International Doctoral School Algorithmic Decision Theory: MCDA and MOO

Lecture 4: Multiobjective Combinatorial Optimization

Matthias Ehrgott

Department of Engineering Science, The University of Auckland, New Zealand
Laboratoire d'Informatique de Nantes Atlantique, CNRS, Université de Nantes, France

MCDA and MOO, Han sur Lesse, September 17 – 21 2007

Overview

Overview

Overview

Overview

Overview

Overview

Overview

Overview

Mathematical Formulation

$$\begin{aligned} \min z(x) &= Cx \\ \text{subject to } Ax &= b \\ x &\in \{0,1\}^n \end{aligned}$$

$$\begin{aligned} x \in \{0,1\}^n &\longrightarrow n \text{ variables, } i = 1,\ldots,n \\ C \in \mathbb{Z}^{p\times n} &\longrightarrow p \text{ objective functions, } k = 1,\ldots,p \\ A \in \mathbb{Z}^{m\times n} &\longrightarrow m \text{ constraints, } j = 1,\ldots,m \end{aligned}$$

Combinatorial structure: paths, trees, flows, tours, etc.

Mathematical Formulation

$$\begin{aligned} \min z(x) &= Cx \\ \text{subject to } Ax &= b \\ x &\in \{0,1\}^n \end{aligned}$$

$x \in \{0,1\}^n \longrightarrow n$ variables, $i = 1, \ldots, n$

$C \in \mathbb{Z}^{p \times n} \longrightarrow p$ objective functions, $k = 1, \ldots, p$

$A \in \mathbb{Z}^{m \times n} \longrightarrow m$ constraints, $j = 1, \ldots, m$

Combinatorial structure: paths, trees, flows, tours, etc.

Mathematical Formulation

$$\begin{aligned} \min z(x) &= Cx \\ \text{subject to } Ax &= b \\ x &\in \{0,1\}^n \end{aligned}$$

$$\begin{aligned} x \in \{0,1\}^n &\longrightarrow n \text{ variables, } i = 1, \ldots, n \\ C \in \mathbb{Z}^{p \times n} &\longrightarrow p \text{ objective functions, } k = 1, \ldots, p \\ A \in \mathbb{Z}^{m \times n} &\longrightarrow m \text{ constraints, } j = 1, \ldots, m \end{aligned}$$

Combinatorial structure: paths, trees, flows, tours, etc.

Mathematical Formulation

$$\begin{aligned} \min z(x) &= Cx \\ \text{subject to } Ax &= b \\ x &\in \{0,1\}^n \end{aligned}$$

$$\begin{aligned} x \in \{0,1\}^n &\longrightarrow n \text{ variables}, i = 1, \ldots, n \\ C \in \mathbb{Z}^{p\times n} &\longrightarrow p \text{ objective functions}, k = 1, \ldots, p \\ A \in \mathbb{Z}^{m\times n} &\longrightarrow m \text{ constraints}, j = 1, \ldots, m \end{aligned}$$

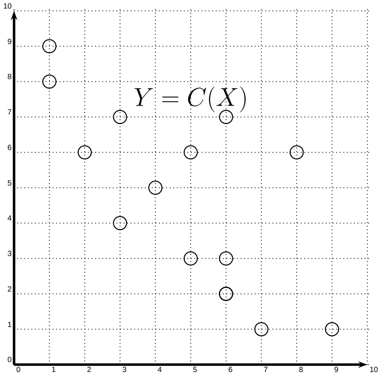
Combinatorial structure: paths, trees, flows, tours, etc.

Feasible Sets

- $X = \{x \in \{0,1\}^n : Ax = b\}$
  feasible set in decision space
- $Y = z(X) = \{Cx : x \in X\}$
  feasible set in objective space
- $\text{conv}(Y) + \mathbb{R}^p_{\geqq}$

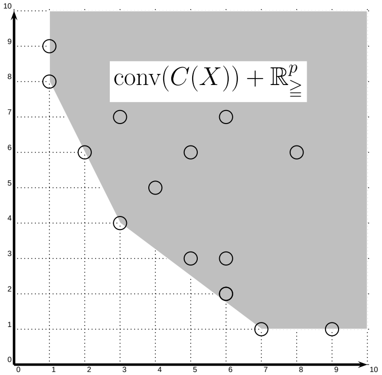
Feasible Sets

- $X = \{x \in \{0,1\}^n : Ax = b\}$ feasible set in decision space
- $Y = z(X) = \{Cx : x \in X\}$ feasible set in objective space
- $\text{conv}(Y) + \mathbb{R}^p_{\geqq}$



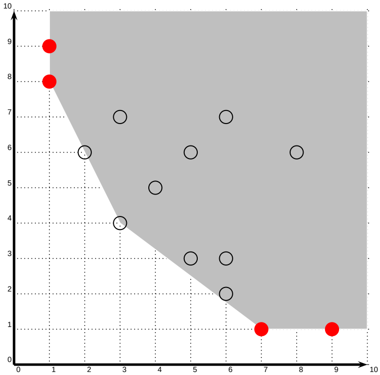
Feasible Sets

- $X = \{x \in \{0,1\}^n : Ax = b\}$ feasible set in decision space
- $Y = z(X) = \{Cx : x \in X\}$ feasible set in objective space
- $\text{conv}(Y) + \mathbb{R}^p_{\geqq}$



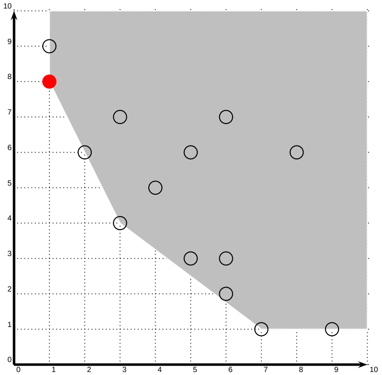
Lexicographic Optimality

- Individual minima
  $z_k(\hat{x}) \leqq z_k(x)$ for all $x \in X$
- Lexicographic optimality (1)
  $z(\hat{x}) \leqq_{lex} z(x)$ for all $x \in X$
- Lexicographic optimality (2)
  $z^\pi(\hat{x}) \leqq_{lex} z^\pi(x)$ for all $x \in X$ and some permutation z^π of $(z_1, \ldots, z_p)$



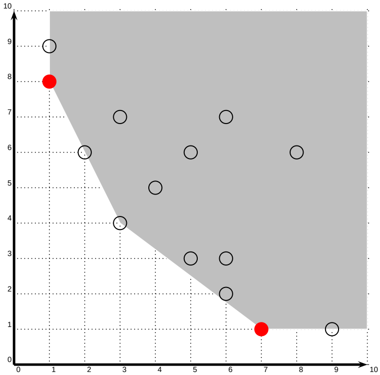
Lexicographic Optimality

- Individual minima
  $z_k(\hat{x}) \leqq z_k(x)$ for all $x \in X$
- Lexicographic optimality (1)
  $z(\hat{x}) \leqq_{lex} z(x)$ for all $x \in X$
- Lexicographic optimality (2)
  $z^{\pi}(\hat{x}) \leqq_{lex} z^{\pi}(x)$ for all $x \in X$ and some permutation z^{π} of $(z_1, \ldots, z_p)$



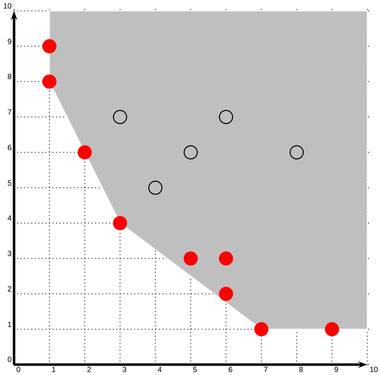
Lexicographic Optimality

- Individual minima
  $z_k(\hat{x}) \leqq z_k(x)$ for all $x \in X$
- Lexicographic optimality (1)
  $z(\hat{x}) \leqq_{lex} z(x)$ for all $x \in X$
- Lexicographic optimality (2)
  $z^\pi(\hat{x}) \leqq_{lex} z^\pi(x)$ for all $x \in X$ and some permutation z^π of $(z_1, \ldots, z_p)$



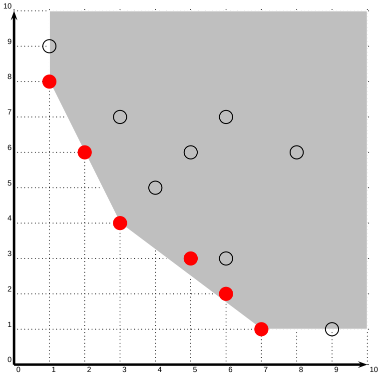
Efficient Solutions

- Weakly efficient solutions X_{wE}:
  there is no x with $z(x) < z(\hat{x})$
  $z(\hat{x})$ is weakly nondominated
  $Y_{wN} := z(X_{wN})$
- Efficient solutions X_E:
  there is no x with $z(x) \leq z(\hat{x})$
  $z(\hat{x})$ is nondominated
  $Y_N := z(X_E)$



Efficient Solutions

- Weakly efficient solutions X_{wE}:
  there is no x with $z(x) < z(\hat{x})$
  $z(\hat{x})$ is weakly nondominated
  $Y_{wN} := z(X_{wN})$
- Efficient solutions X_E:
  there is no x with $z(x) \leq z(\hat{x})$
  $z(\hat{x})$ is nondominated
  $Y_N := z(X_E)$

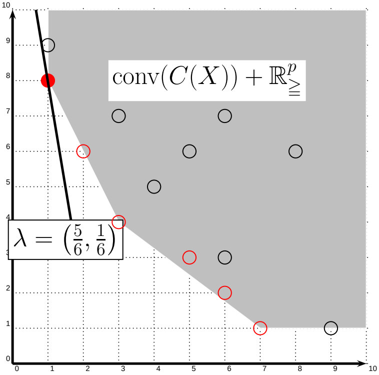


Efficient Solutions

- Supported efficient solutions X_{SE}: There is $\lambda > 0$ with $\lambda^T C\hat{x} \leqq \lambda^T Cx$ for all $x \in X$
  - $C\hat{x}$ is extreme point of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE1}$
  - $C\hat{x}$ is in relative interior of face of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE2}$
- Nonsupported efficient solutions X_{NE}: $C\hat{x}$ is in interior of $\text{conv}(Y) + \mathbb{R}^p_{\geqq}$

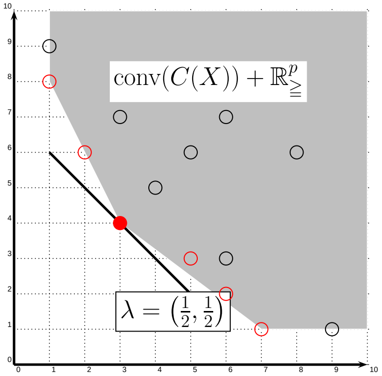
Efficient Solutions

- Supported efficient solutions X_{SE}: There is $\lambda > 0$ with $\lambda^T C\hat{x} \leqq \lambda^T Cx$ for all $x \in X$
  - $C\hat{x}$ is extreme point of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE1}$
  - $C\hat{x}$ is in relative interior of face of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE2}$
- Nonsupported efficient solutions X_{NE}: $C\hat{x}$ is in interior of $\text{conv}(Y) + \mathbb{R}^p_{\geqq}$



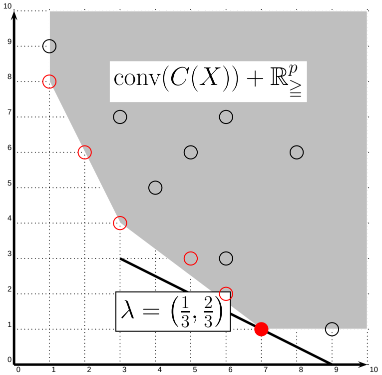
Efficient Solutions

- Supported efficient solutions X_{SE}: There is $\lambda > 0$ with $\lambda^T C\hat{x} \leqq \lambda^T Cx$ for all $x \in X$
  - $C\hat{x}$ is extreme point of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE1}$
  - $C\hat{x}$ is in relative interior of face of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE2}$
- Nonsupported efficient solutions X_{NE}: $C\hat{x}$ is in interior of $\text{conv}(Y) + \mathbb{R}^p_{\geqq}$



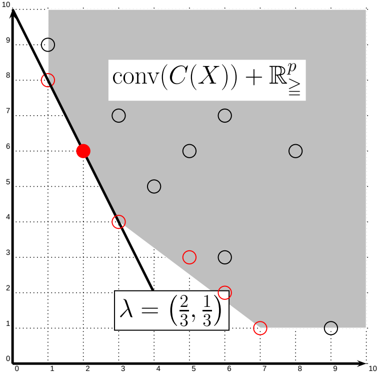
Efficient Solutions

- Supported efficient solutions
  X_{SE}: There is $\lambda > 0$ with $\lambda^T C\hat{x} \leqq \lambda^T Cx$ for all $x \in X$
  - $C\hat{x}$ is extreme point of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE1}$
  - $C\hat{x}$ is in relative interior of face of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE2}$
- Nonsupported efficient solutions
  X_{NE}: $C\hat{x}$ is in interior of $\text{conv}(Y) + \mathbb{R}^p_{\geqq}$



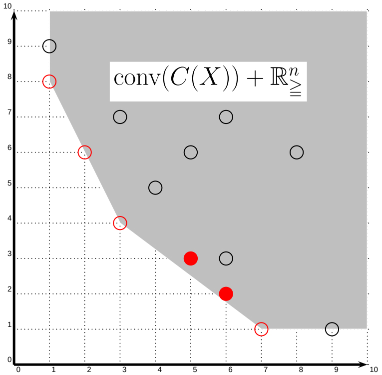
Efficient Solutions

- Supported efficient solutions
  X_{SE}: There is $\lambda > 0$ with $\lambda^T C\hat{x} \leqq \lambda^T Cx$ for all $x \in X$
  - $C\hat{x}$ is extreme point of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE1}$
  - $C\hat{x}$ is in relative interior of face of $\text{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE2}$
- Nonsupported efficient solutions
  X_{NE}: $C\hat{x}$ is in interior of $\text{conv}(Y) + \mathbb{R}^p_{\geqq}$

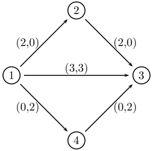


Efficient Solutions

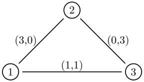
- Supported efficient solutions X_{SE}: There is $\lambda > 0$ with $\lambda^T C\hat{x} \leqq \lambda^T Cx$ for all $x \in X$
  - $C\hat{x}$ is extreme point of $\mathrm{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE1}$
  - $C\hat{x}$ is in relative interior of face of $\mathrm{conv}(Y) + \mathbb{R}^p_{\geqq} \rightarrow X_{SE2}$
- Nonsupported efficient solutions X_{NE}: $C\hat{x}$ is in interior of $\mathrm{conv}(Y) + \mathbb{R}^p_{\geqq}$



