Bicriteria Optimization Approach to Analyze Incorporation of Biofuel and Carbon Capture Technologies

Ali Öztürk, Metin Türkay
College of Industrial Engineering, Koç University, Sariyer, Istanbul, 34450, Turkey
{alozturk@ku.edu.tr, mturkay@ku.edu.tr}

Supplementary Material 1:
The Two-Phase Algorithm to generate the efficient set for BOMILPs
AIChE Journal

In this supplementary material, the details of the two-phase algorithm, an illustrative example and and extensive comparison with other methods on benchmark problems are presented.

1. The Two-Phase Algorithm

The proposed algorithm consists of an initialization step and two main phases which are Phase-1 and Phase-2. In Phase-1 the supported efficient solutions are generated by using the weighted sum approach. In Phase-2, the Tchebycheff approach and two sub models are used to determine the non-supported efficient solutions. In the algorithms the following parameters are used.

- SE: Supported efficient points in the objective space.
- NSE: Non-supported efficient points in the objective space.
- LS: Intervals to be explored in Phase-1.
- DLS : Intervals to be explored in Phase-2.
- z^{int}, z^{end}: Extreme efficient points.
- $\varepsilon_1, \varepsilon_2$: Allowable search intervals.

Algorithm 1: Initialization Phase

Initialize θ
$SE \leftarrow \emptyset$, $NSE \leftarrow \emptyset$, $LS \leftarrow \emptyset$, $DLS \leftarrow \emptyset$
$\lambda_1 \leftarrow 1$, $\lambda_2 \leftarrow 0$
$z_1^*, z_2^* \leftarrow -\infty$; $z_1^{**}, z_2^{**} \leftarrow \infty$
$z \leftarrow$ `WeightedSumModel`(z^*, z^{**}, λ)
$\lambda_1 \leftarrow 0$, $\lambda_2 \leftarrow 1$
$z^{int} \leftarrow$ `WeightedSumModel`(z^*, z, λ)
$z \leftarrow$ `WeightedSumModel`(z^*, z^{**}, λ)
$\lambda_1 \leftarrow 1$, $\lambda_2 \leftarrow 0$
$z^{end} \leftarrow$ `WeightedSumModel`(z^*, z, λ)
if $(z_{int} = z_{end})$ **then**
    $SE \leftarrow LE \cup z^{int}$
    Stop
else
    $SE \leftarrow \{z^{int}, z^{end}\}$
    $LS \leftarrow [z^{int}\ z^{end}]$
    $\varepsilon_1 \leftarrow (z_1^{end} - z_1^{int})/\theta$
    $\varepsilon_2 \leftarrow (z_2^{int} - z_2^{end})/\theta$
end

Initially, it is necessary to obtain extreme efficient points. To do that, an initialization phase is introduced before Phase-1 and Phase-2. The steps of the initialization phase are given in Algorithm 1.

In Algorithm 1, `WeightedSumModel` method is used to determine extreme efficient points. This function solves P_{we} model with three inputs which are z^* (lower bound), z^{**} (upper bound), λ (weights) and returns an efficient solution.

After the initialization phase, the supported efficient solutions can be obtained by using Algorithm 2.

In Phase-1, starting from the extreme efficient points interval all intervals in the LS list is investigated. For each interval $[z^l\ z^u]$, `WeightedSumModel` is used to obtain supported efficient solution. There may not be any supported efficient solution for some search intervals, in that case the method returns either z^l or z^u. The algorithm is terminated when the search interval list is empty. Phase-1 of the proposed method is followed by the Phase-2. In Phase-2, all non-supported efficient solutions are determined. The steps of the Phase-2 are given in Algorithm 3.

Algorithm 2: Phase-1

input : z_{int}, z_{end}, LS
output: SE, DLS

while $LS \neq \emptyset$ **do**
  Pick an interval INT from the LS, $INT \leftarrow [z^l\ z^u]$
  $LS \leftarrow LS \backslash INT$
  $DLS \leftarrow DLS \cup [z^l\ z^u]$
  Calculate z^* and z^{**} from Equation (**??**) and (**??**), respectively
  Calculate λ_1 and λ_2 from Equation (**??**) and (**??**), respectively
  $z^{PWe} \leftarrow$ `weightedSumModel(`z^*, z^{**}, λ`)`
  if $(z^{PWe} \neq z^l)$ **and** $(z^{PWe} \neq z^u)$ **then**
    $SE \leftarrow SE \cup z^{PWe}$
    $LS \leftarrow LS \cup \{[z^l\ z^{PWe}], [z^{PWe}\ z^u]\}$
    $DLS \leftarrow DLS \backslash [z^l\ z^u]$
  end
end
return SE, DLS

