On the computation of the nondominated set by scalarizations with adaptive parameter selection

Kerstin Dächert

joint work with Kathrin Klamroth (University of Wuppertal, Germany)

Stipendiary of the German Academic Exchange Service (DAAD)
Host institute: Université Paris-Dauphine - LAMSADE

23/06/2015

JFRO Université Paris VI

Outline

1. Introduction
   - Notation and definitions
   - Scalarizations
   - Parametric algorithm
   - Bicriteria case, relevant literature
2. Systematic, redundancy-free decomposition of the search region
   - Generic decomposition, redundancy
   - Split criterion to avoid redundancy
   - Linear worst-case bound in the tricriteria case
3. Numerical study
4. Conclusion

Notation

- Multicriteria optimization problem:

$$\min_{x \in X} \; [f_1(x), \dots, f_m(x)]^\top$$

with	$f_i : X \to \mathbb{R},\ i = 1, \dots, m$	objectives
	$m \in \mathbb{N}, m \geq 2$	number of objectives
	$X \subseteq \mathbb{R}^n$	feasible set

- Formulation in the image space $Z := f(X)$:

$$\min_{z \in Z} \; [z_1, \dots, z_m]^\top$$

- In this talk Z discrete, finite

Notation

- Multicriteria optimization problem:

$$\min_{x \in X} \; [f_1(x), \dots, f_m(x)]^\top$$

with	$f_i : X \to \mathbb{R}$, $i = 1, \dots, m$	objectives
	$m \in \mathbb{N}, m \geq 2$	number of objectives
	$X \subseteq \mathbb{R}^n$	feasible set

- Formulation in the image space $Z := f(X)$:

$$\min_{z \in Z} \; [z_1, \dots, z_m]^\top$$

- In this talk Z discrete, finite

Notation

- Multicriteria optimization problem:

$$\min_{x \in X} \ [f_1(x), \dots, f_m(x)]^\top$$

with

$f_i : X \to \mathbb{R},\ i = 1, \dots, m$	objectives
$m \in \mathbb{N}, m \geq 2$	number of objectives
$X \subseteq \mathbb{R}^n$	feasible set

- Formulation in the image space $Z := f(X)$:

$$\min_{z \in Z} \ [z_1, \dots, z_m]^\top$$

- In this talk Z discrete, finite

Concepts of optimality

Definition (Efficiency, Nondominance)

$\bar{x} \in X$ efficient (Pareto-optimal), $f(\bar{x})$ nondominated

$:\Leftrightarrow \nexists x \in X : f(x) \leq f(\bar{x})$, i.e. $f_i(x) \leq f_i(\bar{x})$ for all $i = 1, \ldots, m$ and $f_j(x) < f_j(\bar{x})$ for at least one $j \in \{1, \ldots, m\}$

Geometrically:

$$\left(\{f(\bar{x})\} - \mathbb{R}^m_{\geqq}\right) \cap Z = \{f(\bar{x})\}$$

with

$$\mathbb{R}^m_{\geqq} := \{z \in \mathbb{R}^m : z_i \geq 0, i = 1, \ldots, m\}$$

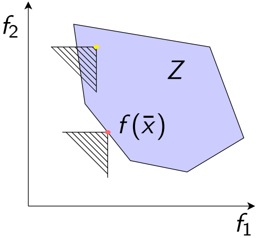
Z_N set of nondominated points, X_E set of efficient solutions

Concepts of optimality

Definition (Efficiency, Nondominance)

$\bar{x} \in X$ efficient (Pareto-optimal), $f(\bar{x})$ nondominated

$:\Leftrightarrow \nexists x \in X : f(x) \leq f(\bar{x})$, i.e. $f_i(x) \leq f_i(\bar{x})$ for all $i = 1, \ldots, m$ and $f_j(x) < f_j(\bar{x})$ for at least one $j \in \{1, \ldots, m\}$



Geometrically:

$$\left(\{f(\bar{x})\} - \mathbb{R}^m_{\geqq}\right) \cap Z = \{f(\bar{x})\}$$

with

$$\mathbb{R}^m_{\geqq} := \{z \in \mathbb{R}^m : z_i \geq 0, i = 1, \ldots, m\}$$

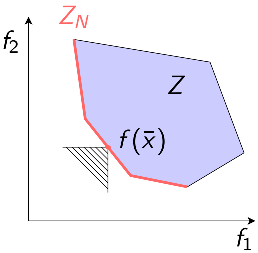
Z_N set of nondominated points, X_E set of efficient solutions

Concepts of optimality

Definition (Efficiency, Nondominance)

$\bar{x} \in X$ efficient (Pareto-optimal), $f(\bar{x})$ nondominated

$:\Leftrightarrow \nexists x \in X : f(x) \leq f(\bar{x})$, i.e. $f_i(x) \leq f_i(\bar{x})$ for all $i = 1, \ldots, m$ and $f_j(x) < f_j(\bar{x})$ for at least one $j \in \{1, \ldots, m\}$



Geometrically:

$$\left(\{f(\bar{x})\} - \mathbb{R}^m_{\geqq}\right) \cap Z = \{f(\bar{x})\}$$

with

$$\mathbb{R}^m_{\geqq} := \{z \in \mathbb{R}^m : z_i \geq 0, i = 1, \ldots, m\}$$

Z_N set of nondominated points, X_E set of efficient solutions

Concepts of optimality (cont.)

Definition (Weak efficiency, weak nondominance)

$\bar{x} \in X$ weakly efficient, $f(\bar{x})$ weakly nondominated

$:\Leftrightarrow \nexists x \in X : f_i(x) < f_i(\bar{x})$ for all $i = 1, \dots, m$

Geometrically:

$$(\{f(\bar{x})\} - \mathbb{R}^m_>) \cap Z = \emptyset$$

with

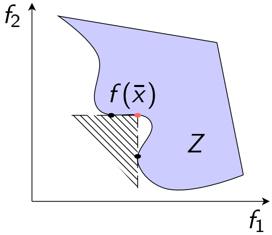
$$\mathbb{R}^m_> := \{z \in \mathbb{R}^m : z_i > 0, i = 1, \dots, m\}$$

Concepts of optimality (cont.)

Definition (Weak efficiency, weak nondominance)

$\bar{x} \in X$ weakly efficient, $f(\bar{x})$ weakly nondominated

$:\Leftrightarrow \nexists x \in X : f_i(x) < f_i(\bar{x})$ for all $i = 1, \ldots, m$



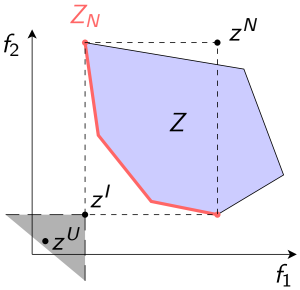
Geometrically:

$$(\{f(\bar{x})\} - \mathbb{R}^m_>) \cap Z = \emptyset$$

with

$$\mathbb{R}^m_> := \{z \in \mathbb{R}^m : z_i > 0, i = 1, \ldots, m\}$$

Bounds on the nondominated set



- Ideal point $z^I \in \mathbb{R}^m$:

$$z_i^I := \min_{x \in X_E} f_i(x) = \min_{x \in X} f_i(x) \quad \forall\, i = 1, \ldots, m$$

- Utopian point $z^U \in \mathbb{R}^m$:

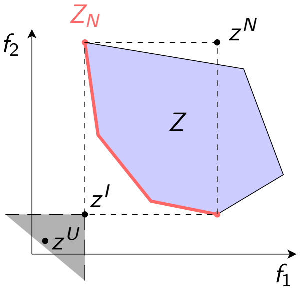
$$z^U \in \left(\{z^I\} - \mathbb{R}^m_>\right)$$

- Nadir point $z^N \in \mathbb{R}^m$:

$$z_i^N := \max_{x \in X_E} f_i(x) \quad \forall\, i = 1, \ldots, m$$

(difficult to compute for $m \geq 3$!)

Bounds on the nondominated set



- Ideal point $z^I \in \mathbb{R}^m$:

$$z_i^I := \min_{x \in X_E} f_i(x) = \min_{x \in X} f_i(x) \quad \forall\, i = 1, \dots, m$$

- Utopian point $z^U \in \mathbb{R}^m$:

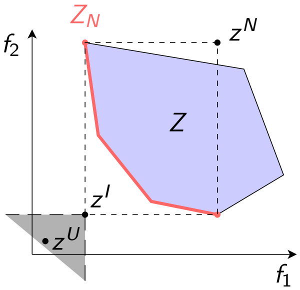
$$z^U \in \left(\{z^I\} - \mathbb{R}^m_>\right)$$

- Nadir point $z^N \in \mathbb{R}^m$:

$$z_i^N := \max_{x \in X_E} f_i(x) \quad \forall\, i = 1, \dots, m$$

(difficult to compute for $m \geq 3$!)

Bounds on the nondominated set



- Ideal point $z^I \in \mathbb{R}^m$:

$$z_i^I := \min_{x \in X_E} f_i(x)$$
$$= \min_{x \in X} f_i(x) \quad \forall\, i = 1, \dots, m$$

- Utopian point $z^U \in \mathbb{R}^m$:

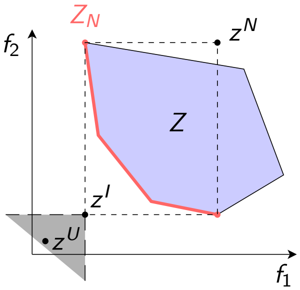
$$z^U \in \left(\{z^I\} - \mathbb{R}^m_>\right)$$

- Nadir point $z^N \in \mathbb{R}^m$:

$$z_i^N := \max_{x \in X_E} f_i(x) \quad \forall\, i = 1, \dots, m$$

(difficult to compute for $m \geq 3$!)

Bounds on the nondominated set



- Ideal point $z^I \in \mathbb{R}^m$:

$$z_i^I := \min_{x \in X_E} f_i(x)$$
$$= \min_{x \in X} f_i(x) \quad \forall\, i = 1, \dots, m$$

- Utopian point $z^U \in \mathbb{R}^m$:

$$z^U \in \left(\{z^I\} - \mathbb{R}^m_{>}\right)$$

- Nadir point $z^N \in \mathbb{R}^m$:

$$z_i^N := \max_{x \in X_E} f_i(x) \quad \forall\, i = 1, \dots, m$$

(difficult to compute for $m \geq 3$!)

