $\bar{B}$  is a box of the current decomposition, then, additionally, the set

$$S'_1(z^*) := \{z \in \bar{B} : z_1 < z_1^*\} = \{z \in \mathbb{R}^m : z_1 < z_1^*, z_i < u_i(\bar{B}) \forall i = 2, \dots, m\}$$

cannot contain any further points, as this would contradict the optimality of  $z^*$  in (21). Note that  $S'_1(z^*)$  depends on  $\bar{B}$  as well as on the component  $i$  with respect to which the  $\varepsilon$ -constraint problem is minimized. We choose  $i = 1$  without loss of generality. Also note that an optimal solution of (21) is only weakly efficient, in general. Hence, in order to guarantee that  $z^*$  is nondominated, a lexicographic (see, e.g., Laumanns

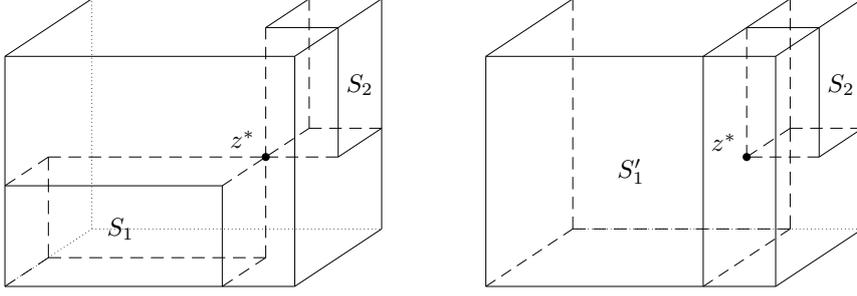


Figure 9: Reduction of the search region for  $m = 3$ : Solely based on nondominance of  $z^*$  (left) and when taking into account that  $z^*$  is obtained as optimal point of a corresponding  $\varepsilon$ -constraint method (right)

et al., 2006), a two-stage (see, e.g., Kirlik and Sayın, 2014) or an augmented (see, e.g., Lokman and Köksalan, 2013)  $\varepsilon$ -constraint method should be employed. When solving (21), two cases might occur. If  $z_1^* \geq u_1(\bar{B})$ , then  $\bar{B} \subseteq S_1'(z^*) = \emptyset$  holds. This case corresponds to the situation in which the current subproblem is infeasible, see Line 7 in Algorithm 2. Box  $\bar{B}$  is removed from the decomposition and a new iteration starts. Otherwise, i.e., if  $z_1^* < u_1(\bar{B})$ ,  $z^* \in \bar{B}$  holds. In Figure 9, an example of the sets  $S_1$ ,  $S_1'$  and  $S_2$  for  $z^* \in \bar{B}$  is depicted.

We consider now the implications of this additional reduction of the search region in combination with the  $v$ -split algorithm. Let box  $\bar{B} \in \mathcal{B}_s$  be the currently selected box, and let  $z^s$  denote the nondominated point obtained in iteration  $s$ ,  $z^s \in \bar{B}$ . Then, the set  $S_1'(z^s) := \{z \in \bar{B} : z_1 < z_1^s\}$  corresponds to the box obtained by a split of  $\bar{B}$  with respect to the first component. Since  $S_1'(z^s)$  is empty,  $\bar{B}$  does not need to be split with respect to the first component. A split with respect to all other components  $i \in \{2, 3\}$  is performed according to the  $v$ -split criterion, i.e., if and only if  $z_i^s \geq v_i(\bar{B})$  holds for  $i \in \{2, 3\}$ . For all other boxes  $B \in \bar{\mathcal{B}}_s \setminus \{\bar{B}\}$  the usual  $v$ -split criterion is employed with respect to all components. In particular, a box  $B \in \bar{\mathcal{B}}_s \setminus \{\bar{B}\}$  must be split with respect to the first component whenever  $z_1^s \geq v_1(B)$  holds.