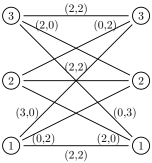
Nonsupported Effcient Solutions

- Shortest path: 

$Y_N = \left\{ \begin{pmatrix} 4 \\ 0 \end{pmatrix}, \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \begin{pmatrix} 0 \\ 4 \end{pmatrix} \right\}$

- Spanning tree: 

$Y_N = \left\{ \begin{pmatrix} 4 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \begin{pmatrix} 1 \\ 4 \end{pmatrix} \right\}$

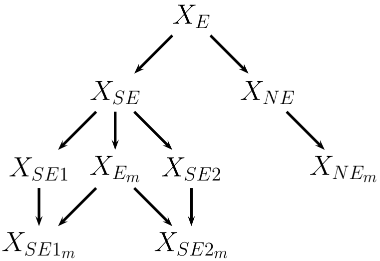
- Assignment: 

$Y_N = \left\{ \begin{pmatrix} 7 \\ 0 \end{pmatrix}, \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \begin{pmatrix} 0 \\ 7 \end{pmatrix} \right\}$

Classification of Efficient Sets

Hansen 1979:

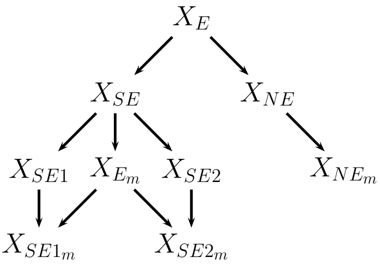
- $x^1, x^2 \in X_E$ are equivalent if $Cx^1 = Cx^2$
- Complete set: $\hat{X} \subset X_E$ such that for all $y \in Y_N$ there is $x \in \hat{X}$ with $z(x) = y$
- Minimal complete set contains no equivalent solutions
- Maximal complete set contains all equivalent solutions



Classification of Efficient Sets

Hansen 1979:

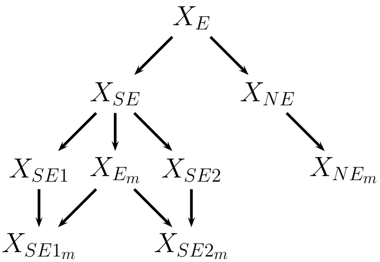
- $x^1, x^2 \in X_E$ are equivalent if $Cx^1 = Cx^2$
- Complete set: $\hat{X} \subset X_E$ such that for all $y \in Y_N$ there is $x \in \hat{X}$ with $z(x) = y$
- Minimal complete set contains no equivalent solutions
- Maximal complete set contains all equivalent solutions



Classification of Efficient Sets

Hansen 1979:

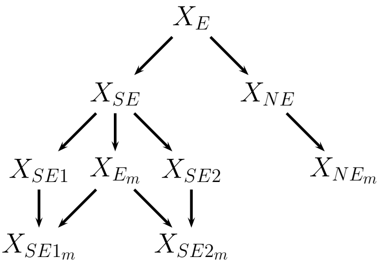
- $x^1, x^2 \in X_E$ are equivalent if $Cx^1 = Cx^2$
- Complete set: $\hat{X} \subset X_E$ such that for all $y \in Y_N$ there is $x \in \hat{X}$ with $z(x) = y$
- Minimal complete set contains no equivalent solutions
- Maximal complete set contains all equivalent solutions



Classification of Efficient Sets

Hansen 1979:

- $x^1, x^2 \in X_E$ are equivalent if $Cx^1 = Cx^2$
- Complete set: $\hat{X} \subset X_E$ such that for all $y \in Y_N$ there is $x \in \hat{X}$ with $z(x) = y$
- Minimal complete set contains no equivalent solutions
- Maximal complete set contains all equivalent solutions



Overview

MOCO Problems Are Hard

- Decision problem: Given $b \in \mathbb{Z}^p$: Does there exist $x \in X$ such that $Cx \leqq b$?
- Counting problem: Given $b \in \mathbb{Z}^p$: How many $x \in X$ satisfy $Cx \leqq b$?
- How many efficient solutions (nondominated points) do exist?
- KNAPSACK: Given $a^1, a^2 \in \mathbb{Z}^n$ and $b_1, b_2 \in \mathbb{Z}$, does there exist $x \in \{0,1\}^n$ such that $(a^1)^T x \leqq b_1$ and $(a^2)^T x \geqq b_2$?
- KNAPSACK is NP-complete and #P-complete

MOCO Problems Are Hard

- Decision problem: Given $b \in \mathbb{Z}^p$: Does there exist $x \in X$ such that $Cx \leqq b$?
- Counting problem: Given $b \in \mathbb{Z}^p$: How many $x \in X$ satisfy $Cx \leqq b$?
- How many efficient solutions (nondominated points) do exist?
- KNAPSACK: Given $a^1, a^2 \in \mathbb{Z}^n$ and $b_1, b_2 \in \mathbb{Z}$, does there exist $x \in \{0,1\}^n$ such that $(a^1)^T x \leqq b_1$ and $(a^2)^T x \geqq b_2$?
- KNAPSACK is NP-complete and #P-complete

MOCO Problems Are Hard

- Decision problem: Given $b \in \mathbb{Z}^p$: Does there exist $x \in X$ such that $Cx \leqq b$?
- Counting problem: Given $b \in \mathbb{Z}^p$: How many $x \in X$ satisfy $Cx \leqq b$?
- How many efficient solutions (nondominated points) do exist?
- KNAPSACK: Given $a^1, a^2 \in \mathbb{Z}^n$ and $b_1, b_2 \in \mathbb{Z}$, does there exist $x \in \{0,1\}^n$ such that $(a^1)^T x \leqq b_1$ and $(a^2)^T x \geqq b_2$?
- KNAPSACK is NP-complete and #P-complete

MOCO Problems Are Hard

- Decision problem: Given $b \in \mathbb{Z}^p$: Does there exist $x \in X$ such that $Cx \leqq b$?
- Counting problem: Given $b \in \mathbb{Z}^p$: How many $x \in X$ satisfy $Cx \leqq b$?
- How many efficient solutions (nondominated points) do exist?
- KNAPSACK: Given $a^1, a^2 \in \mathbb{Z}^n$ and $b_1, b_2 \in \mathbb{Z}$, does there exist $x \in \{0,1\}^n$ such that $(a^1)^T x \leqq b_1$ and $(a^2)^T x \geqq b_2$?
- KNAPSACK is NP-complete and #P-complete

MOCO Problems Are Hard

- Decision problem: Given $b \in \mathbb{Z}^p$: Does there exist $x \in X$ such that $Cx \leqq b$?
- Counting problem: Given $b \in \mathbb{Z}^p$: How many $x \in X$ satisfy $Cx \leqq b$?
- How many efficient solutions (nondominated points) do exist?
- KNAPSACK: Given $a^1, a^2 \in \mathbb{Z}^n$ and $b_1, b_2 \in \mathbb{Z}$, does there exist $x \in \{0, 1\}^n$ such that $(a^1)^T x \leqq b_1$ and $(a^2)^T x \geqq b_2$?
- KNAPSACK is NP-complete and #P-complete

The Unconstrained MOCO Problem

Observation

Multiobjective combinatorial optimization problems are NP-hard, #P-complete, and intractable.

$$\min \sum_{i=1}^{n} c_i^k x_i \qquad k = 1, \ldots, p$$
$$\text{subject to } x_i \in \{0, 1\} \qquad i = 1, \ldots, n$$

The Unconstrained MOCO Problem

Observation

Multiobjective combinatorial optimization problems are NP-hard, #P-complete, and intractable.

$$\begin{aligned} \min \sum_{i=1}^{n} c_i^k x_i \qquad & k = 1, \ldots, p \\ \text{subject to } x_i \in \{0, 1\} \qquad & i = 1, \ldots, n \end{aligned}$$

The Unconstrained MOCO Problem

- Does there exist $x \in \{0,1\}^n$ such that $(c^1)^T x \leqq d_1$ and $(c^2)^T x \leqq d_2$?
- With an instance of KNAPSACK $c^1 := a^1$, $d_1 = b_1$, $c^2 := -a^2$, $d_2 := -b_2$ is a parsimonious transformation
- With $c_i^k := (-1)^k 2^{i-1}$ it holds $Y = Y_N$

The Unconstrained MOCO Problem

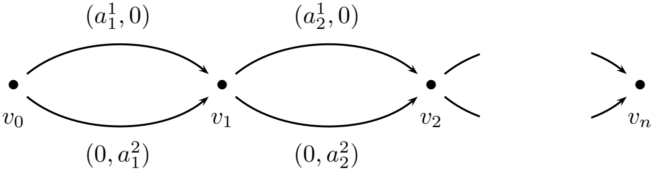
- Does there exist $x \in \{0,1\}^n$ such that $(c^1)^T x \leqq d_1$ and $(c^2)^T x \leqq d_2$?
- With an instance of KNAPSACK $c^1 := a^1$, $d_1 = b_1$, $c^2 := -a^2$, $d_2 := -b_2$ is a parsimonious transformation
- With $c_i^k := (-1)^k 2^{i-1}$ it holds $Y = Y_N$

The Unconstrained MOCO Problem

- Does there exist $x \in \{0,1\}^n$ such that $(c^1)^T x \leqq d_1$ and $(c^2)^T x \leqq d_2$?
- With an instance of KNAPSACK $c^1 := a^1$, $d_1 = b_1$, $c^2 := -a^2$, $d_2 := -b_2$ is a parsimonious transformation
- With $c_i^k := (-1)^k 2^{i-1}$ it holds $Y = Y_N$

Multiobjective Shortest Path Problem

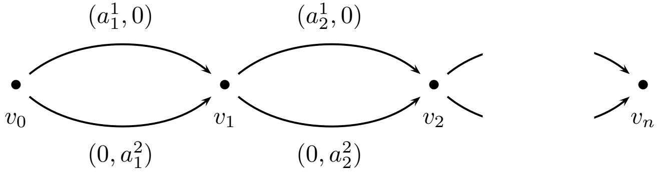
- NP-hard: Hansen 1979



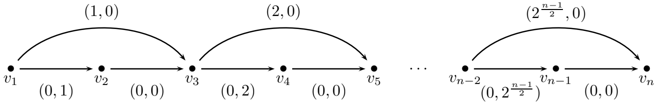
- Exponentially many efficient paths

Multiobjective Shortest Path Problem

- NP-hard: Hansen 1979



- Exponentially many efficient paths



Other Examples

- Assignment (NP-hard: Serafini 1986, #P-hard: Neumayer 1994)
- Spanning tree (Intractable: Hamacher and Ruhe 1994)
- Network flow (Intractable: Ruhe 1988)

Other Examples

- Assignment (NP-hard: Serafini 1986, #P-hard: Neumayer 1994)
- Spanning tree (Intractable: Hamacher and Ruhe 1994)
- Network flow (Intractable: Ruhe 1988)

Other Examples

- Assignment (NP-hard: Serafini 1986, #P-hard: Neumayer 1994)
- Spanning tree (Intractable: Hamacher and Ruhe 1994)
- Network flow (Intractable: Ruhe 1988)

Number of Efficient Solutions

- Intractable: X_E, even Y_{SN}, can be exponential in the size of the instance
- Number of nondominated points for biobjective shortest path and assignment problems (Raith 2007, Przybylski 2006)

Shortest path			Assignment	
Nodes	Edges	$\lvert Y_N \rvert$	n	$\lvert Y_N \rvert$
4,902	19,596	6	10	13
4,902	19,596	1,594	20	82
3,000	33,224	15	40	243
14,000	153,742	17	60	470
330,386	1,202,458	21	80	671
330,386	1,202,458	24	100	947

Number of Efficient Solutions

- Intractable: X_E, even Y_{SN}, can be exponential in the size of the instance
- Number of nondominated points for biobjective shortest path and assignment problems (Raith 2007, Przybylski 2006)

Shortest path			Assignment	
Nodes	Edges	$\vert Y_N \vert$	n	$\vert Y_N \vert$
4,902	19,596	6	10	13
4,902	19,596	1,594	20	82
3,000	33,224	15	40	243
14,000	153,742	17	60	470
330,386	1,202,458	21	80	671
330,386	1,202,458	24	100	947

Number of Efficient Solutions

Empirically often

- $|X_{NE}|$ grows exponentially with instance size
- $|X_{SE}|$ grows polynomially with instance size
- but this depends on numerical values of C

Number of Efficient Solutions

Empirically often

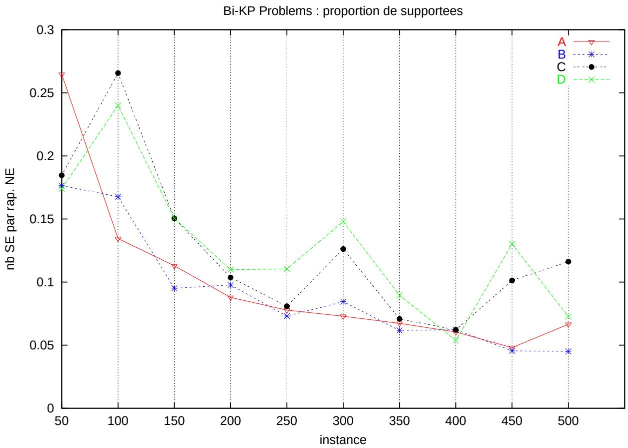
- $|X_{NE}|$ grows exponentially with instance size
- $|X_{SE}|$ grows polynomially with instance size
- but this depends on numerical values of C

Number of Efficient Solutions

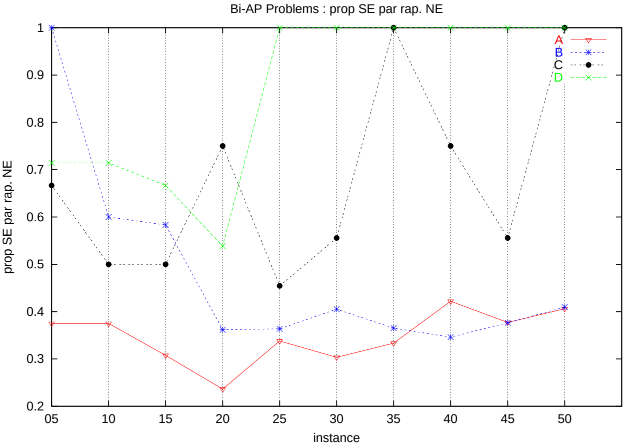
Empirically often

- $|X_{NE}|$ grows exponentially with instance size
- $|X_{SE}|$ grows polynomially with instance size
- but this depends on numerical values of C

Number of Efficient Solutions



Number of Efficient Solutions



Overview

The Multiobjective Shortest Path Problem

- Digraph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with arc costs $c_{ij}^k, k = 1, \ldots, p, (i,j) \in \mathcal{A}$
- Given origin $s \in \mathcal{V}$, destination $t \in \mathcal{V}$ find efficient paths from s to t:

$$\min_{P \in \mathcal{P}} \sum_{(i,j) \in P} c_{ij}$$

  where $\mathcal{P}$ is set of all s-t paths
- Assume that all $c_{ij}^k \geqq 0$

Proposition

Let P_{st} be an efficient path from s to t. Then any subpath P_{uv} from u to v, where u and v are vertices on P_{st} is an efficient path from u to v.