Solution concepts

Scalarization

Convert vector-valued into scalar-valued problem

Classic scalarization methods:

1. Weighted Sum Method
2. ε-constraint method
3. Weighted Tchebycheff method

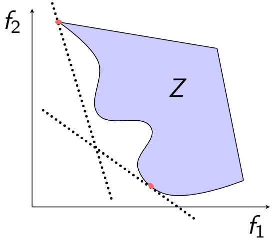
Weighted Sum Method

Formulation (Gass & Saaty, 1955):

$$\min_{x \in X} \sum_{i=1}^{m} \lambda_i f_i(x) \qquad \text{(WS)}$$

with

$$\lambda_i \geq 0, i = 1, \ldots, m, \sum_{i=1}^{m} \lambda_i = 1$$



Properties:

- Every optimal solution of (WS) is weakly efficient, for $\lambda \in \mathbb{R}^m_>$ efficient
- For every nondominated point $f(x) \in \partial\text{conv}(Z)$ exists $\lambda \in \mathbb{R}^m_\geq$ such that x optimal solution of (WS)

$\Rightarrow$ (WS) not suited for non-convex and discrete problems

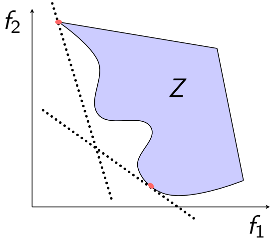
Weighted Sum Method

Formulation (Gass & Saaty, 1955):

$$\min_{x \in X} \sum_{i=1}^{m} \lambda_i f_i(x) \qquad \text{(WS)}$$

with

$$\lambda_i \geq 0, i = 1, \ldots, m, \sum_{i=1}^{m} \lambda_i = 1$$



Properties:

- Every optimal solution of (WS) is weakly efficient, for $\lambda \in \mathbb{R}^m_>$ efficient
- For every nondominated point $f(x) \in \partial\text{conv}(Z)$ exists $\lambda \in \mathbb{R}^m_\geqq$ such that x optimal solution of (WS)

$\Rightarrow$ (WS) not suited for non-convex and discrete problems

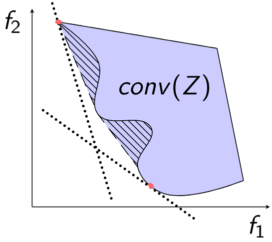
Weighted Sum Method

Formulation (Gass & Saaty, 1955):

$$\min_{x \in X} \sum_{i=1}^{m} \lambda_i f_i(x) \qquad \text{(WS)}$$

with

$$\lambda_i \geq 0, i = 1, \ldots, m, \sum_{i=1}^{m} \lambda_i = 1$$



Properties:

- Every optimal solution of (WS) is weakly efficient, for $\lambda \in \mathbb{R}^m_>$ efficient
- For every nondominated point $f(x) \in \partial\text{conv}(Z)$ exists $\lambda \in \mathbb{R}^m_{\geqq}$ such that x optimal solution of (WS)

$\Rightarrow$ (WS) not suited for non-convex and discrete problems

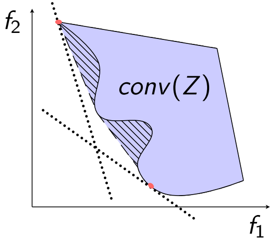
Weighted Sum Method

Formulation (Gass & Saaty, 1955):

$$\min_{x \in X} \sum_{i=1}^{m} \lambda_i f_i(x) \qquad \text{(WS)}$$

with

$$\lambda_i \geq 0, i = 1, \dots, m, \sum_{i=1}^{m} \lambda_i = 1$$



Properties:

- Every optimal solution of (WS) is weakly efficient, for $\lambda \in \mathbb{R}^m_>$ efficient
- For every nondominated point $f(x) \in \partial\text{conv}(Z)$ exists $\lambda \in \mathbb{R}^m_{\geqq}$ such that x optimal solution of (WS)

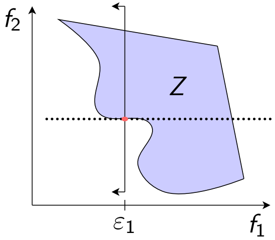
  $\Rightarrow$ (WS) not suited for non-convex and discrete problems

ε-Constraint Method

Formulation (Haimes et al., 1971):

$$\begin{array}{ll} \min & f_i(x) \\ \text{s.t.} & f_k(x) \leq \varepsilon_k \; \forall\, k \neq i \\ & x \in X \end{array} \qquad \text{(EC)}$$

with $\varepsilon \in \mathbb{R}^m$, $i \in \{1, \ldots, m\}$ arbitrary



Properties:

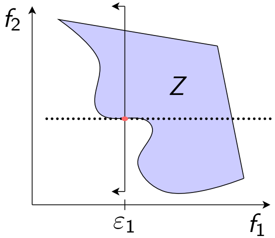
- Every optimal solution of (EC) is weakly efficient
- For every nondominated point $f(x)$ exists $\varepsilon \in \mathbb{R}^m$ such that x optimal solution of (EC)

ε-Constraint Method

Formulation (Haimes et al., 1971):

$$\begin{array}{ll} \min & f_i(x) \\ \text{s.t.} & f_k(x) \leq \varepsilon_k \; \forall\, k \neq i \\ & x \in X \end{array} \qquad \text{(EC)}$$

with $\varepsilon \in \mathbb{R}^m$, $i \in \{1, \ldots, m\}$ arbitrary



Properties:

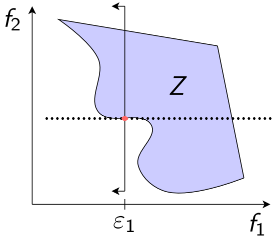
- Every optimal solution of (EC) is weakly efficient
- For every nondominated point $f(x)$ exists $\varepsilon \in \mathbb{R}^m$ such that x optimal solution of (EC)

ε-Constraint Method

Formulation (Haimes et al., 1971):

$$\begin{array}{ll} \min & f_i(x) \\ \text{s.t.} & f_k(x) \leq \varepsilon_k \ \forall\, k \neq i \\ & x \in X \end{array} \qquad \text{(EC)}$$

with $\varepsilon \in \mathbb{R}^m$, $i \in \{1, \ldots, m\}$ arbitrary



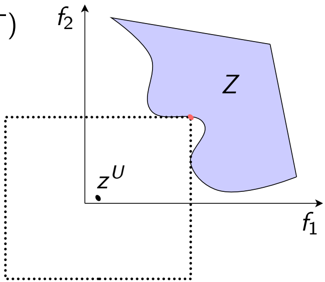
Properties:

- Every optimal solution of (EC) is weakly efficient
- For every nondominated point $f(x)$ exists $\varepsilon \in \mathbb{R}^m$ such that x optimal solution of (EC)

Weighted Tchebycheff Method

Formulation (Bowman, 1976):

$$\min_{x \in X} \max_{i=1,\ldots,m} \left\{ w_i \left| f_i(x) - z_i^U \right| \right\} \qquad \text{(WT)}$$



with

$w_i > 0, i = 1, \ldots, m,$

$\sum_{i=1}^{m} w_i = 1,$

z^U utopian point

Properties:

- Every optimal solution of (WT) is weakly efficient
- For every nondominated point $f(x)$ exists $w \in \mathbb{R}^m_>$ such that x optimal solution of (WT)

Weighted Tchebycheff Method

Formulation (Bowman, 1976):

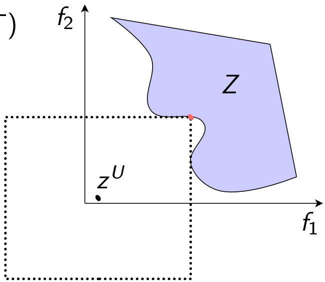
$$\min_{x \in X} \max_{i=1,\dots,m} \left\{ w_i \left| f_i(x) - z_i^U \right| \right\} \qquad \text{(WT)}$$

with

$w_i > 0, i = 1, \dots, m,$

$\sum_{i=1}^{m} w_i = 1,$

z^U utopian point

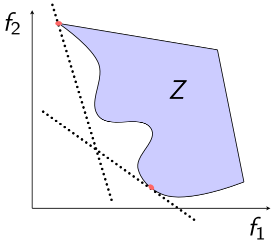


Properties:

- Every optimal solution of (WT) is weakly efficient
- For every nondominated point $f(x)$ exists $w \in \mathbb{R}^m_>$ such that x optimal solution of (WT)

Subproblem and Parametric Algorithm

- One scalarized problem yields (at most) one nondominated point
- Vary parameters in a systematic way in order to obtain Z_N (or a subset of it)



We use the following definitions:

Definition (Subproblem)

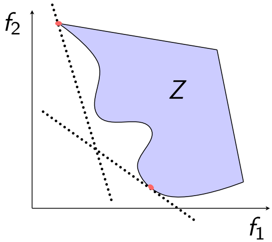
= Scalarized problem with a certain parameter choice

Definition (Parametric algorithm)

= Iterative solution of subproblems with different parameter choices

Subproblem and Parametric Algorithm

- One scalarized problem yields (at most) one nondominated point
- Vary parameters in a systematic way in order to obtain Z_N (or a subset of it)



We use the following definitions:

Definition (Subproblem)

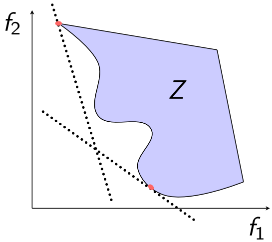
= Scalarized problem with a certain parameter choice

Definition (Parametric algorithm)

= Iterative solution of subproblems with different parameter choices

Subproblem and Parametric Algorithm

- One scalarized problem yields (at most) one nondominated point
- Vary parameters in a systematic way in order to obtain Z_N (or a subset of it)



We use the following definitions:

Definition (Subproblem)

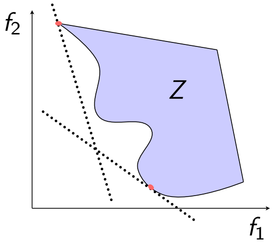
= Scalarized problem with a certain parameter choice

Definition (Parametric algorithm)

= Iterative solution of subproblems with different parameter choices

Subproblem and Parametric Algorithm

- One scalarized problem yields (at most) one nondominated point
- Vary parameters in a systematic way in order to obtain Z_N (or a subset of it)



We use the following definitions:

Definition (Subproblem)

= Scalarized problem with a certain parameter choice

Definition (Parametric algorithm)

= Iterative solution of subproblems with different parameter choices

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

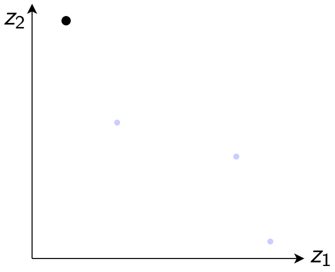
Parametric algorithm in the discrete, bicriteria case (well-known):

Example with four nondominated points

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

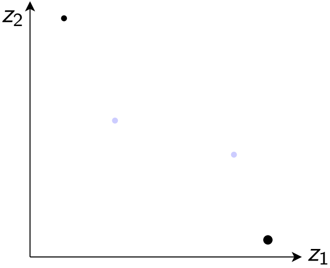


Compute lexicographic minima, determine bounds

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

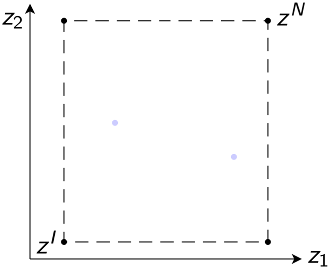


Compute lexicographic minima, determine bounds

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

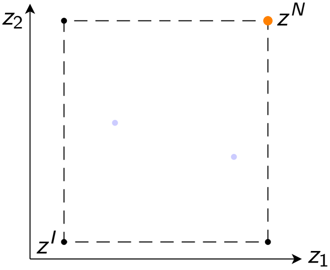


Initial search region

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

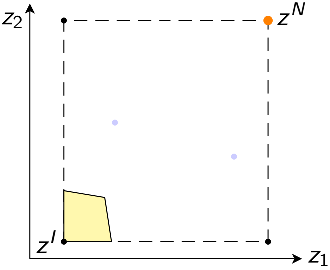


Select rectangle, given by its (local) upper bound

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

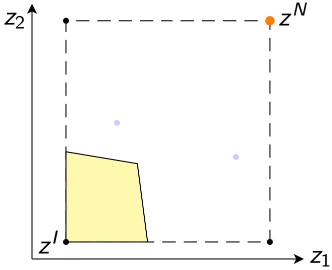


Solve subproblem
(e.g. augmented weighted Tchebycheff problem with z^I as reference point)