In this phase of the algorithm, non-supported efficient solutions are determined by using the P_{Tch}, P_{Ep1} and P_{Ep2} models. Similar to Phase-1, all search intervals in the DLS list is explored. For each interval $[z^l\ z^u]$, `TchebycheffModel` and two submodel methods that corresponds to `PEpModel1` and `PEpModel2` are used to obtain non-supported efficient solutions. The `TchebycheffModel` function gets z^* (lower bound), z^{**} (upper bound) as inputs and returns α^*. Then, functions `PEpModel1` and `PEpModel2` are used to obtain z^{Pep1} and z^{Pep2}, respectively. Four different cases are controlled to determine the efficient solution(s). Phase-2 is terminated when the DLS list is empty.

2. Illustrative Example

In this subsection, the proposed two-phase algorithm is applied to bicriteria knapsack problem to show how the method generates supported and non-supported efficient solutions. The bicriteria knapsack problem consists of an integer capacity $W > 0$ with n objects. Each object j has a positive integer weight w_j and two non-negative integer profits which are v_j^1 and v_j^2. Decision variables (x_j) represents whether the object j is selected or not. The mathematical model of the bicriteria knapsack problem is as follows;

Algorithm 3: Phase-2

input : $DLS, \varepsilon_1, \varepsilon_2$
output: NSE

Initialize α^0
while $(DLS \neq \emptyset)$ **do**
  Pick an interval INT from the DLS, $INT \leftarrow [z^l\ z^u]$
  $DLS \leftarrow DLS \backslash INT$
  if $((z_1^u - z_1^l) > \varepsilon_1)$ **or** $((z_2^l - z_2^u) > \varepsilon_2)$ **then**
    Calculate z^* and z^{**} from Equation (**??**) and (**??**), respectively
    $\alpha^* \leftarrow$ `TchebycheffModel` (z^*, z^{**})
    if $(\alpha^* < \alpha^0)$ **then**
      $z^{PEp1} \leftarrow$ `PEpModel1` (z^*, z^{**}, α^*)
      $z^{PEp2} \leftarrow$ `PEpModel2` (z^*, z^{**}, α^*)
      if $(z^{PEp1} = z^{PEp2})$ **then**
        `// Solutions are equal`
        $DLS \leftarrow DLS \cup \{[z^l\ z^{PEp1}], [z^{PEp1}\ z^u]\}$
        $NSE \leftarrow NSE \cup z^{PEp1}$
      end
      if $(z^{PEp2} \Delta z^{PEp1})$ **then**
        `//` z^{PEp2} `dominates` z^{PEp1}
        $DLS \leftarrow DLS \cup \{[z^l\ z^{PEp2}], [z^{PEp2}\ z^u]\}$
        $NSE \leftarrow NSE \cup z^{PEp2}$
      end
      if $(z^{PEp1} \Delta z^{PEp2})$ **then**
        `//` z^{PEp1} `dominates` z^{PEp2}
        $DLS \leftarrow DLS \cup \{[z^l\ z^{PEp1}], [z^{PEp1}\ z^u]\}$
        $NSE \leftarrow NSE \cup z^{PEp1}$
      end
      if $(\sim (z^{PEp1} \Delta z^{PEp2})$ **and** $\sim (z^{PEp2} \Delta z^{PEp1}))$ **then**
        `// Both solutions are efficient`
        $DLS \leftarrow DLS \cup \{[z^l\ z^{PEp1}], [z^{PEp2}\ z^u]\}$
        $NSE \leftarrow NSE \cup z^{PEp1}$
        $NSE \leftarrow NSE \cup z^{PEp2}$
      end
    end
  end
end
return NSE

$$\max \left(\sum_{j=1}^{n} v_j^1 x_j, \sum_{j=1}^{n} v_j^2 x_j \right) \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} w_j x_j \leq W \tag{2}$$

$$x^j \in \{0,1\} \quad j = 1, \ldots, n \tag{3}$$

Equation (1) is the objective function of the model that maximizes both first and second objective functions. Equation (2) is the capacity constraint, the total weight of selected objects has to be less then or equal to the knapsack's capacity. Equation (3) represents of binary integrality constraints.