The Multiobjective Shortest Path Problem

- Digraph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with arc costs $c_{ij}^k, k = 1, \ldots, p, (i,j) \in \mathcal{A}$
- Given origin $s \in \mathcal{V}$, destination $t \in \mathcal{V}$ find efficient paths from s to t:

$$\min_{P \in \mathcal{P}} \sum_{(i,j) \in P} c_{ij}$$

where $\mathcal{P}$ is set of all s-t paths
- Assume that all $c_{ij}^k \geqq 0$

Proposition

Let P_{st} be an efficient path from s to t. Then any subpath P_{uv} from u to v, where u and v are vertices on P_{st} is an efficient path from u to v.

The Multiobjective Shortest Path Problem

- Digraph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with arc costs $c_{ij}^k, k = 1, \dots, p, (i,j) \in \mathcal{A}$
- Given origin $s \in \mathcal{V}$, destination $t \in \mathcal{V}$ find efficient paths from s to t:

$$\min_{P \in \mathcal{P}} \sum_{(i,j) \in P} c_{ij}$$

  where $\mathcal{P}$ is set of all s-t paths
- Assume that all $c_{ij}^k \geqq 0$

Proposition

Let P_{st} be an efficient path from s to t. Then any subpath P_{uv} from u to v, where u and v are vertices on P_{st} is an efficient path from u to v.

The Multiobjective Shortest Path Problem

- Digraph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with arc costs $c_{ij}^k, k = 1, \ldots, p, (i,j) \in \mathcal{A}$
- Given origin $s \in \mathcal{V}$, destination $t \in \mathcal{V}$ find efficient paths from s to t:

$$\min_{P \in \mathcal{P}} \sum_{(i,j) \in P} c_{ij}$$

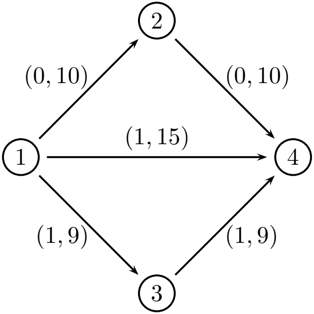
  where $\mathcal{P}$ is set of all s-t paths
- Assume that all $c_{ij}^k \geqq 0$

Proposition

Let P_{st} be an efficient path from s to t. Then any subpath P_{uv} from u to v, where u and v are vertices on P_{st} is an efficient path from u to v.

The Multiobjective Shortest Path Problem

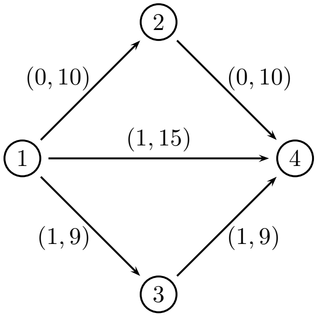
Concatenations of efficient paths need not be efficient!



1-3 is efficient, 3-4 is efficient, 1-3-4 is not

The Multiobjective Shortest Path Problem

Concatenations of efficient paths need not be efficient!



1-3 is efficient, 3-4 is efficient, 1-3-4 is not

The Multiobjective Shortest Path Problem

For a labelling algorithm we need

- Sets of nondominated labels at each node
- Store all costs, the predecessor, the label number at the predecessor, the number of the current label
- A list of permanent and temporary labels
- Make sure that a permanent label defines an efficient path: Choose the lexicographically smallest label from temporary list

Lemma

If P_1 and P_2 are two paths between nodes s and t and $c(P_1) \leq c(P_2)$ then $c(P_1) <_{lex} c(P_2)$.

The Multiobjective Shortest Path Problem

For a labelling algorithm we need

- Sets of nondominated labels at each node
- Store all costs, the predecessor, the label number at the predecessor, the number of the current label
- A list of permanent and temporary labels
- Make sure that a permanent label defines an efficient path: Choose the lexicographically smallest label from temporary list

Lemma

If P_1 and P_2 are two paths between nodes s and t and $c(P_1) \leq c(P_2)$ then $c(P_1) <_{lex} c(P_2)$.

The Multiobjective Shortest Path Problem

For a labelling algorithm we need

- Sets of nondominated labels at each node
- Store all costs, the predecessor, the label number at the predecessor, the number of the current label
- A list of permanent and temporary labels
- Make sure that a permanent label defines an efficient path: Choose the lexicographically smallest label from temporary list

Lemma

If P_1 and P_2 are two paths between nodes s and t and $c(P_1) \leq c(P_2)$ then $c(P_1) <_{lex} c(P_2)$.

The Multiobjective Shortest Path Problem

For a labelling algorithm we need

- Sets of nondominated labels at each node
- Store all costs, the predecessor, the label number at the predecessor, the number of the current label
- A list of permanent and temporary labels
- Make sure that a permanent label defines an efficient path: Choose the lexicographically smallest label from temporary list

Lemma

If P_1 and P_2 are two paths between nodes s and t and $c(P_1) \leq c(P_2)$ then $c(P_1) <_{lex} c(P_2)$.

The Multiobjective Shortest Path Problem

For a labelling algorithm we need

- Sets of nondominated labels at each node
- Store all costs, the predecessor, the label number at the predecessor, the number of the current label
- A list of permanent and temporary labels
- Make sure that a permanent label defines an efficient path: Choose the lexicographically smallest label from temporary list

Lemma

If P_1 and P_2 are two paths between nodes s and t and $c(P_1) \leq c(P_2)$ then $c(P_1) <_{lex} c(P_2)$.

Algorithm (Multiobjective label setting algorithm)

Input: A digraph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with p arc costs.

Initialization: Create label $L = (0, \ldots, 0, 0, 0, 1)$ at node s and let $\mathcal{TL} := \{L\}$.

While $\mathcal{TL} \neq \emptyset$ do

Let label $L = (c^1, \ldots, c^p, v_h, l, k)$ of node v_i be the lexicographically smallest label in $\mathcal{TL}$.

Remove L from $\mathcal{TL}$ and add it to $\mathcal{PL}$.

For all $v_j \in \mathcal{V}$ such that $(v_i, v_j) \in \mathcal{A}$ do

Create label $L' = (c^1 + c^1(v_i, v_j), \ldots, c^p + c^p(v_i, v_j), v_i, k, t)$ as the next label at node v_j and add it to $\mathcal{TL}$.

Delete all temporary labels of node v_j dominated by L', delete L' if it is dominated by another label of node v_j.

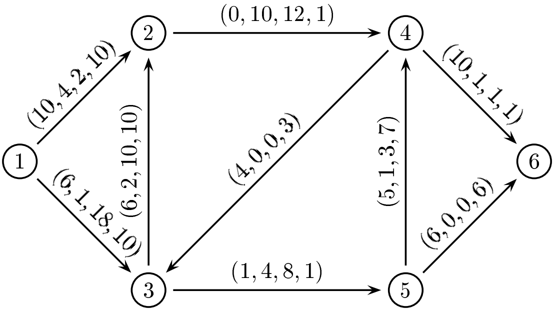
End for.

End while.

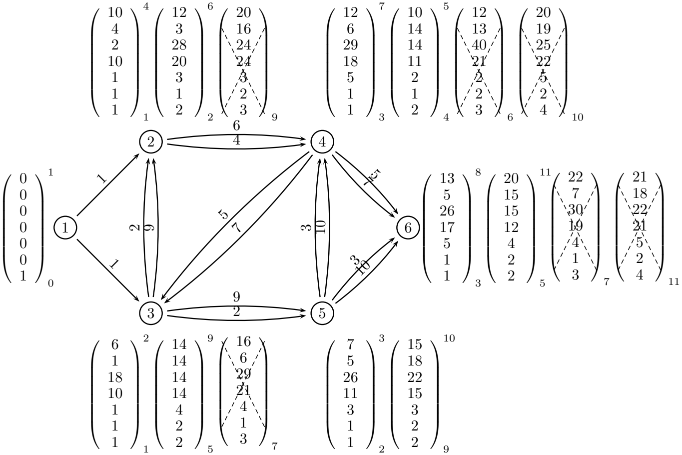
Use the predecessor labels in the permanent labels to recover all efficient paths from s to other nodes of $\mathcal{G}$.

Output: All efficient paths from node s to all other nodes of $\mathcal{G}$.

Multiobjective Label Setting Algorithm



Multiobjective Label Setting Algorithm



Multiobjective Label Correcting Algorithm

- Label setting fails if negative arc lengths are permitted
- Negative cycles C
  - Case 1: If $\sum_{a \in C} c_a^k < 0$ and $\sum_{a \in C} c_a^j > 0$ for $j \neq k$ there are infinitely many efficient paths
  - Case 2: If $\sum_{a \in C} c_a \leq 0$ there is no efficient path
- A label correcting algorithm is required
- Let $\mathcal{L}(i, k)$ be set of labels at node i in iteration k

Multiobjective Label Correcting Algorithm

- Label setting fails if negative arc lengths are permitted
- Negative cycles C
  - Case 1: If $\sum_{a \in C} c_a^k < 0$ and $\sum_{a \in C} c_a^j > 0$ for $j \neq k$ there are infinitely many efficient paths
  - Case 2: If $\sum_{a \in C} c_a \leqq 0$ there is no efficient path
- A label correcting algorithm is required
- Let $\mathcal{L}(i, k)$ be set of labels at node i in iteration k

Multiobjective Label Correcting Algorithm

- Label setting fails if negative arc lengths are permitted
- Negative cycles C
  - Case 1: If $\sum_{a \in C} c_a^k < 0$ and $\sum_{a \in C} c_a^j > 0$ for $j \neq k$ there are infinitely many efficient paths
  - Case 2: If $\sum_{a \in C} c_a \leq 0$ there is no efficient path
- A label correcting algorithm is required
- Let $\mathcal{L}(i, k)$ be set of labels at node i in iteration k

Multiobjective Label Correcting Algorithm

- Label setting fails if negative arc lengths are permitted
- Negative cycles C
  - Case 1: If $\sum_{a \in C} c_a^k < 0$ and $\sum_{a \in C} c_a^j > 0$ for $j \neq k$ there are infinitely many efficient paths
  - Case 2: If $\sum_{a \in C} c_a \leq 0$ there is no efficient path
- A label correcting algorithm is required
- Let $\mathcal{L}(i, k)$ be set of labels at node i in iteration k

Multiobjective Label Correcting Algorithm

- Label setting fails if negative arc lengths are permitted
- Negative cycles C
  - Case 1: If $\sum_{a \in C} c_a^k < 0$ and $\sum_{a \in C} c_a^j > 0$ for $j \neq k$ there are infinitely many efficient paths
  - Case 2: If $\sum_{a \in C} c_a \leqq 0$ there is no efficient path
- A label correcting algorithm is required
- Let $\mathcal{L}(i, k)$ be set of labels at node i in iteration k

Multiobjective Label Correcting Algorithm

- Label setting fails if negative arc lengths are permitted
- Negative cycles C
  - Case 1: If $\sum_{a \in C} c_a^k < 0$ and $\sum_{a \in C} c_a^j > 0$ for $j \neq k$ there are infinitely many efficient paths
  - Case 2: If $\sum_{a \in C} c_a \leq 0$ there is no efficient path
- A label correcting algorithm is required
- Let $\mathcal{L}(i, k)$ be set of labels at node i in iteration k

Algorithm (Multiobjective label correcting algorithm)

Input: A digraph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with p arc costs.

Initialization: Set $d_{ii} := (0, \ldots, 0)$ for $i = 1, \ldots, n$.
Set $d_{ij} := (\infty, \ldots, \infty)$ if $v_i \neq v_j$ and $(v_i, v_j) \notin \mathcal{A}$.
Set $d_{ij} := (c^1(v_i, v_j), \ldots, c^p(v_i, v_j))$ otherwise.
Set $\mathcal{L}(i, 1) := \{d_{1i}\}, i = 1, \ldots, n$.

For $k := 1$ to $n - 1$ do

For $i := 1$ to n do

$$\mathcal{L}(i, k+1) := \min \bigcup_{j=1}^{n} \{d_{ji} + l_j^k : l_j^k \in \mathcal{L}(j, k)\}$$

End for.
If $\mathcal{L}(i, k+1) = \mathcal{L}(i, k)$ for all $i = 1, \ldots, n$ then

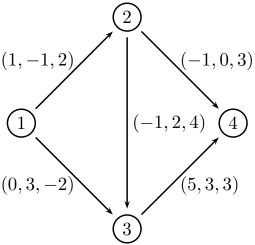
If $k = n - 1$ then STOP, a negative cycle exists.

STOP

End for.

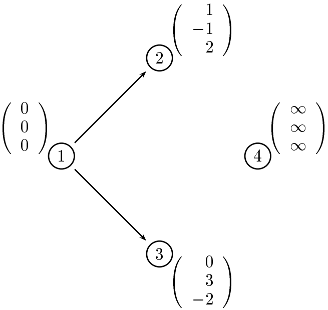
Output: All efficient paths from node v_1 to all other nodes.