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

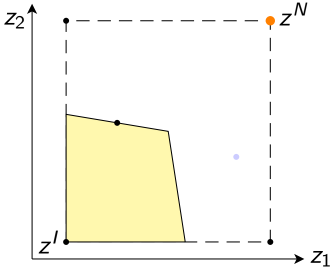


Solve subproblem
(e.g. augmented weighted Tchebycheff problem with z^I as reference point)

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

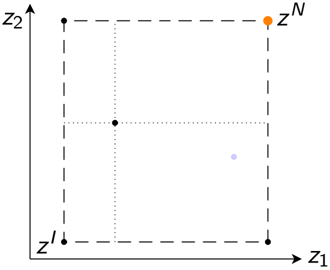


Solve subproblem
(e.g. augmented weighted Tchebycheff problem with z^I as reference point)

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

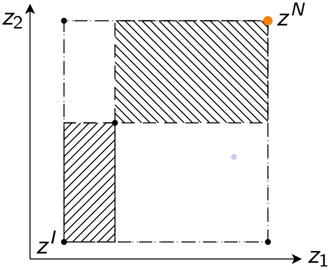


New nondominated point found

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

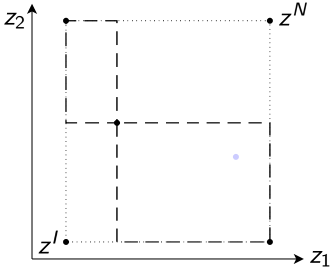


Exclude sets that cannot contain further nondominated points

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

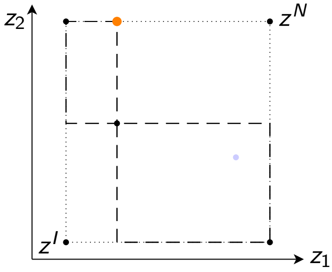


Reduce search region
(split former rectangle into two new rectangles)

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

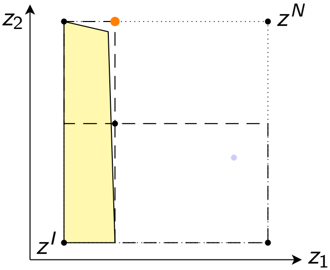


New iteration:
select rectangle

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

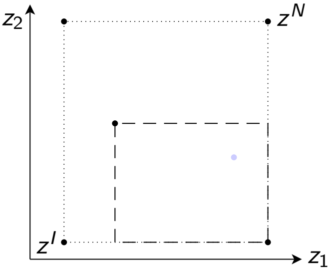


Solve subproblem
(no new nondominated point found)

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

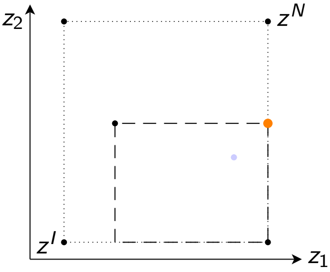


Reduce search region
(remove investigated subset)

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

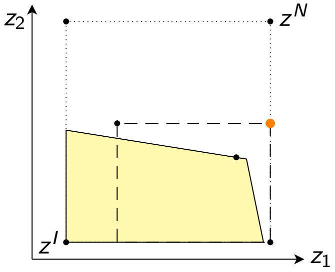


New iteration:
select rectangle

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

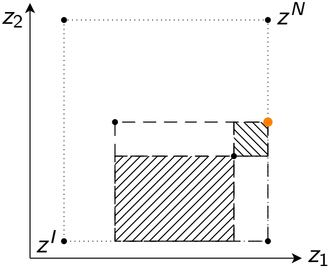


Solve subproblem

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

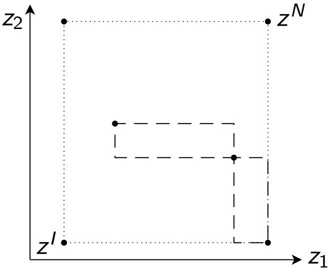


Decompose search region

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

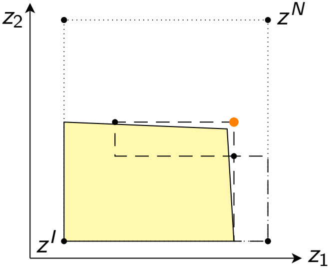


Decompose search region

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

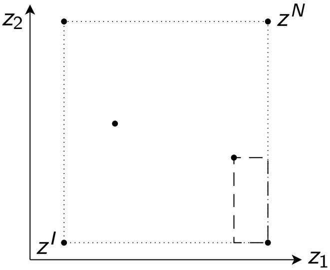


Select rectangle,
solve subproblem

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

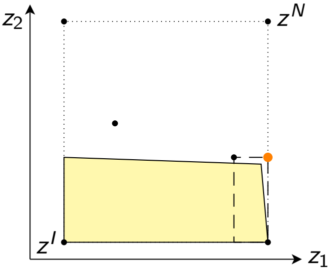


Reduce search region

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

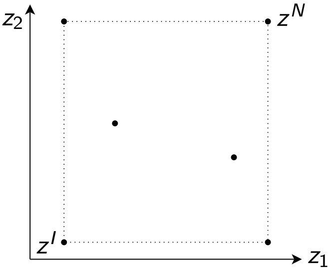


Select rectangle,
solve subproblem

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):

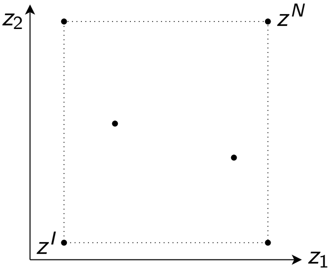


Reduce search region,
no rectangles left $\Rightarrow$ done

Adaptive Parametric Algorithm

- determine parameters during parametric algorithm
- dependent on nondominated points that are already known

Parametric algorithm in the discrete, bicriteria case (well-known):



Particular interest in the discrete case: How many subproblems do we have to solve?

$$N + (N - 1) = 2N - 1$$

(independent of scalarization!)

Literature

	Reference	Scalarization	Subproblems
$m = 2$	Aneja & Nair (1979) Chalmet et al. (1986) Ralphs et al. (2006)	WS EC WT/AWT	$2N - 1$

$m \geq 2$	Laumanns et al. (2006) Özlen & Azizoğlu (2009) Lokman & Köksalan (2013) Ozlen et al. (2014) Kirlik & Sayın (2014)	EC	$\mathcal{O}(N^{m-1})$

Literature

	Reference	Scalarization	Subproblems
$m = 2$	Aneja & Nair (1979)	WS	$2N - 1$
	Chalmet et al. (1986)	EC	
	Ralphs et al. (2006)	WT/AWT	

$m \geq 2$	Laumanns et al. (2006)	EC	$\mathcal{O}(N^{m-1})$
	Özlen & Azizoğlu (2009)		
	Lokman & Köksalan (2013)		
	Ozlen et al. (2014)		
	Kirlik & Sayın (2014)		

Case $m = 3$

- Best known worst-case bound on number of subproblems: $\mathcal{O}(N^2)$
- All algorithms use ε-constraint method as scalarization
- Numerical studies in the literature suggest that less than $\mathcal{O}(N^2)$ subproblems are needed
- Open question: Linear worst-case bound?

Contribution:

New adaptive parametric algorithm

- $\mathcal{O}(N)$ subproblems for $m = 3$
- independent of particular scalarization

Case $m = 3$

- Best known worst-case bound on number of subproblems: $\mathcal{O}(N^2)$
- All algorithms use ε-constraint method as scalarization
- Numerical studies in the literature suggest that less than $\mathcal{O}(N^2)$ subproblems are needed
- Open question: Linear worst-case bound?

Contribution:

New adaptive parametric algorithm

- $\mathcal{O}(N)$ subproblems for $m = 3$
- independent of particular scalarization

Case $m = 3$

- Best known worst-case bound on number of subproblems: $\mathcal{O}(N^2)$
- All algorithms use ε-constraint method as scalarization
- Numerical studies in the literature suggest that less than $\mathcal{O}(N^2)$ subproblems are needed
- Open question: Linear worst-case bound?

Contribution:

New adaptive parametric algorithm

- $\mathcal{O}(N)$ subproblems for $m = 3$
- independent of particular scalarization

Case $m = 3$

- Best known worst-case bound on number of subproblems: $\mathcal{O}(N^2)$
- All algorithms use ε-constraint method as scalarization
- Numerical studies in the literature suggest that less than $\mathcal{O}(N^2)$ subproblems are needed
- Open question: Linear worst-case bound?

Contribution:

New adaptive parametric algorithm

- $\mathcal{O}(N)$ subproblems for $m = 3$
- independent of particular scalarization

Case $m = 3$

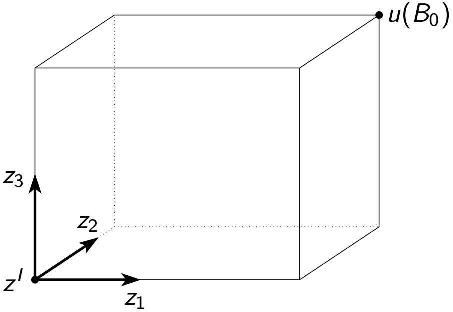
- Best known worst-case bound on number of subproblems: $\mathcal{O}(N^2)$
- All algorithms use ε-constraint method as scalarization
- Numerical studies in the literature suggest that less than $\mathcal{O}(N^2)$ subproblems are needed
- Open question: Linear worst-case bound?

