Solving "Easy" Multi-objective Combinatorial Optimisation Problems

Andrea Raith and Matthias Ehrgott

Department of Engineering Science, The University of Auckland

OptALI Summer School, Auckland, 14-18 Feb 2011

Outline

Strategy: Extend Exisiting Algorithms

Straight-Forward Extension of the Single-Objective Algorithm Possible for Simple Problems

- Multi-objective Shortest Path
- Multi-objective Spanning Tree

Multi-objective Shortest Path Problems

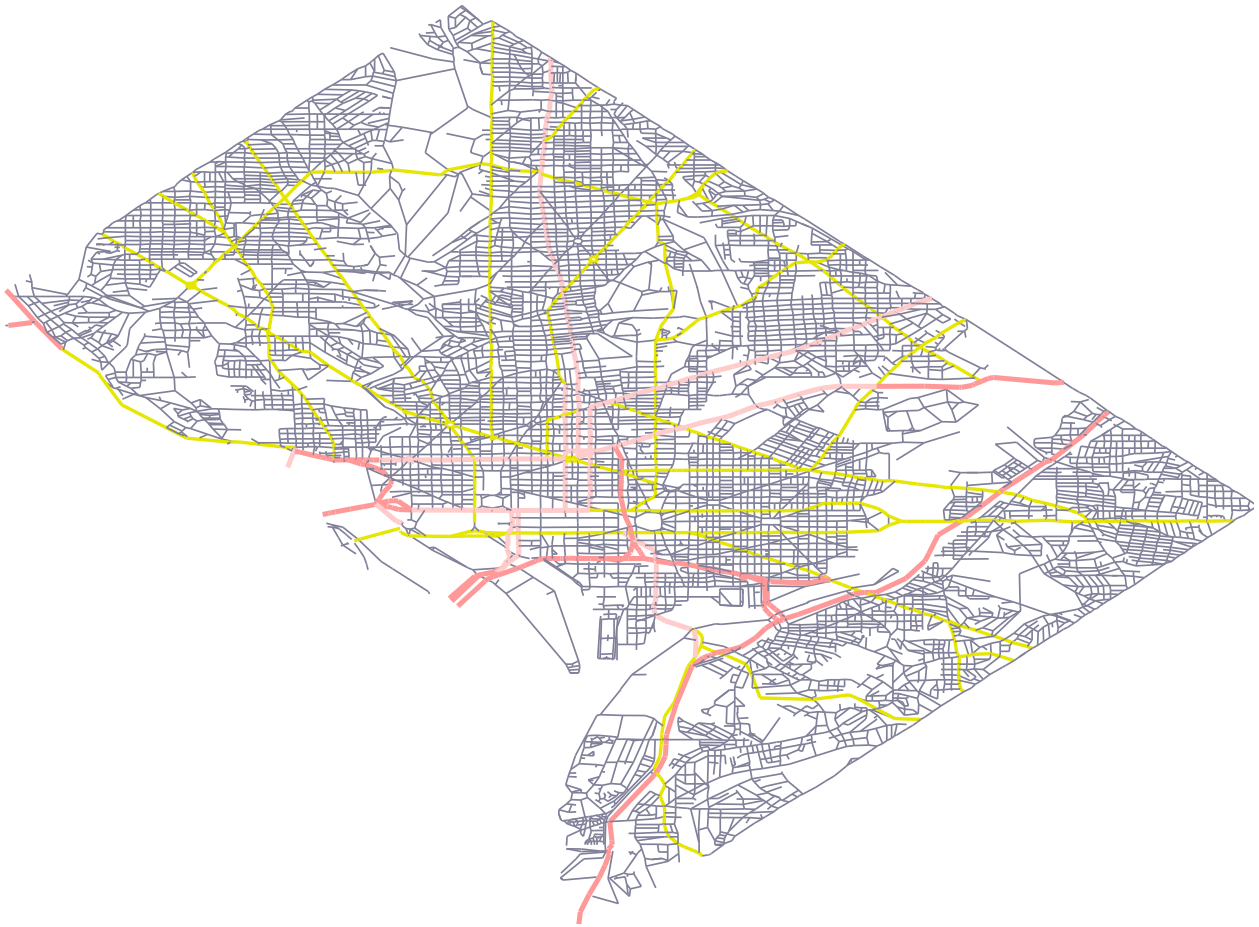
Definition

- $G = (N, A)$ a *directed network* with nodes $N = \{1, \ldots, n\}$ and arcs $A \subseteq N \times N$
- Pos. costs $c_{ij} = (c_{ij}^1, \ldots, c_{ij}^q) \in \mathbb{N}^q$ for each $(i, j) \in A$, $q \geq 2$
- *Path* in G from node $s \in N$ to node $t \in N$ is sequence $\{(s = i_0, i_1), (i_1, i_2), \ldots, (i_{l-1}, i_l = t)\}$ of arcs in A
- $\mathcal{P}_{st}$: Set of paths connecting s, t

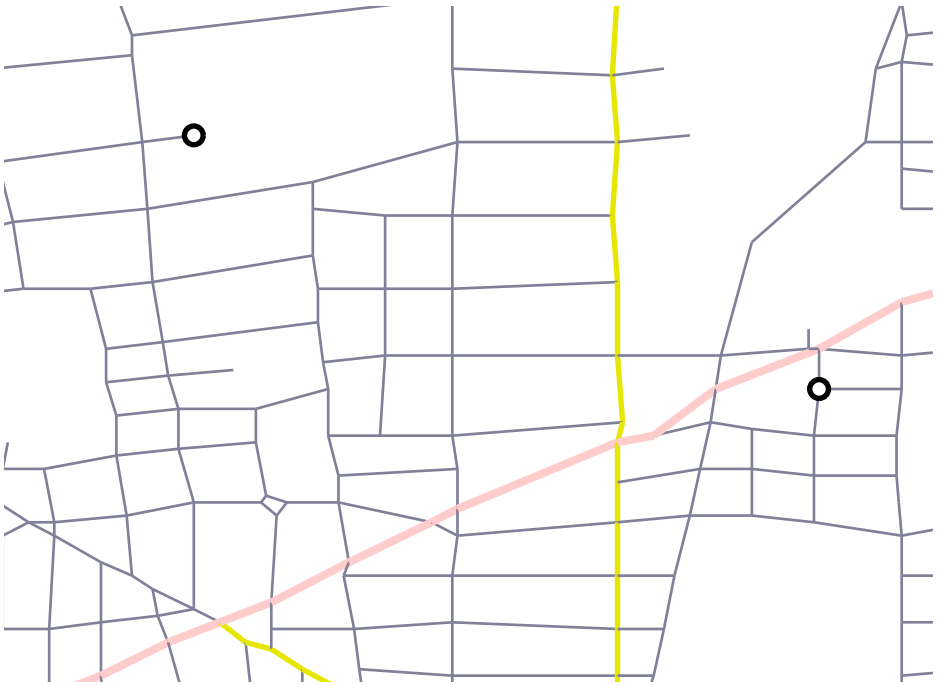
Multi-objective Shortest Path Problem

$$\begin{aligned} \min \quad & z(p) = \begin{cases} z_1(p) = \sum_{(i,j)\in p} c_{ij}^1 \\ \ldots \\ z_q(p) = \sum_{(i,j)\in p} c_{ij}^q \end{cases} \\ \text{s.t.} \quad & p \in \mathcal{P}_{st} \end{aligned}$$

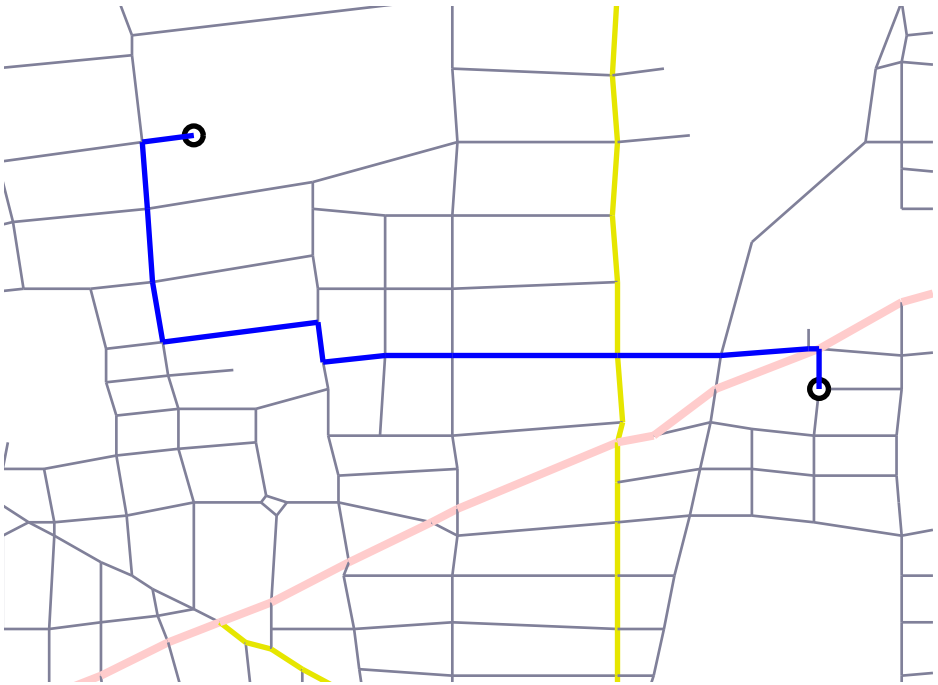
An Example in a Road Network of Washington DC

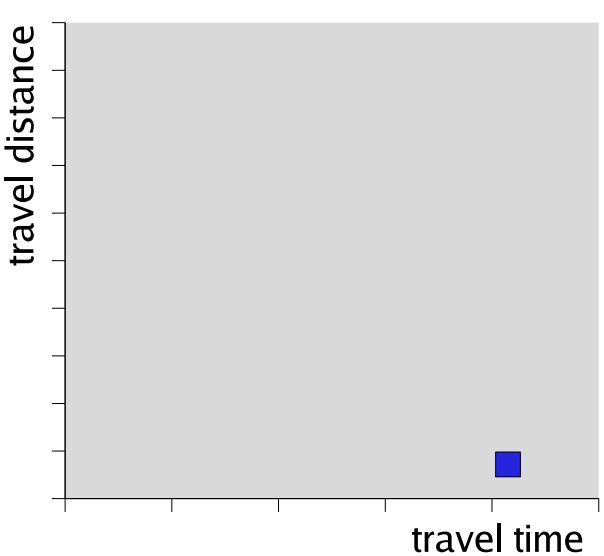


An Example in a Road Network of Washington DC

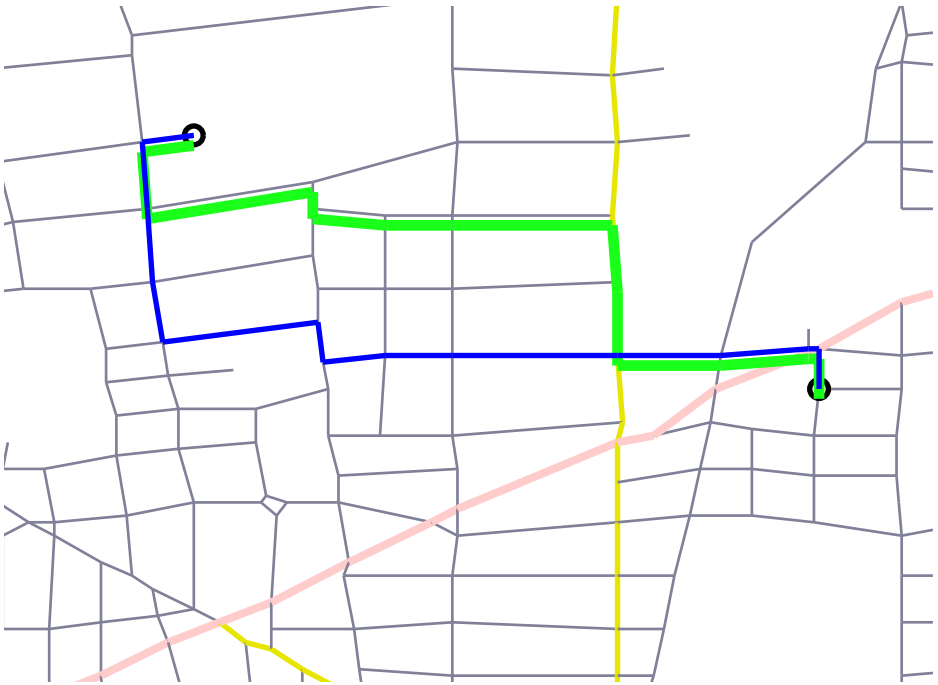


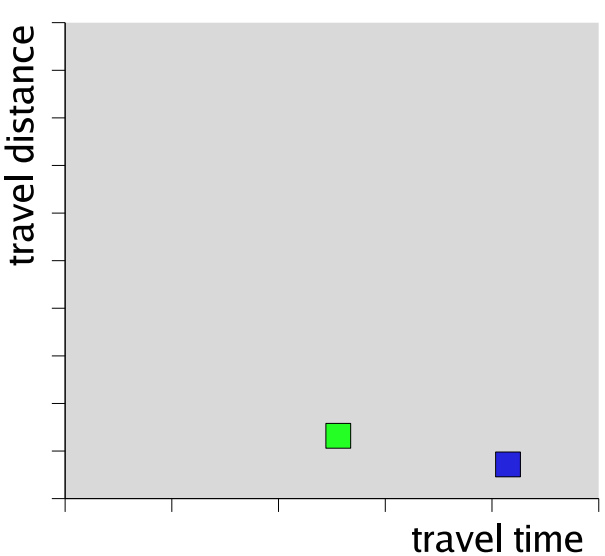
An Example in a Road Network of Washington DC