An instance of the bicriteria knapsack problem is given below where the number items is $n = 10$ and v_1, v_2 and w parameters are generated uniformly distributed in the range of $[1, 100]$. The knapsack capacity is calculated by using Equation (4).

$$W = \left\lfloor \frac{\sum_{j=1}^{n} w_j}{2} \right\rfloor \tag{4}$$

The bicriteria knapsack problems instance is given below where the parameters of the instance are obtained as explained above.

$$\max \ f_1(x) = 42x_1 + 47x_2 + 46x_3 + 37x_4 + 16x_5 + 54x_6 + 94x_7 + 37x_8 + 76x_9 + 59x_{10}$$

$$\max \ f_2(x) = 90x_1 + 75x_2 + 78x_3 + 31x_4 + 80x_5 + 23x_6 + 11x_7 + 6x_8 + 97x_9 + 16x_{10}$$

s.t.

$$14x_1 + 80x_2 + 73x_3 + 89x_4 + 66x_5 + 48x_6 + 53x_7 + 4x_8 + 27x_9 + 80x_{10} \leq 267$$

$$x_j \in \{0,1\}, \quad j = 1, \ldots, 10$$

The proposed two-phase algorithm is a generic method and the criterion functions in the algorithm is defined in minimization form. As the bicriteria knapsack problem's criteria functions are in maximization form, we first transformed it into minimization form ($f_1'(x) = -f_1(x)$ and $f_2'(x) = -f_2(x)$) then generate the efficient set for this instance.

For the given instance, the extreme efficient points are determined which are $z^{int} = [-362 \ -243]$ and $z^{end} = [-264 - 426]$. A new supported efficient point $[-342 \ -357]$ is obtained with convex combination of extreme efficient solutions. A final efficient point is