Contribution:

New adaptive parametric algorithm

- $\mathcal{O}(N)$ subproblems for $m = 3$
- independent of particular scalarization

Decomposition of search region for $m \geq 2$

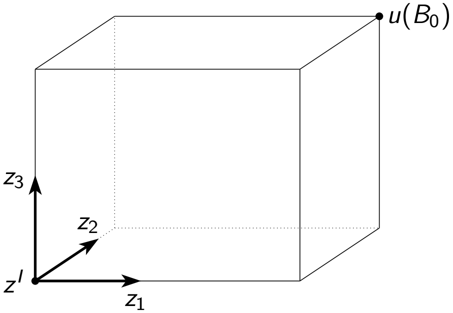


Initial search region (box)

$$B_0 := \{z \in \mathbb{R}^m \, : \, z^I \leqq z < u\}$$

with $u_i := \max_{x \in X}\{f_i(x)\} + \delta, \; i = 1, \ldots, m, \; \delta > 0$

Decomposition of search region for $m \geq 2$

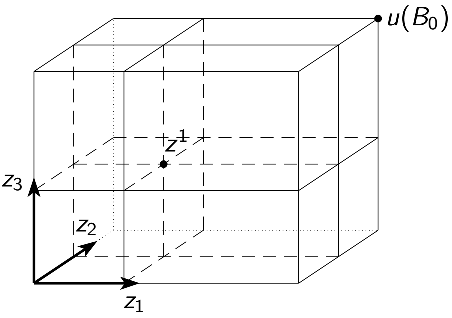


Initial search region (box)

$$B_0 := \{z \in \mathbb{R}^m \, : \, z^I \leqq z < u\}$$

$\rightsquigarrow$ Note: Every box B characterized by $u(B)$

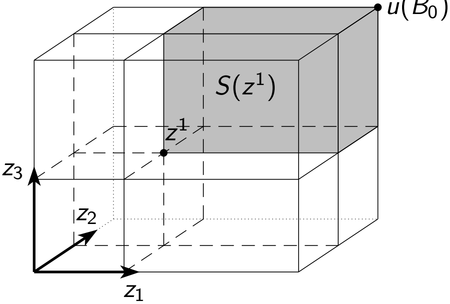
Decomposition of search region for $m \geq 2$



Solve subproblem in $B_0 \rightsquigarrow z^1 \in Z_N \cap B_0$

Insertion of z^1 into B_0

Decomposition of search region for $m \geq 2$

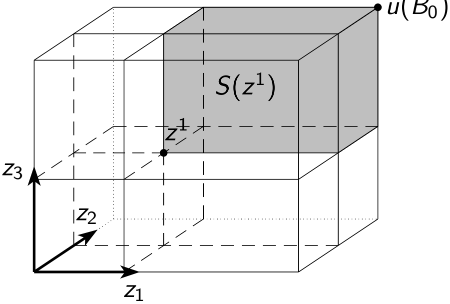


By definition of nondominance:

$$Z_N \cap S(z^1) = \{z^1\} \quad \text{with} \quad S(z^1) := \{z \in B_0 : z \geqq z^1\}$$

$\Rightarrow$ All $z \in Z_N \setminus \{z^1\}$ contained in $B_0 \setminus S(z^1)$

Decomposition of search region for $m \geq 2$

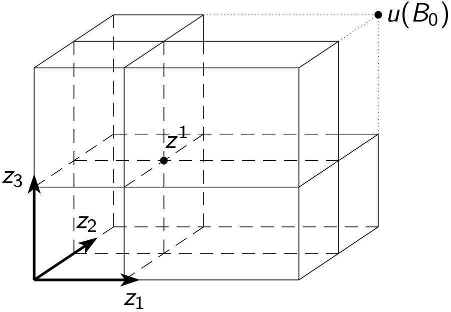


By definition of nondominance:

$$Z_N \cap S(z^1) = \{z^1\} \quad \text{with} \quad S(z^1) := \{z \in B_0 : z \geqq z^1\}$$

$\Rightarrow$ All $z \in Z_N \setminus \{z^1\}$ contained in $B_0 \setminus S(z^1)$

Decomposition of search region for $m \geq 2$

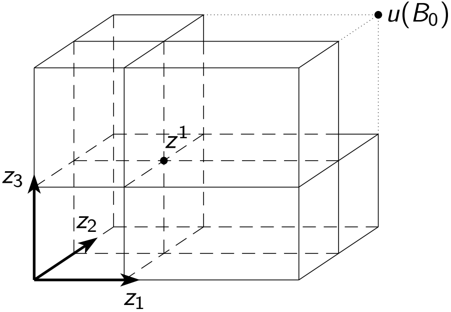


By definition of nondominance:

$$Z_N \cap S(z^1) = \{z^1\} \quad \text{with} \quad S(z^1) := \{z \in B_0 : z \geqq z^1\}$$

$\Rightarrow$ All $z \in Z_N \backslash \{z^1\}$ contained in $B_0 \backslash S(z^1)$

Decomposition of search region for $m \geq 2$

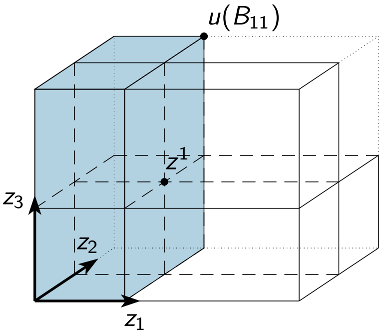


Representation of $B_0 \backslash S(z^1)$ by $\bigcup_{i=1}^{m} B_{1,i}$ with

$$B_{1,i} := \{z \in B_0 : z_i < z_i^1\}, \quad i = 1, \ldots, m,$$

i.e. $u_i(B_{1,i}) := z_i^1$, $u_j(B_{1,i}) := u_j(B_0)\ \forall j \neq i$

Decomposition of search region for $m \geq 2$

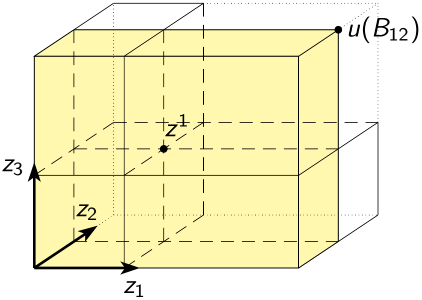


Representation of $B_0 \backslash S(z^1)$ by $\bigcup_{i=1}^m B_{1,i}$ with

$$B_{1,i} := \{z \in B_0 : z_i < z_i^1\}, \quad i = 1, \ldots, m,$$

i.e. $u_i(B_{1,i}) := z_i^1$, $u_j(B_{1,i}) := u_j(B_0)\ \forall j \neq i$

Decomposition of search region for $m \geq 2$

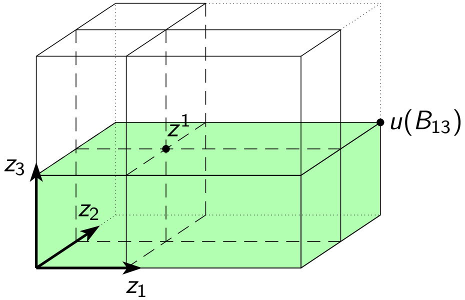


Representation of $B_0 \backslash S(z^1)$ by $\bigcup_{i=1}^{m} B_{1,i}$ with

$$B_{1,i} := \{z \in B_0 : z_i < z_i^1\}, \quad i = 1, \ldots, m,$$

i.e. $u_i(B_{1,i}) := z_i^1$, $u_j(B_{1,i}) := u_j(B_0) \;\; \forall j \neq i$

Decomposition of search region for $m \geq 2$

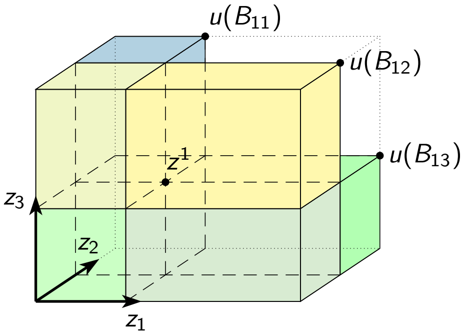


Representation of $B_0 \setminus S(z^1)$ by $\bigcup_{i=1}^{m} B_{1,i}$ with

$$B_{1,i} := \{z \in B_0 : z_i < z_i^1\}, \quad i = 1, \ldots, m,$$

i.e. $u_i(B_{1,i}) := z_i^1$, $u_j(B_{1,i}) := u_j(B_0)\ \ \forall j \neq i$

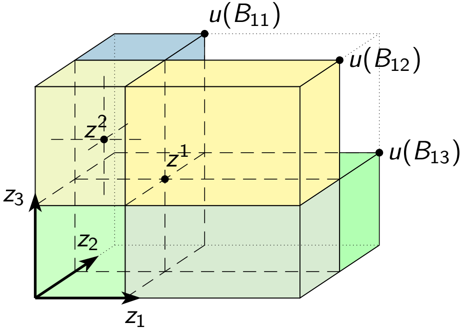
Decomposition of search region for $m \geq 2$



$\Rightarrow$ Decomposition of $B_0 \setminus S(z^1)$ into m (non-disjoint) subboxes

(see Dhaenens et al. (2010), Przybylski et al. (2010))

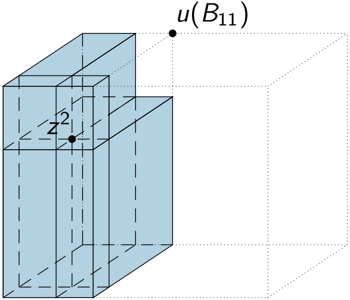
Redundancy for $m \geq 3$



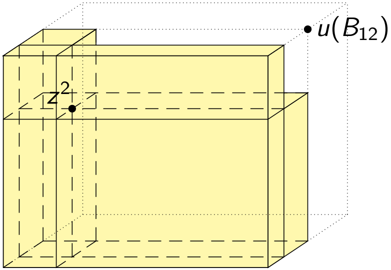
Let $z^2 \in (B_{11} \cap B_{12})$

$\Rightarrow$ Split B_{11} and B_{12} into 3 new boxes, resp.

Redundancy for $m \geq 3$



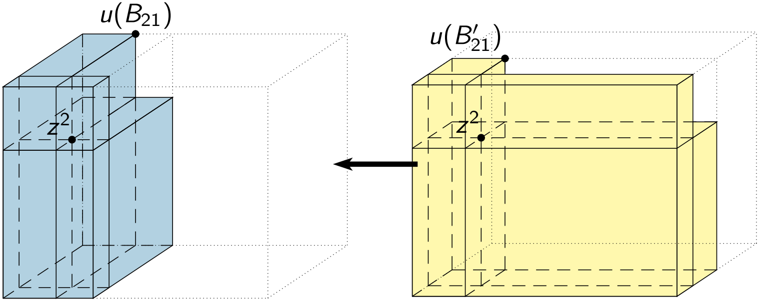
B_{21}, B_{22}, B_{23}



$B'_{21}, B'_{22}, B'_{23}$

Split of B_{11} and B_{12} wrt. z^2

Redundancy for $m \geq 3$

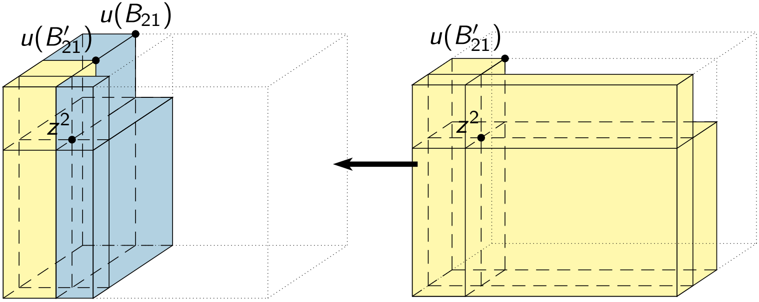


$\underline{\mathbf{B_{21}}}, B_{22}, B_{23}$ $\underline{\mathbf{B'_{21}}}, B'_{22}, B'_{23}$