In order to benefit from the fact that the set  $S_1'(z^s)$  can be excluded additionally from the search region, we must guarantee that the box resulting from a split of  $\bar{B}$  with respect to the first component would have been part of the decomposition. According to the definition of the  $v$ -split, this is the case if  $z_1^s \geq v_1(\bar{B})$  holds. A sufficient criterion to guarantee that  $z_1^s \geq v_1(\bar{B})$  holds is to select a box  $\bar{B}$  that does not have a neighbor in  $\mathcal{B}_s$  with respect to  $i = 1$ , i.e.,  $B_1^s(\bar{B}) = \emptyset$ . Equivalently, we might select a box  $\bar{B}$  which satisfies  $v_1(\bar{B}) = \min\{v_1(B) : B \in \mathcal{B}_s\}$ . This means that we replace Line 5 in Algorithm 2 by ‘choose  $\bar{B} \in \mathcal{B}_s$  such that  $v_1(\bar{B}) = \min\{v_1(B) : B \in \mathcal{B}_s\}$ ’. If a box with minimal value  $v_1$  is selected according to this rule at the beginning of each iteration, then one box is saved in each iteration in which a new nondominated point, that does not equal the ideal point in the first component, is generated in the selected box. Therefore, we obtain  $2|Z_N| - 1$  as new upper bound on the number of subproblems to

be solved in the tricriteria case.

## 5 Numerical results

For our tests we use five instances of a tricriteria multidimensional knapsack problem, i.e., a zero-one knapsack problem with three objectives and three constraints. The considered instances have already been employed for numerical experiments, e.g., in Laumanns et al. (2005) and Ozlen et al. (2014), wherefore we regard them as a good benchmark. The five instances correspond to five different numbers of (knapsack) items  $n = 10, 20, 30, 40, 50$ . The respective cardinality of the nondominated set is 9, 61, 195, 389 and 1048, as reported in Laumanns et al. (2005) and Ozlen et al. (2014) and verified by our algorithms. Note that we generated and saved the nondominated set of every instance once. For all methods presented in the following we always compare the respective generated representation with the saved nondominated set in order to verify that the complete nondominated set is computed correctly. The computational platform for our study is a compute server with 4x Intel Xeon E7540 CPUs (2.0 GHz) and 128 GB of memory. All algorithms are (re)implemented in MATLAB R2013a and call IBM ILOG CPLEX Optimization Studio Version 12.5 to solve the subproblems. We turned off the option of CPLEX to parallelize.

### 5.1 Validation of the $v$ -Split Algorithm

We test Algorithm 2 in combination with a weighted Tchebycheff method (WT) and an  $\varepsilon$ -constraint method (EC). In particular, we are interested in the question whether the upper bounds on the number of subproblems  $3|Z_N|-2$  (WT) and  $2|Z_N|-1$  (EC), which were derived in Sections 3 and 4, can be validated numerically. Both scalarizations are tested in an augmented and a two-stage formulation, see Steuer and Choo (1983) in case of the weighted Tchebycheff scalarization. Note that typically it makes a difference for computational time whether an augmented or a two-stage approach is used. This is caused by the fact that in the latter, two integer problems are solved in every subproblem in which the first stage yields a feasible solution. In contrast, when an augmented method is used, only one integer problem per subproblem is solved. The parameters of all scalarizations are set adaptively dependent on the upper bound vector of the box that is selected in the current iteration. For the two-stage variant, we solve (21) in the first stage and

$$\min \left\{ \sum_{i=1}^m z_i : z_j \leq z_j^*, j = 1, \dots, m \right\}$$

in the second stage, where  $z^*$  denotes the weakly nondominated point obtained in the first stage. For the augmented variant we exploit the given integrality in order to determine a suitable augmentation parameter. For a detailed description we refer to Dächert (2014).

In Table 1, the results are reported. The CPU times (in seconds) are averaged over three independent runs. To facilitate the comparison of the number of subproblems

$n$	$ Z_N $		Algorithm 2 (WT)		Algorithm 2 (EC)	
			CPU	#SP	CPU	#SP
10	9	TS	10.03	25	7.97	17
	(25/17)*	A	7.81		6.09	
20	61	TS	56.42	181	43.29	121
	(181/121)*	A	42.72		30.02	
30	195	TS	213.31	583	163.15	389
	(583/389)*	A	163.29		114.39	
40	389	TS	464.47	1165	361.74	777
	(1165/777)*	A	361.01		257.64	
50	1048	TS	1552.56	3142	1369.89	2095
	(3142/2095)*	A	1174.90		1012.15	

Table 1: Average CPU times (in seconds) and number of subproblems solved by Algorithm 2 in combination with a weighted Tchebycheff method (WT) and an  $\varepsilon$ -constraint method (EC). Each scalarization is evaluated in a two-stage (TS) and an augmented (A) formulation, respectively. In the second column, additionally to  $|Z_N|$ , the theoretical upper bounds  $3|Z_N| - 2$  (WT) and  $2|Z_N| - 1$  (EC) are given in parentheses (\*) for better comparison.

with the values  $3|Z_N| - 2$  and  $2|Z_N| - 1$ , respectively, we indicate these values in parentheses in the second column of Table 1.