Multiobjective Label Correcting Algorithm

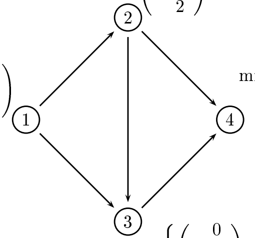


$$d_{ij} = \begin{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} & \begin{pmatrix} 1 \\ -1 \\ 2 \end{pmatrix} & \begin{pmatrix} 0 \\ 3 \\ -2 \end{pmatrix} & \begin{pmatrix} \infty \\ \infty \\ \infty \end{pmatrix} \\ \begin{pmatrix} \infty \\ \infty \\ \infty \end{pmatrix} & \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} & \begin{pmatrix} -1 \\ 2 \\ 4 \end{pmatrix} & \begin{pmatrix} -1 \\ 0 \\ 3 \end{pmatrix} \\ \begin{pmatrix} \infty \\ \infty \\ \infty \end{pmatrix} & \begin{pmatrix} \infty \\ \infty \\ \infty \end{pmatrix} & \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} & \begin{pmatrix} 5 \\ 3 \\ 3 \end{pmatrix} \\ \begin{pmatrix} \infty \\ \infty \\ \infty \end{pmatrix} & \begin{pmatrix} \infty \\ \infty \\ \infty \end{pmatrix} & \begin{pmatrix} \infty \\ \infty \\ \infty \end{pmatrix} & \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \end{pmatrix}$$

Multiobjective Label Correcting Algorithm



Multiobjective Label Correcting Algorithm

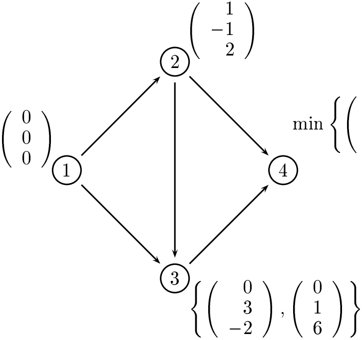


$$\begin{pmatrix}0\\0\\0\end{pmatrix} \quad \begin{pmatrix}1\\-1\\2\end{pmatrix}$$

$$\min\left\{\begin{pmatrix}1\\-1\\2\end{pmatrix}+\begin{pmatrix}-1\\0\\3\end{pmatrix},\begin{pmatrix}0\\3\\-2\end{pmatrix}+\begin{pmatrix}5\\3\\3\end{pmatrix}\right\} = \left\{\begin{pmatrix}0\\-1\\5\end{pmatrix},\begin{pmatrix}5\\6\\1\end{pmatrix}\right\}$$

$$\min\left\{\begin{pmatrix}0\\3\\-2\end{pmatrix},\begin{pmatrix}1\\-1\\2\end{pmatrix}+\begin{pmatrix}-1\\2\\4\end{pmatrix}\right\}=\left\{\begin{pmatrix}0\\3\\-2\end{pmatrix},\begin{pmatrix}0\\1\\6\end{pmatrix}\right\}$$

Multiobjective Label Correcting Algorithm



$$\min\left\{\begin{pmatrix}0\\-1\\5\end{pmatrix},\begin{pmatrix}5\\6\\1\end{pmatrix},\begin{pmatrix}0\\1\\6\end{pmatrix}+\begin{pmatrix}5\\3\\3\end{pmatrix}\right\}$$
$$=\left\{\begin{pmatrix}0\\-1\\5\end{pmatrix},\begin{pmatrix}5\\6\\1\end{pmatrix}\right\}$$

The Multiobjective Spanning Tree Problem

- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with edge costs $c_{ij}^k, k = 1, \ldots, p; (i,j) \in \mathcal{E}$
- Find efficient spanning trees of $\mathcal{G}$:

$$\min_{T \in \mathcal{T}} \sum_{[i,j] \in T} c_{ij}$$

where $\mathcal{T}$ is set of all spanning trees of $\mathcal{G}$

The Multiobjective Spanning Tree Problem

- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with edge costs $c_{ij}^k, k = 1, \ldots, p; (i,j) \in \mathcal{E}$
- Find efficient spanning trees of $\mathcal{G}$:

$$\min_{T \in \mathcal{T}} \sum_{[i,j] \in T} c_{ij}$$

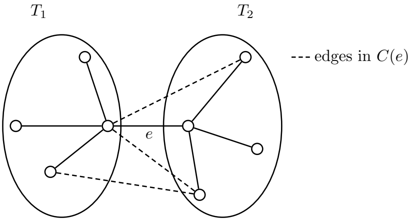
where $\mathcal{T}$ is set of all spanning trees of $\mathcal{G}$

The Multiobjective Spanning Tree Problem

Theorem (Hamacher and Ruhe 1994)

T efficient spanning tree of $\mathcal{G}$

1. *Let $e \in \mathcal{E}(T)$ be an edge of T. Let $(\mathcal{V}(T_1), \mathcal{E}(T_1))$ and $(\mathcal{V}(T_2), \mathcal{E}(T_2))$ be the two connected components of $\mathcal{G} \setminus \{e\}$. Let $C(e) := \{f = (v_i, v_j) \in \mathcal{E} : v_i \in \mathcal{V}(T_1), v_j \in \mathcal{V}(T_2)\}$ be the cut defined by deleting e. Then $c(e) \in \min\{c(f) : f \in C(e)\}$.*

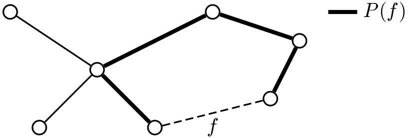


The Multiobjective Spanning Tree Problem

Theorem (Hamacher and Ruhe 1994)

T efficient spanning tree of $\mathcal{G}$

1. *Let $f \in \mathcal{E} \setminus \mathcal{E}(T)$ and let $P(f)$ be the unique path in T connecting the end nodes of f. Then $c(f) \leq c(e)$ does not hold for any $e \in P(f)$.*



The Multiobjective Spanning Tree Problem

Algorithm (Prim's spanning tree algorithm)

Input: *A graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *with* p *edge costs.*

$\mathcal{T}_0 := \{(\{1\}, \emptyset)\}$

For $k := 1$ *to* $n-1$ *do*

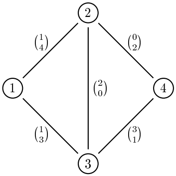
$\mathcal{T}_k := \{\mathcal{E}(T) \cup \{e_j\} : T \in \mathcal{T}_{k-1}, e_j \in argmin\{c(e) = c([v_i, v_j]) : v_i \in \mathcal{V}(T), v_j \in \mathcal{V} \setminus \mathcal{V}(T)\}\}$

End for

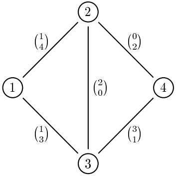
$\mathcal{T}_{n-1} := argmin\{c(T) : T \in \mathcal{T}_{n-1}\}$

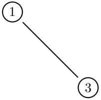
Output: $\mathcal{T}_{n-1}$, *all efficient spanning trees of* $\mathcal{G}$.

The Multiobjective Spanning Tree Problem

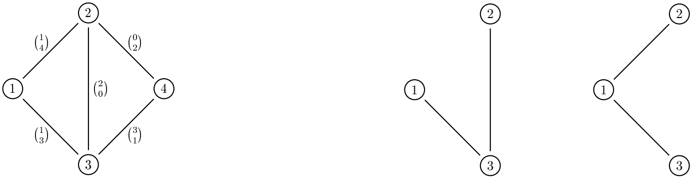


The Multiobjective Spanning Tree Problem

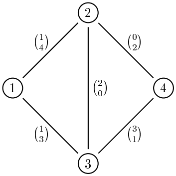


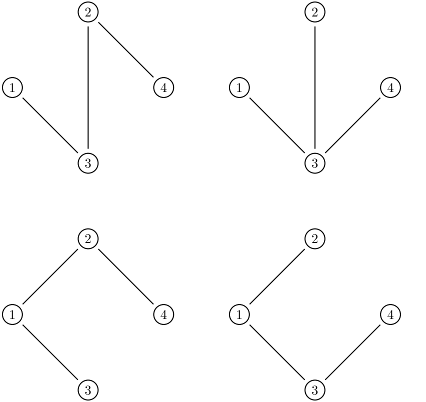


The Multiobjective Spanning Tree Problem

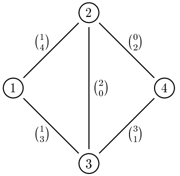


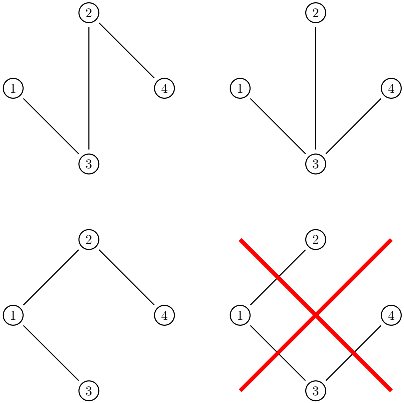
The Multiobjective Spanning Tree Problem





The Multiobjective Spanning Tree Problem





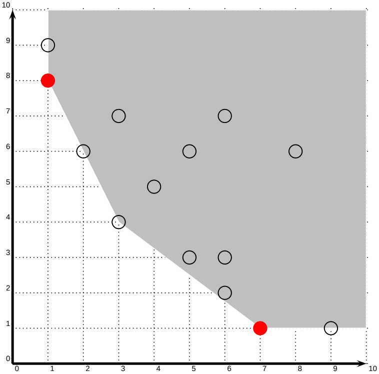
Overview

The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)

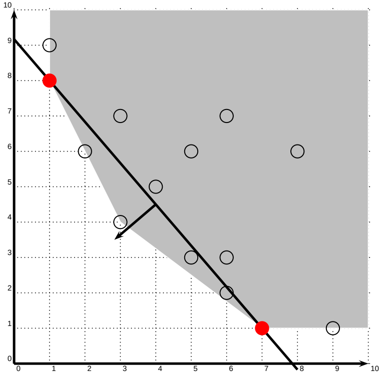
The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)



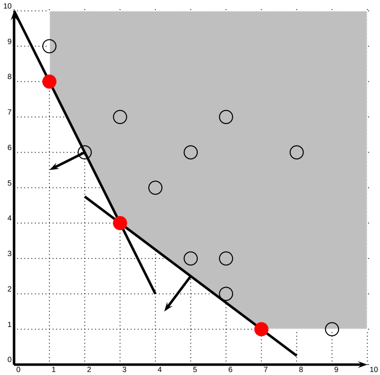
The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)



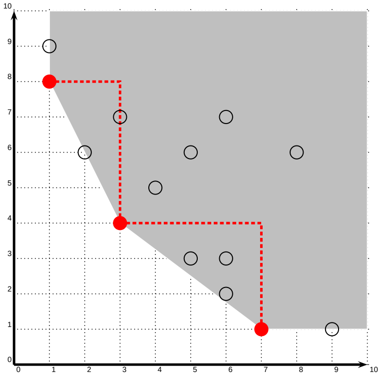
The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)



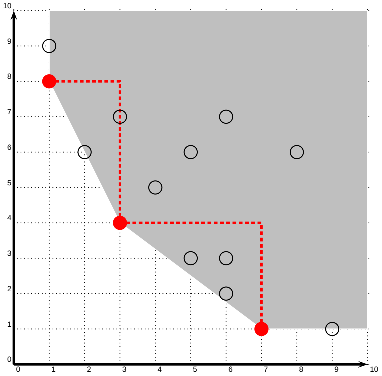
The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)



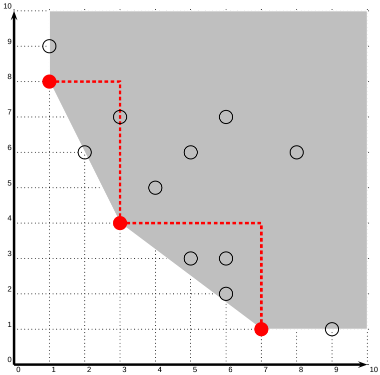
The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)



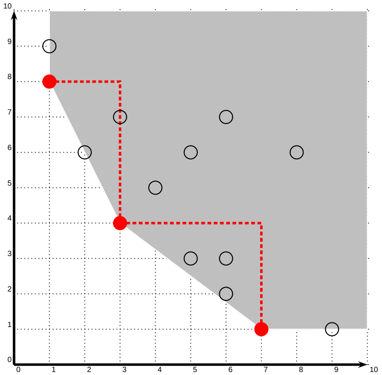
The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min\limits_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)



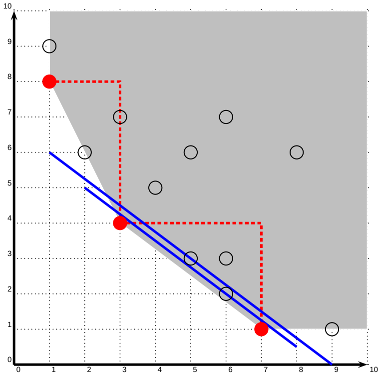
The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)



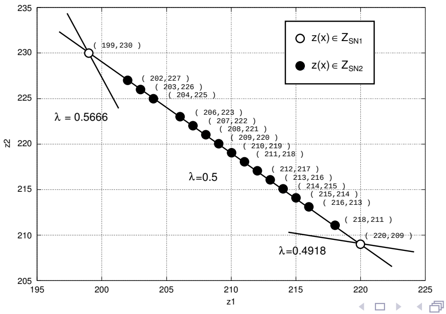
The Two Phase Method with 2 Objectives

- Phase 1: Compute $X_{SE(1)}$
  1. Find lexicographic solutions
  2. Recursively:
     Calculate λ
     Solve $\min_{x \in X} \lambda^T Cx$
- Phase 2: Compute X_{NE}
  1. Solve by triangle
  2. Use neighborhood (wrong)
  3. Use constraints (bad)
  4. Use variable fixing (possible)
  5. Use ranking (good)



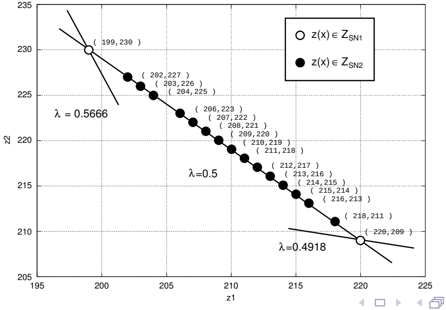
Enumeration Problems

- Finding maximal complete set:
  - Enumeration to find all optimal solutions of $\min_{x \in X} \lambda^T Cx$
  - Enumeration to find all $x \in X_{NE}$ with $Cx = y \in Y_{ND}$
- Finding minimal complete set:
  - Enumeration to find X_{SE2}



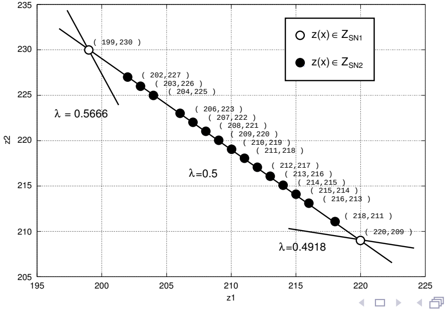
Enumeration Problems

- Finding maximal complete set:
  - Enumeration to find all optimal solutions of $\min_{x \in X} \lambda^T Cx$
  - Enumeration to find all $x \in X_{NE}$ with $Cx = y \in Y_{ND}$
- Finding minimal complete set:
  - Enumeration to find X_{SE2}



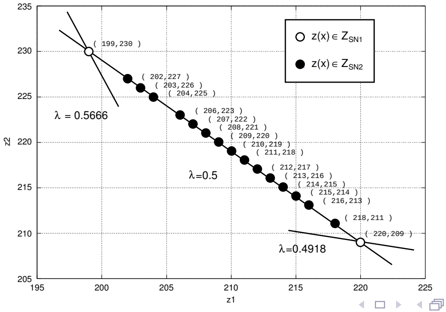
Enumeration Problems

- Finding maximal complete set:
  - Enumeration to find all optimal solutions of $\min_{x \in X} \lambda^T Cx$
  - Enumeration to find all $x \in X_{NE}$ with $Cx = y \in Y_{ND}$
- Finding minimal complete set:
  - Enumeration to find X_{SE2}