Split wrt. $i = 1$:

$B'_{21} \subseteq B_{21} \iff u(B'_{21}) \leq u(B_{21})$

Redundancy for $m \geq 3$

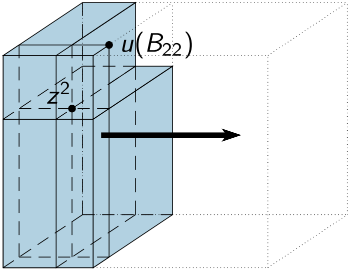


$\underline{\mathbf{B_{21}}}, B_{22}, B_{23}$ $\quad\quad$ $\underline{\mathbf{B'_{21}}}, B'_{22}, B'_{23}$

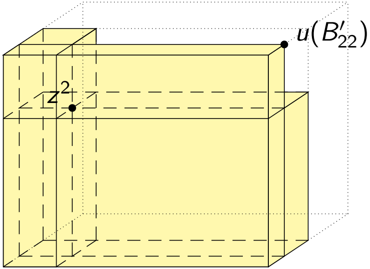
Split wrt. $i = 1$:

$$B'_{21} \subseteq B_{21} \iff u(B'_{21}) \leqq u(B_{21})$$

Redundancy for $m \geq 3$



$B_{21}, \underline{\mathbf{B_{22}}}, B_{23}$

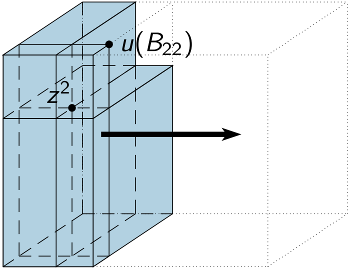


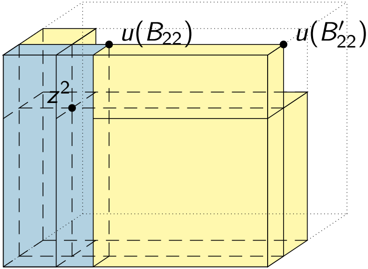
$B'_{21}, \underline{\mathbf{B'_{22}}}, B'_{23}$

Split wrt. $i = 2$:

$$B_{22} \subseteq B'_{22} \iff u(B_{22}) \leq u(B'_{22})$$

Redundancy for $m \geq 3$





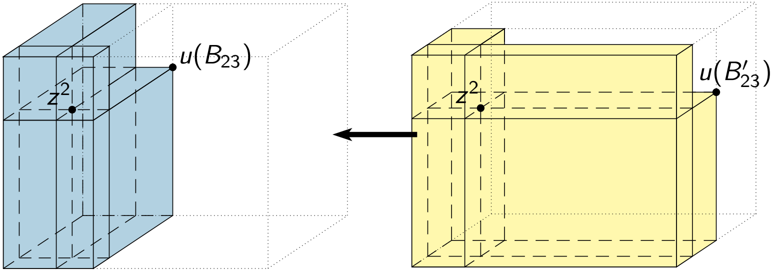
$B_{21}, \underline{\mathbf{B_{22}}}, B_{23}$

$B'_{21}, \underline{\mathbf{B'_{22}}}, B'_{23}$

Split wrt. $i = 2$:

$$B_{22} \subseteq B'_{22} \iff u(B_{22}) \leqq u(B'_{22})$$

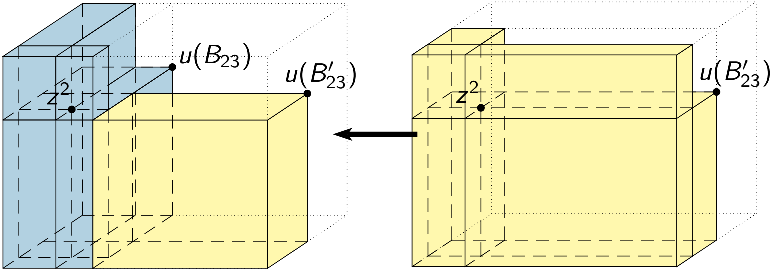
Redundancy for $m \geq 3$



$B_{21}, B_{22}, \underline{\mathbf{B_{23}}}$ $B'_{21}, B'_{22}, \underline{\mathbf{B'_{23}}}$

Split wrt. $i = 3$:
no redundancy

Redundancy for $m \geq 3$



$B_{21}, B_{22}, \underline{\mathbf{B_{23}}}$ $B'_{21}, B'_{22}, \underline{\mathbf{B'_{23}}}$

Split wrt. $i = 3$:
no redundancy

Redundancy

Generic split produces redundant boxes

- Example: already in 2nd iteration, two of the six new boxes redundant
- if redundant boxes are kept in decomposition
  - additional, unnecessary subproblems are solved
  - increases running time of algorithm

$\Rightarrow$ avoid redundant boxes

Identifying redundant boxes:

1. compare upper bounds $u(B)$ pairwise, remove redundant ones (Przybylski et al. (2010))
2. detect redundant boxes before their creation, i.e. only generate non-redundant boxes

Redundancy

Generic split produces redundant boxes

- Example: already in 2nd iteration, two of the six new boxes redundant
- if redundant boxes are kept in decomposition
  - additional, unnecessary subproblems are solved
  - increases running time of algorithm

$\Rightarrow$ avoid redundant boxes

Identifying redundant boxes:

1. compare upper bounds $u(B)$ pairwise, remove redundant ones (Przybylski et al. (2010))
2. detect redundant boxes before their creation, i.e. only generate non-redundant boxes

Redundancy

Generic split produces redundant boxes

- Example: already in 2nd iteration, two of the six new boxes redundant
- if redundant boxes are kept in decomposition
  - additional, unnecessary subproblems are solved
  - increases running time of algorithm

$\Rightarrow$ avoid redundant boxes

Identifying redundant boxes:

1. compare upper bounds $u(B)$ pairwise, remove redundant ones (Przybylski et al. (2010))
2. detect redundant boxes before their creation, i.e. only generate non-redundant boxes

Redundancy

Generic split produces redundant boxes

- Example: already in 2nd iteration, two of the six new boxes redundant
- if redundant boxes are kept in decomposition
  - additional, unnecessary subproblems are solved
  - increases running time of algorithm

$\Rightarrow$ avoid redundant boxes

Identifying redundant boxes:

1. compare upper bounds $u(B)$ pairwise, remove redundant ones (Przybylski et al. (2010))
2. detect redundant boxes before their creation, i.e. only generate non-redundant boxes

Redundancy

Generic split produces redundant boxes

- Example: already in 2nd iteration, two of the six new boxes redundant
- if redundant boxes are kept in decomposition
  - additional, unnecessary subproblems are solved
  - increases running time of algorithm

$\Rightarrow$ avoid redundant boxes

Identifying redundant boxes:

1. compare upper bounds $u(B)$ pairwise, remove redundant ones (Przybylski et al. (2010))
2. detect redundant boxes before their creation, i.e. only generate non-redundant boxes

Redundancy

Generic split produces redundant boxes

- Example: already in 2nd iteration, two of the six new boxes redundant
- if redundant boxes are kept in decomposition
  - additional, unnecessary subproblems are solved
  - increases running time of algorithm

$\Rightarrow$ avoid redundant boxes

Identifying redundant boxes:

1. compare upper bounds $u(B)$ pairwise, remove redundant ones (Przybylski et al. (2010))
2. detect redundant boxes before their creation, i.e. only generate non-redundant boxes

Redundancy

Generic split produces redundant boxes

- Example: already in 2nd iteration, two of the six new boxes redundant
- if redundant boxes are kept in decomposition
  - additional, unnecessary subproblems are solved
  - increases running time of algorithm

$\Rightarrow$ avoid redundant boxes

Identifying redundant boxes:

1. compare upper bounds $u(B)$ pairwise, remove redundant ones (Przybylski et al. (2010))
2. detect redundant boxes before their creation, i.e. only generate non-redundant boxes

Redundancy

Generic split produces redundant boxes

- Example: already in 2nd iteration, two of the six new boxes redundant
- if redundant boxes are kept in decomposition
  - additional, unnecessary subproblems are solved
  - increases running time of algorithm

$\Rightarrow$ avoid redundant boxes

Identifying redundant boxes:

1. compare upper bounds $u(B)$ pairwise, remove redundant ones (Przybylski et al. (2010))
2. detect redundant boxes before their creation, i.e. only generate non-redundant boxes

Individual subsets

Observation:

B non-redundant $\iff$ B contains non-empty subset which is not part of any other box of the decomposition

Definition (Individual subsets)

For every $\bar{B} \in \mathcal{B}_s$, the set

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

is called individual subset of $\bar{B}$.

Individual subsets

Observation:

B non-redundant $\iff$ B contains non-empty subset which is not part of any other box of the decomposition

Definition (Individual subsets)

For every $\bar{B} \in \mathcal{B}_s$, the set

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

is called individual subset of $\bar{B}$.

Individual subsets

Observation:

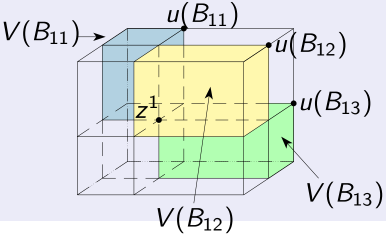
B non-redundant $\iff$ B contains non-empty subset which is not part of any other box of the decomposition

Definition (Individual subsets)

For every $\bar{B} \in \mathcal{B}_s$, the set

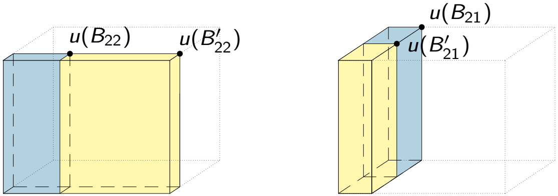
$$V(\bar{B}) := \bar{B} \setminus \left(\bigcup_{B \in \mathcal{B}_s \setminus \{\bar{B}\}} B \right)$$

is called individual subset of $\bar{B}$.