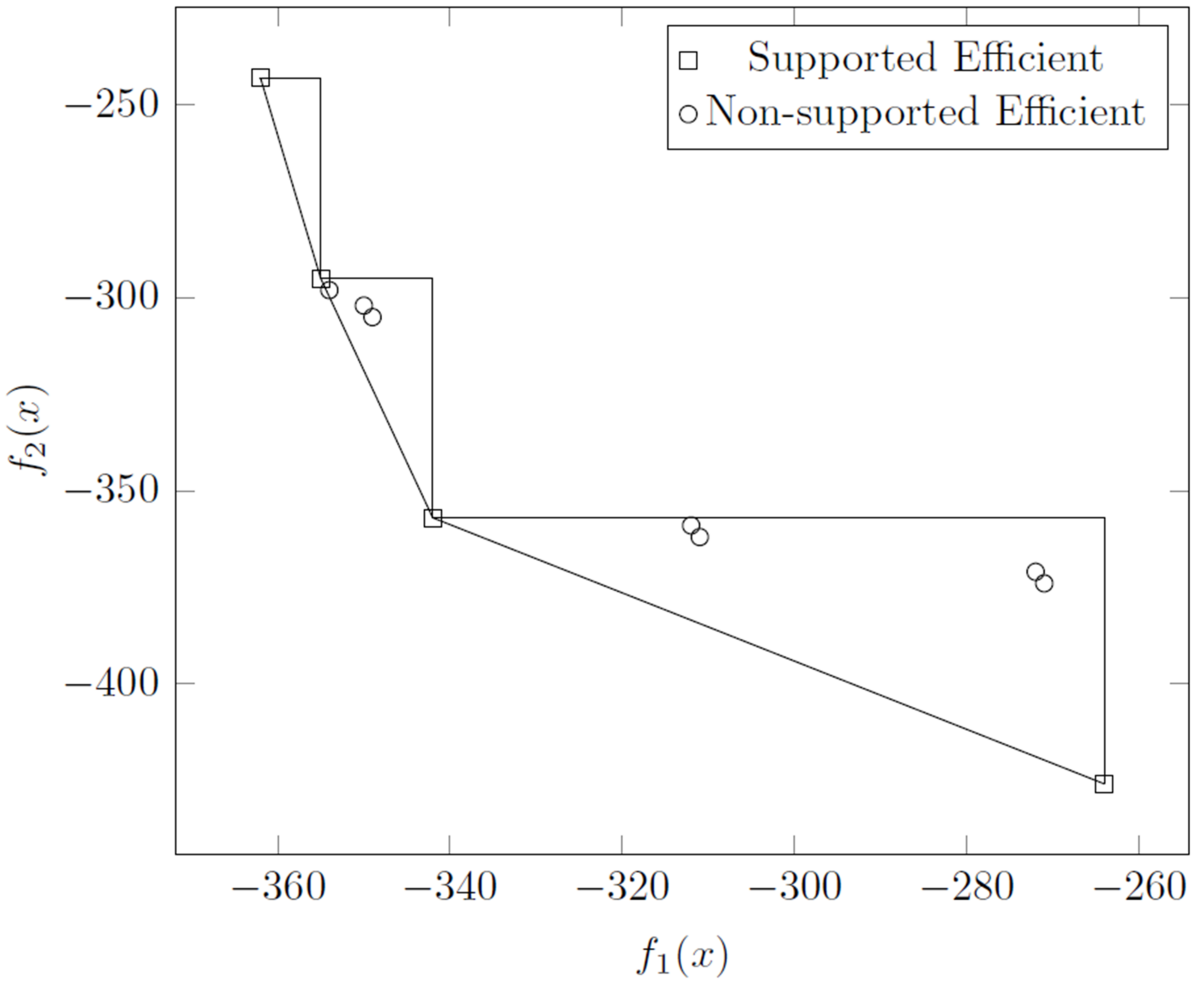

Figure 1: Supported and non-supported efficient solutions for the illustrative example.

$[-355\ -295]$ which is obtained by using weighted sum model where $z^l = [-362\ -243]$ and $z^u = [-342\ -357]$. The supported efficient points are shown in Figure S1.

After the determination of supported efficient solutions, the two-phase algorithm continues to generate the efficient set by obtaining non-supported efficient solutions. First non-supported efficient point $[-302\ -369]$ is determined when $z^l = [-342\ -357]$ and $z^u = [-264\ -426]$. Other non-supported efficient points are as follows $[-271\ -374]$, $[-272\ -371]$, $[-311\ -362]$, $[-312\ -359]$, $[-349\ -305]$, $[-354\ -298]$, $[-350\ -302]$. Non-supported efficient points are also shown in Figure S 1.

3. Scalarization Methods for Bicriteria Optimization

Several methods have been proposed in the literature to obtain the efficient set for biobjective optimization problems:

- The weighted-sum method (Geoffrion, 1968)
- The ϵ-Constraint method (Haimes et al., 1971)
- Augmented ϵ-Constraint method (Mavrotas, 2009)
- The weighted Tchebycheff method (Bowman, 1976)
- Thw augmented-Weighted Tchebycheff method (Steuer and Choo, 1983)
- Quadratic function method (Neumayer and Schweigert, 1994)
- Two-stage sub problems (Neumayer and Schweigert, 1994)

A brief description of each of these methods are given below.

3.1 The Weighted-sum Method

- Weighted sum model is defined as follows,

$$\mathbf{P_{We}} \quad \min \quad \sum_{j=1}^{p} \lambda_j f_j(x) \tag{5}$$

$$\text{s.t.} \quad x \in \mathcal{X} \tag{6}$$

$$\sum_{j=1}^{p} \lambda_j = 1 \tag{7}$$

$$0 \leq \lambda_j \leq 1 \quad j = 1, \ldots, p \tag{8}$$

- Optimal solution of the P_{We} is efficient (Geoffrion, 1968).
- Only supported efficient solution can be obtained the weighted-sum method.

3.2 The ϵ-constraint Method

- The $\epsilon-$constraint was proposed by (Haimes et al., 1971).
- It is based on scalarization where one of the objective functions is minimized while other objective functions are bounded from above with additional constraints.
- It is one of the most widely used methods for generating the efficient set. However, it generates only the supported efficient solutions.

$$\begin{aligned} \mathbf{P}_{\epsilon} \quad \min \quad & f_k(x) && (9)\\ \text{s.t.} \quad & f_j(x) \leq \epsilon_j \quad j = 1, \ldots, p;\ j \neq k && (10)\\ & x \in \mathcal{X} && (11) \end{aligned}$$

Theorem 1. *For any $\epsilon_k \in \mathbb{R}^{p-1}$ the following statements hold.*

- *If $x^* \in \mathcal{X}$ is optimal solution of P_ϵ, then x^* is weakly efficient solution.*
- *If $x^* \in \mathcal{X}$ is unique optimal solution of P_ϵ, then x^* is efficient solution.*

$$\begin{aligned} \mathbf{P}_{\mathbf{lex}-\epsilon} \quad \mathbf{Lex}\ \min \quad & (f_1(x), f_2(x)) && (12)\\ \text{s.t.} \quad & f_j(x) \leq \epsilon_j \quad j = 1, 2 && (13)\\ & x \in \mathcal{X} && (14) \end{aligned}$$

The optimal solution ($x^* \in \mathcal{X}$) of $P_{lex-\epsilon}$ is efficient.