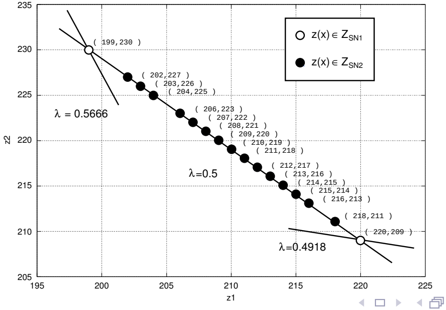
Enumeration Problems

- Finding maximal complete set:
  - Enumeration to find all optimal solutions of $\min_{x \in X} \lambda^T Cx$
  - Enumeration to find all $x \in X_{NE}$ with $Cx = y \in Y_{ND}$
- Finding minimal complete set:
  - Enumeration to find X_{SE2}



Enumeration Problems

- Finding maximal complete set:
  - Enumeration to find all optimal solutions of $\min_{x \in X} \lambda^T Cx$
  - Enumeration to find all $x \in X_{NE}$ with $Cx = y \in Y_{ND}$
- Finding minimal complete set:
  - Enumeration to find X_{SE2}

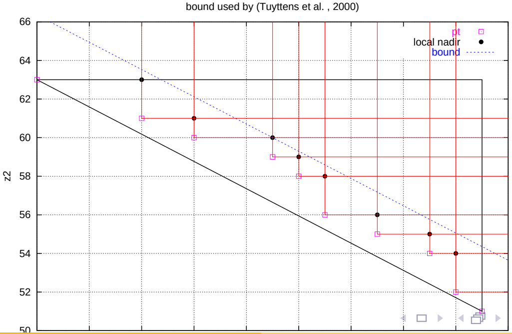


Bounds on $\lambda^T Cx$ in Phase Two

$\{x^i : 0 \leqq i \leqq q\}$ candidates for X_{NE} sorted by increasing z^1 in $\Delta(x^r, x^s)$

$$\gamma := \max_{i=0}^{q-1}\{\lambda_1 z_1(x^{i+1}) + \lambda_2 z_2(x^i)\}$$

$$\beta_0 := \max\left\{\gamma, \lambda^1 z^1(x^0) + \lambda^2 z^2(x^r), \lambda^1 z^1(x^s) + \lambda^2 z^2(x^q)\right\}$$



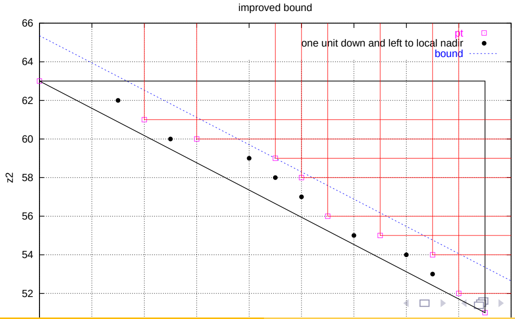
Bounds on $\lambda^T Cx$ in Phase Two

$\{x^i : 0 \leqq i \leqq q\}$ candidates for X_{NE} sorted by increasing z^1 in $\Delta(x^r, x^s)$

$$\delta_1 := \max_{i=0}^{q} \{\lambda^1 z^1(x^i) + \lambda^2 z^2(x^i)\}$$

$$\delta_2 := \max_{i=1}^{q} \{\lambda^1 (z^1(x^i) - 1) + \lambda^2 (z^2(x^{i-1}) - 1)\}$$

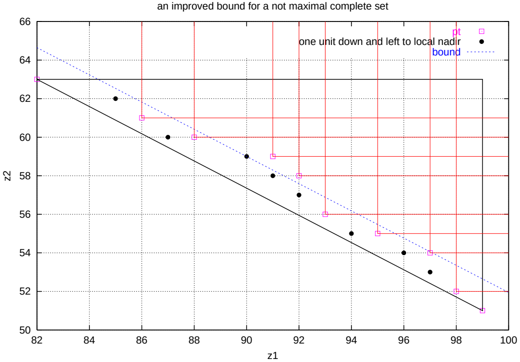
$$\beta_1 := \max \left\{ \delta_1, \delta_2, \lambda^1(z^1(x^0) - 1) + \lambda^2(z^2(x^r) - 1), \lambda^1(z^1(x^s) - 1) + \lambda^2(z^2(x^q) - 1) \right\}$$



Bounds on $\lambda^T Cx$ in Phase Two

$\{x^i : 0 \leqq i \leqq q\}$ candidates for X_{NE} sorted by increasing z^1 in $\Delta(x^r, x^s)$

$$\beta_2 := \max\left\{\delta_2, \lambda^1(z^1(x^0) - 1) + \lambda^2(z^2(x^r) - 1), \lambda^1(z^1(x^s) - 1) + \lambda^2(z^2(x^n) - 1)\right\}$$



Two Phase Algorithm for Biobjective Assignment

Przybylski et al. 2006:

- Hungarian Method for $\min_{x \in X} \lambda^T Cx$
- Enumeration of all optimal solutions of $\min_{x \in X} \lambda^T Cx$ (Fukuda and Matsui 1992)
- Ranking of (non-optimal) solutions of $\min_{x \in X} \lambda^T Cx$ (Chegireddy and Hamacher 1987)

Two Phase Algorithm for Biobjective Assignment

Przybylski et al. 2006:

- Hungarian Method for $\min_{x \in X} \lambda^T Cx$
- Enumeration of all optimal solutions of $\min_{x \in X} \lambda^T Cx$ (Fukuda and Matsui 1992)
- Ranking of (non-optimal) solutions of $\min_{x \in X} \lambda^T Cx$ (Chegireddy and Hamacher 1987)

Two Phase Algorithm for Biobjective Assignment

Przybylski et al. 2006:

- Hungarian Method for $\min_{x \in X} \lambda^T Cx$
- Enumeration of all optimal solutions of $\min_{x \in X} \lambda^T Cx$ (Fukuda and Matsui 1992)
- Ranking of (non-optimal) solutions of $\min_{x \in X} \lambda^T Cx$ (Chegireddy and Hamacher 1987)

Two Phase Algorithm for Biobjective Assignment

Przybylski et al. 2006:

- Hungarian Method for $\min_{x \in X} \lambda^T Cx$
- Enumeration of all optimal solutions of $\min_{x \in X} \lambda^T Cx$ (Fukuda and Matsui 1992)
- Ranking of (non-optimal) solutions of $\min_{x \in X} \lambda^T Cx$ (Chegireddy and Hamacher 1987)

Two Phase Algorithm for Biobjective Assignment

Results for 100×100 (1 GHz, 512 MB RAM):

Range of c_{ij}^k	Variable Fixing	Seek & Cut	Ranking
$[0, 20]$	14049.17	2251.72	228.26
$[0, 40]$	$\times$	17441.35	225.06
$[0, 60]$	$\times$	38553.18	399.65
$[0, 80]$	$\times$	53747.45	721.08
$[0, 100]$	$\times$	60227.31	711.97

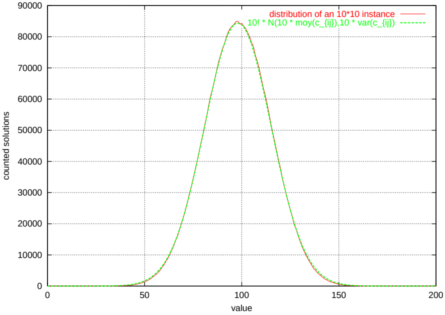
Two Phase Algorithm for Biobjective Assignment

Comparison with CPLEX 9.0 using constraints (3.4 GHz, 4 GB RAM)

Range of c_{ij}^k	CPLEX 9.0	Ranking
$[0, 20]$	200.63	85.58
$[0, 40]$	512.96	83.63
$[0, 60]$	1730.65	149.73
$[0, 80]$	3766.00	274.06
$[0, 100]$	4822.00	275.09

Two Phase Algorithm for Biobjective Assignment

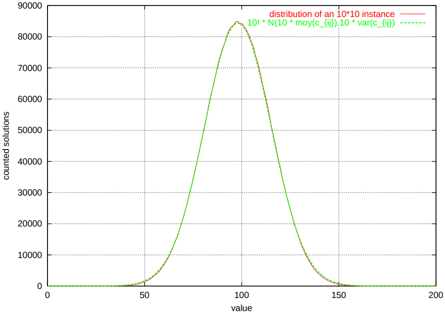
- Objective values of an AP with $c_{ij} \in \{0, \dots, r-1\}$



- Proof by Przybylski and Bourdon 2006:
$\mu = \frac{n(r-1)}{2}, \sigma^2 = \frac{n(r^2-1)}{12}$

Two Phase Algorithm for Biobjective Assignment

- Objective values of an AP with $c_{ij} \in \{0, \dots, r-1\}$



- Proof by Przybylski and Bourdon 2006:
$\mu = \frac{n(r-1)}{2}, \sigma^2 = \frac{n(r^2-1)}{12}$

The Two Phase Method with 3 Objectives

- Hyperplane defined by 3 points, possibly 6 lexicographically optimal points
- Normal vector defined by 3 nondominated points may not be positive

$$C^1 = \begin{pmatrix} 3 & 6 & 4 & 5 \\ 2 & 3 & 5 & 4 \\ 3 & 5 & 4 & 2 \\ 4 & 5 & 3 & 6 \end{pmatrix}, C^2 = \begin{pmatrix} 2 & 3 & 5 & 4 \\ 5 & 3 & 4 & 3 \\ 5 & 2 & 6 & 4 \\ 4 & 5 & 2 & 5 \end{pmatrix}, C^3 = \begin{pmatrix} 4 & 2 & 4 & 2 \\ 4 & 2 & 4 & 6 \\ 4 & 2 & 6 & 3 \\ 2 & 4 & 5 & 3 \end{pmatrix}$$

- 3 lexicographic points (11,11,14), (15,9,17) and (19,14,10)
- Normal vector $\lambda = (1, -40, -28)$ yields (16,20,16)
- Normal vector $\lambda = (-1, 40, 28)$ yields (11,11,14), (15,9,17) and (19,14,10)
- Supported non-dominated point (13,16,11) is not found

The Two Phase Method with 3 Objectives

- Hyperplane defined by 3 points, possibly 6 lexicographically optimal points
- Normal vector defined by 3 nondominated points may not be positive

$$C^1 = \begin{pmatrix} 3 & 6 & 4 & 5 \\ 2 & 3 & 5 & 4 \\ 3 & 5 & 4 & 2 \\ 4 & 5 & 3 & 6 \end{pmatrix}, C^2 = \begin{pmatrix} 2 & 3 & 5 & 4 \\ 5 & 3 & 4 & 3 \\ 5 & 2 & 6 & 4 \\ 4 & 5 & 2 & 5 \end{pmatrix}, C^3 = \begin{pmatrix} 4 & 2 & 4 & 2 \\ 4 & 2 & 4 & 6 \\ 4 & 2 & 6 & 3 \\ 2 & 4 & 5 & 3 \end{pmatrix}$$

- 3 lexicographic points (11,11,14), (15,9,17) and (19,14,10)
- Normal vector $\lambda = (1, -40, -28)$ yields (16,20,16)
- Normal vector $\lambda = (-1, 40, 28)$ yields (11,11,14), (15,9,17) and (19,14,10)
- Supported non-dominated point (13,16,11) is not found

The Two Phase Method with 3 Objectives

- Hyperplane defined by 3 points, possibly 6 lexicographically optimal points
- Normal vector defined by 3 nondominated points may not be positive

$$C^1 = \begin{pmatrix} 3 & 6 & 4 & 5 \\ 2 & 3 & 5 & 4 \\ 3 & 5 & 4 & 2 \\ 4 & 5 & 3 & 6 \end{pmatrix}, C^2 = \begin{pmatrix} 2 & 3 & 5 & 4 \\ 5 & 3 & 4 & 3 \\ 5 & 2 & 6 & 4 \\ 4 & 5 & 2 & 5 \end{pmatrix}, C^3 = \begin{pmatrix} 4 & 2 & 4 & 2 \\ 4 & 2 & 4 & 6 \\ 4 & 2 & 6 & 3 \\ 2 & 4 & 5 & 3 \end{pmatrix}$$

- 3 lexicographic points (11,11,14), (15,9,17) and (19,14,10)
- Normal vector $\lambda = (1, -40, -28)$ yields (16,20,16)
- Normal vector $\lambda = (-1, 40, 28)$ yields (11,11,14), (15,9,17) and (19,14,10)
- Supported non-dominated point (13,16,11) is not found

The Two Phase Method with 3 Objectives

- Hyperplane defined by 3 points, possibly 6 lexicographically optimal points
- Normal vector defined by 3 nondominated points may not be positive

$$C^1 = \begin{pmatrix} 3 & 6 & 4 & 5 \\ 2 & 3 & 5 & 4 \\ 3 & 5 & 4 & 2 \\ 4 & 5 & 3 & 6 \end{pmatrix}, C^2 = \begin{pmatrix} 2 & 3 & 5 & 4 \\ 5 & 3 & 4 & 3 \\ 5 & 2 & 6 & 4 \\ 4 & 5 & 2 & 5 \end{pmatrix}, C^3 = \begin{pmatrix} 4 & 2 & 4 & 2 \\ 4 & 2 & 4 & 6 \\ 4 & 2 & 6 & 3 \\ 2 & 4 & 5 & 3 \end{pmatrix}$$

- 3 lexicographic points (11,11,14), (15,9,17) and (19,14,10)
- Normal vector $\lambda = (1, -40, -28)$ yields (16,20,16)
- Normal vector $\lambda = (-1, 40, 28)$ yields (11,11,14), (15,9,17) and (19,14,10)
- Supported non-dominated point (13,16,11) is not found

The Two Phase Method with 3 Objectives

- Hyperplane defined by 3 points, possibly 6 lexicographically optimal points
- Normal vector defined by 3 nondominated points may not be positive

$$C^1 = \begin{pmatrix} 3 & 6 & 4 & 5 \\ 2 & 3 & 5 & 4 \\ 3 & 5 & 4 & 2 \\ 4 & 5 & 3 & 6 \end{pmatrix}, C^2 = \begin{pmatrix} 2 & 3 & 5 & 4 \\ 5 & 3 & 4 & 3 \\ 5 & 2 & 6 & 4 \\ 4 & 5 & 2 & 5 \end{pmatrix}, C^3 = \begin{pmatrix} 4 & 2 & 4 & 2 \\ 4 & 2 & 4 & 6 \\ 4 & 2 & 6 & 3 \\ 2 & 4 & 5 & 3 \end{pmatrix}$$

- 3 lexicographic points (11,11,14), (15,9,17) and (19,14,10)
- Normal vector $\lambda = (1, -40, -28)$ yields (16,20,16)
- Normal vector $\lambda = (-1, 40, 28)$ yields (11,11,14), (15,9,17) and (19,14,10)
- Supported non-dominated point (13,16,11) is not found