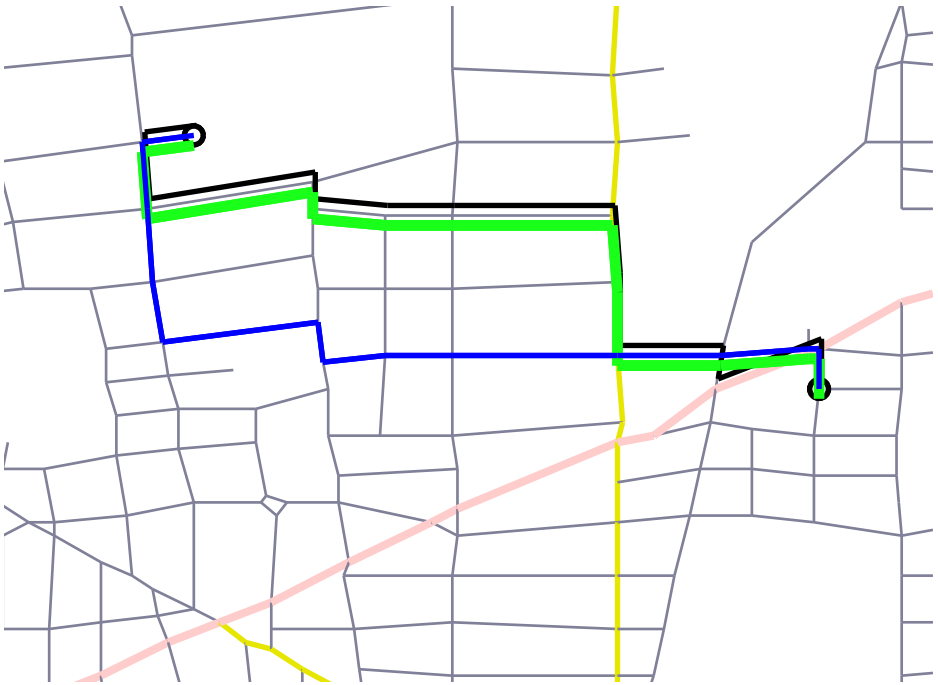


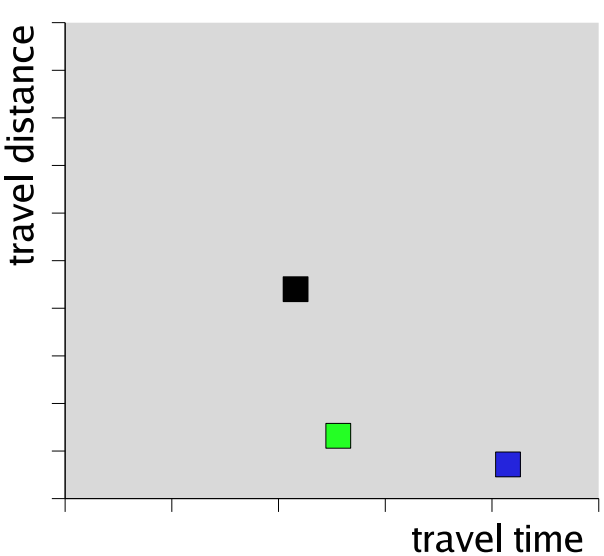
An Example in a Road Network of Washington DC



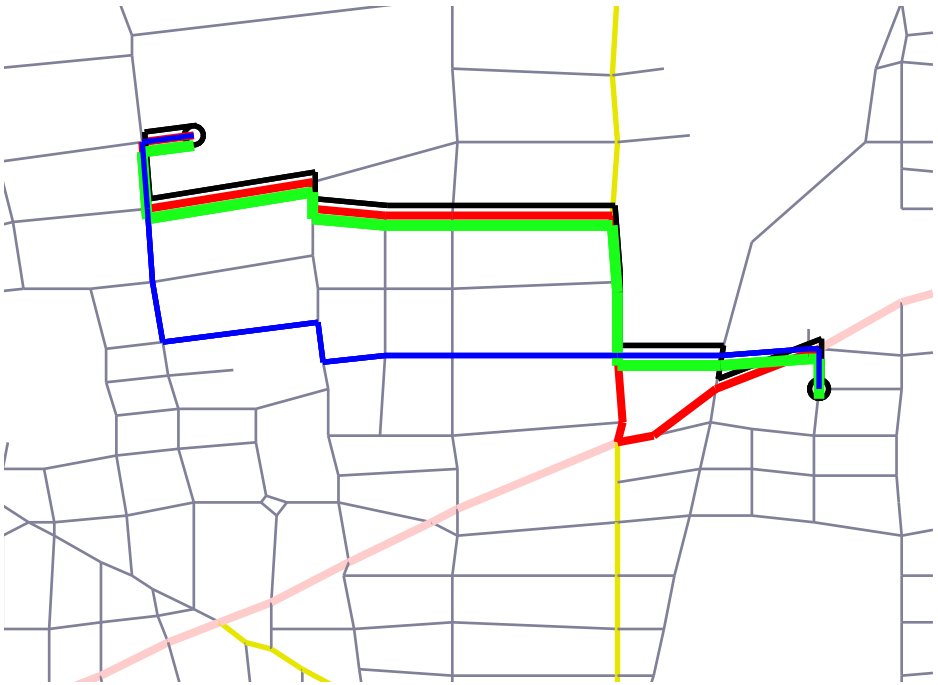


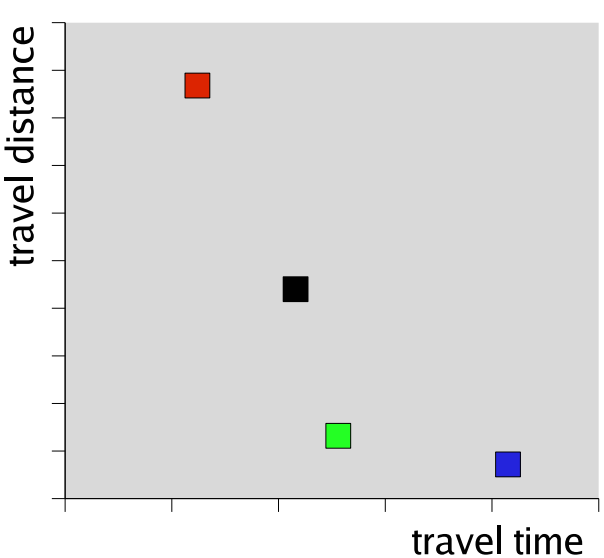
An Example in a Road Network of Washington DC



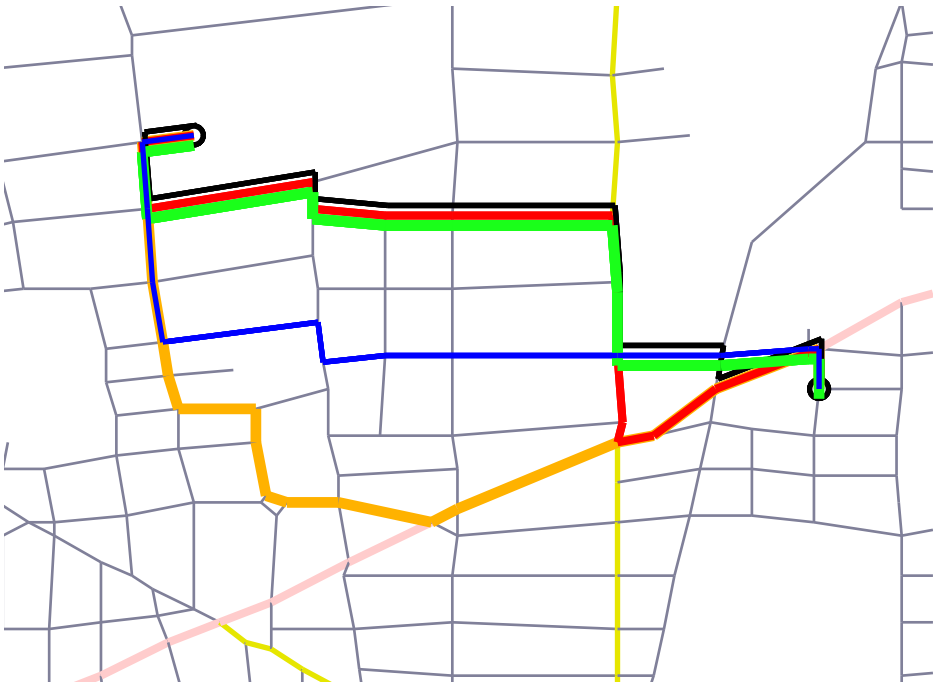


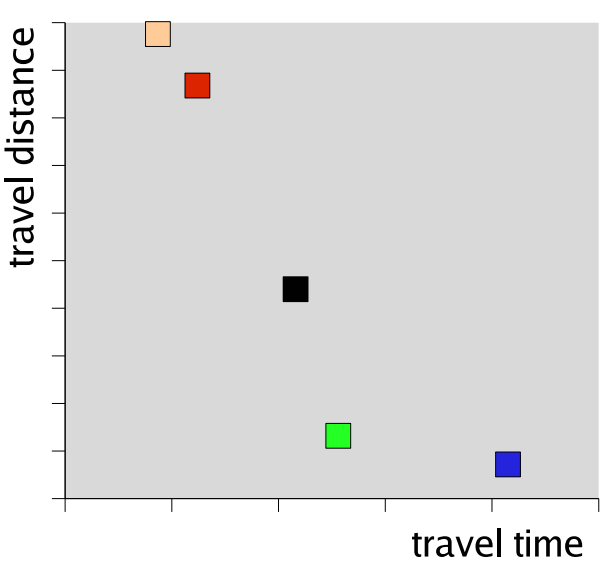
An Example in a Road Network of Washington DC





An Example in a Road Network of Washington DC





Examples of Multi-objective Shortest Path Problems

- Shipping of hazardous materials
- Routing airplanes through convective weather
- Scheduling of earth-observing satellites
- Public transport journey planner

Solving Shortest Path Problems: Label Setting

Label Setting Algorithm (Dijkstra)

1: {Find shortest path from s to t in G with cost c.}
2: label $l_s = (0)$ at s, other nodes un-labeled.
3: $L = \{s\}$ {nodes with tent. labels}, $M = \emptyset$ {fixed nodes}
4: **while** $L \neq \emptyset$ and $t \notin M$ **do**
5: Select $i \in L$ with $l_i = \min\{l_j : j \in L\}$
6: **for all** $(i, j) \in \mathcal{A}$ **do**
7: **if** $j \notin L \cup M$ **then**
8: create $l_j = l_i + c_{ij}$ and $L = L \cup \{j\}$
9: **else if** $l_i + c_{ij} < l_j$ **then**
10: set $l_j = l_i + c_{ij}$ {Shorter path s to j via i}
11: **end if**
12: **end for**
13: $L = L \setminus \{i\}, M = M \cup \{i\}$ {move i from L to M}
14: **end while**

Label Setting for Multi-objective Shortest Path Problems

Main Differences

single-objective	multi-objective
One path s to i, i.e. one label per node	multiple efficient paths from s to each i.
Selection of min cost unmarked label	selection of label that cannot be dominated, e.g. lex-min, min sum.
Comparison of pairs of labels $l_i + c_{ij} < l_j$	Merging of extended label $l_i + c_{ij}$ into set of labels at node j.
Can stop algorithm once $t \in M$	Can only stop when $L = \emptyset$

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s to t in G with cost c.}
label $l_s = (0)$ at s, other nodes un-labeled.
$L = \{s\}$ {nodes with tent. labels}, $M_i = \emptyset$ {fixed nodes},
while $L \neq \emptyset$ and $t \notin M$ **do**
  Select $i \in L$ with $l_i = \min\{l_j : j \in L\}$
  for all $(i, j) \in \mathcal{A}$ **do**
    if $j \notin L \cup M$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost c.}
label $l_s = (0)$ at s, other nodes un-labeled.
$L = \{s\}$ {nodes with tent. labels}, $M_i = \emptyset$ {fixed nodes},
while $L \neq \emptyset$ and $t \notin M$ **do**
  Select $i \in L$ with $l_i = \min\{l_j : j \in L\}$
  for all $(i, j) \in \mathcal{A}$ **do**
    if $j \notin L \cup M$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0)$ at s, other nodes un-labeled.
$L = \{s\}$ {nodes with tent. labels}, $M_i = \emptyset$ {fixed nodes},
while $L \neq \emptyset$ and $t \notin M$ **do**
  Select $i \in L$ with $l_i = \min\{l_j : j \in L\}$
  for all $(i, j) \in \mathcal{A}$ **do**
    if $j \notin L \cup M$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L = \{s\}$ {nodes with tent. labels}, $M_i = \emptyset$ {fixed nodes},
while $L \neq \emptyset$ and $t \notin M$ **do**
  Select $i \in L$ with $l_i = \min\{l_j : j \in L\}$
  for all $(i, j) \in \mathcal{A}$ **do**
    if $j \notin L \cup M$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed nodes},
while $L \neq \emptyset$ and $t \notin M$ **do**
  Select $i \in L$ with $l_i = \min\{l_j : j \in L\}$
  for all $(i, j) \in \mathcal{A}$ **do**
    if $j \notin L \cup M$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed labels},
while $L \neq \emptyset$ and $t \notin M$ **do**
  Select $i \in L$ with $l_i = \min\{l_j : j \in L\}$
  for all $(i, j) \in \mathcal{A}$ **do**
    if $j \notin L \cup M$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed labels},
while $\bigcup_{i \in \mathcal{N}} L_i \neq \emptyset$ **do**
    Select $i \in L$ with $l_i = \min\{l_j : j \in L\}$
    for all $(i, j) \in \mathcal{A}$ **do**
        if $j \notin L \cup M$ **then**
            create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
        else if $l_i + c_{ij} < l_j$ **then**
            set $l_j = l_i + c_{ij}$
        end if
    end for
    $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed labels},
while $\bigcup_{i \in \mathcal{N}} L_i \neq \emptyset$ **do**
  Select $l_i \in L_i$ for some i guaranteed to remain non-dom.
  for all $(i, j) \in \mathcal{A}$ **do**
    if $j \notin L \cup M$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed labels},
while $\bigcup_{i \in \mathcal{N}} L_i \neq \emptyset$ **do**
  Select $l_i \in L_i$ for some i guaranteed to remain non-dom.
  for all $(i, j) \in \mathcal{A}$ **do**
    if $j \notin L \cup M$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed labels},
while $\bigcup_{i \in \mathcal{N}} L_i \neq \emptyset$ **do**
  Select $l_i \in L_i$ for some i guaranteed to remain non-dom.
  for all $(i, j) \in \mathcal{A}$ **do**
    if $L_j \cup M_j = \emptyset$ **then**
      create $l_j = l_i + c_{ij}$ and $L_j = L_j \cup \{l_j\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed labels},
while $\bigcup_{i \in \mathcal{N}} L_i \neq \emptyset$ **do**
  Select $l_i \in L_i$ for some i guaranteed to remain non-dom.
  for all $(i, j) \in \mathcal{A}$ **do**
    if $L_j \cup M_j = \emptyset$ **then**
      create $L_j = \{l_i + c_{ij}\}$
    else if $l_i + c_{ij} < l_j$ **then**
      set $l_j = l_i + c_{ij}$
    end if
  end for
  $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed labels},
while $\bigcup_{i \in \mathcal{N}} L_i \neq \emptyset$ **do**
    Select $l_i \in L_i$ for some i guaranteed to remain non-dom.
    for all $(i, j) \in \mathcal{A}$ **do**
        if $L_j \cup M_j = \emptyset$ **then**
            create $L_j = \{l_i + c_{ij}\}$
        else
            Merge $L_j \cup M_j$ and $l_i + c_{ij}$
        end if
    end for
    $L = L \setminus \{i\}, M = M \cup \{i\}$
end while

