

# A tolerance function for the multiobjective set covering problem

Lakmali Weerasena<sup>1</sup> · Margaret M. Wiecek<sup>2</sup>

Received: 3 July 2017 / Accepted: 24 April 2018 / Published online: 3 May 2018  
© Springer-Verlag GmbH Germany, part of Springer Nature 2018

**Abstract** The multiobjective set covering problem (MOSCP), an NP-hard combinatorial optimization problem, has received limited attention in the literature from the perspective of approximating its Pareto set. The available algorithms for approximating the Pareto set do not provide a bound for the approximation error. In this study, a polynomial-time algorithm is proposed to approximate an element in the weak Pareto set of the MOSCP with a quality that is known. A tolerance function is defined to identify the approximation quality and is derived for the proposed algorithm. It is shown that the tolerance function depends on the characteristics of the problem and the weight vector that is used for computing the approximation. For a set of weight vectors, the algorithm approximates a subset of the weak Pareto set of the MOSCP.

**Keywords** Approximation algorithm · Approximation error · Representation · Combinatorial optimization · Efficient solution · Max-ordering

## 1 Introduction

Multiobjective combinatorial optimization (MOCO) problems involve optimizing more than one objective function on a finite set of feasible solutions. Some well-known MOCO problems include the traveling salesman problem (TSP), the set covering problem (SCP), the minimum spanning tree problem (MSTP), and the knapsack prob-

---

✉ Lakmali Weerasena  
lakmali-weerasena@utc.edu

Margaret M. Wiecek  
wmalgor@clemson.edu

<sup>1</sup> University of Tennessee at Chattanooga, 615 McCallie Ave, Chattanooga, TN 37403, USA

<sup>2</sup> Clemson University, O-110 Martin Hall, Clemson, SC 29634, USA

lem (KP). During the past decades the interest in solving MOCO problems has been growing and surveys summarizing those efforts are given by Ehrgott [6], Ehrgott and Gandibleux [8], and Ulungu and Teghem [24]. Because there may not exist a single optimal solution to a MOCO problem as the objective functions are in conflict with each other, a solution set is defined based on the concept of Pareto optimality. Solving a MOCO problem is then understood as computing the elements in the Pareto set. In this paper, attention is given to the multiobjective set covering problem (MOSCP) that has received limited attention in the literature from the perspective of the approximation of its Pareto set.

The MOSCP has the same structure as the well-known single objective set covering problem (SOSCP). An instance of the SCP consists of a finite set of the items and a family of subsets of the items so that every item belongs to at least one of the subsets in the family. When we consider the SOSCP, each set in the family has a positive scalar cost. The goal of the SOSCP is to determine a subset of sets, among the sets in the family, so that all items are included by at least one set in the subset and the total costs of the selected sets is minimized. When there are  $p$  scalar costs for each set in the family, the SCP is called the MOSCP.

Some real-world applications such as the emergency medical service problem [5] or the reserve site selection problem [18,27] are modelled as the bi-objective SCP (BOSCP). For example, in the reserve site selection problem a reserve system containing some etiological species is partitioned into reserve sites. The species can be considered as items and the reserve sites can be represented by subsets. The problem is to find a subset of reserve sites covering a given set of species in a reserve system.

The SCP is in the category of NP problems and it is shown to be NP-complete by Karp [14]. Therefore, the SOSCP and MOSCP are also NP-hard problems. For NP-hard problems, we are typically interested in finding a near optimal solution, that is, a solution that yields the objective value that is worse than the optimal objective value by a factor of  $\rho > 0$ . An algorithm providing a near optimal solution with a factor  $\rho$  is called a  $\rho$ -approximation algorithm. Chvátal [4] and Vazirani [26] propose polynomial-time approximation algorithms for the SOSCP. Chvátal's [4] algorithm has the factor  $\rho$  being a function of the cardinality of the largest subset while Vazirani's [26] algorithm has the factor equal to  $\log m$  where  $m$  is the number of items. The SOSCP is a well-studied problem and different methods have been proposed in the literature to address it ([3, 19], and others).