Consider first the number of subproblems solved. From Table 1 we see that Algorithm 2 (WT) requires exactly  $3|Z_N| - 2$  and Algorithm 2 (EC) exactly  $2|Z_N| - 1$  subproblems for all problem sizes and for both formulations, i.e., for a two-stage (TS) and an augmented (A) formulation. Hence, the predicted upper bound on the number of subproblems is met precisely.

Regarding computational times in Table 1 we observe that all variants using (EC) are considerably faster than the variants using (WT) as the former solve about one third less subproblems compared to the latter. However, the savings with respect to computational time are not proportional to the savings with respect to the number of subproblems, in general. Recall from Section 4 that in order to achieve a saving with respect to the number of subproblems when using the  $\varepsilon$ -constraint method, the box at the beginning of each iteration cannot be selected arbitrarily, but a box with minimal value  $v_1$  must be identified. This causes an additional computational effort.

## 5.2 Comparison of Three Recent Algorithms to the New Algorithm

We additionally compare Algorithm 2 to three recent algorithms for generating complete representations for discrete multicriteria optimization problems with finite non-dominated set. These comprise the second algorithm stated in Lokman and Köksalan (2013), the approach of Kirlik and Sayın (2014) and the method of Ozlen et al. (2014). All three methods employ an  $\varepsilon$ -constraint scalarization, however, each in a different variant: Lokman and Köksalan (2013) use an augmented, Kirlik and Sayın (2014) a two-stage and Ozlen et al. (2014) a lexicographic  $\varepsilon$ -constraint method. In order to make the comparison in our numerical study as fair as possible, we implement and test

all algorithms with both, a two-stage and an augmented formulation, where the augmentation parameter is set adaptively according to the formulas presented in Dächert (2014). Note, however, that thereby we extend and/or modify the original algorithms of Lokman and Köksalan (2013), Kirlik and Sayın (2014) and Ozlen et al. (2014). Dependent on the formulation and particular parameters (as the augmentation parameter) used, the order in which the points are generated might change. However, this does not affect the general functionality of the respective algorithms.

In the literature, further methods to compute complete representations of discrete multicriteria optimization problems are presented, see, e.g., Tenfelde-Podehl (2003), Sylva and Crema (2004), Laumanns et al. (2005), Laumanns et al. (2006) and Özlen and Azizoglu (2009). However, all these approaches have been reported to be outperformed by at least one of the three methods that we incorporate into this numerical study.

We reimplement the algorithms of Lokman and Köksalan (LK), Kirlik and Sayın (KS) and Özlen, Burton and MacRae (OBM) with the following slight modifications. The algorithm of Lokman and Köksalan (2013) is originally formulated for problems in maximization format. For the sake of simplicity, we implement it for minimization problems. Moreover, as recommended in Lokman and Köksalan (2013), we keep the list of current nondominated points sorted, as, thereby, better computational times are obtained. In (LK) and (OBM), the right-hand side vectors of previously solved subproblems are saved as well as the corresponding results, i.e., a (nondominated) point or a value indicating infeasibility. Before solving a subproblem, the list of bounds is scanned to find a so-called relaxation. If a relaxed problem exists and it is either infeasible or the saved point is feasible for the current subproblem, then the current subproblem does not need to be solved since the solution of the relaxation is also valid for the considered subproblem. In this case, the bounds of the current subproblem should not be saved, as they do not contribute new information and, clearly, the shorter the list of bounds is, the better computational times can be expected. In the implementation of (KS) we change a detail with respect to the pseudocode given in Kirlik and Sayın (2014). When a new nondominated point is generated, all cells of the decomposition are checked twice in Kirlik and Sayın (2014): first, to identify the cells to be split, secondly, to remove cells that can not contain further nondominated points. We combine both checks, which are performed within two independent procedures in the original pseudocode, into one by removing cells that can not contain further nondominated points immediately after or during the split. In our implementation, this slight modification led to a huge saving of computational time.

The CPU times and the number of subproblems solved by all methods and for all instances are given in Table 2. Again, the given CPU times are averaged over three independent runs.