Solving Multi-objective Shortest Path Problems

Multi-objective Label Setting Algorithm [e.g. 9; 10; 17; 18]

{Find shortest path from s ~~to t~~ in G with cost $(c^1, \ldots, c^q)$.}
label $l_s = (0, \ldots, 0)$ at s, other nodes un-labeled.
$L_s = \{l_s\}, L_i = \emptyset$ {tent. labels at i}, $M_i = \emptyset$ {fixed labels},
while $\bigcup_{i \in \mathcal{N}} L_i \neq \emptyset$ **do**
  Select $l_i \in L_i$ for some i guaranteed to remain non-dom.
  for all $(i, j) \in \mathcal{A}$ **do**
    if $L_j \cup M_j = \emptyset$ **then**
      create $L_j = \{l_i + c_{ij}\}$
    else
      Merge $L_j \cup M_j$ and $l_i + c_{ij}$
    end if
  end for
  $L_i = L_i \setminus \{l_i\}, M_i = M_i \cup \{l_i\}$
end while

Label Correcting

Single-objective label correcting

- Nodes *marked* when their labels are created or change
- Keep list of marked nodes, e.g. in FIFO order
- Re-visit marked nodes, extend label along outgoing arcs and un-mark them
- Continue until no more marked labels exist

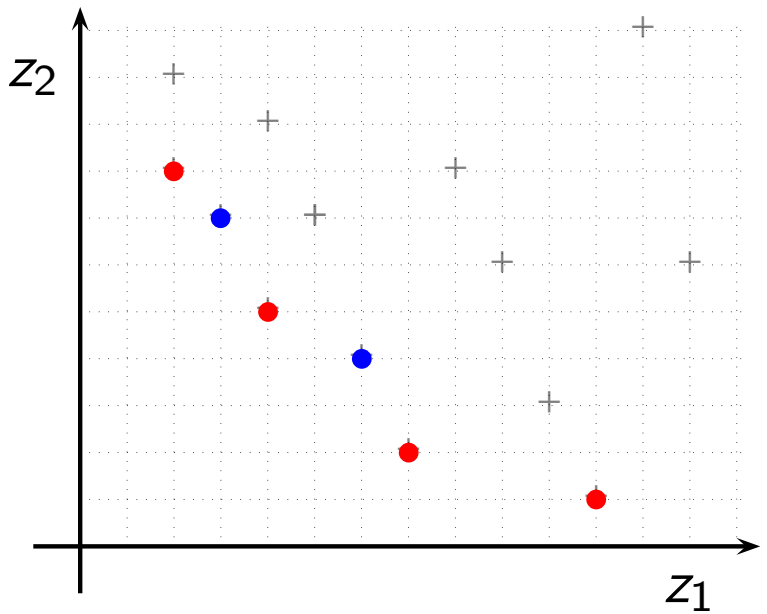
Multi-objective label correcting [2; 16; 12]

- Similar to the above: node-selection
- Alternative: label-selection (considering individual labels)

Advantages: No need to extract "minimum" label; For node-selection merge *sets* of labels, which may be faster

Outline

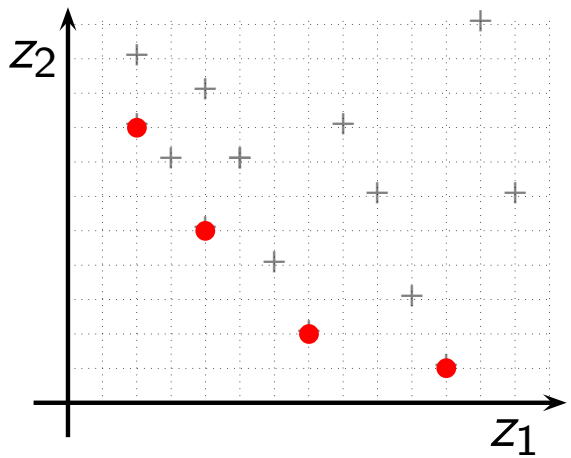
Two Phase Method to Identify "Easy" and "Difficult" Solutions Separately



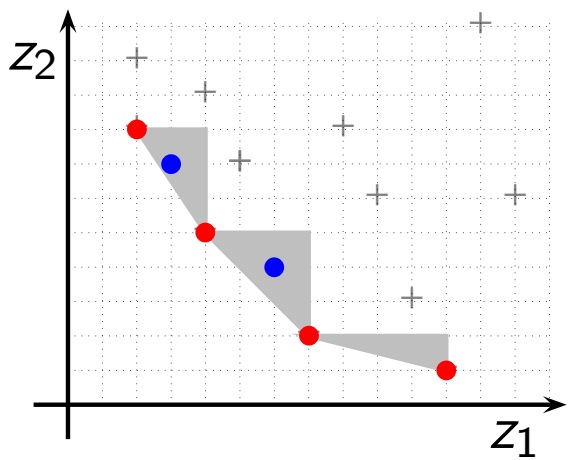
Easy: supported extreme points (red)
Difficult: non-supported and non-extreme supported points (blue)

The Two Phase Method [19]

In phase 1 all extreme supported solutions are computed



In phase 2 all other solutions are computed



Bi-objective Integer Minimum Cost Flow (BIMFC) Problem

Definition

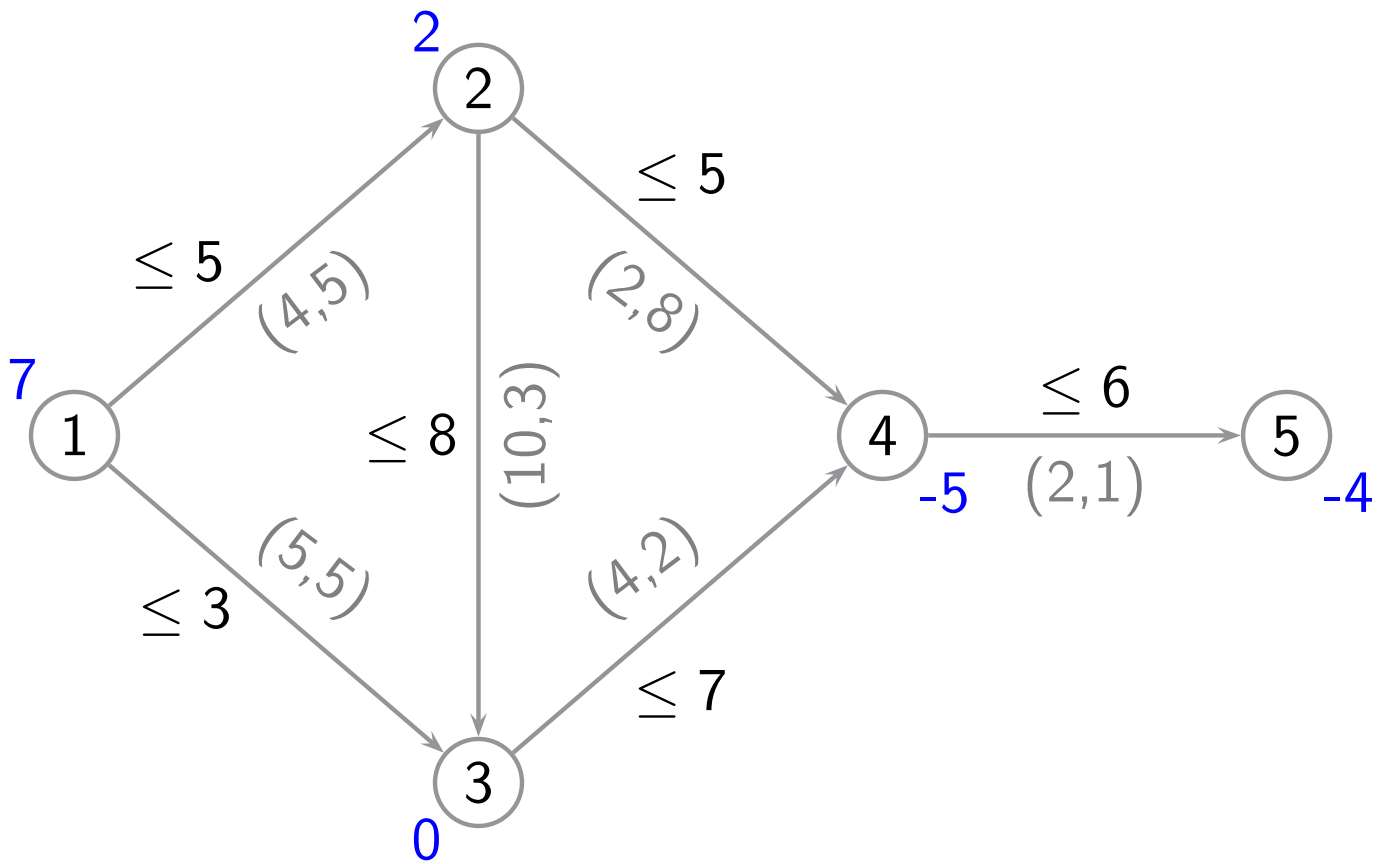
- $G = (N, A)$ with nodes $N = \{1, \ldots, n\}$ and arcs $A \subseteq N \times N$
- costs $c_{ij} = (c_{ij}^1, c_{ij}^2) \in \mathbb{Z}^2$ for each $(i, j) \in A$
- lower and upper bounds $l_{ij} = 0$ and $u_{ij} > 0$ for each arc (i, j)
- flow balance b_i for each node i

BIMFC

$$\begin{aligned}
\min \quad & z(x) = \begin{cases} z_1(x) = \sum_{(i,j)\in A} c_{ij}^1 x_{ij} \\ z_2(x) = \sum_{(i,j)\in A} c_{ij}^2 x_{ij} \end{cases} \\
\text{s.t.} \quad & \sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ij} = b_i, \forall i \in N \\
& 0 \leq x_{ij} \leq u_{ij}, \forall (i,j) \in A \\
& x_{ij} \text{ integer}
\end{aligned}$$

Bi-objective Network Flow - Example

Bi-objective network flow problem: Shipping a single commodity through the network



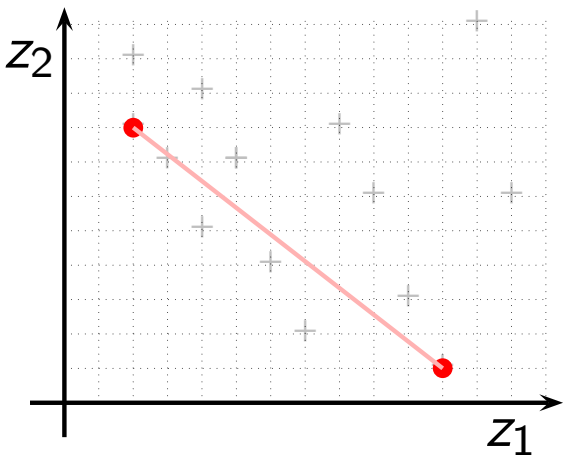
Solving Phase 1

In Phase 1 supported (extreme) solutions need to be computed.

1. Dichotomic Appoach [e.g. 1; 4; 6]:
   - Supported (extreme) solutions can be obtained by repeatedly solving a single-objective version of the problem at hand with different weighting factors.
   - Dichotomic strategy of selecting weighting factors
2. Problem specific approach:
   - Modify existing solution algorithm for specific problem to obtain supported (extreme) solutions.
   - Example: Network simplex for BIMCF problems [e.g. 12; 13]

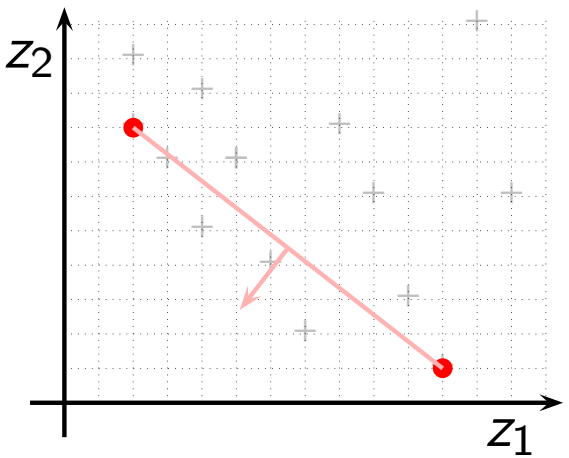
Phase 1 - Dichotomic Approach

- Compute both $lex(2,1)$- and $lex(1,2)$-best solutions $\bar{x}, \hat{x}$ and determine weights $\lambda^1 = z_2(\hat{x}) - z_2(\bar{x})$ and $\lambda^2 = z_1(\bar{x}) - z_1(\hat{x})$



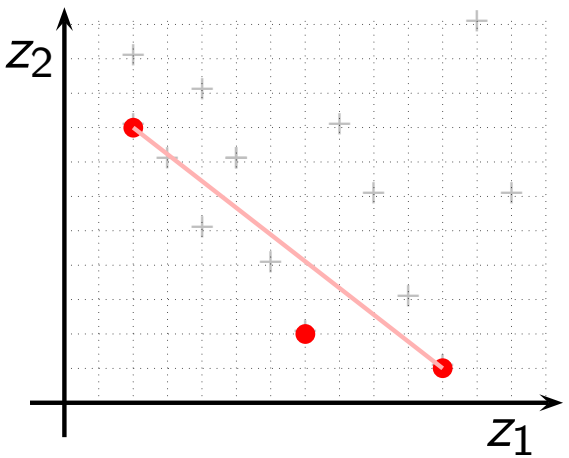
Phase 1 - Dichotomic Approach

- Compute both $lex(2,1)$- and $lex(1,2)$-best solutions $\bar{x}, \hat{x}$ and determine weights
  $\lambda^1 = z_2(\hat{x}) - z_2(\bar{x})$ and
  $\lambda^2 = z_1(\bar{x}) - z_1(\hat{x})$
- Determine optimal solution of problem with weighted sum objective $\min \lambda_1 z^1 + \lambda_2 z^2$.