3.3 The Augmented ϵ-constraint Method

- The objective function of the $\epsilon-$constraint method is turned into augmented form by (Mavrotas, 2009).

- For $\rho > 0$ (sufficiently small, e.g. $\rho = 0.0001$) the augmented ϵ-constraint model is defined as follows.

$$\mathbf{P_{Aug-\epsilon}} \quad \min \quad f_k(x) + \rho \sum_{\substack{j=1,\ldots,p \\ j \neq k}} s_j \tag{15}$$

$$\text{s.t.} \quad f_j(x) + s_j = \epsilon_j \quad j = 1, \ldots, p;\ j \neq k \tag{16}$$

$$s_j \in \mathbb{R}^+ \tag{17}$$

$$x \in \mathcal{X} \tag{18}$$

The optimal solution ($x^* \in \mathcal{X}$) of $P_{Aug-\epsilon}$ is efficient.

3.4 The Weighted Tchebycheff Method

- The weighted Tchebycheff problem was introduced by (Bowman, 1976).
- The main idea of the method is to find a solution as close as possible to ideal point.
- For the weight vector $w \geq 0$ the weighted Tchebycheff model is defined as follows.

$$\mathbf{P_{Tch}} \quad \min \max \quad \left(w_j | f_j(x) - f_j^I|\right) \quad i = 1, \ldots, p \tag{19}$$

$$\text{s.t.} \quad x \in \mathcal{X} \tag{20}$$

- The problem can be solved by adding p variables and constraints to linearize the max term of the objective function.
- The solution of the problem is weakly efficient.

3.5 The Augmented-weighted Tchebycheff Method

- The augmented weighted Tchebycheff was suggested by (Steuer and Choo, 1983).
- In this case, distance between ideal point and feasible objective region is minimized.
- For the weight vector $w \geq 0$ and $\rho > 0$ (sufficiently small, e.g. $\rho = 0.0001$) the augmented weighted Tchebycheff model is defined as follows.

$$\mathbf{P_{Aug}} \quad \min \max_{j=1,\ldots,p} \quad \left(w_j |f_j(x) - f_j^I| + \rho \sum_{j=1}^{p} |f_j(x) - f_j^I| \right) \tag{21}$$

$$\text{s.t.} \quad x \in \mathcal{X} \tag{22}$$

- The solution of the problem is efficient.

3.6 Quadratic Function Method

- The main of this method is to hyperbola instead of norm.
- The input data are two efficient points $p, q \in \mathbb{Z}^2$.
- Using these efficient points parameters a, b and c are determined as follows,

$$a = (p_1 - q_1 + \frac{1}{2})(q_2 - p_2 + \frac{1}{2}) - \frac{1}{4} \tag{23}$$

$$b = (p_2 - q_2 - \frac{1}{2})(p_1 - q_1 + \frac{1}{2})q_1 + \frac{1}{4}p_1 \tag{24}$$

$$c = (p_2 - q_2 - \frac{1}{2})(p_1 - q_1 + \frac{1}{2})p_2 + \frac{1}{4}q_2 \tag{25}$$

$$\mathbf{P_{quad}} \quad \min \quad af_1(x)f_2(x) + bf_2(x) + cf_1(x) \tag{26}$$

$$\text{s.t.} \quad p_1 \leq f_1(x) \leq q_1 \tag{27}$$

$$p_2 \leq f_2(x) \leq q_2 \tag{28}$$

$$x \in \mathcal{X} \tag{29}$$

- The optimal solution of P_{quad} is an efficient solution Neumayer and Schweigert (1994).
- In some cases objective function of the P_{quad} may be non-convex.

3.7 Max Ordering

- This approach is based on a two step procedure based on max ordering Neumayer and Schweigert (1994).