Individual subsets

- Maintaining non-redundant boxes $\iff$ maintaining boxes with non-empty individual subset



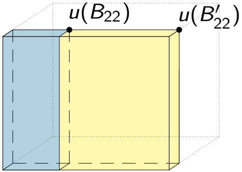
- Goal: Derive explicit representation of $V(B)$ ($\rightsquigarrow$ split criterion)

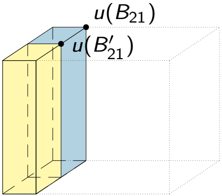
Preliminary technical assumption

For all $z, \bar{z} \in Z_N, z \neq \bar{z}$, let $z_i \neq \bar{z}_i$ for all $i = 1, \ldots, m$

Individual subsets

- Maintaining non-redundant boxes $\iff$ maintaining boxes with non-empty individual subset





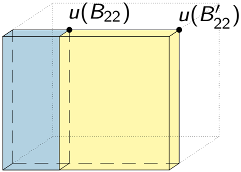
- Goal: Derive explicit representation of $V(B)$ ($\rightsquigarrow$ split criterion)

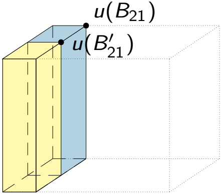
Preliminary technical assumption

For all $z, \bar{z} \in Z_N, z \neq \bar{z}$, let $z_i \neq \bar{z}_i$ for all $i = 1, \dots, m$

Individual subsets

- Maintaining non-redundant boxes $\iff$ maintaining boxes with non-empty individual subset





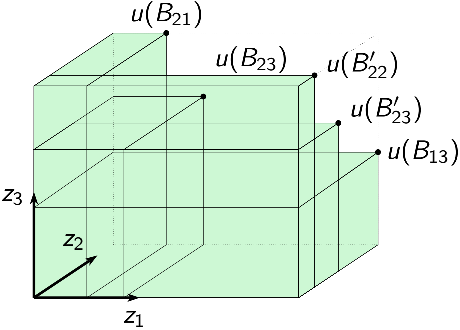
- Goal: Derive explicit representation of $V(B)$ ($\rightsquigarrow$ split criterion)

Preliminary technical assumption

For all $z, \bar{z} \in Z_N, z \neq \bar{z}$, let $z_i \neq \bar{z}_i$ for all $i = 1, \ldots, m$

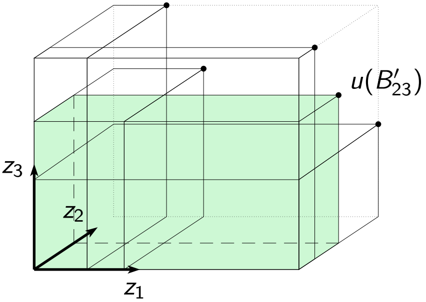
Representation of $V(B)$

- Individual subsets are determined by other boxes
- Idea: For each component exists exactly one box that limits $V(B)$



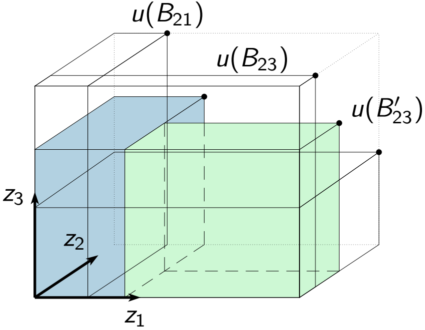
Representation of $V(B)$

- Individual subsets are determined by other boxes
- Idea: For each component exists exactly one box that limits $V(B)$



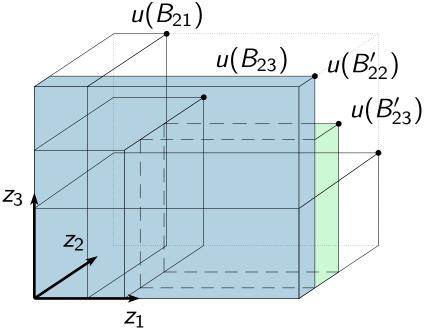
Representation of $V(B)$

- Individual subsets are determined by other boxes
- Idea: For each component exists exactly one box that limits $V(B)$



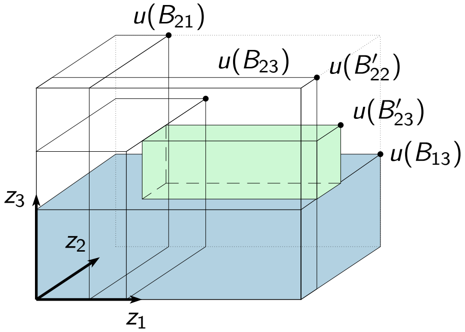
Representation of $V(B)$

- Individual subsets are determined by other boxes
- Idea: For each component exists exactly one box that limits $V(B)$



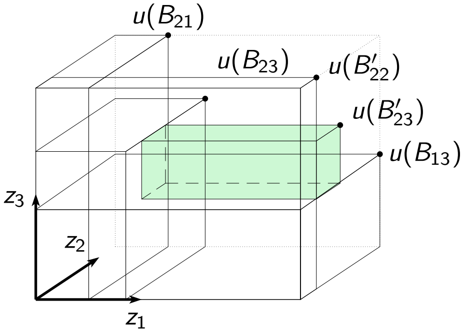
Representation of $V(B)$

- Individual subsets are determined by other boxes
- Idea: For each component exists exactly one box that limits $V(B)$



Representation of $V(B)$

- Individual subsets are determined by other boxes
- Idea: For each component exists exactly one box that limits $V(B)$



Lemma (Existence of unique neighbor for $m = 3$)

For every $\bar{B} \in \mathcal{B}_s$ and for every $i \in \{1, 2, 3\}$ with $u_i(\bar{B}) > \min_{B \in \mathcal{B}_s}\{u_i(B)\}$ there exists a unique $\hat{B} \in \mathcal{B}_s$ such that

$$\begin{aligned} u_i(\hat{B}) &< u_i(\bar{B}) \\ u_j(\hat{B}) &> u_j(\bar{B}) \quad \text{for some } j \neq i \\ u_k(\hat{B}) &= u_k(\bar{B}) \quad \text{for } k \neq i, j \end{aligned}$$

and $u_i(\hat{B})$ maximal with these properties.

Remark: This result can be generalized to higher dimensions.

Representation of $V(B)$ for $m = 3$

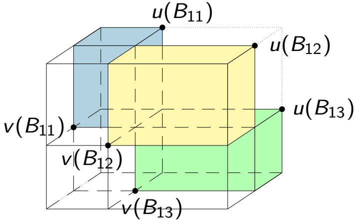
The individual subsets $V(B), B \in \mathcal{B}_s$, can be represented by

$$V(B) = \{z \in B : v(B) \leqq z\}$$

with

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

and $B_i^s(B)$ denoting the neighbor of B wrt. i in iteration s



For example

$$v(B_{12}) = \begin{pmatrix} u_1(B_{11}) \\ z_2^I \\ u_3(B_{13}) \end{pmatrix}$$

Representation of $V(B)$ for $m = 3$

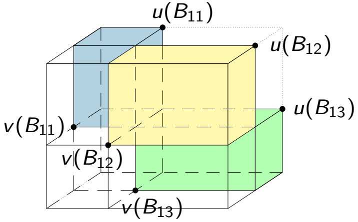
The individual subsets $V(B), B \in \mathcal{B}_s$, can be represented by

$$V(B) = \{z \in B : v(B) \leqq z\}$$

with

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

and $B_i^s(B)$ denoting the neighbor of B wrt. i in iteration s



For example

$$v(B_{12}) = \begin{pmatrix} u_1(B_{11}) \\ z_2^I \\ u_3(B_{13}) \end{pmatrix}$$

The v-split-criterion to avoid redundancy

Recall:

Split box B wrt component $i \iff V(B_i) \neq \emptyset$

Lemma

Let $z^s \in B$, i.e. $z^s < u(B)$, and let B_i be the box obtained from B by a split wrt component $i \in \{1, 2, 3\}$.

$$\text{Then } B_i \text{ is non-redundant} \iff z^s_i \geq v_i(B).$$

The v-split-criterion to avoid redundancy

Recall:

Split box B wrt component $i \iff V(B_i) \neq \emptyset$

Lemma

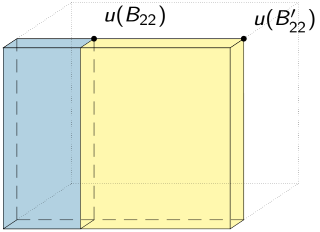
Let $z^s \in B$, i.e. $z^s < u(B)$, and let B_i be the box obtained from B by a split wrt component $i \in \{1, 2, 3\}$.

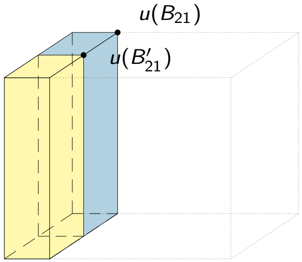
$$\text{Then } B_i \text{ is non-redundant} \iff z_i^s \geq v_i(B).$$

Example revisited

Generic split: Two redundant boxes

- B_{22} (Split of B_{11} wrt. $i = 2$)
- B'_{21} (Split of B_{12} wrt. $i = 1$)



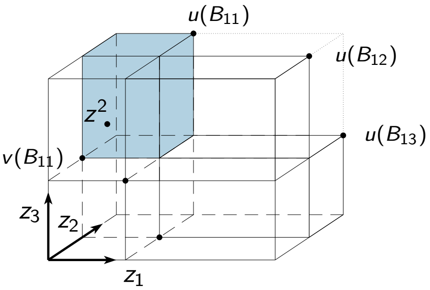


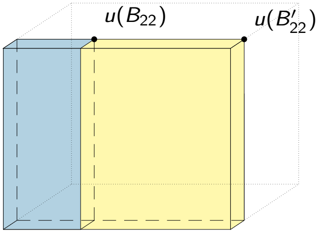
Example revisited

v-Split in B_{11}:

$$\begin{array}{ccll} z_1^2 & > & v_1(B_{11}) & \checkmark \\ z_2^2 & < & v_2(B_{11}) & \\ z_3^2 & > & v_3(B_{11}) & \checkmark \end{array}$$

$\Rightarrow$ Split B_{11} wrt. $i = 1$ and $i = 3$ (Redundant box B_{22} not generated!)



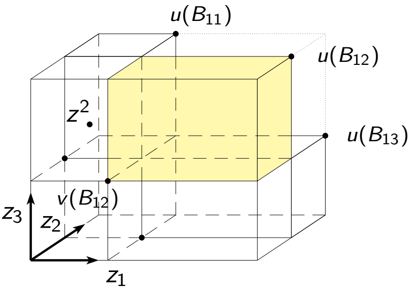


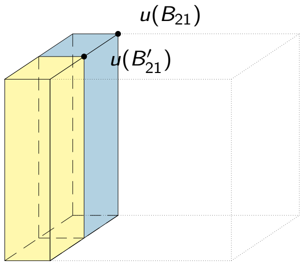
Example revisited

v-Split in B_{12}:

$$\begin{array}{cclc} z_1^2 & < & v_1(B_{12}) & \\ z_2^2 & > & v_2(B_{12}) & \checkmark \\ z_3^2 & > & v_3(B_{12}) & \checkmark \end{array}$$

$\Rightarrow$ Split B_{12} wrt. $i = 2$ and $i = 3$ (Redundant box B'_{21} not generated!)

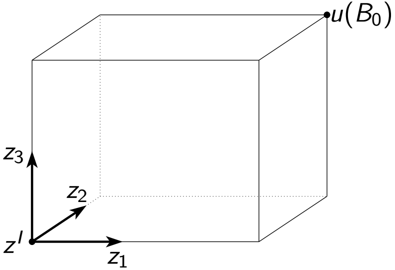




Use v-split to derive worst-case linear bound

Observation from example:

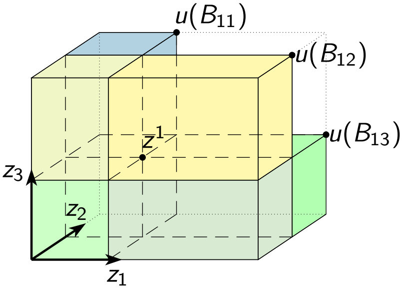
- Initialization: one box
- After 1st iteration: 3 boxes (+2)
- After 2nd iteration: 5 boxes (+2)
- …?