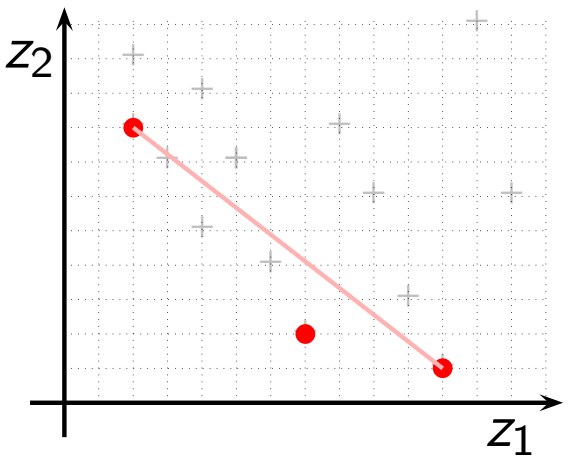
Phase 1 - Dichotomic Approach

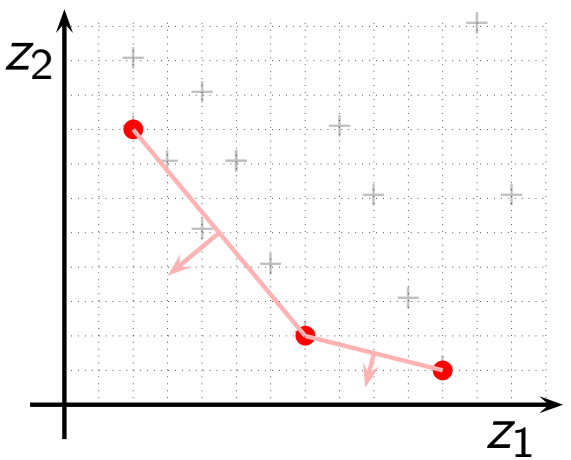
- Compute both $lex(2,1)$- and $lex(1,2)$-best solutions $\bar{x}, \hat{x}$ and determine weights
  $\lambda^1 = z_2(\hat{x}) - z_2(\bar{x})$ and
  $\lambda^2 = z_1(\bar{x}) - z_1(\hat{x})$
- Determine optimal solution of problem with weighted sum objective $\min \lambda_1 z^1 + \lambda_2 z^2$.



Phase 1 - Dichotomic Approach

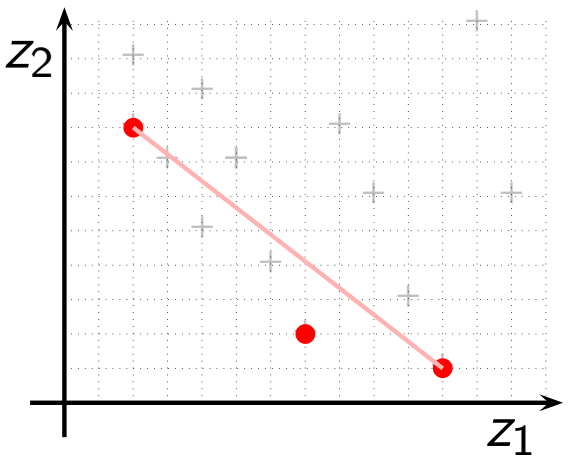
- Compute both $lex(2,1)$- and $lex(1,2)$-best solutions $\bar{x}, \hat{x}$ and determine weights $\lambda^1 = z_2(\hat{x}) - z_2(\bar{x})$ and $\lambda^2 = z_1(\bar{x}) - z_1(\hat{x})$
- Determine optimal solution of problem with weighted sum objective $\min \lambda_1 z^1 + \lambda_2 z^2$.
- Define two subproblems and find optimal solutions in each one.

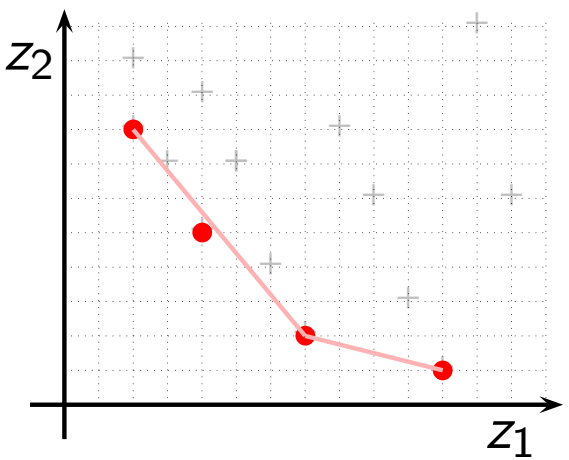




Phase 1 - Dichotomic Approach

- Compute both $lex(2,1)$- and $lex(1,2)$-best solutions $\bar{x}, \hat{x}$ and determine weights $\lambda^1 = z_2(\hat{x}) - z_2(\bar{x})$ and $\lambda^2 = z_1(\bar{x}) - z_1(\hat{x})$
- Determine optimal solution of problem with weighted sum objective $\min \lambda_1 z^1 + \lambda_2 z^2$.
- Define two subproblems and find optimal solutions in each one.
- etc.

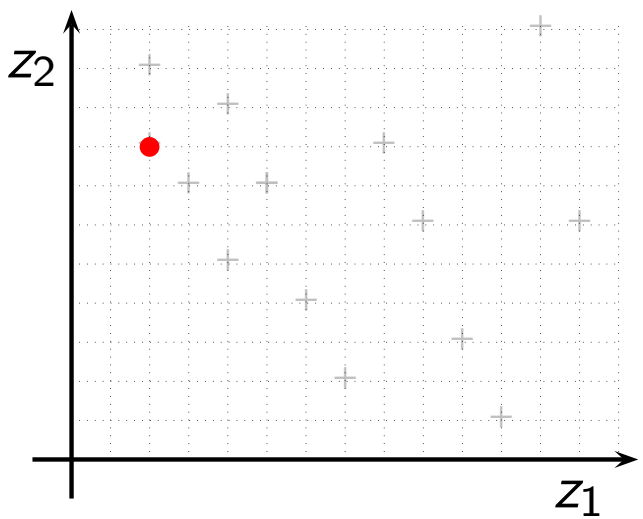




Phase 1 - Problem Specific for BIMCF

Adapt network simplex algorithm for BIMCF: Parametric network simplex [13–15].

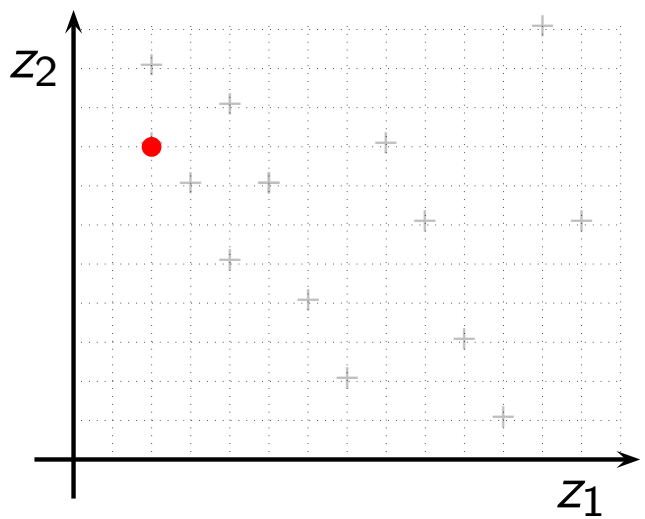
- Compute the $lex(1, 2)$-best solution



Phase 1 - Problem Specific for BIMCF

Adapt network simplex algorithm for BIMCF: Parametric network simplex [13–15].

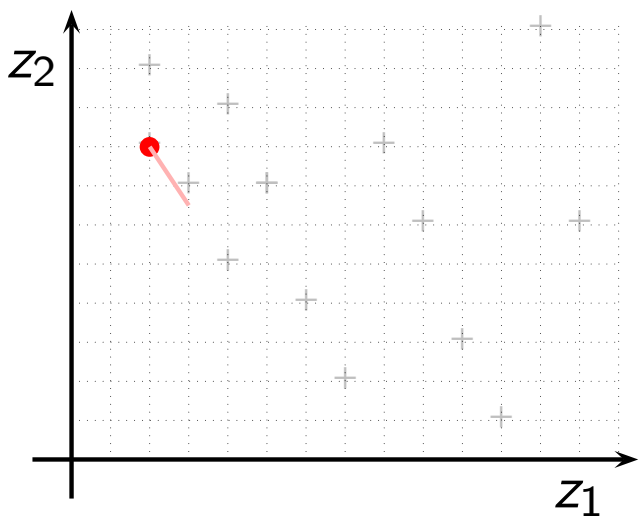
- Compute the $lex(1, 2)$-best solution
- Run the network simplex algorithm with the objective to find $lex(2, 1)$-best solution



Phase 1 - Problem Specific for BIMCF

Adapt network simplex algorithm for BIMCF: Parametric network simplex [13–15].

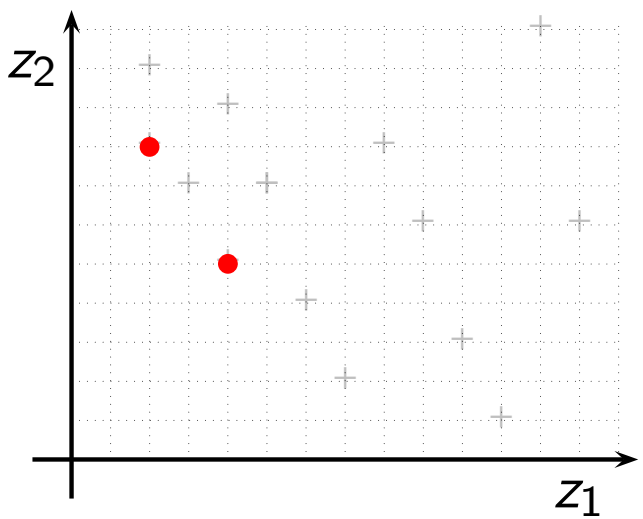
- Compute the $lex(1,2)$-best solution
- Run the network simplex algorithm with the objective to find $lex(2,1)$-best solution
  - choose basic entering arc with minimum ratio μ of deterioration of z_1 and improvement of z_2.



Phase 1 - Problem Specific for BIMCF

Adapt network simplex algorithm for BIMCF: Parametric network simplex [13–15].

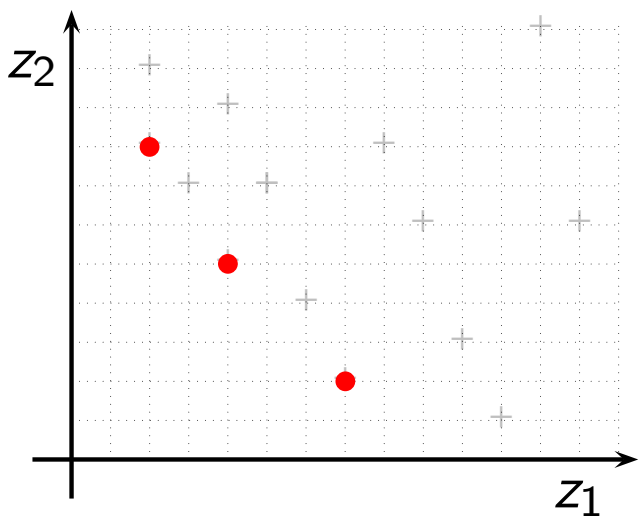
- Compute the $lex(1,2)$-best solution
- Run the network simplex algorithm with the objective to find $lex(2,1)$-best solution
  - choose basic entering arc with minimum ratio μ of deterioration of z_1 and improvement of z_2.
  - whenever μ changes, we have an extreme supported solution.



Phase 1 - Problem Specific for BIMCF

Adapt network simplex algorithm for BIMCF: Parametric network simplex [13–15].

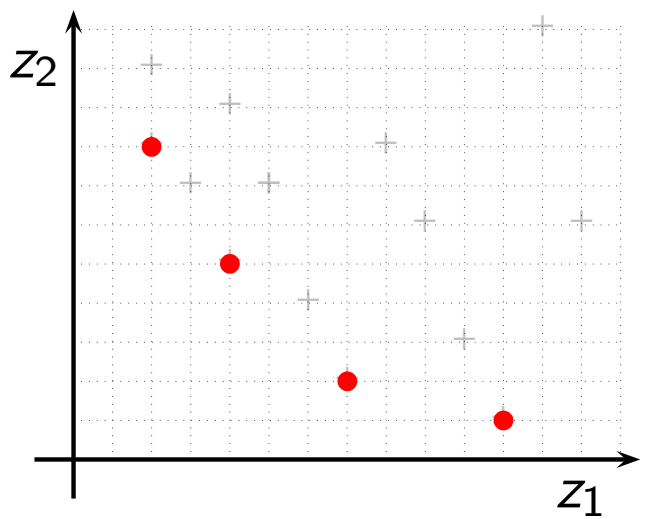
- Compute the $lex(1,2)$-best solution
- Run the network simplex algorithm with the objective to find $lex(2,1)$-best solution
  - choose basic entering arc with minimum ratio μ of deterioration of z_1 and improvement of z_2.
  - whenever μ changes, we have an extreme supported solution.