The Two Phase Method with 3 Objectives

- Hyperplane defined by 3 points, possibly 6 lexicographically optimal points
- Normal vector defined by 3 nondominated points may not be positive

$$C^1 = \begin{pmatrix} 3 & 6 & 4 & 5 \\ 2 & 3 & 5 & 4 \\ 3 & 5 & 4 & 2 \\ 4 & 5 & 3 & 6 \end{pmatrix}, C^2 = \begin{pmatrix} 2 & 3 & 5 & 4 \\ 5 & 3 & 4 & 3 \\ 5 & 2 & 6 & 4 \\ 4 & 5 & 2 & 5 \end{pmatrix}, C^3 = \begin{pmatrix} 4 & 2 & 4 & 2 \\ 4 & 2 & 4 & 6 \\ 4 & 2 & 6 & 3 \\ 2 & 4 & 5 & 3 \end{pmatrix}$$

- 3 lexicographic points (11,11,14), (15,9,17) and (19,14,10)
- Normal vector $\lambda = (1, -40, -28)$ yields (16,20,16)
- Normal vector $\lambda = (-1, 40, 28)$ yields (11,11,14), (15,9,17) and (19,14,10)
- Supported non-dominated point (13,16,11) is not found

Weight Space Decomposition

$$W^0 := \left\{ \lambda : \lambda_1 > 0, \ldots, \lambda_p > 0, \lambda_p = 1 - \sum_{k=1}^{p-1} \lambda_k \right\}$$

$$W^0(y) := \{ \lambda \in W^0 : \lambda^T y \leqq \lambda^t Y' \text{ for all } y' \in Y \}$$

Proposition

1. *If y is supported but not extreme then $W^0(y) = \bigcap_{i=1}^{k} W^0(y^i)$ where $y^i, i \in \{1, \ldots, k\}$ are the extreme points of the face of conv(Y) that contains y in its relative interior.*
2. *Let y be a supported point, then $W^0(y) = \{\lambda \in W^0 : \lambda^T y \leqq \lambda^T y'$ for all supported extreme points y'.*
3. *$W^0(y)$ is a nonempty convex polytope if y is supported.*

Weight Space Decomposition

$$W^0 := \left\{\lambda : \lambda_1 > 0, \ldots, \lambda_p > 0, \lambda_p = 1 - \sum_{k=1}^{p-1} \lambda_k\right\}$$

$$W^0(y) := \{\lambda \in W^0 : \lambda^T y \leqq \lambda^t Y' \text{ for all } y' \in Y\}$$

Proposition

1. *If y is supported but not extreme then $W^0(y) = \bigcap_{i=1}^k W^0(y^i)$ where $y^i, i \in \{1, \ldots, k\}$ are the extreme points of the face of $\text{conv}(Y)$ that contains y in its relative interior.*
2. *Let y be a supported point, then $W^0(y) = \{\lambda \in W^0 : \lambda^T y \leqq \lambda^T y'$ for all supported extreme points y'.*
3. *$W^0(y)$ is a nonempty convex polytope if y is supported.*

Weight Space Decomposition

$$W^0 := \left\{\lambda : \lambda_1 > 0, \ldots, \lambda_p > 0, \lambda_p = 1 - \sum_{k=1}^{p-1} \lambda_k \right\}$$

$$W^0(y) := \{\lambda \in W^0 : \lambda^T y \leqq \lambda^t Y' \text{ for all } y' \in Y\}$$

Proposition

1. *If y is supported but not extreme then* $W^0(y) = \bigcap_{i=1}^{k} W^0(y^i)$ *where* $y^i, i \in \{1, \ldots, k\}$ *are the extreme points of the face of* $conv(Y)$ *that contains y in its relative interior.*
2. *Let y be a supported point, then* $W^0(y) = \{\lambda \in W^0 : \lambda^T y \leqq \lambda^T y'$ *for all supported extreme points* y'.
3. $W^0(y)$ *is a nonempty convex polytope if y is supported.*

Weight Space Decomposition

$$W^0 := \left\{\lambda : \lambda_1 > 0, \ldots, \lambda_p > 0, \lambda_p = 1 - \sum_{k=1}^{p-1} \lambda_k\right\}$$

$$W^0(y) := \{\lambda \in W^0 : \lambda^T y \leqq \lambda^t Y' \text{ for all } y' \in Y\}$$

Proposition

1. *If y is supported but not extreme then $W^0(y) = \bigcap_{i=1}^k W^0(y^i)$ where $y^i, i \in \{1, \ldots, k\}$ are the extreme points of the face of $conv(Y)$ that contains y in its relative interior.*
2. *Let y be a supported point, then $W^0(y) = \{\lambda \in W^0 : \lambda^T y \leqq \lambda^T y'$ for all supported extreme points y'.*
3. *$W^0(y)$ is a nonempty convex polytope if y is supported.*

Weight Space Decomposition

Proposition

1. *If y is a supported extreme point then $W^0(y)$ is a polytope of dim $p-1$.*
2. *Let y^1 and y^2 be two supported points and $W^0(y^1) \cap W^0(y^2) \neq \emptyset$ then $W^0(y^1)$ and $W^0(y^2)$ have a common face.*

Two supported extreme points y^1 and y^2 are adjacent if $W^0(y^1) \cap W^0(y^2)$ is a polytope of dimension $p-2$

Proposition

Let $\{y^1, \ldots, y^n\}$ be the set of supported extreme points, then $W^0 = \bigcup_{i=1}^n W^0(y^i)$.

Weight Space Decomposition

Proposition

1. *If y is a supported extreme point then $W^0(y)$ is a polytope of dim $p-1$.*
2. *Let y^1 and y^2 be two supported points and $W^0(y^1) \cap W^0(y^2) \neq \emptyset$ then $W^0(y^1)$ and $W^0(y^2)$ have a common face.*

Two supported extreme points y^1 and y^2 are adjacent if $W^0(y^1) \cap W^0(y^2)$ is a polytope of dimension $p-2$

Proposition

Let $\{y^1, \ldots, y^n\}$ be the set of supported extreme points, then $W^0 = \bigcup_{i=1}^{n} W^0(y^i)$.

Weight Space Decomposition

Proposition

1. *If y is a supported extreme point then $W^0(y)$ is a polytope of dim $p-1$.*
2. *Let y^1 and y^2 be two supported points and $W^0(y^1) \cap W^0(y^2) \neq \emptyset$ then $W^0(y^1)$ and $W^0(y^2)$ have a common face.*

Two supported extreme points y^1 and y^2 are adjacent if $W^0(y^1) \cap W^0(y^2)$ is a polytope of dimension $p-2$

Proposition

Let $\{y^1, \ldots, y^n\}$ be the set of supported extreme points, then $W^0 = \bigcup_{i=1}^n W^0(y^i)$.

Weight Space Decomposition

Proposition

1. *If y is a supported extreme point then $W^0(y)$ is a polytope of dim $p-1$.*
2. *Let y^1 and y^2 be two supported points and $W^0(y^1) \cap W^0(y^2) \neq \emptyset$ then $W^0(y^1)$ and $W^0(y^2)$ have a common face.*

Two supported extreme points y^1 and y^2 are adjacent if $W^0(y^1) \cap W^0(y^2)$ is a polytope of dimension $p-2$

Proposition

Let $\{y^1, \dots, y^n\}$ be the set of supported extreme points, then $W^0 = \bigcup_{i=1}^{n} W^0(y^i)$.

Weight Space Decomposition

Proposition (Optimality Condition)

If S is a set of supported points then

$$Y_{SN1} \subseteq S \iff W^0 = \bigcup_{y \in S} W^0(y).$$

- Let S be a set of supported points
- Let $W_p^0(y) = \{\lambda \in W^0 : \langle \lambda, y \rangle \leq \langle \lambda, y^* \rangle \text{ for all } y^* \in S\}$
- $W^0(y) \subseteq W_p^0(y)$ for all $y \in S$
- $W^0 = \bigcup_{y \in S} W_p^0(y)$

Weight Space Decomposition

Proposition (Optimality Condition)

If S is a set of supported points then

$$Y_{SN1} \subseteq S \iff W^0 = \bigcup_{y \in S} W^0(y).$$

- Let S be a set of supported points
- Let $W_p^0(y) = \{\lambda \in W^0 : \langle \lambda, y \rangle \leq \langle \lambda, y^* \rangle \text{ for all } y^* \in S\}$
- $W^0(y) \subseteq W_p^0(y)$ for all $y \in S$
- $W^0 = \bigcup_{y \in S} W_p^0(y)$

Weight Space Decomposition

Proposition (Optimality Condition)

If S is a set of supported points then

$$Y_{SN1} \subseteq S \iff W^0 = \bigcup_{y \in S} W^0(y).$$

- Let S be a set of supported points
- Let $W_p^0(y) = \{\lambda \in W^0 : \langle \lambda, y \rangle \leq \langle \lambda, y^* \rangle \text{ for all } y^* \in S\}$
- $W^0(y) \subseteq W_p^0(y)$ for all $y \in S$
- $W^0 = \bigcup_{y \in S} W_p^0(y)$

Weight Space Decomposition

Proposition (Optimality Condition)

If S is a set of supported points then

$$Y_{SN1} \subseteq S \iff W^0 = \bigcup_{y \in S} W^0(y).$$

- Let S be a set of supported points
- Let $W^0_p(y) = \{\lambda \in W^0 : \langle \lambda, y \rangle \leq \langle \lambda, y^* \rangle \text{ for all } y^* \in S\}$
- $W^0(y) \subseteq W^0_p(y)$ for all $y \in S$
- $W^0 = \bigcup_{y \in S} W^0_p(y)$

Weight Space Decomposition

Proposition (Optimality Condition)

If S is a set of supported points then

$$Y_{SN1} \subseteq S \Longleftrightarrow W^0 = \bigcup_{y \in S} W^0(y).$$

- Let S be a set of supported points
- Let $W_p^0(y) = \{\lambda \in W^0 : \langle \lambda, y \rangle \leq \langle \lambda, y^* \rangle \text{ for all } y^* \in S\}$
- $W^0(y) \subseteq W_p^0(y)$ for all $y \in S$
- $W^0 = \bigcup_{y \in S} W_p^0(y)$

Algorithm (Phase 1 with 3 Objectives)

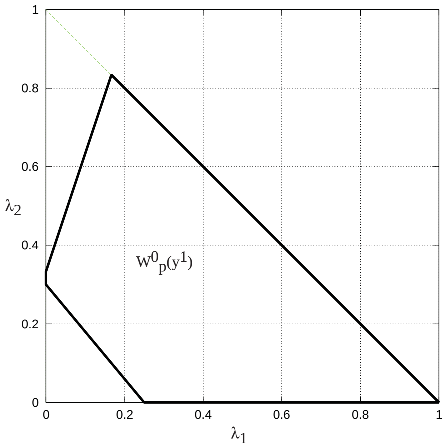
1. *Initialize S with the lexicographically optimal points*
2. *For all $\hat{y} \in S$*
   - *Compute $W_p^0(\hat{y})$*
   - *Investigate all facets F of $W_p^0(\hat{y})$ defined by $\lambda^T \hat{y} = \lambda^T y'$ for $y' \in S$ to determine whether F is also a facet of $W^0(\hat{y})$*
     - *If $\hat{y}$ minimizes $\lambda^T y$ for all $\lambda \in F$ then $\hat{y}$ and y' are adjacent and F is the common face of $W^0(\hat{y})$ and $W^0(y')$*
     - *If there are $y^* \in Y$ and $\lambda \in F$ such that $\lambda^T y^* < \lambda^T y$ then $W^0(\hat{y}) \subset W_p^0(\hat{y})$ and y^* is added to S and $W_p^0(\hat{y})$ is updated*

An Example

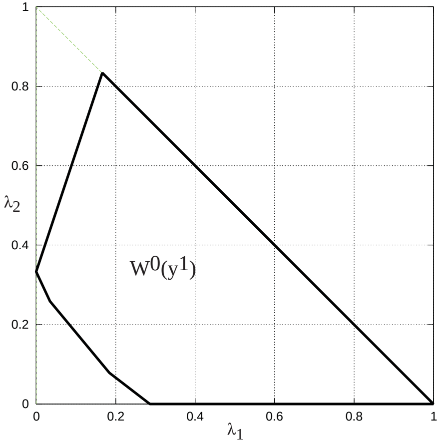
$$C^1 = \begin{pmatrix} 2 & 5 & 4 & 7 \\ 3 & 3 & 5 & 7 \\ 3 & 8 & 4 & 2 \\ 6 & 5 & 2 & 5 \end{pmatrix}, \; C^2 = \begin{pmatrix} 3 & 3 & 6 & 2 \\ 5 & 3 & 7 & 3 \\ 5 & 2 & 7 & 4 \\ 4 & 6 & 3 & 5 \end{pmatrix}, \; C^3 = \begin{pmatrix} 4 & 2 & 5 & 3 \\ 5 & 3 & 4 & 3 \\ 4 & 3 & 5 & 2 \\ 6 & 4 & 7 & 3 \end{pmatrix}$$

- Lexicographically optimal points: $y^1 = (9, 13, 16)$, $y^2 = (19, 11, 17)$, $y^3 = (18, 20, 13)$
- $S = \{y^1, y^2, y^3\}$