The MOSCP has not received as much attention as the SOSCP and only a few studies are found in the literature. Liu [15] proposes a heuristic algorithm generating only one solution of the MOSCP. Saxena and Arora [23] formulate the SCP with quadratic objective functions and develop a heuristic enumeration technique for solving the MOSCP. The authors convert the quadratic objective functions to linear objective functions by assuming that all objective functions are differentiable and use the Gomory cut technique to get the efficient solutions. Jaskiewicz [12, 13] proposes an algorithm called the Pareto memetic algorithm (PMA) and provides a comparative study of multiobjective metaheuristics for the BOSCP. The performance of the PMA is compared with nine well-known multiobjective metaheuristics and it is concluded that the performance depends on the problem structure. Prins et al. [22] propose a heuristic-based two-phase method (TPM) to find the Pareto set of the BOSCP. In the

first phase, the scalarized SCP is solved with a heuristic algorithm to find a subset of the Pareto set called the supported Pareto set. In the second phase, the Pareto points located between two supported Pareto points are found using a heuristic algorithm. This heuristic optimizes one objective function at a time and requires that the SOSCP be reformulated by Lagrangian relaxation. Lust et al. [16] and Lust and Tuytens [17] adapt a very large-scale neighborhood search [1] for the MOSCP and compare the performance of the adaptation, a two-phase local search algorithm (2PPLS), with the PMA and TPM for the BOSCP. The performance of their algorithm also depends on the problem structure. Weerasena et al. [28] propose a heuristic algorithm that partitions the feasible region of the MOSCP into smaller subregions based on the neighbors of a reference solution and then explores each subregion for the Pareto points of the MOSCP. They compare the performance of their algorithm with the PMA, TPM and 2PPLS for the BOSCP. In general, their algorithm performs better than the TPM and PMA and is competitive with 2PPLS. Florios and Mavrotas [10] propose a method to find the Pareto set of multiobjective integer programs, called AUGMECON2, to produce all Pareto points of the MOSCP. The authors present their results on small size BOSCPs.

In summary, all the above studies but one propose heuristic approaches to obtaining the Pareto points of the MOSCP. The method by Florios and Mavrotas [10] is the only exact approach. As the MOSCP is an NP-hard problem, from the computational point of view the right direction of research is to approximate the Pareto set or its elements. However, neither the authors quoted above claim that their methods approximate the Pareto set or its elements, nor they provide performance guarantee for their algorithms. Contrary to those approaches, the objective of this paper is to propose an algorithm for approximating a point in the Pareto set of the MOSCP and compute the quality of this approximation.

To accomplish this, we follow the concept of Pareto set approximation by Papadimitriou and Yannakakis [21] and its generalization by Vanderpooten et al. [25]. The former recognize that the Pareto set of a MOCO problem is typically exponential in its size and therefore, finding all Pareto points is computationally infeasible. Even for two objective functions, determining whether a point belongs to the Pareto set is an NP problem. They propose an approximation of the Pareto set which they call the  $(1 + \epsilon)$ -approximate Pareto set and define the approximation as a set of solutions such that for every Pareto point there exists another point within a factor of  $(1 + \epsilon)$  where  $\epsilon > 0$ . This definition has already been considered for other NP-hard MOCO problems such as the TSP [2] and the KP [9], but not yet for the MOSCP. Vanderpooten et al. [25] define sets representing the Pareto set and propose the concept of tolerance function as a tool for modeling representation quality.

Using these authors' terminology, in this paper we develop an algorithm to approximate a weak Pareto point of the MOSCP with the quality given by an associated tolerance function. The algorithm appears to be the first approach in the literature to provide the approximation along with a known tolerance function, a feature not available for the existing methods. In the algorithm, the weighted-max-ordering formulation [7] of the MOSCP serves as the single objective optimization problem whose optimal solution provides the approximation. For a set of weight vectors, the algorithm approximates a subset of the weak Pareto set of the MOSCP.

The paper is organized as follows. In Sect. 2, the formulation and terminology of the MOSCP are provided. An exact approach to computing the Pareto points and the concept of representation of the Pareto set for the MOSCP are also included. The algorithm and the derivation of the tolerance function are presented in Sect. 3. The computational work is presented in Sect. 4 and the paper is concluded in Sect. 5.