Phase 1 - Problem Specific for BIMCF

Adapt network simplex algorithm for BIMCF: Parametric network simplex [13–15].

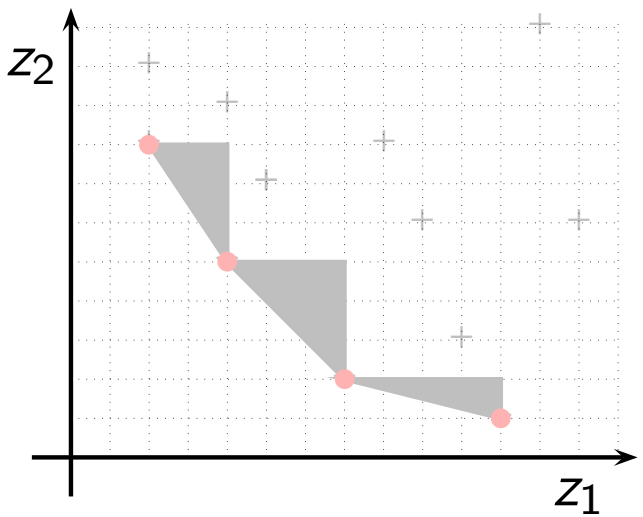
- Compute the $lex(1,2)$-best solution
- Run the network simplex algorithm with the objective to find $lex(2,1)$-best solution
  - choose basic entering arc with minimum ratio μ of deterioration of z_1 and improvement of z_2.
  - whenever μ changes, we have an extreme supported solution.
- until $lex(2,1)$-best solution is obtained.



Phase 2

Identify Remaining (Non-supported) Solutions

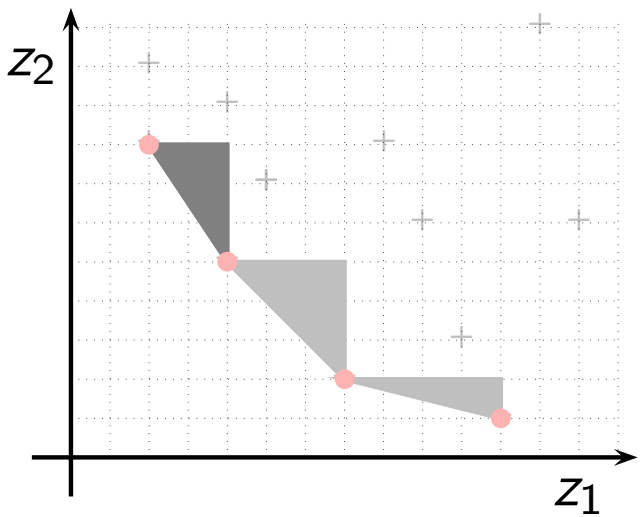
- Need to fill the gaps between solutions found in Phase 1.
- Restrict search to triangles defined by two neighbouring extreme non-domianted points
- Avoid breaking the problem structure, e.g. by introducting additional constraints.



Phase 2

Identify Remaining (Non-supported) Solutions

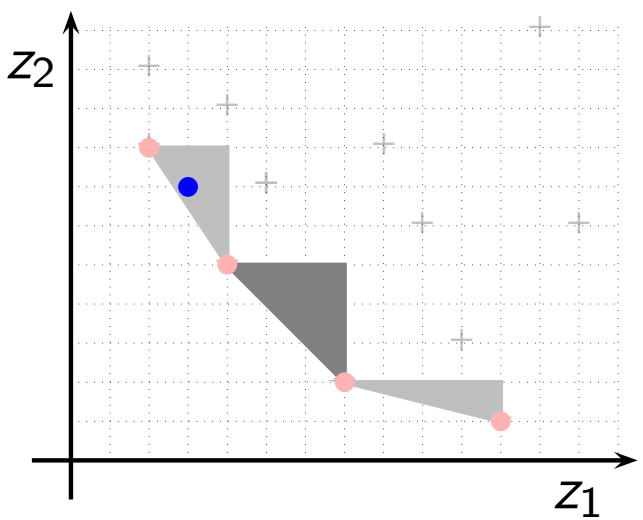
- Need to fill the gaps between solutions found in Phase 1.
- Restrict search to triangles defined by two neighbouring extreme non-domianted points
- Avoid breaking the problem structure, e.g. by introducting additional constraints.



Phase 2

Identify Remaining (Non-supported) Solutions

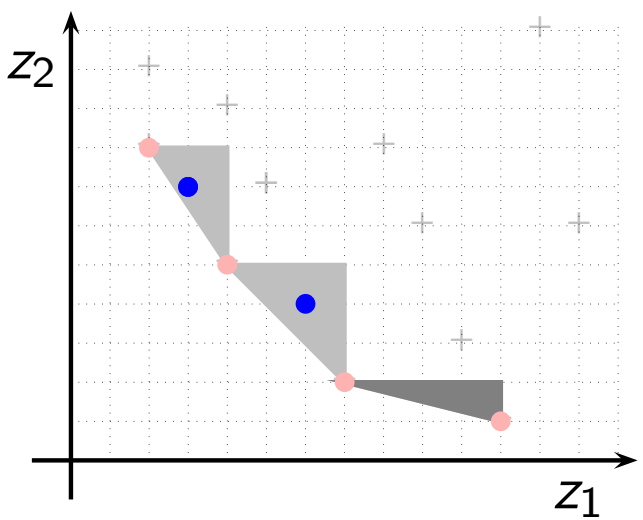
- Need to fill the gaps between solutions found in Phase 1.
- Restrict search to triangles defined by two neighbouring extreme non-domianted points
- Avoid breaking the problem structure, e.g. by introducting additional constraints.



Phase 2

Identify Remaining (Non-supported) Solutions

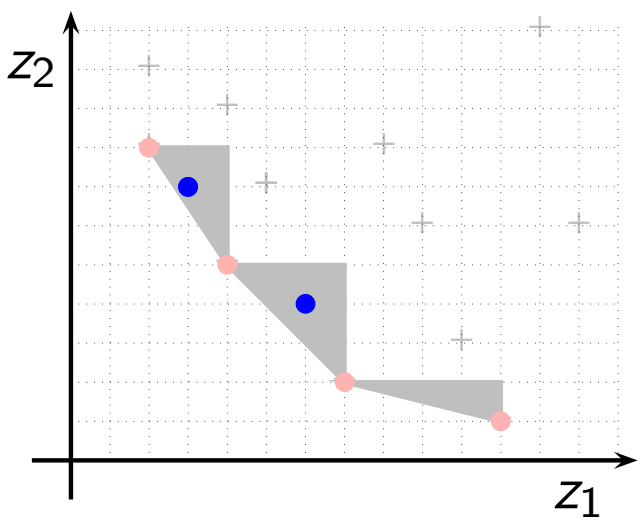
- Need to fill the gaps between solutions found in Phase 1.
- Restrict search to triangles defined by two neighbouring extreme non-domianted points
- Avoid breaking the problem structure, e.g. by introducting additional constraints.



Phase 2

Identify Remaining (Non-supported) Solutions

- Need to fill the gaps between solutions found in Phase 1.
- Restrict search to triangles defined by two neighbouring extreme non-domianted points
- Avoid breaking the problem structure, e.g. by introducting additional constraints.



Phase 2 - Ranking

Ranking

- Consider single objective problem with weighted sum objective

$$\min c_\lambda(x) = \lambda^1 z_1(x) + \lambda^2 z_2(x)$$

- Ranking algorithm computes $x^1, x^2, x^3, \ldots, x^k$

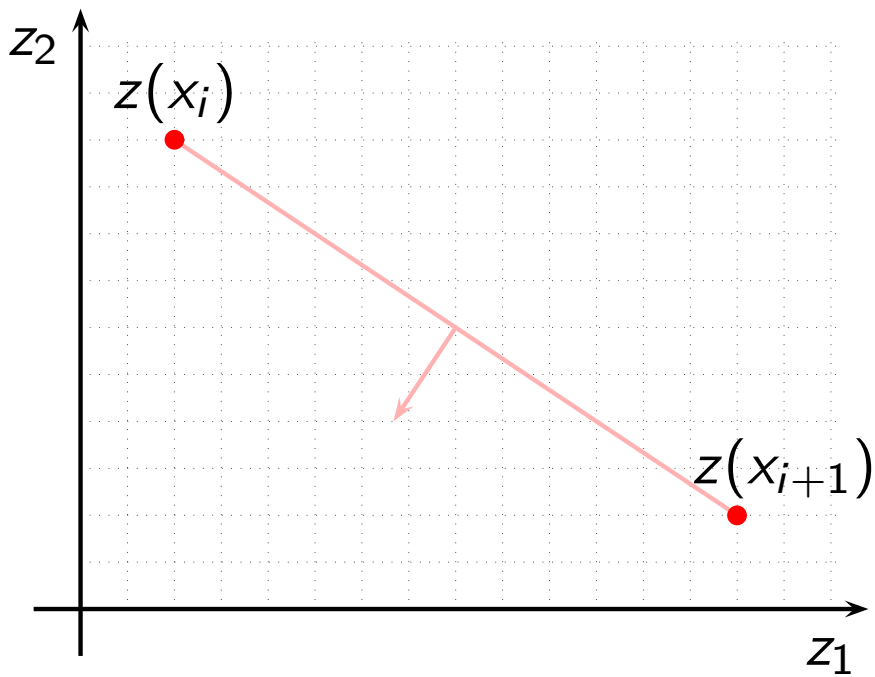
$$\text{with} \quad c_\lambda(x^1) \leqq c_\lambda(x^2) \leqq c_\lambda(x^3) \leqq \ldots \leqq c_\lambda(x^k).$$

- Continue computing x^{k+1} until it can be guaranteed that all efficient non-supported points are found.

Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

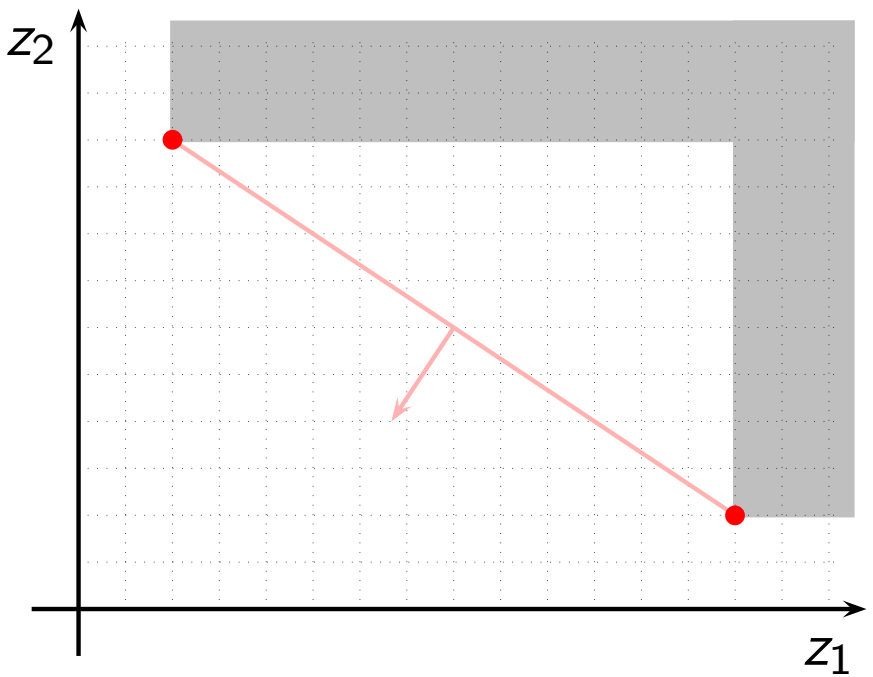
- Solve problem with objective c_λ



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

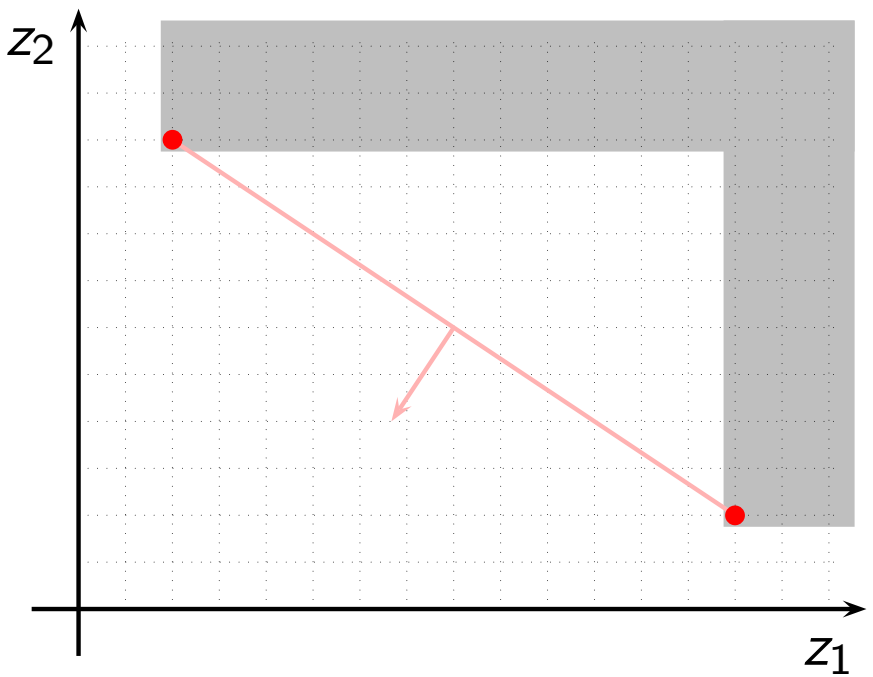
- Solve problem with objective c_λ
- Restrict to triangle