For a weight vector $w \geq 0$, the subproblem P_w is as follows;

$$\mathbf{P_w} \quad \min \max_{\mathbf{j=1,\ldots,p}} \quad w_j f_j(x) \tag{30}$$

$$\text{s.t.} \quad x \in \mathcal{X} \tag{31}$$

Let optimal solution of P_w as z^*, then the second subproblem Q_w defined as follows,

$$\mathbf{Q_w} \quad \min \quad \sum_{j=1}^{p} f_j(x) \tag{32}$$

$$\text{s.t.} \quad w_j f_j(x) \leq z^* \quad j = 1, \ldots, p \tag{33}$$

$$x \in \mathcal{X} \tag{34}$$

The optimal solution of P_w and Q_w is an efficient solution.

4. Computational Results

In this section, we provide comparative performance analysis of the proposed two-stage algorithm with the other algorithms available in the literature. We select the most widely used benchmark problems for testing the effectiveness of different algorithms. All experiments presented here were performed on a dual core Xeon 3.33 GHz CPU with 32 Gb RAM. Algorithms are coded in C++ with the IBM CPLEX 12.1 callable library.

4.1 The Assignment Problem

- The bicriteria assignment problem consists of two non-negative integer assignment costs which are c_{ij}^1 and c_{ij}^2.
- $x_{ij} = \begin{cases} 1 & \text{If } i \text{ is assigned to } j. \\ 0 & \text{otherwise} \end{cases}$
- The mathematical model of the bicriteria assignment problem is as follows;

$$\mathbf{P_{Ap}} \quad \min \quad \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^1 x_{ij} \tag{35}$$

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^2 x_{ij} \tag{36}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, \ldots, n \tag{37}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad j = 1, \ldots, n \tag{38}$$

$$x_{ij} \in \{0, 1\} \quad \forall (i, j) \tag{39}$$

- The benchmark problems for the bicriteria assignment problem instances taken from Przybylski et al. (2008).
- For sizes $n = 10$ to $n = 100$ with an increment of 10, 10 instances each.

- The objective function coefficients were generated in the range of $[0, 20]$, $[0, 40]$, $[0, 60]$.
- The results are shown in three tables.
- In the tables, n is the number of tasks to be assigned and $|\mathcal{X}_{SE}|$ corresponds to the number of supported efficient solutions.

Table 1: Average CPU times (sec) of tested algorithms on the assignment problem ($[0, 20]$ range).

n	$\lvert\mathcal{X}_{SE}\rvert$	$P_{mAug-\epsilon}$	P_{Aug}	P_{Tch}	Max Ord.	Two-Phase
10	29	0.13	1.42	3.97	4.09	0.80
20	49	0.53	7.21	18.61	20.81	5.90
30	98	1.85	31.72	82.23	92.64	28.27
40	106	2.51	62.66	153.52	181.61	56.96
50	117	3.57	120.21	285.50	338.27	110.28
60	181	9.20	337.56	737.49	938.64	355.32
70	176	14.63	527.67	1092.71	1243.09	561.35
80	190	26.01	851.61	1667.01	1927.94	925.66
90	216	56.52	1514.89	2782.79	3130.79	1702.12
100	230	277.97	2548.71	4131.25	4706.44	3185.88

Table 2: Average CPU times (sec) of tested algorithms on the assignment problem ($[0, 40]$ range).

n	$\lvert\mathcal{X}_{SE}\rvert$	$P_{mAug-\epsilon}$	P_{Aug}	P_{Tch}	Max Ord.	Two-Phase
10	21	0.10	0.99	2.53	2.42	0.56
20	66	0.63	8.91	24.17	25.11	7.30
30	109	2.11	33.45	84.94	96.74	29.82
40	186	5.07	114.78	272.71	347.77	104.34
50	216	8.77	234.98	536.95	690.18	215.57
60	253	14.64	468.61	1006.66	1330.05	493.27
70	331	22.37	1010.31	2178.18	2633.49	1074.79
80	355	33.98	1581.67	3284.27	3835.96	1719.20
90	432	55.35	2854.61	5418.72	6662.81	3207.42
100	429	72.38	3884.95	7158.59	8687.52	4856.18

- The results indicate that $P_{mAug-\epsilon}$ is the fastest algorithm. However, $P_{mAug-\epsilon}$ is a single stage scalarization method and does not guarantee that all identified solutions

Table 3: Average CPU times (sec) of tested algorithms on the assignment problem ($[0, 60]$ range).