## 2 Problem formulation

In the MOSCP, there is a set of  $m$  items,  $E = \{e_1, e_2, \dots, e_m\}$  with the index set  $I = \{i : i = 1, 2, \dots, m\}$ , and a set of  $n$  subsets of  $E$ ,  $S = \{S_1, S_2, \dots, S_n\}$  with the index set of  $J = \{j : j = 1, 2, \dots, n\}$ . The items are grouped into subsets of  $E$  and an item  $e_i$  in  $E$  is covered by a set  $S_j$  provided  $e_i$  in  $S_j$ . An instance of the SCP is given by the sets  $E$  and  $S$ . The binary coefficient  $a_{ij}$  is equal to 1 if an item  $e_i$  is covered by a set  $S_j$  and otherwise  $a_{ij}$  is equal to 0 for  $i \in I$  and  $j \in J$ . A cover is defined as a sub-collection  $\{S_j : J \in J^* \subseteq J\}$  which is a subset of  $S$  such that all items of  $E$  are covered, where  $J^*$  is the index set of selected sets for the sub-collection.

Let  $x \in \mathcal{R}^n$  be the decision variable defined as follows,

$$x_j = \begin{cases} 1 & \text{if } S_j \text{ is selected for a cover} \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } j \in J.$$

As mentioned in the Introduction, in the SCP each item must be covered by at least one set, i.e., a cover is sought to cover all items. Thus the feasible region  $X$  is defined as  $X = \{x \in \mathcal{R}^n : \sum_{j \in J} a_{ij} x_j \geq 1 \text{ for } i \in I \text{ and } x_j \in \{0, 1\} \text{ for } j \in J\}$ .

Every feasible vector  $x \in X$  is associated with a cover and vice versa.

The MOSCP has  $p$  conflicting objectives. Let  $c_j^q > 0$  be the cost of a set  $S_j$  with respect to objective  $q$  for  $q = 1, 2, \dots, p$ . The cost of a cover with respect to objective  $q$  is given by  $\sum_{j \in J^*} c_j^q$ . In the MOSCP the goal is to find a cover such that the costs with respect to all objective functions are minimized.

The MOSCP can be presented as follows:

$$\min z(x) = \left[ z_1(x) = \sum_{j=1}^n c_j^1 x_j, \quad z_2(x) = \sum_{j=1}^n c_j^2 x_j, \quad \dots, \quad z_p(x) = \sum_{j=1}^n c_j^p x_j \right]$$

subject to  $x \in X$ .

(2.1)

*Example 1* Let  $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$  be a set of items and  $S_1 = \{e_1, e_2\}$ ,  $S_2 = \{e_1, e_2, e_3, e_4\}$ ,  $S_3 = \{e_3, e_4, e_5, e_6\}$ ,  $S_4 = \{e_1, e_3, e_6\}$  be four sets with cost vectors  $c^1 = (2, 1, 4, 3)^T$  and  $c^2 = (1, 3, 5, 1)^T$ , respectively. For example, the collection  $\{S_2, S_3\}$  covers all items in  $S$  and thus forms a cover.

### 2.1 Preliminaries and basic definitions

Let  $\mathcal{R}^p$  be a finite dimensional Euclidean vector space. We first introduce some basic notations. For  $y^1, y^2 \in \mathcal{R}^p$ , to define an ordering relation on  $\mathcal{R}^p$ , the following notation will be used for  $p > 1$ .

1.  $y^1 \leq y^2$  if  $y^1_k \leq y^2_k$  for all  $k = 1, 2, \dots, p$ ;
2.  $y^1 \leq y^2$  if  $y^1_k \leq y^2_k$  for all  $k = 1, 2, \dots, p$  and  $y^1 \neq y^2$ ;
3.  $y^1 < y^2$  if  $y^1_k < y^2_k$  for all  $k = 1, 2, \dots, p$ .

In particular, using componentwise orders, the nonnegative orthant of  $\mathcal{R}^p$  is defined as  $\mathcal{R}^p_{\leq} = \{y \in \mathcal{R}^p : y \geq 0\}$ , the nonzero orthant of  $\mathcal{R}^p$  is defined as  $\mathcal{R}^p_{\neq} = \{y \in \mathcal{R}^p : y \geq 0\}$  and positive orthant of  $\mathcal{R}^p$  is defined as  $\mathcal{R}^p_{>} = \{y \in \mathcal{R}^p : y > 0\}$ .

Solving the MOSCP is understood as finding its efficient solutions and Pareto outcomes.

**Definition 1** A point  $x^* \in X$  is called

1. a weakly efficient solution of the MOSCP if there does not exist  $x \in X$  such that  $z(x) < z(x^*)$ .
2. an efficient solution of the MOSCP if there does not exist  $x \in X$  such that  $z(x) \leq z(x^*)$ .