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

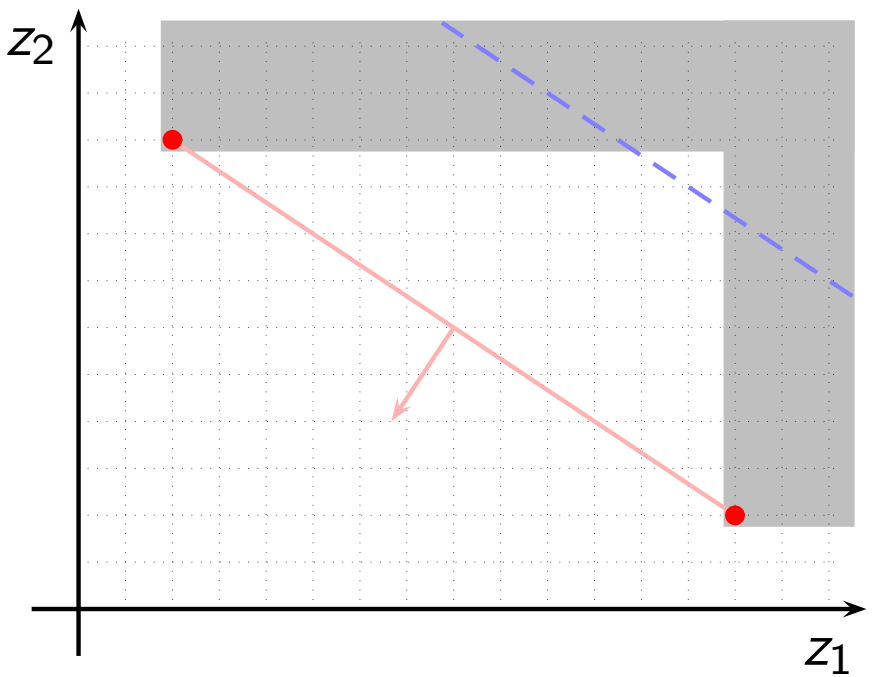
- Solve problem with objective c_λ
- Restrict to triangle



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

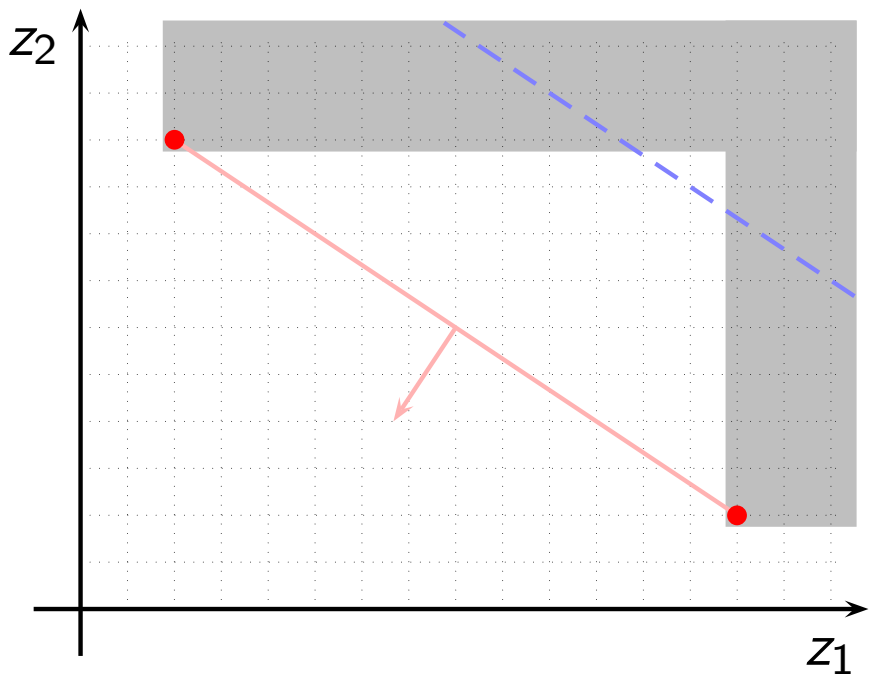
- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

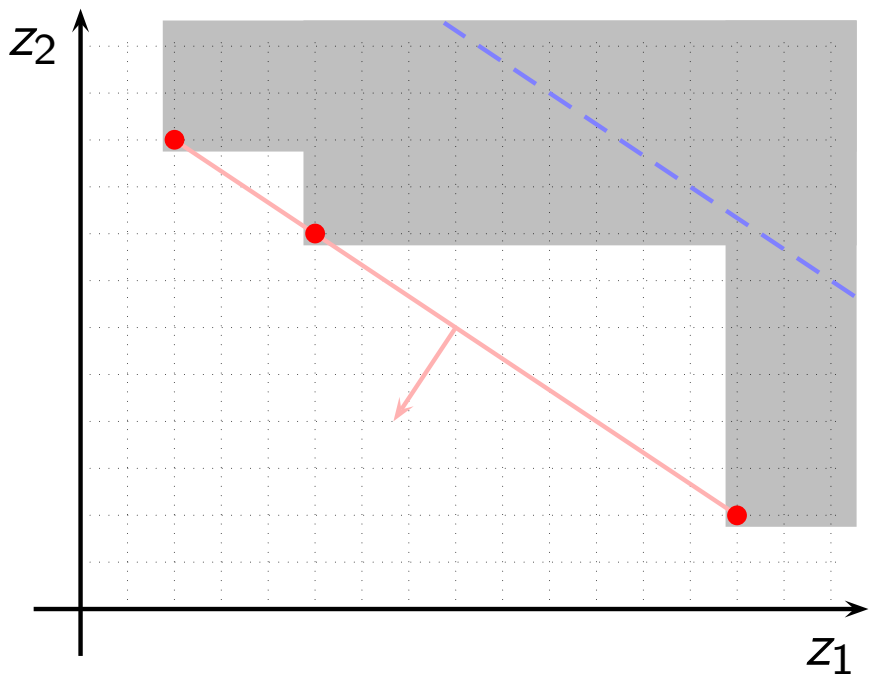
- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ
- Rank flows x while $c_\lambda(x) \leqq u_\lambda$
  If x is efficient: save it and update u_λ.



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

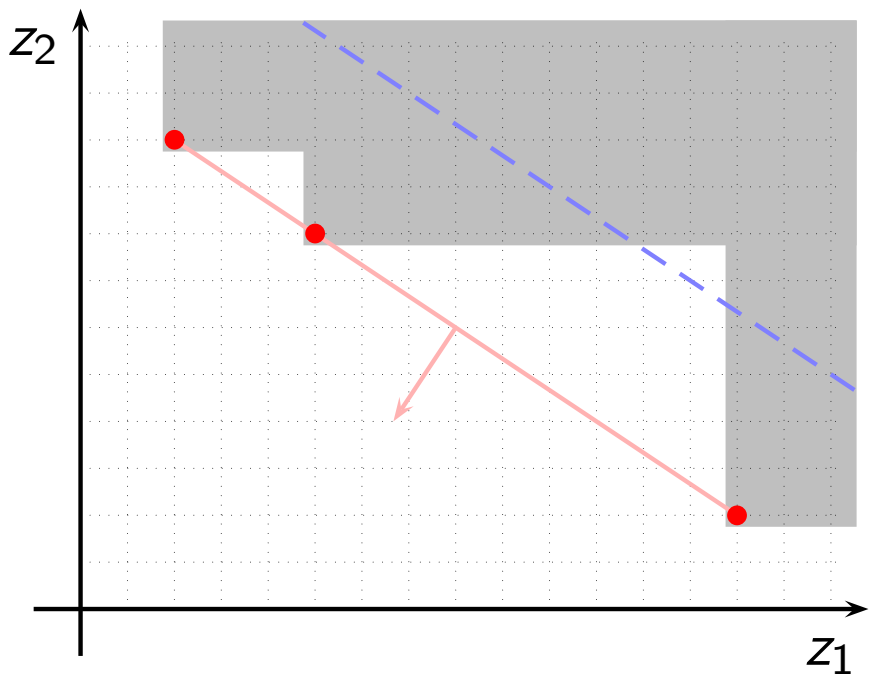
- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ
- Rank flows x while $c_\lambda(x) \leqq u_\lambda$
  If x is efficient: save it and update u_λ.



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

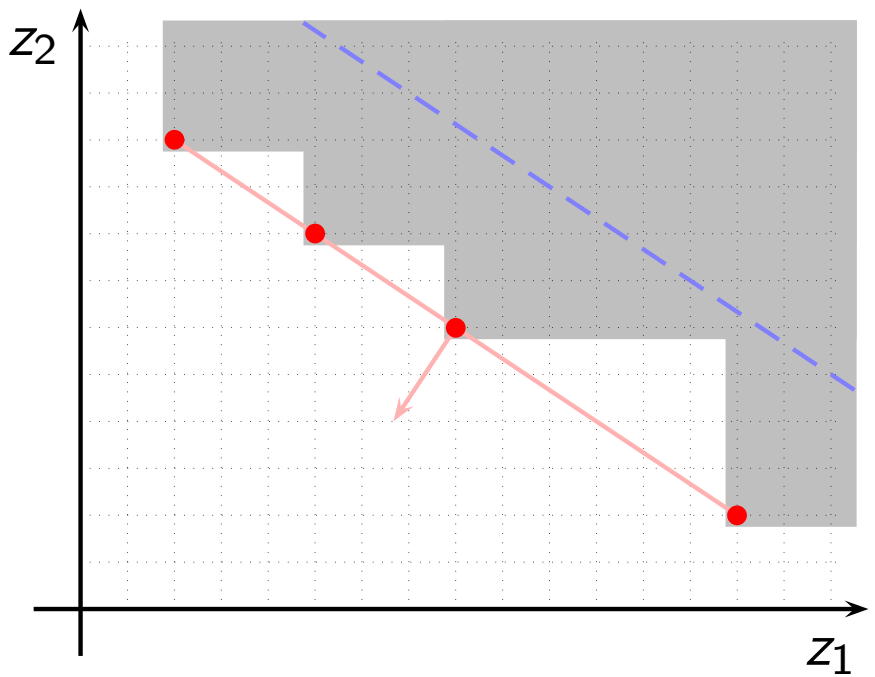
- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ
- Rank flows x while $c_\lambda(x) \leqq u_\lambda$
  If x is efficient: save it and update u_λ.



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

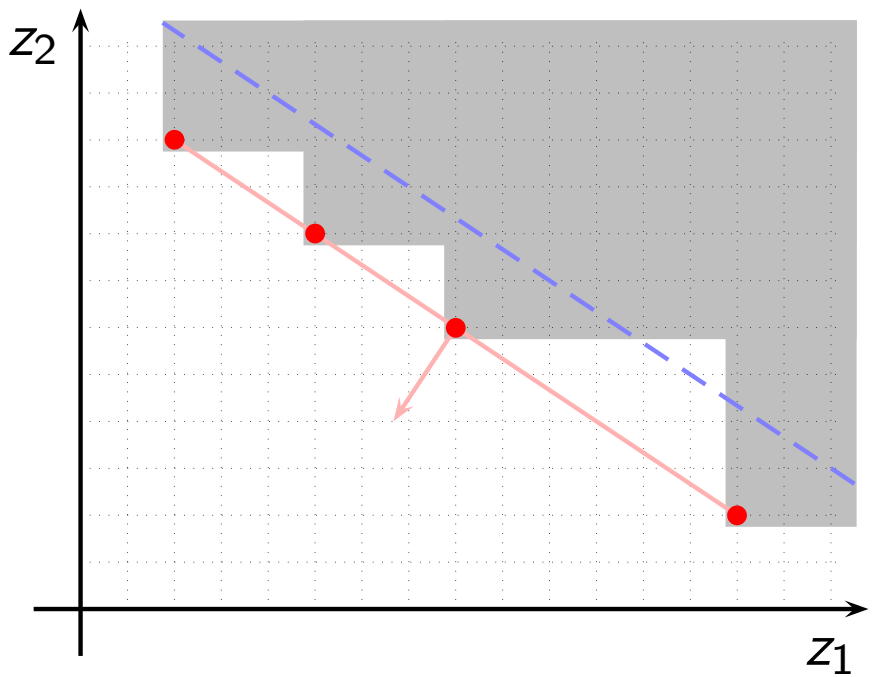
- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ
- Rank flows x while $c_\lambda(x) \leqq u_\lambda$
  If x is efficient: save it and update u_λ.



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

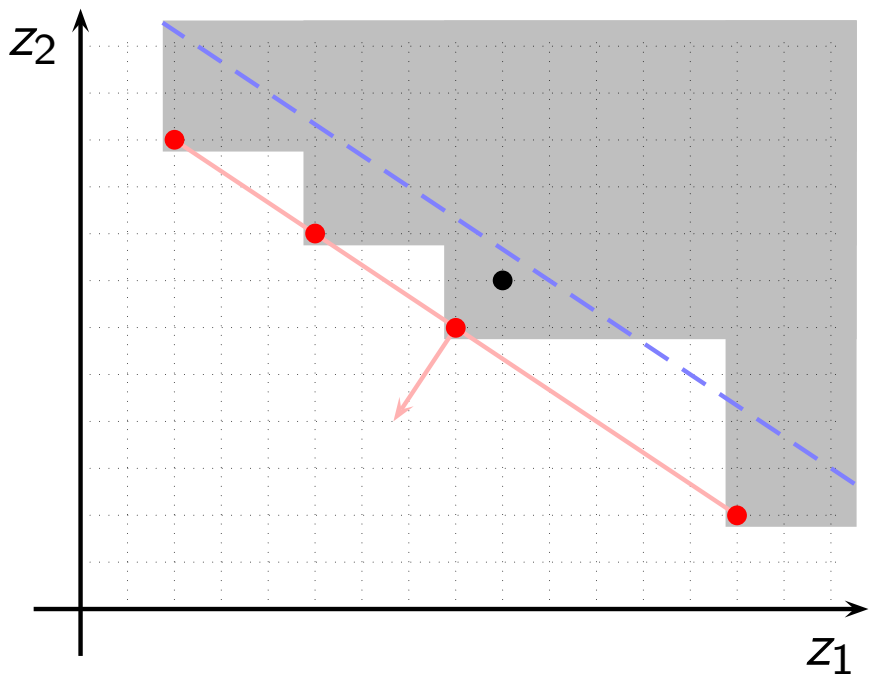
- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ
- Rank flows x while $c_\lambda(x) \leqq u_\lambda$
  If x is efficient: save it and update u_λ.