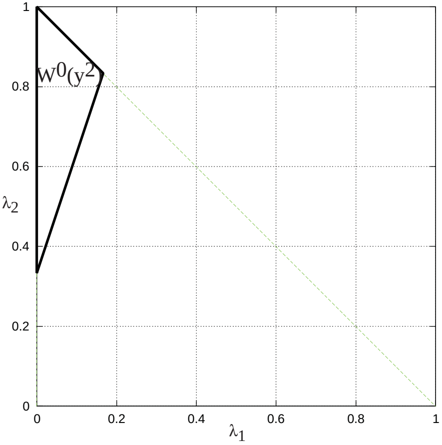
An Example



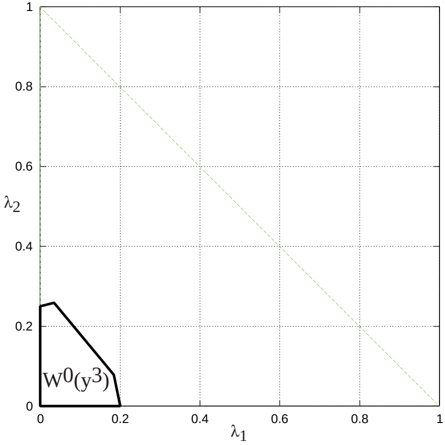
An Example



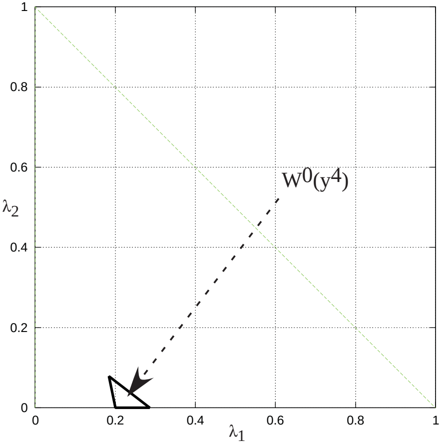
An Example



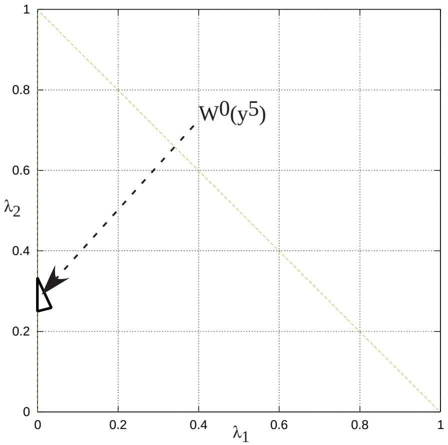
An Example



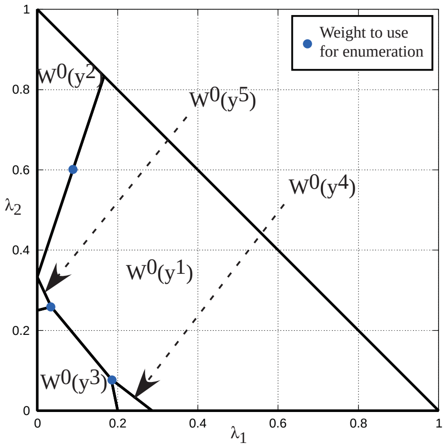
An Example



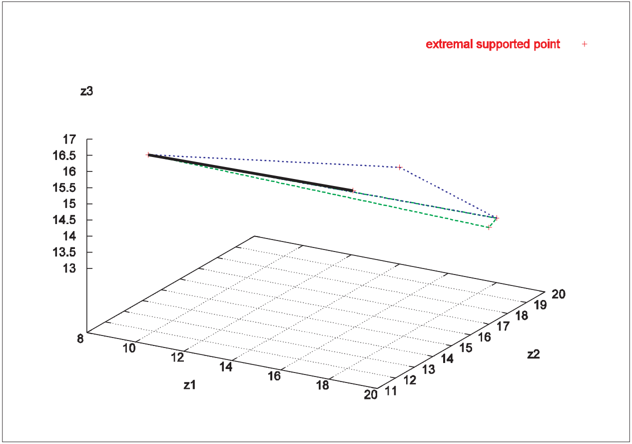
An Example



An Example


1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1
λ_2
λ_1
Weight to use for enumeration
$W^0(y^2)$
$W^0(y^5)$
$W^0(y^4)$
$W^0(y^1)$
$W^0(y^3)$

An Example


extremal supported point
z3
17
16.5
16
15.5
15
14.5
14
13.5
13
8
10
12
14
16
18
20
z1
11
12
13
14
15
16
17
18
19
20
z2

Numerical Results

Size	$\lvert Y_N \rvert$	SC 2004	T-P 2003	LZT 2005	2 Phase
5	12	0.15	0.04	0.15	0.00
10	221	99865.00	97.30	41.70	0.08
15	483	×	544.53	172.29	0.36
20	1942	×	×	1607.92	4.51
25	3750	×	×	5218.00	30.13
30	5195	×	×	15579.00	55.87
35	10498	×	×	101751.00	109.96
40	14733	×	×	×	229.05
45	23941	×	×	×	471.60
50	29193	×	×	×	802.68

Comments

- Two phase method respects problem structure
- Successful for assignment, shortest path, network flow
- Requires ranking algorithm, which is available for most polynomially solvable problems

Comments

- Two phase method respects problem structure
- Successful for assignment, shortest path, network flow
- Requires ranking algorithm, which is available for most polynomially solvable problems

Comments

- Two phase method respects problem structure
- Successful for assignment, shortest path, network flow
- Requires ranking algorithm, which is available for most polynomially solvable problems

Overview

Principle and Properties of Scalarization

Convert multiobjective problem to (parameterized) single objective problem and solve repeatedly with different parameter values

Desirable properties of scalarizations: (Wierzbicki 1984)

- Correctness: Optimal solutions are (weakly) efficient
- Completeness: All efficient solutions can be found
- Computability: Scalarization is not harder than single objective version of problem (theory and practice)
- Linearity: Scalarization has linear formulation

Principle and Properties of Scalarization

Convert multiobjective problem to (parameterized) single objective problem and solve repeatedly with different parameter values

Desirable properties of scalarizations: (Wierzbicki 1984)

- Correctness: Optimal solutions are (weakly) efficient
- Completeness: All efficient solutions can be found
- Computability: Scalarization is not harder than single objective version of problem (theory and practice)
- Linearity: Scalarization has linear formulation

Principle and Properties of Scalarization

Convert multiobjective problem to (parameterized) single objective problem and solve repeatedly with different parameter values

Desirable properties of scalarizations: (Wierzbicki 1984)

- Correctness: Optimal solutions are (weakly) efficient
- Completeness: All efficient solutions can be found
- Computability: Scalarization is not harder than single objective version of problem (theory and practice)
- Linearity: Scalarization has linear formulation

Principle and Properties of Scalarization

Convert multiobjective problem to (parameterized) single objective problem and solve repeatedly with different parameter values

Desirable properties of scalarizations: (Wierzbicki 1984)

- Correctness: Optimal solutions are (weakly) efficient
- Completeness: All efficient solutions can be found
- Computability: Scalarization is not harder than single objective version of problem (theory and practice)
- Linearity: Scalarization has linear formulation

Principle and Properties of Scalarization

Convert multiobjective problem to (parameterized) single objective problem and solve repeatedly with different parameter values

Desirable properties of scalarizations: (Wierzbicki 1984)

- Correctness: Optimal solutions are (weakly) efficient
- Completeness: All efficient solutions can be found
- Computability: Scalarization is not harder than single objective version of problem (theory and practice)
- Linearity: Scalarization has linear formulation

Principle and Properties of Scalarization

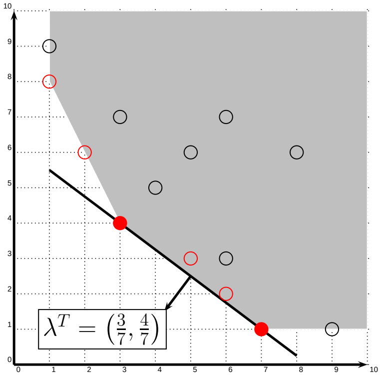
Convert multiobjective problem to (parameterized) single objective problem and solve repeatedly with different parameter values

Desirable properties of scalarizations: (Wierzbicki 1984)

- Correctness: Optimal solutions are (weakly) efficient
- Completeness: All efficient solutions can be found
- Computability: Scalarization is not harder than single objective version of problem (theory and practice)
- Linearity: Scalarization has linear formulation

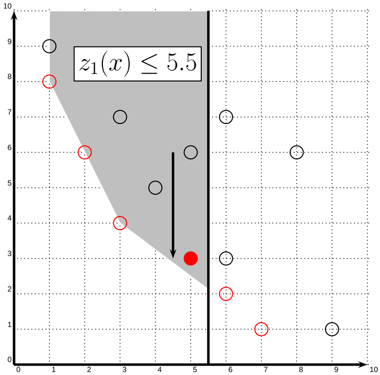
Scalarization Methods

- Weighted sum:
  $$\min_{x \in X} \left\{ \lambda^T z(x) \right\}$$
- ε-constraint:
  $$\min_{x \in X} \{ z_l(x) : z_k(x) \leq \varepsilon_k, k \neq l \}$$
- Weighted Chebychev:
  $$\min_{x \in X} \left\{ \max_{k=1,\ldots,p} \nu_k (z_k(x) - y_k^I) \right\}$$



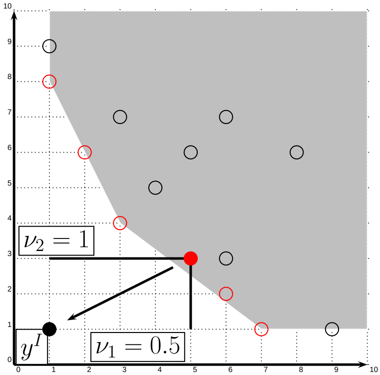
Scalarization Methods

- Weighted sum:
  $\min_{x \in X} \left\{ \lambda^T z(x) \right\}$
- ε-constraint:
  $\min_{x \in X} \{ z_l(x) : z_k(x) \leq \varepsilon_k, k \neq l \}$
- Weighted Chebychev:
  $\min_{x \in X} \left\{ \max_{k=1,\ldots,p} \nu_k (z_k(x) - y_k^I) \right\}$



Scalarization Methods

- Weighted sum:
  $$\min_{x \in X} \left\{ \lambda^T z(x) \right\}$$
- ε-constraint:
  $$\min_{x \in X} \{ z_l(x) : z_k(x) \leq \varepsilon_k, k \neq l \}$$
- Weighted Chebychev:
  $$\min_{x \in X} \left\{ \max_{k=1,\dots,p} \nu_k (z_k(x) - y^I_k) \right\}$$



General Formulation

$$\min_{x \in X} \left\{ \max_{k=1}^{p} [\nu_k(c_k x - \rho_k)] + \sum_{k=1}^{p} [\lambda_k(c_k x - \rho_k)] \right\}$$

$$\text{subject to} \quad c_k x \leq \varepsilon_k \qquad k = 1, \ldots, p$$

Includes	Correct	Complete	Computable	Linear
Weighted sum	+	-	+	+
ε-constraint	+	+	-	+
Benson	+	+	-	+
Chebychev	+	(+)	(-)	+
Max-ordering	+	+	-	+
Reference point	+	(+)	(-)	+

General Formulation

$$\min_{x \in X} \left\{ \max_{k=1}^{p} [\nu_k(c_k x - \rho_k)] + \sum_{k=1}^{p} [\lambda_k(c_k x - \rho_k)] \right\}$$

$$\text{subject to} \quad c_k x \leq \varepsilon_k \qquad k = 1, \ldots, p$$

Includes	Correct	Complete	Computable	Linear
Weighted sum	+	-	+	+
ε-constraint	+	+	-	+
Benson	+	+	-	+
Chebychev	+	(+)	(-)	+
Max-ordering	+	+	-	+
Reference point	+	(+)	(-)	+

General Formulation

$$\min_{x \in X} \left\{ \max_{k=1}^{p} [\nu_k (c_k x - \rho_k)] + \sum_{k=1}^{p} [\lambda_k (c_k x - \rho_k)] \right\}$$

$$\text{subject to} \quad c_k x \leq \varepsilon_k \qquad k = 1, \ldots, p$$

Includes	Correct	Complete	Computable	Linear
Weighted sum	+	-	+	+
ε-constraint	+	+	-	+
Benson	+	+	-	+
Chebychev	+	(+)	(-)	+
Max-ordering	+	+	-	+
Reference point	+	(+)	(-)	+

General Formulation

$$\min_{x \in X} \left\{ \max_{k=1}^{p} [\nu_k(c_k x - \rho_k)] + \sum_{k=1}^{p} [\lambda_k(c_k x - \rho_k)] \right\}$$

$$\text{subject to} \quad c_k x \leq \varepsilon_k \qquad k = 1, \ldots, p$$

Includes	Correct	Complete	Computable	Linear
Weighted sum	+	-	+	+
ε-constraint	+	+	-	+
Benson	+	+	-	+
Chebychev	+	(+)	(-)	+
Max-ordering	+	+	-	+
Reference point	+	(+)	(-)	+

General Formulation

$$\min_{x \in X} \left\{ \max_{k=1}^{p} [\nu_k (c_k x - \rho_k)] + \sum_{k=1}^{p} [\lambda_k (c_k x - \rho_k)] \right\}$$

$$\text{subject to} \quad c_k x \leq \varepsilon_k \qquad k = 1, \ldots, p$$

Includes	Correct	Complete	Computable	Linear
Weighted sum	+	-	+	+
ε-constraint	+	+	-	+
Benson	+	+	-	+
Chebychev	+	(+)	(-)	+
Max-ordering	+	+	-	+
Reference point	+	(+)	(-)	+

General Formulation

$$\min_{x \in X} \left\{ \max_{k=1}^{p} [\nu_k(c_k x - \rho_k)] + \sum_{k=1}^{p} [\lambda_k(c_k x - \rho_k)] \right\}$$

$$\text{subject to} \quad c_k x \leq \varepsilon_k \qquad k = 1, \ldots, p$$

Includes	Correct	Complete	Computable	Linear
Weighted sum	+	-	+	+
ε-constraint	+	+	-	+
Benson	+	+	-	+
Chebychev	+	(+)	(-)	+
Max-ordering	+	+	-	+
Reference point	+	(+)	(-)	+

General Formulation

$$\min_{x \in X} \left\{ \max_{k=1}^{p} \left[\nu_k (c_k x - \rho_k)\right] + \sum_{k=1}^{p} \left[\lambda_k (c_k x - \rho_k)\right] \right\}$$

$$\text{subject to} \quad c_k x \leq \varepsilon_k \qquad k = 1, \ldots, p$$

Includes	Correct	Complete	Computable	Linear
Weighted sum	+	-	+	+
ε-constraint	+	+	-	+
Benson	+	+	-	+
Chebychev	+	(+)	(-)	+
Max-ordering	+	+	-	+
Reference point	+	(+)	(-)	+

General Formulation

Theorem (Ehrgott 2005)

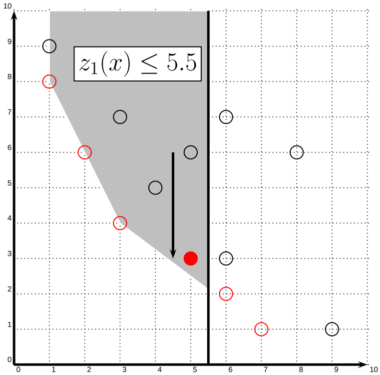
1. *The general scalarization is NP-hard.*
2. *An optimal solution of the Lagrangian dual of the linearized general scalarization is a supported efficient solution.*