The set of all efficient solutions and the set of all weakly efficient solutions are denoted by  $X_E$  and  $X_{wE}$  respectively. The set of all attainable outcomes,  $Y$ , for feasible solutions,  $x \in X$ , is obtained by evaluating the  $p$  objective functions. That is  $Y := z(X) \subset \mathcal{R}^p$ . The image  $z(x) \in Y$  of a (weakly) efficient solution is called a (weak) Pareto outcome. The image of  $(X_{wE}) X_E$  is denoted by  $(z(X_{wE})) z(X_E)$  and is referred to as the (weak) Pareto set. Given the definition of an (weakly) efficient solution of the MOSCP, we define a (weak) Pareto cover for the MOSCP.

**Definition 2** A (weak) Pareto cover is a cover that is associated with an (weakly) efficient solution of the MOSCP.

### 2.2 Finding efficient solutions of the MOSCP

The approximation algorithm we propose in this paper is developed based on the weighted-max-ordering method, an exact method for finding the efficient solutions of multiobjective optimization problems (MOOPs). In this section, we first briefly review this method and include the result needed when proving the accuracy of the algorithm.

#### 2.2.1 Exact method

The underlying concept of the weighted-max-ordering method is to minimize the highest (worst) objective function value,  $z_q$ . The weighted-max-ordering problem associated with the MOSCP can be written as follows:

$$\begin{aligned} \min \quad & \max_{q=1,2,\dots,p} \lambda_q z_q(x) \\ \text{subject to} \quad & x \in X \end{aligned} \tag{2.2}$$

where  $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_p)^T \in \mathcal{R}_{>}^p$ . The following result is useful for the MOOP [7, p. 132] and we state it for the MOSCP.

**Proposition 1** *Let  $x^* \in X$ . Then  $x^* \in X_{wE}$  of the MOSCP if and only if there exists  $\lambda \in \mathcal{R}_{>}^p$  such that  $x^*$  is an optimal solution of problem (2.2).*

The details of an error function for the MOSCP are given in Sect. 2.2.2.

### 2.2.2 Approximation

As mentioned in the Introduction, approximating the Pareto set of a MOCO problem with a performance guarantee is a motivating challenge. The concept of  $(1 + \epsilon)$ -approximate Pareto set for MOCO problems defined by Papadimitriou and Yannakakis [21] and others makes use of a constant  $\epsilon$  to quantify the representation error. To be able to deal with the Pareto set of the MOSCP we define the tolerance function and a related representation as in Vanderpooten et al. [25].

**Definition 3** Let  $Y$  be a set in  $\mathcal{R}^p$ . A vector-valued function  $t : \mathcal{R}^p \rightarrow \mathcal{R}^p$  such that

- (1) for all  $y \in Y$ ,  $y \leq t(y)$ , and
- (2) for all  $y^1, y^2 \in Y$ , if  $y^1 \leq y^2$  then  $t(y^1) \leq t(y^2)$ ,

is called a tolerance function.

*Example 2* Let  $Y \subset \mathcal{R}^2$  be the outcome set associated with the BOSCP in *Example 1*. Then  $Y = \{y^1 = (6, 6)^T, y^2 = (5, 8)^T, y^3 = (7, 9)^T, y^4 = (8, 9)^T, y^5 = (9, 7)^T, y^6 = (10, 10)^T\}$ . A function  $t : \mathcal{R}^2 \rightarrow \mathcal{R}^2$ , defined as  $t(y) = 1.5y$ , satisfies Definition 3 and therefore is a tolerance function.

In the next section we present the main theoretical results of this paper.

## 3 Approximating a Pareto point of the MOSCP

We now develop the algorithm to approximate an element in the weak Pareto set of the MOSCP. We prove that the tolerance  $\epsilon(\cdot)$  yield by the algorithm depends on the parameter  $\lambda$ , where  $\lambda \in \mathcal{R}_{>}^p$ , and the problem data, and that  $t(y) = (1 + \epsilon(\cdot))y$  is a tolerance function as defined in Definition 3.