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

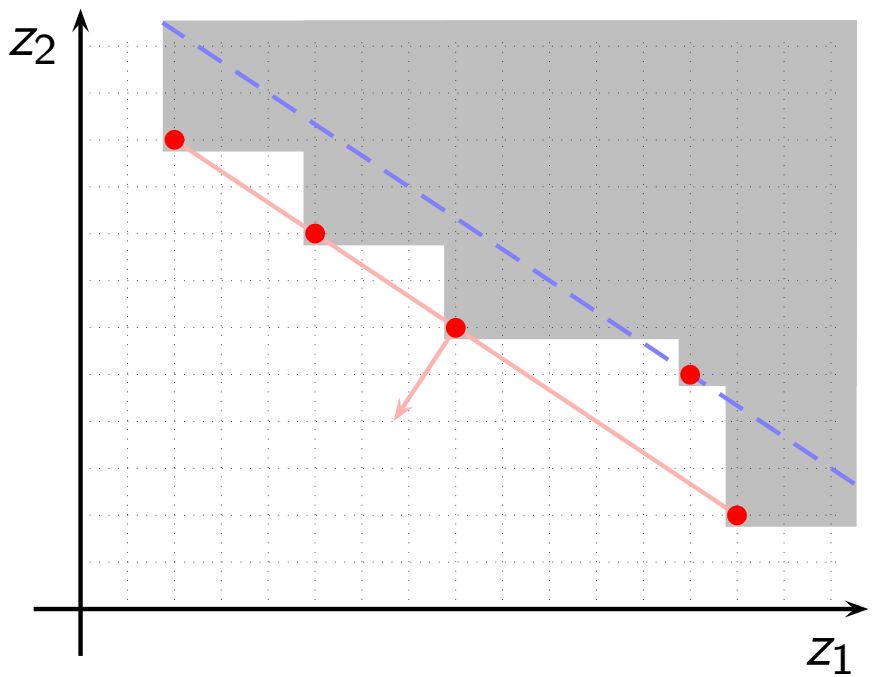
- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ
- Rank flows x while $c_\lambda(x) \leqq u_\lambda$
  If x is efficient: save it and update u_λ.



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

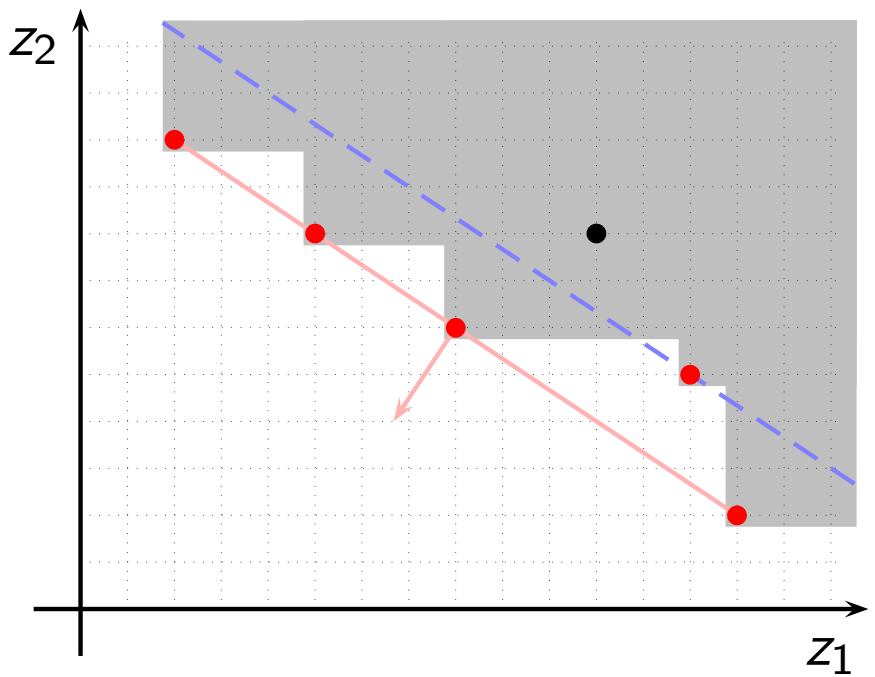
- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ
- Rank flows x while $c_\lambda(x) \leqq u_\lambda$
  If x is efficient: save it and update u_λ.



Ranking All Solutions in a Triangle: Illustration

In each triangle . . . [11–13]

- Solve problem with objective c_λ
- Restrict to triangle
- Find upper bound u_λ on c_λ
- Rank flows x while $c_\lambda(x) \leqq u_\lambda$
  If x is efficient: save it and update u_λ.



Example: Ranking Solutions for BIMCF

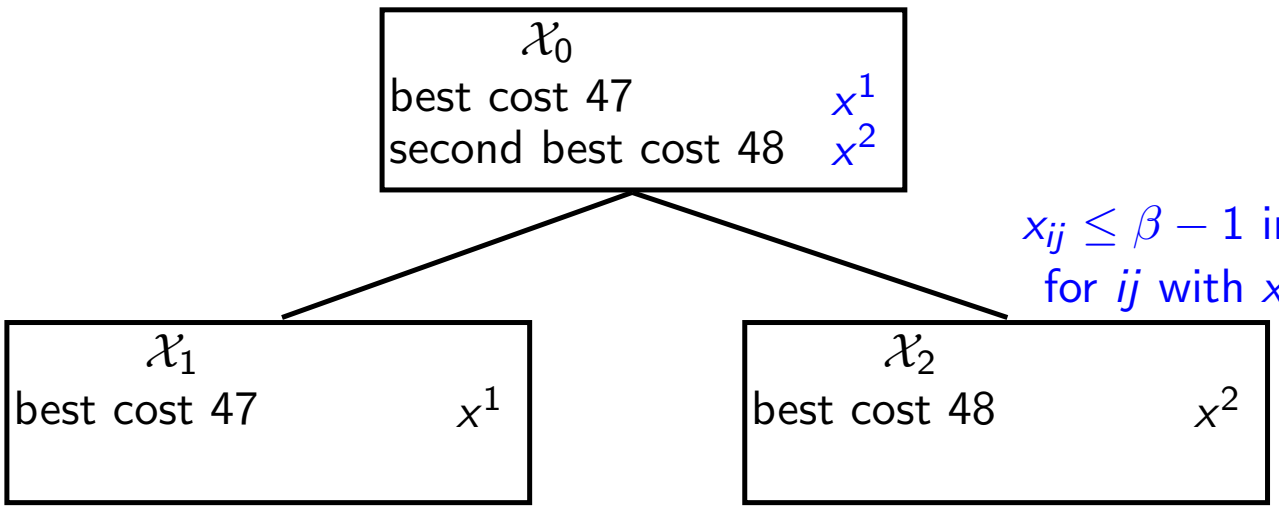
Ranking k best flows by Hamacher [7]

- Identify best x_0 and second best $\bar{x}_0$ feasible solutions in $\mathcal{X}$
- Partition $\mathcal{X}$ so that $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2, \mathcal{X}_1 \cap \mathcal{X}_2 = \textit{emptyset}$ and x_0 is optimal in $\mathcal{X}_1$ and $x_0 \notin \mathcal{X}_2$, and $\bar{x}_0$ is optimal in $\mathcal{X}_2$ and $\bar{x}_0 \notin \mathcal{X}_1$.
- $x_1 = x_0$ and $x_2 = \bar{x}_0$
- Identify second best solution $\bar{x}_1 \in \mathcal{X}_1$ and $\bar{x}_2 \in \mathcal{X}_2$.
- The third best solution is the one with smaller objective value out of $\bar{x}_1$ and $\bar{x}_2$.
- Split corresponding partition further.

Illustration of Ranking by Hamacher [7]

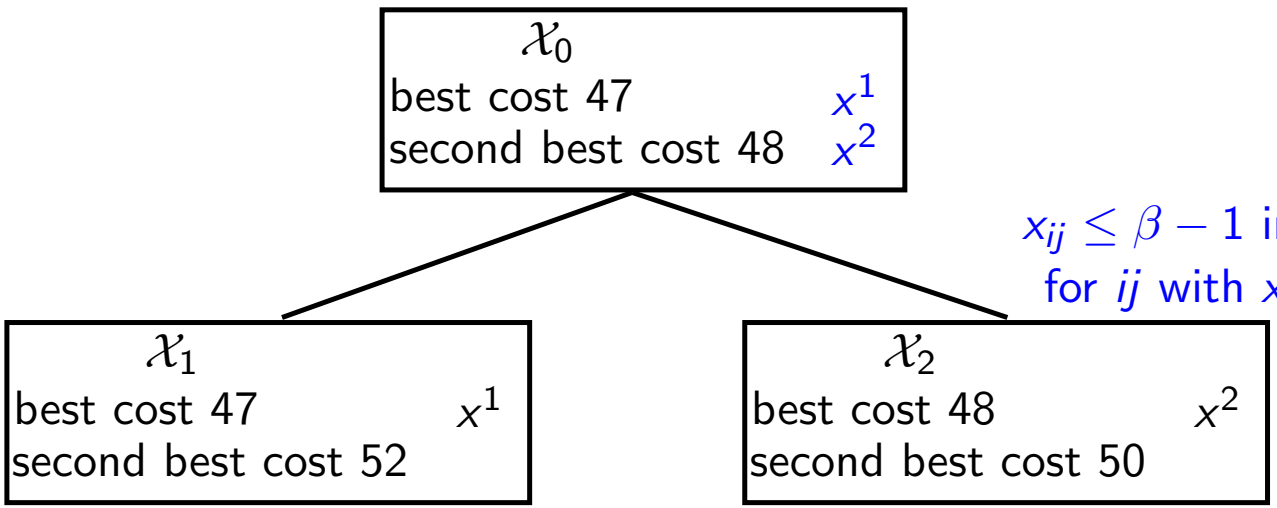
$\mathcal{X}_0$	
best cost 47	x^1
second best cost 48	x^2

Illustration of Ranking by Hamacher [7]



$x_{ij} \leq \beta - 1$ in $\mathcal{X}_1$ and $x_{ij} \geq \beta$ in $\mathcal{X}_2$
for ij with $x^1_{ij} < x^2_{ij}$, $\beta = x^2_{ij}$

Illustration of Ranking by Hamacher [7]



$x_{ij} \leq \beta - 1$ in $\mathcal{X}_1$ and $x_{ij} \geq \beta$ in $\mathcal{X}_2$
for ij with $x^1_{ij} < x^2_{ij}$, $\beta = x^2_{ij}$

Illustration of Ranking by Hamacher [7]

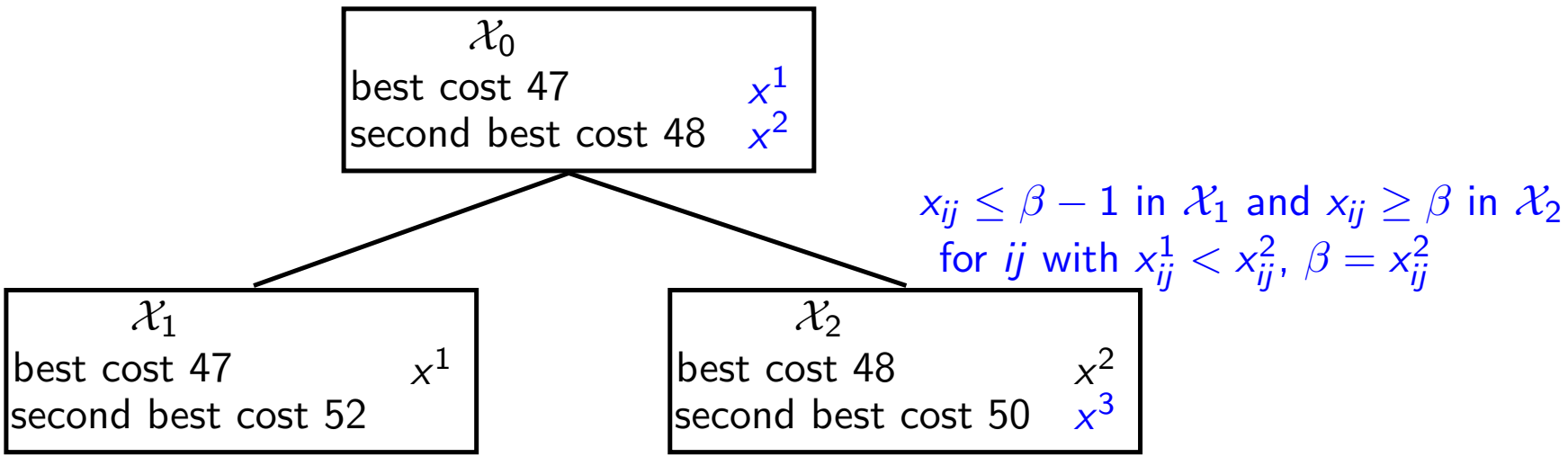


Illustration of Ranking by Hamacher [7]