Use v-split to derive worst-case linear bound

Observation from example:

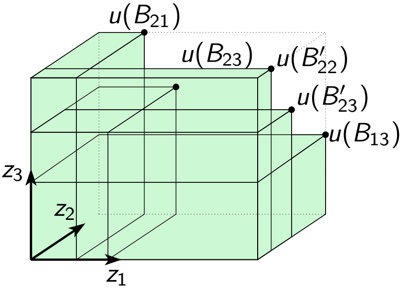
- Initialization: one box
- After 1st iteration: 3 boxes $(+2)$
- After 2nd iteration: 5 boxes $(+2)$
- $\ldots$?



Use v-split to derive worst-case linear bound

Observation from example:

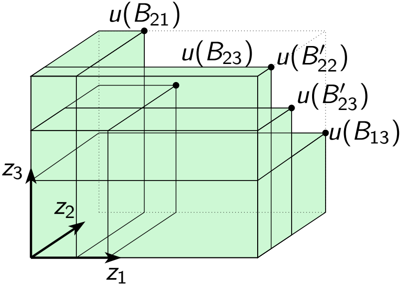
- Initialization: one box
- After 1st iteration: 3 boxes (+2)
- After 2nd iteration: 5 boxes (+2)
- ...?



Use v-split to derive worst-case linear bound

Observation from example:

- Initialization: one box
- After 1st iteration: 3 boxes (+2)
- After 2nd iteration: 5 boxes (+2)
- ...?



Linear bound on the number of subproblems

Lemma

In every iteration $s \geq 1$ of the v-split algorithm, in which a new nondominated point z^s is found, the number of boxes in the decomposition increases by at most two.

Sketch of proof.

Case 1: one box is split $\Rightarrow$ 3 boxes replace one $(-1+3=2\ \checkmark)$

Case 2: more than one box split:

- every box split wrt. at most 2 components
- no pair of boxes split wrt. to the same 2 components
- $\Rightarrow$ at most $2 \cdot 3 - 3 = 3$ additional boxes
- if 3 boxes split wrt. 2 components
- $\Rightarrow$ exists box which is split wrt. no component $(3-1=2\ \checkmark)$

$\square$

Linear bound on the number of subproblems

Lemma

In every iteration $s \geq 1$ of the v-split algorithm, in which a new nondominated point z^s is found, the number of boxes in the decomposition increases by at most two.

Sketch of proof.

- Case 1: one box is split $\Rightarrow$ 3 boxes replace one $(-1 + 3 = 2\ \checkmark)$
- Case 2: more than one box split:
  - every box split wrt. at most 2 components
  - no pair of boxes split wrt. to the same 2 components
  - $\Rightarrow$ at most $2 \cdot 3 - 3 = 3$ additional boxes
  - if 3 boxes split wrt. 2 components
  - $\Rightarrow$ exists box which is split wrt. no component $(3 - 1 = 2\ \checkmark)$

$\square$

Linear bound on the number of subproblems

Lemma

In every iteration $s \geq 1$ of the v-split algorithm, in which a new nondominated point z^s is found, the number of boxes in the decomposition increases by at most two.

Sketch of proof.

Case 1: one box is split $\Rightarrow$ 3 boxes replace one $(-1+3=2\ \checkmark)$

Case 2: more than one box split:

- every box split wrt. at most 2 components
- no pair of boxes split wrt. to the same 2 components
  $\Rightarrow$ at most $2 \cdot 3 - 3 = 3$ additional boxes
- if 3 boxes split wrt. 2 components
  $\Rightarrow$ exists box which is split wrt. no component $(3-1=2\ \checkmark)$

$\square$

Linear bound on the number of subproblems

Lemma

In every iteration $s \geq 1$ of the v-split algorithm, in which a new nondominated point z^s is found, the number of boxes in the decomposition increases by at most two.

Sketch of proof.

Case 1: one box is split $\Rightarrow$ 3 boxes replace one $(-1+3=2\ \checkmark)$

Case 2: more than one box split:

- every box split wrt. at most 2 components
- no pair of boxes split wrt. to the same 2 components
  $\Rightarrow$ at most $2 \cdot 3 - 3 = 3$ additional boxes
- if 3 boxes split wrt. 2 components
  $\Rightarrow$ exists box which is split wrt. no component $(3-1=2\ \checkmark)$

□

Linear bound on the number of subproblems

Lemma

In every iteration $s \geq 1$ of the v-split algorithm, in which a new nondominated point z^s is found, the number of boxes in the decomposition increases by at most two.

Sketch of proof.

Case 1: one box is split $\Rightarrow$ 3 boxes replace one $(-1+3=2\ \checkmark)$

Case 2: more than one box split:

- every box split wrt. at most 2 components
- no pair of boxes split wrt. to the same 2 components
  $\Rightarrow$ at most $2 \cdot 3 - 3 = 3$ additional boxes
- if 3 boxes split wrt. 2 components
  $\Rightarrow$ exists box which is split wrt. no component $(3-1=2\ \checkmark)$

$\square$

Linear bound on the number of subproblems

Lemma

In every iteration $s \geq 1$ of the v-split algorithm, in which a new nondominated point z^s is found, the number of boxes in the decomposition increases by at most two.

Sketch of proof.

Case 1: one box is split $\Rightarrow$ 3 boxes replace one $(-1+3=2\ \checkmark)$

Case 2: more than one box split:

- every box split wrt. at most 2 components
- no pair of boxes split wrt. to the same 2 components
  $\Rightarrow$ at most $2 \cdot 3 - 3 = 3$ additional boxes
- if 3 boxes split wrt. 2 components
  $\Rightarrow$ exists box which is split wrt. no component $(3-1=2\ \checkmark)$

□

Linear bound on the number of subproblems

Lemma

In every iteration $s \geq 1$ of the v-split algorithm, in which a new nondominated point z^s is found, the number of boxes in the decomposition increases by at most two.

Sketch of proof.

Case 1: one box is split $\Rightarrow$ 3 boxes replace one $(-1+3=2\ \checkmark)$

Case 2: more than one box split:

- every box split wrt. at most 2 components
- no pair of boxes split wrt. to the same 2 components
  $\Rightarrow$ at most $2 \cdot 3 - 3 = 3$ additional boxes
- if 3 boxes split wrt. 2 components
  $\Rightarrow$ exists box which is split wrt. no component $(3-1=2\ \checkmark)$

$\square$

Theorem

For Z_N finite ($N = |Z_N|$) and given appropriate initial search region with $lb = z^I$, the v-split algorithm requires at most $3N - 2$ subproblems in order to generate the entire nondominated set.

Sketch of proof.

- in every iteration one subproblem solved
  $\Rightarrow$ number of subproblems equals number of iterations
- for every nondominated point generated
  $\Rightarrow$ number of boxes increases by at most two (previous Lemma)
- every nondominated point is generated exactly once,
  every empty box is investigated exactly once
  $\Rightarrow$ at most $3N$ boxes explored
- plus initial box $\Rightarrow 3N + 1$
- if $z_i^s = z_i^I$ for $i \in \{1, 2, 3\}$, no box created wrt. $i \Rightarrow 3N - 2$

$\square$

Theorem

For Z_N finite ($N = |Z_N|$) and given appropriate initial search region with $lb = z^I$, the v-split algorithm requires at most $3N - 2$ subproblems in order to generate the entire nondominated set.

Sketch of proof.

- in every iteration one subproblem solved
  $\Rightarrow$ number of subproblems equals number of iterations
- for every nondominated point generated
  $\Rightarrow$ number of boxes increases by at most two (previous Lemma)
- every nondominated point is generated exactly once,
  every empty box is investigated exactly once
  $\Rightarrow$ at most $3N$ boxes explored
- plus initial box $\Rightarrow 3N + 1$
- if $z_i^s = z_i^I$ for $i \in \{1, 2, 3\}$, no box created wrt. $i \Rightarrow 3N - 2$

$\square$

Theorem

For Z_N finite ($N = |Z_N|$) and given appropriate initial search region with $lb = z^I$, the v-split algorithm requires at most $3N - 2$ subproblems in order to generate the entire nondominated set.

Sketch of proof.

- in every iteration one subproblem solved
  $\Rightarrow$ number of subproblems equals number of iterations
- for every nondominated point generated
  $\Rightarrow$ number of boxes increases by at most two (previous Lemma)
- every nondominated point is generated exactly once,
  every empty box is investigated exactly once
  $\Rightarrow$ at most $3N$ boxes explored
- plus initial box $\Rightarrow 3N + 1$
- if $z_i^s = z_i^I$ for $i \in \{1, 2, 3\}$, no box created wrt. $i \Rightarrow 3N - 2$

$\square$

Theorem

For Z_N finite ($N = |Z_N|$) and given appropriate initial search region with $lb = z^I$, the v-split algorithm requires at most $3N - 2$ subproblems in order to generate the entire nondominated set.

Sketch of proof.

- in every iteration one subproblem solved
  $\Rightarrow$ number of subproblems equals number of iterations
- for every nondominated point generated
  $\Rightarrow$ number of boxes increases by at most two (previous Lemma)
- every nondominated point is generated exactly once,
  every empty box is investigated exactly once
  $\Rightarrow$ at most $3N$ boxes explored
- plus initial box $\Rightarrow 3N + 1$
- if $z_i^s = z_i^I$ for $i \in \{1, 2, 3\}$, no box created wrt. $i \Rightarrow 3N - 2$

$\square$

Theorem

For Z_N finite ($N = |Z_N|$) and given appropriate initial search region with $lb = z^I$, the v-split algorithm requires at most $3N - 2$ subproblems in order to generate the entire nondominated set.

Sketch of proof.

- in every iteration one subproblem solved
  $\Rightarrow$ number of subproblems equals number of iterations
- for every nondominated point generated
  $\Rightarrow$ number of boxes increases by at most two (previous Lemma)
- every nondominated point is generated exactly once,
  every empty box is investigated exactly once
  $\Rightarrow$ at most $3N$ boxes explored
- plus initial box $\Rightarrow 3N + 1$
- if $z_i^s = z_i^I$ for $i \in \{1, 2, 3\}$, no box created wrt. $i \Rightarrow 3N - 2$

$\square$

Theorem

For Z_N finite ($N = |Z_N|$) and given appropriate initial search region with $lb = z^I$, the v-split algorithm requires at most $3N - 2$ subproblems in order to generate the entire nondominated set.

Sketch of proof.

- in every iteration one subproblem solved
  $\Rightarrow$ number of subproblems equals number of iterations
- for every nondominated point generated
  $\Rightarrow$ number of boxes increases by at most two (previous Lemma)
- every nondominated point is generated exactly once,
  every empty box is investigated exactly once
  $\Rightarrow$ at most $3N$ boxes explored
- plus initial box $\Rightarrow 3N + 1$
- if $z_i^s = z_i^I$ for $i \in \{1, 2, 3\}$, no box created wrt. $i \Rightarrow 3N - 2$

$\square$

Theorem

For Z_N finite ($N = |Z_N|$) and given appropriate initial search region with $lb = z^I$, the v-split algorithm requires at most $3N - 2$ subproblems in order to generate the entire nondominated set.