Regarding the number of subproblems solved, we observe that (KS) generates a complete representation within the lowest number of subproblems among all compared methods in all instances. Method (LK) requires the largest number of subproblems in all instances. While methods (OBM), (KS) and Algorithm 2 (EC), except (OBM) for  $n = 10$ , solve at most  $2|Z_N| - 1$  subproblems, (LK) exceeds this bound in all instances. These results go in line with the results of Lokman and Köksalan (2013), who state that

$n$	$ Z_N $		LK		KS		OBM		Algorithm 2 (EC)	
			CPU	#SP	CPU	#SP	CPU	#SP	CPU	#SP
10	9	TS	9.48	20	8.50	17	8.85	19	7.97	17
		A	6.67		6.07		6.46		6.09	
20	61	TS	53.04	127	50.08	115	48.50	117	43.29	121
		A	31.76	128	30.26		28.83		30.02	
30	195	TS	267.88	468	242.42	373	197.05	375	163.15	389
		A	159.12	464	155.89	372	110.33	374	114.39	
40	389	TS	657.58	852	701.95	739	430.84	741	361.74	777
		A	445.07		516.15	738	246.68	740	257.64	
50	1048	TS	4772.89	2193	4174.48	1913	1533.93	1915	1369.89	2095
		A	4129.47	2200	3603.67	1914	945.35	1916	1012.15	

Table 2: Average CPU times (in seconds) and number of subproblems solved by three state of the art algorithms and Algorithm 2 (EC). Each scalarization is evaluated in a two-stage (TS) and an augmented (A) formulation, respectively.

they solved on average 2.08 subproblems per nondominated point in their numerical study for a classic (one-dimensional) tricriteria knapsack problem. Our results also coincide with the results of Kirlik and Sayın (2014), who state that they required on average 1.97 and at most 1.99 subproblems per nondominated point with their algorithm when it was applied to a classic tricriteria knapsack problem. In our study, (KS) even performs better. In the worst case ( $n = 30$ ) less than 1.92 subproblems per nondominated point are solved. The results of (OBM) can be compared directly with the results reported in Ozlen et al. (2014), as they solve the same problem with the same instances. In their numerical study, 46, 333, 1204, 2357 and 6001 subproblems are solved for  $n = 10, \dots, 50$ , respectively. Interestingly, we obtain a considerably smaller number of subproblems with our reimplementation in all instances. A possible reason for this mismatch might be the scalarization used. While we apply (OBM) in combination with a two-stage and an augmented scalarization, a lexicographic  $\varepsilon$ -constraint scalarization is used in Ozlen et al. (2014), which might lead to a higher number of subproblems.

Considering CPU times, we obtain a slightly different picture. For the small instance  $n = 10$ , the CPU times of all methods are quite close. For all other problem sizes, the best CPU times are clearly obtained by Algorithm 2 (EC) and (OBM). When the augmented formulation is used, (OBM) consumes less CPU time than Algorithm 2 (EC). When the two-stage formulation is used, Algorithm 2 (EC) outperforms (OBM) for all problem sizes.

Both other methods, i.e., (LK) and (KS) require considerably more CPU time than (OBM) and Algorithm 2 (EC) for  $n = 20, 30, 40, 50$ . Besides  $n = 40$ , (LK) performs worst. As (LK) solves more subproblems than all other methods, this result is not surprising. In contrast, the rather bad performance of (KS) is not expected with regard to the fact that (KS) solves the lowest number of subproblems in basically all instances. The reason lies in the huge number of cells, which are maintained in (KS) and which are scanned several times during each iteration. This computational effort

is reflected in the CPU times.

We summarize that our new algorithm based on the  $v$ -split generates complete representations within the predicted number of subproblems. Moreover, it competes with state of the art algorithms.

## 6 Conclusion

In this paper, we positively answer the question whether there exists an algorithm which generates the entire nondominated set of a problem with more than two objectives by solving a number of subproblems which depends linearly on the number of nondominated points. We construct an algorithm which requires a linear number of subproblems for tricriteria problems. This is achieved by avoiding the generation of redundant boxes and by using neighborhood properties between the boxes. Further research should analyze whether and how the concept of individual subsets can be transferred to problems with more than three criteria. Moreover, the presented algorithm can be improved further by using the neighborhood properties also for identifying all boxes containing a current point. Thereby, no exhaustive search is needed in each iteration and the boxes can be updated more efficiently. Finally, with slight modifications, the new algorithm can also be used if only a representative subset of the nondominated set shall be generated.

## 7 Acknowledgement

The final publication is available at Springer via <http://dx.doi.org/10.1007/s10898-014-0205-z>.