n	$\lvert\mathcal{X}_{SE}\rvert$	$P_{mAug-\epsilon}$	P_{Aug}	P_{Tch}	Max Ord.	Two-Phase
10	29	0.13	1.42	3.97	4.09	0.80
20	49	0.53	7.21	18.61	20.81	5.90
30	98	1.85	31.72	82.23	92.64	28.27
40	106	2.51	62.66	153.52	181.61	56.96
50	117	3.57	120.21	285.50	338.27	110.28
60	181	9.20	337.56	737.49	938.64	355.32
70	176	14.63	527.67	1092.71	1243.09	561.35
80	190	26.01	851.61	1667.01	1927.94	925.66
90	216	56.52	1514.89	2782.79	3130.79	1702.12
100	230	277.97	2548.71	4131.25	4706.44	3185.88

are efficient. Besides, $P_{mAug-\epsilon}$ may miss some of the efficient solutions depending on the value of the penalty parameter, ρ.

- The performance of P_{Aug} and the Two-Phase algorithms are clearly better than the performance of P_{Tch} and Max Ordering algorithms for all instances.
- The P_{Aug} has a slightly better performance than the Two-Phase algorithms on about half of the instances.

4.2 The Knapsack Problem

- The bicriteria knapsack problem consists of an integer capacity $W > 0$ with n objects.
- Each object j has a positive integer weight w_j and two non-negative integer profits which are v_j^1 and v_j^2.
- $x_j = \begin{cases} 1 & \text{If the object } j \text{ is selected.} \\ 0 & \text{otherwise} \end{cases}$
- The mathematical model of the bicriteria knapsack problem is as follows;

$$\mathbf{P_{Kp}} \quad \max \quad \sum_{j=1}^{n} v_j^1 * x_j \tag{40}$$

$$\max \quad \sum_{j=1}^{n} v_j^2 * x_j \tag{41}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} w_j x_j \leq W \tag{42}$$

$$x_j \in \{0, 1\} \quad j = 1, \ldots, n \tag{43}$$

Table 4: Average CPU times (sec) of tested algorithms on the knapsack problem.

n	$\lvert\mathcal{X}_{SE}\rvert$	$\lvert\mathcal{X}_{E}\rvert$	P_{We}	$P_{lex-\epsilon}$	P_{Tch}
100	18.9	159.3	0.19	3.74	32.90
200	34.6	529.0	0.64	35.82	245.78
300	54.3	1130.7	1.84	152.50	850.51
400	68.4	1713.3	3.01	346.17	1682.42
500	85.1	2537.5	4.77	682.73	2960.41
600	101.6	3593.9	7.28	1280.96	5086.33

Table 5: Average CPU times (sec) of tested algorithms on the knapsack problem (continued).

n	P_{Aug}	Max Ord.	Two-Phase	$P_{mAug-\epsilon}$
100	19.41	16.03	16.87	2.26
200	159.32	167.81	152.49	22.10
300	554.50	431.93	638.53	95.93
400	1111.65	1084.34	1517.71	215.48
500	1911.57	1986.26	2002.35	425.10
600	3149.32	3212.41	3105.25	809.71

- The bicteriteria knapsack problem instances taken from Cristina Bazgan and Vanderpooten (2009).
- $v_1^k \in U[1, 1000]$, $v_2^k \in U[1, 1000]$ and $w^k \in U[1, 1000]$.

$$W = \left\lfloor \frac{\sum_{j=1}^{n} w_j}{2} \right\rfloor \tag{44}$$

- Number of items varies 100 through 600 ($n = 100, 200, \ldots, 600$), and 10 instances are solved for each size.
- Average of 10 instances are reported with respect to number of items
- In the tables, n is the number of items, $\lvert\mathcal{X}_{SE}\rvert$ corresponds to the number of supported efficient solutions and $\lvert\mathcal{X}_{E}\rvert$ is the number of efficient solutions.

- The results indicate that P_{We}, $P_{lex-\epsilon}$ and $P_{mAug-\epsilon}$ are faster than the other algorithms. This expected from single stage algorithms since neither they cannot guarantee that the solutions generated are efficient nor they can generate all efficient solutions for problems with non-convex feasible regions.
- The performance of the Two-Phase is significantly better than P_{Tch} and slightly better than P_{Aug} and Max Ordering algorithms for all instances.