Sketch of proof.

- in every iteration one subproblem solved
  $\Rightarrow$ number of subproblems equals number of iterations
- for every nondominated point generated
  $\Rightarrow$ number of boxes increases by at most two (previous Lemma)
- every nondominated point is generated exactly once,
  every empty box is investigated exactly once
  $\Rightarrow$ at most $3N$ boxes explored
- plus initial box $\Rightarrow 3N + 1$
- if $z_i^s = z_i^I$ for $i \in \{1, 2, 3\}$, no box created wrt. $i \Rightarrow 3N - 2$

$\square$

Remark

Linear bound in line with results from the field of computational geometry:

1. Boissonnat et al. (1998) show that
   - for a set of n points in $\mathbb{R}^m$ the maximum complexity of the union of n axis-parallel hypercubes in $\mathbb{R}^m$ is $\mathcal{O}(n^{\lceil m/2 \rceil})$
   - If all hypercubes have the same size, the complexity can be improved to $\mathcal{O}(n^{\lfloor m/2 \rfloor})$ for $m \geq 2$. It remains $\mathcal{O}(n)$ for $m = 1$.
2. Bringmann (2013) shows that
   - an instance in which all boxes share one common vertex (z^I!) can be transformed into an instance in which all boxes have the same size

However, no algorithm is indicated in Boissonnat et al. (1998) or Bringmann (2013)

Remark

Linear bound in line with results from the field of computational geometry:

1. Boissonnat et al. (1998) show that
   - for a set of n points in $\mathbb{R}^m$ the maximum complexity of the union of n axis-parallel hypercubes in $\mathbb{R}^m$ is $\mathcal{O}(n^{\lceil m/2 \rceil})$
   - If all hypercubes have the same size, the complexity can be improved to $\mathcal{O}(n^{\lfloor m/2 \rfloor})$ for $m \geq 2$. It remains $\mathcal{O}(n)$ for $m = 1$.
2. Bringmann (2013) shows that
   - an instance in which all boxes share one common vertex (z^I!) can be transformed into an instance in which all boxes have the same size

However, no algorithm is indicated in Boissonnat et al. (1998) or Bringmann (2013)

Remark

Linear bound in line with results from the field of computational geometry:

1. Boissonnat et al. (1998) show that
   - for a set of n points in $\mathbb{R}^m$ the maximum complexity of the union of n axis-parallel hypercubes in $\mathbb{R}^m$ is $\mathcal{O}(n^{\lceil m/2 \rceil})$
   - If all hypercubes have the same size, the complexity can be improved to $\mathcal{O}(n^{\lfloor m/2 \rfloor})$ for $m \geq 2$. It remains $\mathcal{O}(n)$ for $m = 1$.
2. Bringmann (2013) shows that
   - an instance in which all boxes share one common vertex (z^I!) can be transformed into an instance in which all boxes have the same size

However, no algorithm is indicated in Boissonnat et al. (1998) or Bringmann (2013)

Remark

Linear bound in line with results from the field of computational geometry:

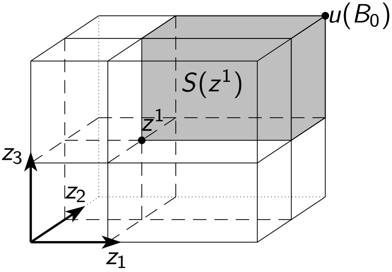
1. Boissonnat et al. (1998) show that
   - for a set of n points in $\mathbb{R}^m$ the maximum complexity of the union of n axis-parallel hypercubes in $\mathbb{R}^m$ is $\mathcal{O}(n^{\lceil m/2 \rceil})$
   - If all hypercubes have the same size, the complexity can be improved to $\mathcal{O}(n^{\lfloor m/2 \rfloor})$ for $m \geq 2$. It remains $\mathcal{O}(n)$ for $m = 1$.
2. Bringmann (2013) shows that
   - an instance in which all boxes share one common vertex (z^I!) can be transformed into an instance in which all boxes have the same size

However, no algorithm is indicated in Boissonnat et al. (1998) or Bringmann (2013)

The ε-constraint scalarization as a special case

Number of subproblems in v-split algorithm can be improved further:

- Assume we minimize wrt first component
- Having obtained z^1, we can additionally exclude $\{z \in B : z_1 < z^1_1\}$, which equals the set obtained by a split of B wrt the first component

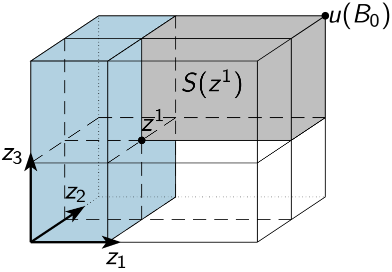


- one box per iteration can be saved if B is selected in an appropriate way
- only $2N - 1$ subproblems are required

The ε-constraint scalarization as a special case

Number of subproblems in v-split algorithm can be improved further:

- Assume we minimize wrt first component
- Having obtained z^1, we can additionally exclude $\{z \in B : z_1 < z_1^1\}$, which equals the set obtained by a split of B wrt the first component

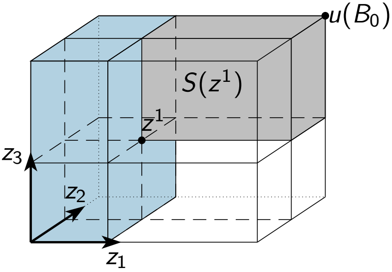


- one box per iteration can be saved if B is selected in an appropriate way
- only $2N - 1$ subproblems are required

The ε-constraint scalarization as a special case

Number of subproblems in v-split algorithm can be improved further:

- Assume we minimize wrt first component
- Having obtained z^1, we can additionally exclude $\{z \in B : z_1 < z_1^1\}$, which equals the set obtained by a split of B wrt the first component

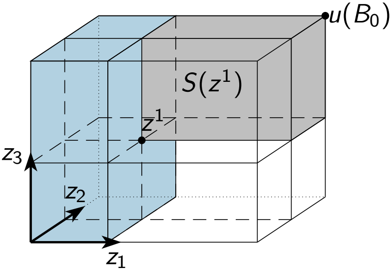


- one box per iteration can be saved if B is selected in an appropriate way
- only $2N - 1$ subproblems are required

The ε-constraint scalarization as a special case

Number of subproblems in v-split algorithm can be improved further:

- Assume we minimize wrt first component
- Having obtained z^1, we can additionally exclude $\{z \in B : z_1 < z_1^1\}$, which equals the set obtained by a split of B wrt the first component



- one box per iteration can be saved if B is selected in an appropriate way
- only $2N - 1$ subproblems are required

Numerical results

Matlab-Implementation of the *v*-split-algorithm with 3D-visualization

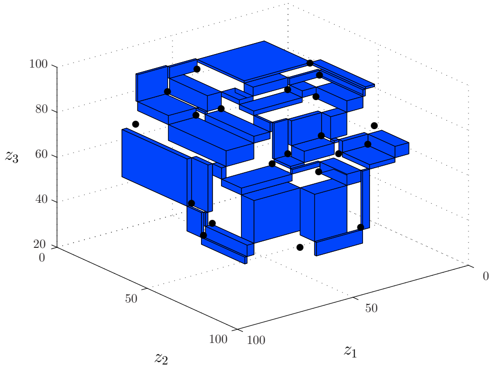


Figure: Example with 21 nondominated points
(Illustration of the individual subsets of all boxes at the end of the algorithm)

Setting

- Multidimensional, tricriteria knapsack problem
- Test problem from Laumanns et al. (2006)
- Original data
- Scalarizations: Weighted Tchebycheff (WT) and ε-Constraint (EC), both in variants Two-stage (TS) and Augmented (A)
- IBM ILOG CPLEX Optimization Studio Version 12.5 (no parallelization)
- MATLAB R2013a
- 4x Intel Xeon E7540 CPUs (2.0 GHz), 128 GB memory

Computational experiments (1)

Validation of theoretical upper bounds $3N - 2$ (WT) and $2N - 1$ (EC)

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

Conclusion

1. New adaptive parametric algorithm for multicriteria, particularly tricriteria optimization problems
2. New split criterion for tricriteria problems avoids redundant boxes
3. Linear worst-case bound on number of subproblems

Ongoing research:

1. Explicit use of neighborhood structure for any number of criteria
2. Generation of representative subsets with quality criteria

Thank you! Questions?

Conclusion

1. New adaptive parametric algorithm for multicriteria, particularly tricriteria optimization problems
2. New split criterion for tricriteria problems avoids redundant boxes
3. Linear worst-case bound on number of subproblems

Ongoing research:

1. Explicit use of neighborhood structure for any number of criteria
2. Generation of representative subsets with quality criteria

Thank you! Questions?

Conclusion

1. New adaptive parametric algorithm for multicriteria, particularly tricriteria optimization problems
2. New split criterion for tricriteria problems avoids redundant boxes
3. Linear worst-case bound on number of subproblems

Ongoing research:

1. Explicit use of neighborhood structure for any number of criteria
2. Generation of representative subsets with quality criteria

Thank you! Questions?

Conclusion

1. New adaptive parametric algorithm for multicriteria, particularly tricriteria optimization problems
2. New split criterion for tricriteria problems avoids redundant boxes
3. Linear worst-case bound on number of subproblems

Ongoing research:

1. Explicit use of neighborhood structure for any number of criteria
2. Generation of representative subsets with quality criteria

Thank you! Questions?

Conclusion

1. New adaptive parametric algorithm for multicriteria, particularly tricriteria optimization problems
2. New split criterion for tricriteria problems avoids redundant boxes
3. Linear worst-case bound on number of subproblems

Ongoing research:

1. Explicit use of neighborhood structure for any number of criteria
2. Generation of representative subsets with quality criteria

Thank you! Questions?

Conclusion

1. New adaptive parametric algorithm for multicriteria, particularly tricriteria optimization problems
2. New split criterion for tricriteria problems avoids redundant boxes
3. Linear worst-case bound on number of subproblems

Ongoing research:

1. Explicit use of neighborhood structure for any number of criteria
2. Generation of representative subsets with quality criteria

Thank you! Questions?

Conclusion

1. New adaptive parametric algorithm for multicriteria, particularly tricriteria optimization problems
2. New split criterion for tricriteria problems avoids redundant boxes
3. Linear worst-case bound on number of subproblems

Ongoing research:

1. Explicit use of neighborhood structure for any number of criteria
2. Generation of representative subsets with quality criteria

Thank you! Questions?

Conclusion

1. New adaptive parametric algorithm for multicriteria, particularly tricriteria optimization problems
2. New split criterion for tricriteria problems avoids redundant boxes
3. Linear worst-case bound on number of subproblems

Ongoing research:

1. Explicit use of neighborhood structure for any number of criteria
2. Generation of representative subsets with quality criteria

Thank you! Questions?