Method of Elastic Constraints

$$\min_{x\in X} c_l x + \sum_{k\neq l} \mu_k w_k$$

$$\text{s.t.} \quad c_k x + v_k - w_k \leq \varepsilon_k \quad k \neq l$$

$$v_k, w_k \geq 0 \quad k \neq l$$

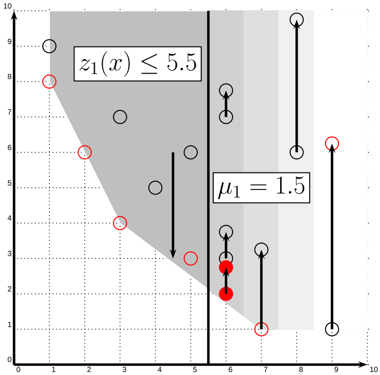


Method of Elastic Constraints

$$\min_{x \in X} c_l x + \sum_{k \neq l} \mu_k w_k$$

$$\text{s.t. } c_k x + v_k - w_k = \varepsilon_k \quad k \neq l$$

$$v_k, w_k \geq 0 \quad k \neq l$$



Method of Elastic Constraints

Theorem (Ehrgott and Ryan 2002)

The method of elastic constraints

- *is correct and complete,*
- *contains the weighted sum and ε-constraint method as special cases,*
- *is NP-hard.*

... but (often) solvable in practice because

- it "respects" problem structure
- it "limits damage" of ε-constraints

Method of Elastic Constraints

Theorem (Ehrgott and Ryan 2002)

The method of elastic constraints

- *is correct and complete,*
- *contains the weighted sum and ε-constraint method as special cases,*
- *is NP-hard.*

... but (often) solvable in practice because

- it "respects" problem structure
- it "limits damage" of ε-constraints

Method of Elastic Constraints

Theorem (Ehrgott and Ryan 2002)

The method of elastic constraints

- *is correct and complete,*
- *contains the weighted sum and ε-constraint method as special cases,*
- *is NP-hard.*

... but (often) solvable in practice because

- it "respects" problem structure
- it "limits damage" of ε-constraints

Method of Elastic Constraints

Theorem (Ehrgott and Ryan 2002)

The method of elastic constraints

- *is correct and complete,*
- *contains the weighted sum and ε-constraint method as special cases,*
- *is NP-hard.*

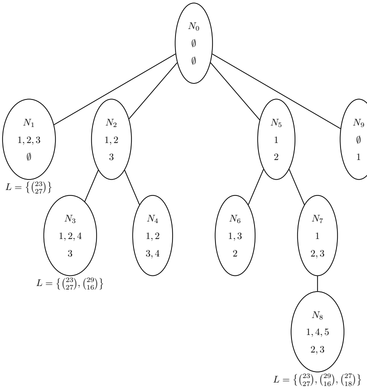
... but (often) solvable in practice because

- it "respects" problem structure
- it "limits damage" of ε-constraints

Overview

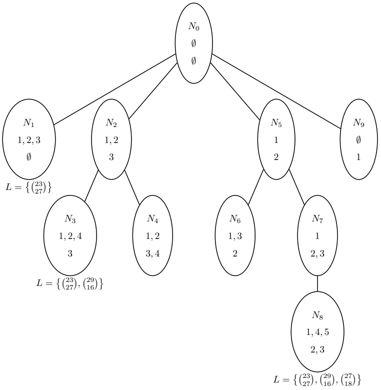
Multicriteria Branch and Bound

- Ulungu and Teghem 1997, Mavrotas and Diakoulaki 2002
- Branching: As in single objective case
- Bounding: Ideal point of problem at node is dominated by efficient solution
- Branching may be very ineffective
- Use lower and upper bound sets



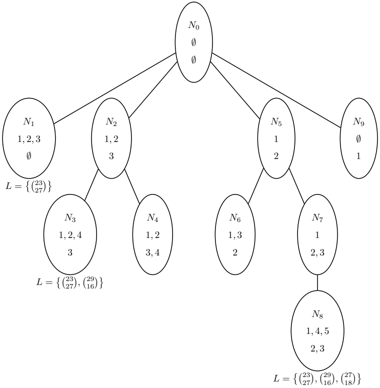
Multicriteria Branch and Bound

- Ulungu and Teghem 1997, Mavrotas and Diakoulaki 2002
- Branching: As in single objective case
- Bounding: Ideal point of problem at node is dominated by efficient solution
- Branching may be very ineffective
- Use lower and upper bound sets



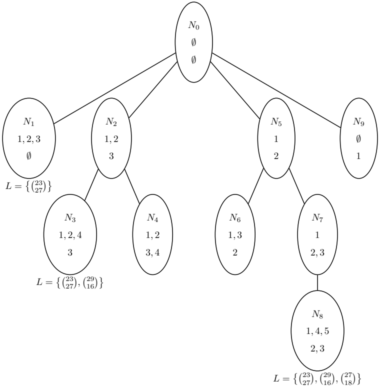
Multicriteria Branch and Bound

- Ulungu and Teghem 1997, Mavrotas and Diakoulaki 2002
- Branching: As in single objective case
- Bounding: Ideal point of problem at node is dominated by efficient solution
- Branching may be very ineffective
- Use lower and upper bound sets



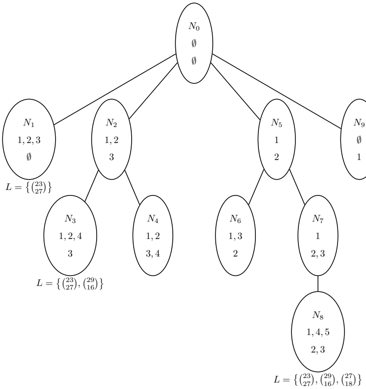
Multicriteria Branch and Bound

- Ulungu and Teghem 1997, Mavrotas and Diakoulaki 2002
- Branching: As in single objective case
- Bounding: Ideal point of problem at node is dominated by efficient solution
- Branching may be very ineffective
- Use lower and upper bound sets

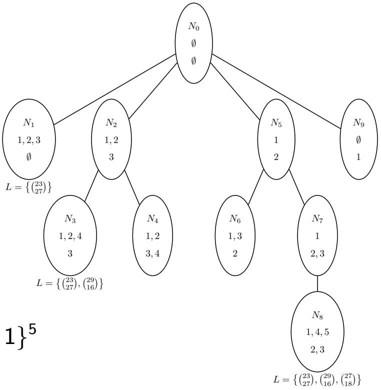


Multicriteria Branch and Bound

- Ulungu and Teghem 1997, Mavrotas and Diakoulaki 2002
- Branching: As in single objective case
- Bounding: Ideal point of problem at node is dominated by efficient solution
- Branching may be very ineffective
- Use lower and upper bound sets



Multicriteria Branch and Bound

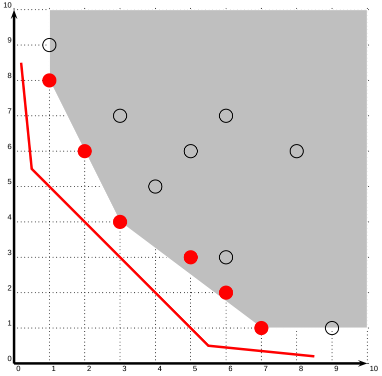


$$
\begin{array}{lrcl}
\max & 11x_1 + 5x_2 + 7x_3 + 13x_4 + 3x_5 & & \\
\max & 9x_1 + 2x_2 + 16x_3 + 5x_4 + 4x_5 & & \\
\text{s. t.} & 4x_1 + 2x_2 + 8x_3 + 7x_4 + 5x_5 & \leq & 16 \\
 & x & \in & \{0,1\}^5
\end{array}
$$

Bound Sets

Ehrgott and Gandibleux 2005:

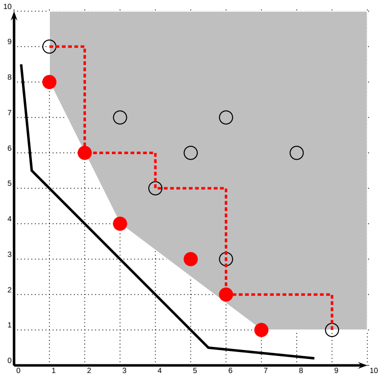
1. Lower bound set L
   - is $\mathbb{R}^p_{\geqq}$-closed
   - is $\mathbb{R}^p_{\geqq}$-bounded
   - $Y_N \subset L + \mathbb{R}^p_{\geqq}$
   - $L \subset \left(L + \mathbb{R}^p_{\geqq}\right)_N$
2. Upper bound set U
   - is $\mathbb{R}^p_{\geqq}$-closed
   - is $\mathbb{R}^p_{\geqq}$-bounded
   - $Y_N \in \mathrm{cl}\left[\left(U + \mathbb{R}^p_{\geqq}\right)^c\right]$
   - $U \subset \left(U + \mathbb{R}^p_{\geqq}\right)_N$

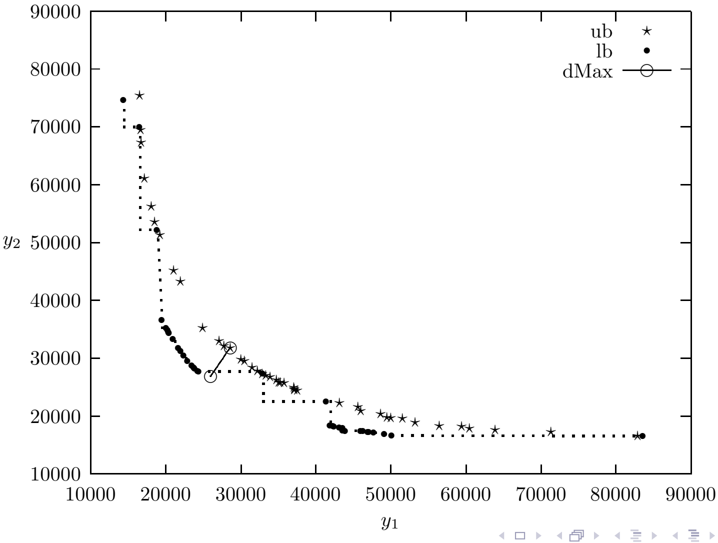


Bound Sets

Ehrgott and Gandibleux 2005:

1. Lower bound set L
   - is $\mathbb{R}^p_{\geqq}$-closed
   - is $\mathbb{R}^p_{\geqq}$-bounded
   - $Y_N \subset L + \mathbb{R}^p_{\geqq}$
   - $L \subset \left(L + \mathbb{R}^p_{\geqq}\right)_N$
2. Upper bound set U
   - is $\mathbb{R}^p_{\geqq}$-closed
   - is $\mathbb{R}^p_{\geqq}$-bounded
   - $Y_N \in \mathrm{cl}\left[\left(U + \mathbb{R}^p_{\geqq}\right)^c\right]$
   - $U \subset \left(U + \mathbb{R}^p_{\geqq}\right)_N$




ub
lb
dMax
90000
80000
70000
60000
50000
40000
30000
20000
10000
y_2
10000
20000
30000
40000
50000
60000
70000
80000
90000
y_1

Overview

Conclusion

- MOCO problems as special multiobjective integer programmes
- Distinguish supported and non-supported efficient solutions
- MOCO problems are NP-hard, #P-hard, intractable
- MOCO problems can be solvable in practice
- Shortest path: Label setting and correcting algorithms transferable, i.e. dynamic programming principles work
- Spanning tree: Prim's (and Kruskal's) algorithm transferable, i.e. greedy algorithm works
- Moderate problems: Two phase method works well, ranking algorithm required (assignment, network flow)
- Difficult problems: Scalarization and branch and bound

Conclusion

- MOCO problems as special multiobjective integer programmes
- Distinguish supported and non-supported efficient solutions
- MOCO problems are NP-hard, #P-hard, intractable
- MOCO problems can be solvable in practice
- Shortest path: Label setting and correcting algorithms transferable, i.e. dynamic programming principles work
- Spanning tree: Prim's (and Kruskal's) algorithm transferable, i.e. greedy algorithm works
- Moderate problems: Two phase method works well, ranking algorithm required (assignment, network flow)
- Difficult problems: Scalarization and branch and bound

Conclusion

- MOCO problems as special multiobjective integer programmes
- Distinguish supported and non-supported efficient solutions
- MOCO problems are NP-hard, #P-hard, intractable
- MOCO problems can be solvable in practice
- Shortest path: Label setting and correcting algorithms transferable, i.e. dynamic programming principles work
- Spanning tree: Prim's (and Kruskal's) algorithm transferable, i.e. greedy algorithm works
- Moderate problems: Two phase method works well, ranking algorithm required (assignment, network flow)
- Difficult problems: Scalarization and branch and bound

Conclusion

- MOCO problems as special multiobjective integer programmes
- Distinguish supported and non-supported efficient solutions
- MOCO problems are NP-hard, #P-hard, intractable
- MOCO problems can be solvable in practice
- Shortest path: Label setting and correcting algorithms transferable, i.e. dynamic programming principles work
- Spanning tree: Prim's (and Kruskal's) algorithm transferable, i.e. greedy algorithm works
- Moderate problems: Two phase method works well, ranking algorithm required (assignment, network flow)
- Difficult problems: Scalarization and branch and bound

Conclusion

- MOCO problems as special multiobjective integer programmes
- Distinguish supported and non-supported efficient solutions
- MOCO problems are NP-hard, #P-hard, intractable
- MOCO problems can be solvable in practice
- Shortest path: Label setting and correcting algorithms transferable, i.e. dynamic programming principles work
- Spanning tree: Prim's (and Kruskal's) algorithm transferable, i.e. greedy algorithm works
- Moderate problems: Two phase method works well, ranking algorithm required (assignment, network flow)
- Difficult problems: Scalarization and branch and bound

Conclusion

- MOCO problems as special multiobjective integer programmes
- Distinguish supported and non-supported efficient solutions
- MOCO problems are NP-hard, #P-hard, intractable
- MOCO problems can be solvable in practice
- Shortest path: Label setting and correcting algorithms transferable, i.e. dynamic programming principles work
- Spanning tree: Prim's (and Kruskal's) algorithm transferable, i.e. greedy algorithm works
- Moderate problems: Two phase method works well, ranking algorithm required (assignment, network flow)
- Difficult problems: Scalarization and branch and bound

Conclusion

- MOCO problems as special multiobjective integer programmes
- Distinguish supported and non-supported efficient solutions
- MOCO problems are NP-hard, #P-hard, intractable
- MOCO problems can be solvable in practice
- Shortest path: Label setting and correcting algorithms transferable, i.e. dynamic programming principles work
- Spanning tree: Prim's (and Kruskal's) algorithm transferable, i.e. greedy algorithm works
- Moderate problems: Two phase method works well, ranking algorithm required (assignment, network flow)
- Difficult problems: Scalarization and branch and bound

Conclusion

- MOCO problems as special multiobjective integer programmes
- Distinguish supported and non-supported efficient solutions
- MOCO problems are NP-hard, #P-hard, intractable
- MOCO problems can be solvable in practice
- Shortest path: Label setting and correcting algorithms transferable, i.e. dynamic programming principles work
- Spanning tree: Prim's (and Kruskal's) algorithm transferable, i.e. greedy algorithm works
- Moderate problems: Two phase method works well, ranking algorithm required (assignment, network flow)
- Difficult problems: Scalarization and branch and bound