## References

- Aneja, Y. P. and Nair, K. P. K. Bicriteria Transportation Problem. *Management Science*, 25:73–78, 1979.
- Chalmet, L., Lemonidis, L., and Elzinga, D. An algorithm for the bi-criterion integer programming problem. *European Journal of Operational Research*, 25:292–300, 1986.
- Dächert, K. *Adaptive Parametric Scalarizations in Multicriteria Optimization*. PhD thesis, University of Wuppertal, Germany, 2014.
- Dächert, K., Gorski, J., and Klamroth, K. An augmented weighted Tchebycheff method with adaptively chosen parameters for discrete bicriteria optimization problems. *Computers and Operations Research*, 39:2929–2943, 2012.
- Dhaenens, C., Lemesre, J., and Talbi, E.-G. K-PPM: A new exact method to solve multi-objective combinatorial optimization problems. *European Journal of Operational Research*, 200:45–53, 2010.

- Ehrgott, M. *Multicriteria Optimization*. Springer, Berlin, 2005.
- Eswaran, P., Ravindran, A., and Moskowitz, H. Algorithms for Nonlinear Integer Bicriterion Problems. *Journal of Optimization Theory and Applications*, 63(2):261–279, 1989.
- Hamacher, H. W., Pedersen, C. R., and Ruzika, S. Finding representative systems for discrete bicriteria optimization problems by box algorithms. *Operations Research Letters* 35, 3:336–344, 2007.
- Kirlik, G. and Sayın, S. A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems. *European Journal of Operational Research*, 232:479–488, 2014.
- Klein, D. and Hannan, E. An algorithm for the multiple objective integer linear programming problem. *European Journal of Operational Research*, 9:378–385, 1982.
- Laumanns, M., Thiele, L., and Zitzler, E. An Adaptive Scheme to Generate the Pareto Front Based on the Epsilon-Constraint Method. In Branke, J., Deb, K., Miettinen, K., and Steuer, R. E., editors, *Practical Approaches to Multi-Objective Optimization*, number 04461 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2005. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany. URL <http://drops.dagstuhl.de/opus/volltexte/2005/246>.
- Laumanns, M., Thiele, L., and Zitzler, E. An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method. *European Journal of Operational Research*, 169:932–942, 2006.
- Lokman, B. and Köksalan, M. Finding all nondominated points of multi-objective integer programs. *Journal of Global Optimization*, 57:347–365, 2013.
- Miettinen, K. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, 1999.
- Özlen, M. and Azizoğlu, M. Multi-objective integer programming: A general approach for generating all non-dominated solutions. *European Journal of Operations Research*, 199:25–35, 2009.
- Ozlen, M., Burton, B. A., and MacRae, C. A. G. Multi-Objective Integer Programming: An Improved Recursive Algorithm. *Journal of Optimization Theory and Applications*, 160(2):470–482, 2014.
- Przybylski, A., Gandibleux, X., and Ehrgott, M. A two phase method for multi-objective integer programming and its application to the assignment problem with three objectives. *Discrete Optimization*, 7:149–165, 2010.
- Ralphs, T., Saltzman, M., and Wiecek, M. M. An improved algorithm for solving biobjective integer programs. *Annals of Operations Research*, 147:43–70, 2006.

- Sayin, S. and Kouvelis, P. The Multiobjective Discrete Optimization Problem: A Weighted Min-Max Two-Stage Optimization Approach and a Bicriteria Algorithm. *Management Science*, 51(10):1572–1581, 2005.
- Solanki, R. S. Generating the Noninferior Set in Mixed Integer Biobjective Linear Programs: An Application to a Location Problem. *Computers and Operations Research*, 18(1):1–15, 1991.
- Steuer, R. E. and Choo, E. An Interactive Weighted Tchebycheff Procedure for Multiple Objective Programming. *Mathematical Programming*, 26:326–344, 1983.
- Sylva, J. and Crema, A. A method for finding the set of non-dominated vectors for multiple objective integer linear programs. *European Journal of Operational Research*, 158:46–55, 2004.
- Tenfelde-Podehl, D. A Recursive Algorithm for Multiobjective Combinatorial Optimization Problems with Q Criteria. Technical report, Institut für Mathematik, Technische Universität Graz, 2003.
- Ulungu, E. L. and Teghem, J. The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems. *Foundations of Computing and Decision Sciences*, 20(2):149–165, 1995.