4.3 Set Covering Problem

- The bicriteria set covering problem consists of a binary parameter a_{ij} whether item i can be covered by set j.
- Each set j has two non-negative integer profits which are c_j^1 and c_j^2.
- $x_j = \begin{cases} 1 & \text{If the set } j \text{ is selected.} \\ 0 & \text{otherwise} \end{cases}$
- The mathematical model of the bicriteria set covering problem is as follows;

$$\mathbf{P_{Scp}} \quad \min \quad \sum_{j=1}^{n} c_j^1 * x_j \tag{45}$$

$$\min \quad \sum_{j=1}^{n} c_j^2 * x_j \tag{46}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad i = 1, \ldots, m \tag{47}$$

$$x_j \in \{0, 1\} \quad j = 1, \ldots, n \tag{48}$$

- Number of items varies 100 through 600 ($n = 100, 200, \ldots, 600$), and 10 instances are solved for each size.
- Average of 10 instances are reported with respect to number of items
- In the tables, m is the number of sets, n is the number of elements, $|\mathcal{X}_{SE}|$ corresponds to the number of supported efficient solutions and $|\mathcal{X}_E|$ is the number of efficient solutions.

Table 6: Average CPU times (sec) of tested algorithms on the set covering problems.

m	n	$\lvert\mathcal{X}_{SE}\rvert$	$\lvert\mathcal{X}_E\rvert$	$P_{mAug-\epsilon}$	P_{Aug}
10	100	11	39	0.17	1.61
40	200	23	107	0.89	9.04
40	400	34	208	1.95	26.61
40	200	14	46	2.08	8.54
60	600	32	257	4.31	55.15
60	600	13	98	24.36	80.23
80	800	38	424	6.56	112.06
80	800	23	132	52.54	179.45
100	1000	24	157	118.13	472.01
100	1000	21	83	31.33	126.57
200	1000	24	274	4631.39	32055.70

Table 7: Average CPU times (sec) of tested algorithms on the set covering problems (continued).

m	n	Two-Phase	P_{Tch}	$P_{lex-\epsilon}$	Max Ord.
10	100	1.03	3.26	0.30	3.54
40	200	8.37	17.01	1.52	16.70
40	400	27.89	51.24	4.53	58.61
40	200	4.83	16.90	2.10	17.71
60	600	58.97	107.10	10.21	121.02
60	600	47.51	171.11	25.24	177.68
80	800	143.63	241.01	18.73	291.07
80	800	110.20	367.09	56.60	372.76
100	1000	408.21	934.13	155.94	946.32
100	1000	73.43	248.54	34.45	262.06
200	1000	44213.60	55034.50	8579.11	54418.70

- The performance of the Two-Phase is significantly better than P_{Tch} and Max Ordering algorithms. P_{Aug}; and Two-Phase have similar performance, in half of the instances the Two-Phase has a better performance than P_{Aug}.

The comparative analysis of different algorithms show that the Two-Phase algorithm consistently generates the efficient set faster across all of the benchmark problem categories. While some algorithms perform very well in one problem type, they show very poor performance with another type. However, the Two-Phase is consistently the fastest or tied for the fastest CPU time.

References

Bowman, V., 1976. On the relationship of the tchebycheff norm and the efficient frontier of multiple-criteria objectives. In: H.Thiriez, S.Zionts (Eds.), Multiple Criteria Decision Making. Vol. 130 of Lecture Notes Econom. Math. Systems. Springer-Verlag, Berlin, Germany, Ch. 53, pp. 76–85.

Cristina Bazgan, H. H., Vanderpooten, D., 2009. Solving efficiently the $0-1$ multi-objective knapsack problem. Computers & Operations Research 36, 260–279.

Geoffrion, A. M., 1968. Proper efficiency and the theory of vector maximization. Journal of Mathematical Analysis and Applications 22 (3), 618–630.

Haimes, Y. Y., Lasdon, L. S., Wismer, D. A., 1971. On a bicriterion formulation of the problems of integrated system identification and system optimization. IEEE Transactions on Systems, Man and Cybernetics 1 (3), 296–297.

Mavrotas, G., 2009. Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. Applied mathematics and computation 213 (2), 455–465.

Neumayer, P., Schweigert, D., 1994. Three algorithms for bicriteria integer linear programs. Operations-Research-Spektrum 16 (4), 267–276.

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

Steuer, R. E., Choo, E. U., 1983. An interactive weighted Tchebycheff procedure for multiple objective programming. Mathematical Programming 26 (3), 326–344.