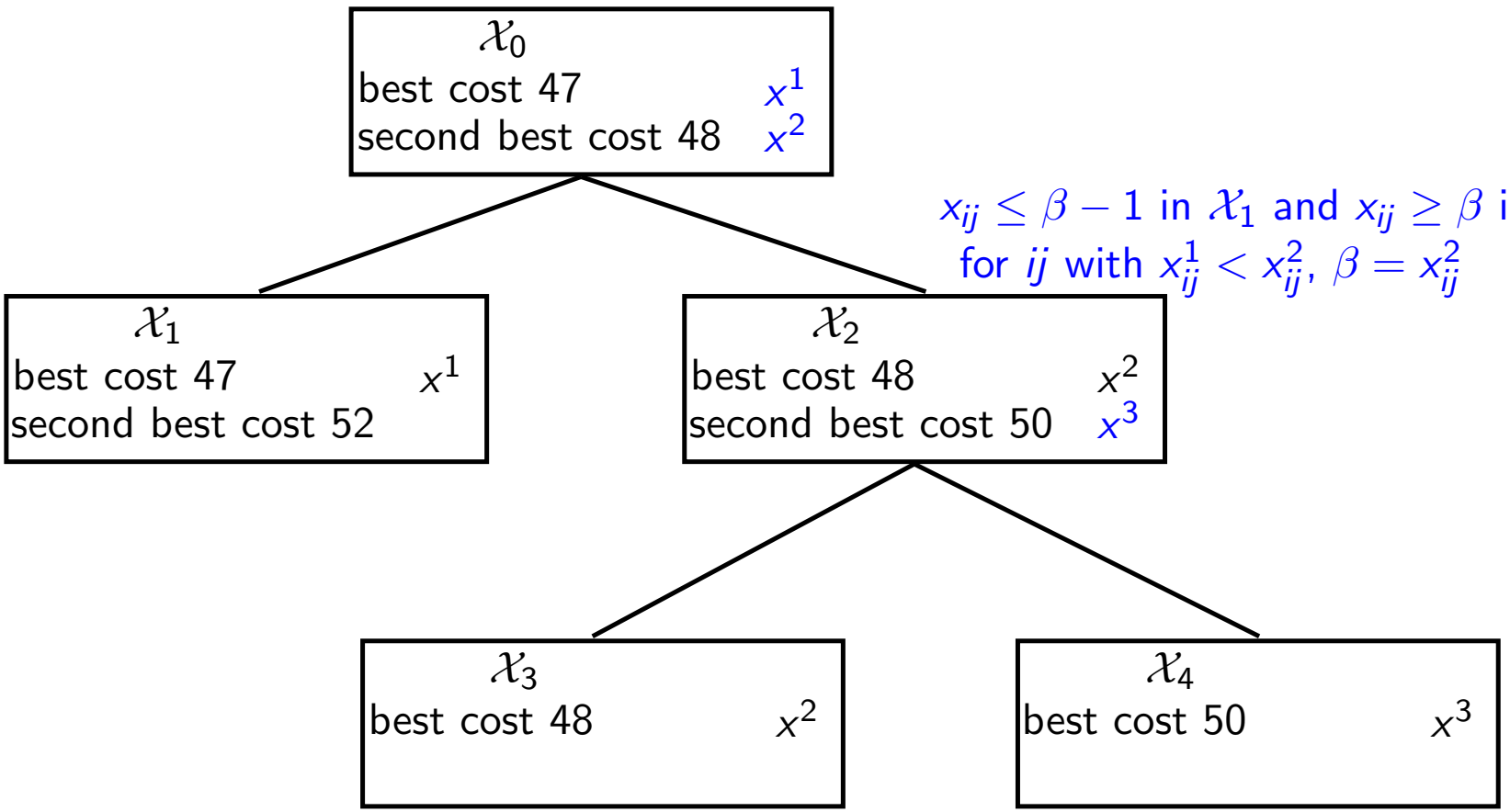


Illustration of Ranking by Hamacher [7]

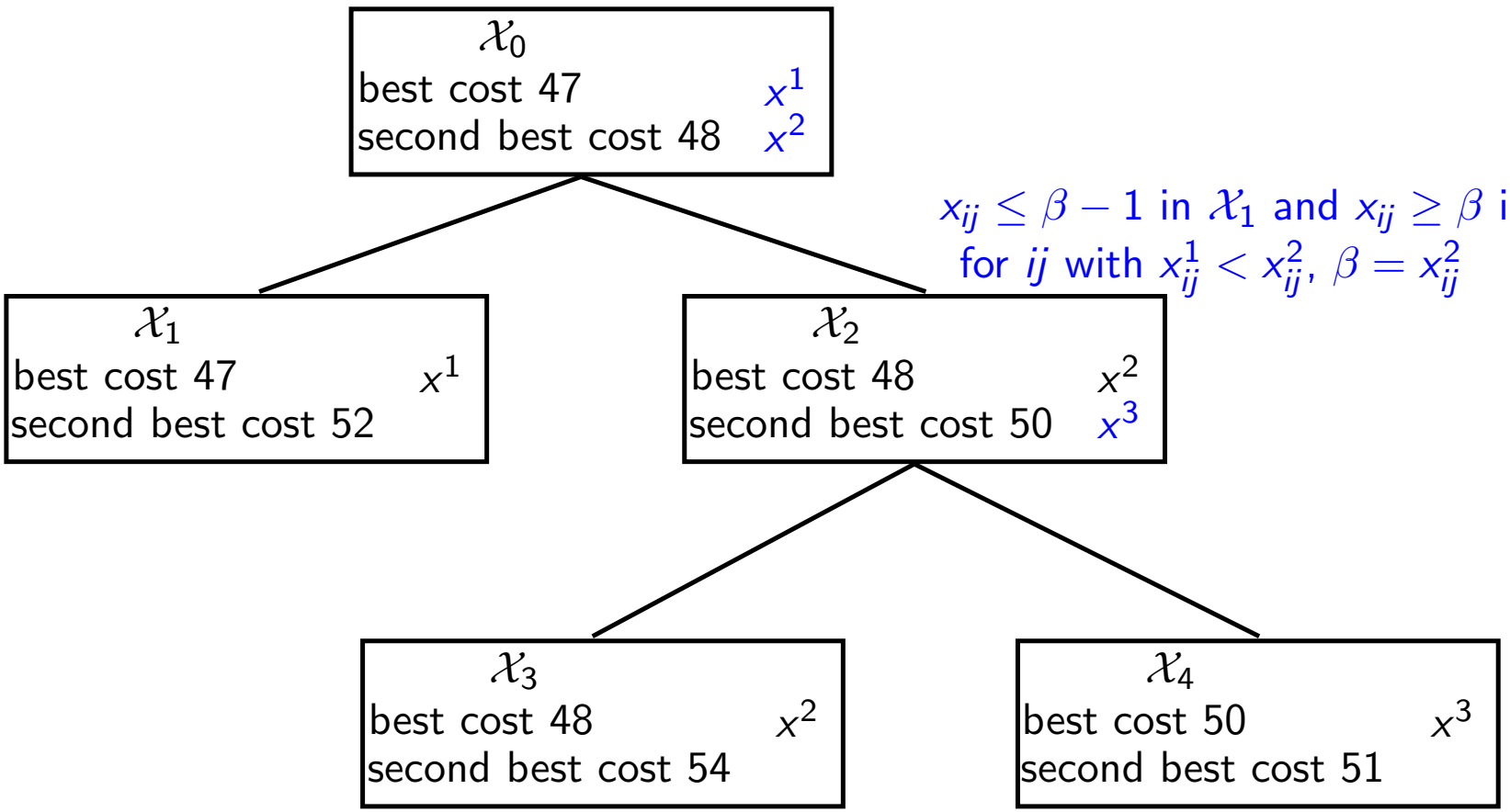
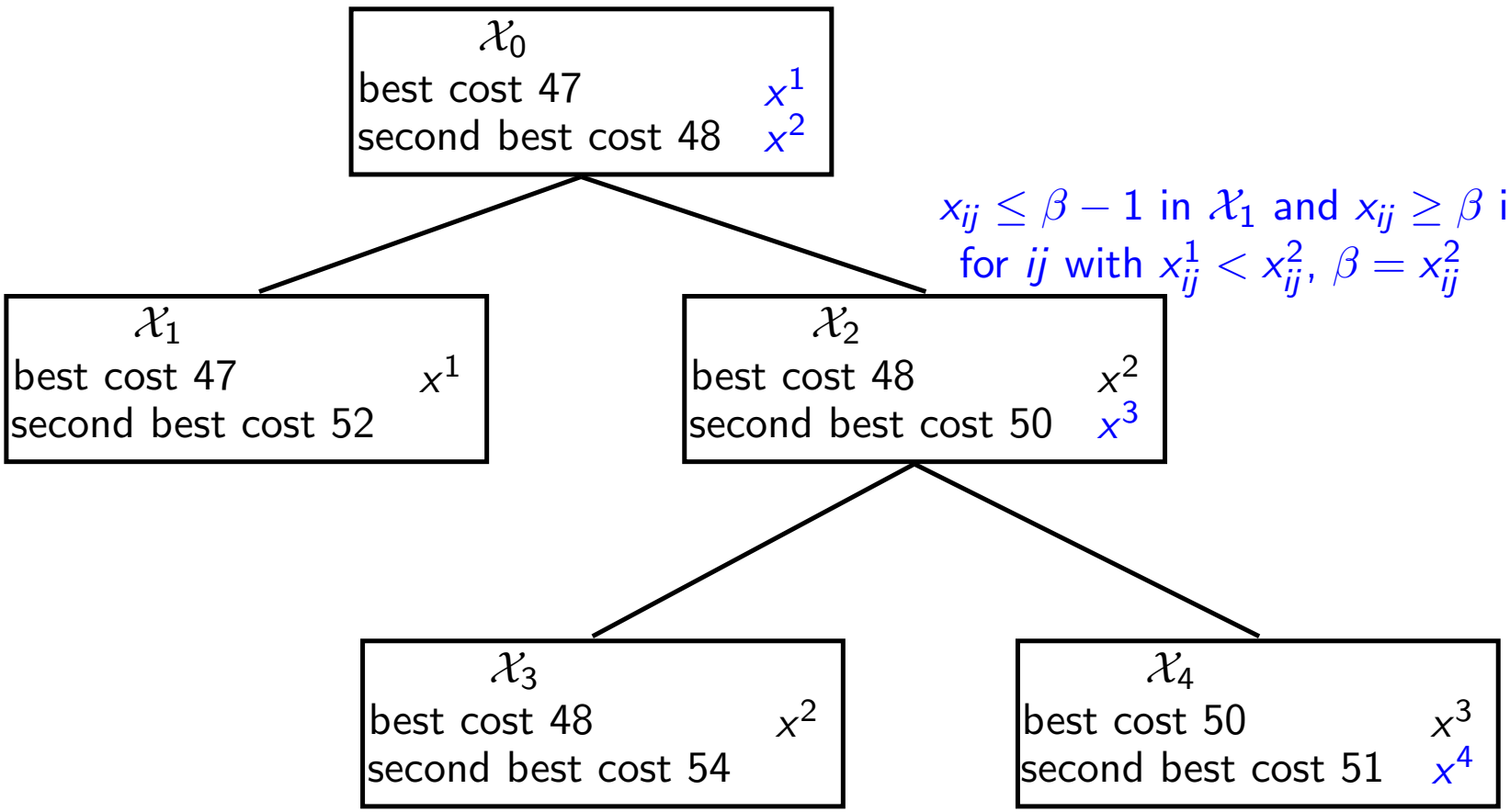


Illustration of Ranking by Hamacher [7]



Summary

- For some easy multi-objective problems it is possible to extend single-objective solution algorithms. Examples are multi-objective shortest path [e.g. 3; 17; 2; 16] and spanning tree problems [e.g. 5; 8].
- The two phase method solves problems when algorithms for the single-objective problem are available and there exists a method of solving phase 2 efficiently, e.g. by ranking. Examples are multi-objective integer minimum cost network flow [13] and assignment problems [11].

Outline

Appendix

[1] Y.P Aneja and K.P.K. Nair. Bicriteria transportation problem. *Management Science*, 25:73–78, 1979.

[2] J. Brumbaugh-Smith and D. Shier. An empirical investigation of some bicriterion shortest path algorithms. *European Journal of Operational Research*, 43(2):216–224, 1989.

[3] J.C.N. Clímaco and E.Q.V. Martins. A bicriterion shortest path problem. *European Journal of Operational Research*, 11: 399–404, 1982.

[4] J.L. Cohon. *Multiobjective Programming and Planning*. Academic Press, New York, 1978.

[5] H.W. Corley. Efficient spanning trees. *Journal of Optimization Theory and Applications*, 45:481–485, 1985.

[6] R.B. Dial. A model and algorithm for multicriteria route-mode choice. *Transportation Research*, 13B:311–316, 1979.

[7] H.W. Hamacher. A note on k best network flows. *Annals of Operations Research*, 57:65–72, 1995.

[8] H.W. Hamacher and G. Ruhe. On spanning tree problems with multiple objectives. *Annals of Operations Research*, 52: 209–230, 1994.

[9] P. Hansen. Bicriterion path problems. In G. Fandel and T. Gal, editors, *Multiple Criteria Decision Making, Theory and Application. Proceedings of the 3rd International Conference, Hagen/Königswinter 1979*, volume 177 of *Lecture Notes in Economics and Mathematical Systems*, pages 109–127. Springer Verlag, Berlin, 1980.

[10] E.Q.V. Martins. On a multicriteria shortest path problem. *European Journal of Operational Research*, 16:236–245, 1984.

[11] A. Przybylski, X. Gandibleux, and M. Ehrgott. Two-phase algorithms for the bi-objective assignment problem. *European Journal of Operational Research*, 185(2):509–533, 2008.

[12] A. Raith and M. Ehrgott. A comparison of solution strategies for biobjective shortest path problems. *Computers & Operations Research*, 36:1299–1331, 2009. doi: 10.1016/j.cor.2008.02.002.

[13] A. Raith and M. Ehrgott. A two-phase algorithm for the biobjective integer minimum cost flow problem. *Computers & Operations Research*, 36:1945–1954, 2009. doi: 10.1016/j.cor.2008.06.008.

[14] A. Sedeño-Noda and C. González-Martín. The biobjective minimum cost flow problem. *European Journal of Operational Research*, 124:591–600, 2000.

[15] A. Sedeño-Noda, C. González-Martín, and J. Gutiérrez. The biobjective undirected two-commodity minimum cost flow problem. *European Journal of Operational Research*, 164: 89–103, 2005.

[16] A.J.V. Skriver and K.A. Andersen. A label correcting approach for solving bicriterion shortest-path problems. *Computers & Operations Research*, 27:507–524, 2000.

[17] C.T. Tung and K.L. Chew. A bicriterion Pareto-optimal path algorithm. *Asia-Pacific Journal of Operational Research*, 5: 166–172, 1988.

[18] C.T. Tung and K.L. Chew. A multicriteria Pareto-optimal path algorithm. *European Journal of Operational Research*, 62:203–209, 1992.

[19] E. L. Ulungu and J. Teghem. The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems. *Foundations of Computing and Decision Sciences*, 20(2):149–165, 1995.