### 3.1 Vector cost effectiveness

We first present the concept of the cost effectiveness of a set. The key idea of our approach is based on the following observation: when we select a set to include in a minimum cost cover, we consider not only the cost of the set but also the coverage of the set, that is, the items this set covers. Refer again to *Example 1*. The set  $S_1$  covers 2 items for 2 unit cost with respect to the first objective, while the set  $S_2$  covers 4 items for 1 unit cost with respect to the same objective. Intuitively, it is clear that selecting  $S_2$  is more beneficial than selecting  $S_1$  since it covers more items for less cost. Therefore, when we construct a minimum cost cover, it seems reasonable to choose

a set having a small cost and a large coverage. Consequently, in the construction of the algorithm, when selecting a set we consider these two aspects: the minimum cost and the maximum coverage. This is equivalent to selecting a set having a small ratio of cost to coverage. The ratio (cost/coverage) is called the cost effectiveness of the set and is denoted by  $\alpha_j$  for a set  $S_j$ . We propose a concept for defining cost effectiveness based on the weighted-max-ordering method which defines the cost effectiveness as a vector of cost effectiveness ratios, where each ratio is the scalar cost effectiveness with respect to one objective. The vector cost effectiveness of a set  $S_j$  is defined as follows:

$$\begin{aligned} \alpha_j &= \left[ \lambda_1 c_j^1 / |S_j|, \dots, \lambda_p c_j^p / |S_j| \right]^T \\ &= [\alpha_j^1, \dots, \alpha_j^p]^T, \end{aligned} \tag{3.1}$$

where  $|S_j|$  is the cardinality of the set  $S_j$  and  $\lambda \in \mathcal{R}_{>}^p$ . We use Definition 4 to select a set to include in a cover with respect to the vector cost effectiveness preference.

**Definition 4** A set  $S_{j_1}$  is preferred to a set  $S_{j_2}$  for  $j_1 \neq j_2$  with respect to the vector cost effectiveness, denoted as  $\alpha_{j_1} \underset{mo}{\succeq} \alpha_{j_2}$ , if

$$\max_{q=1,2,\dots,p} [\alpha_{j_1}^1, \dots, \alpha_{j_1}^p]^T \leq \max_{q=1,2,\dots,p} [\alpha_{j_2}^1, \dots, \alpha_{j_2}^p]^T.$$

### 3.2 Algorithm

The concept of vector cost effectiveness leads to the development of an approximation algorithm that returns a cover for the MOSCP associated with the weight  $\lambda \in \mathcal{R}_{>}^p$ . The algorithm employs Procedure 1 that identifies a preferred sets  $S_{j^*}$  and the associated vector cost effectiveness  $\alpha_{j^*}$  according to Definition 4. The algorithm is presented in generic form and referred to as Algorithm 1. The symbols  $\bar{E}$ ,  $\bar{J}$  and  $C$  denote the set of currently covered items in  $E$ , the index set of selected sets for covering items in  $E$ , and a cover, respectively. We assign  $\alpha_{j^*}$  to the items covered by the set  $S_{j^*}$  as their prices and refer to  $\alpha_{j^*}$  as the price of an item covered by  $S_{j^*}$ . We let  $p(e_i)$  denote the price of item  $e_i$ .

The algorithm approximately solves a collection of SOSCPs obtained from scalarizing the MOSCP with weights  $\lambda \in \mathcal{R}_{>}^p$ . Pseudo-codes of the algorithm and an auxiliary procedure are given below. The outline of Algorithm 1 is the following: the algorithm starts with the empty sets  $\bar{E}$ ,  $\bar{J}$  and the empty set as the current cover, that is, initially  $C = \emptyset$ . In the main step, Procedure 1 is called to calculate the vector cost effectiveness values,  $\alpha_{j_s}$ , for all sets based on the scalarized costs and uncovered items, and to determine a set to be added to the current cover. Once the best set,  $S_{j^*}$ , has been selected based on the Definition 4, the index  $j^*$  is added to the index set,  $\bar{J}$ , and all items in  $S_{j^*}$  are added to  $\bar{E}$ . For each item  $e_i$  covered in this step, the price is set as  $p(e_i) = \alpha_{j^*}$ . Algorithm 1 is run until all items in  $E$  have been covered. Upon termination, the algorithm yields a cover associated with a feasible solution,  $\bar{x}$ , where the components of  $\bar{x}$  identify the sets selected to be in the cover.

**Algorithm 1** Generic Algorithm for the MOSCP

- 
- 1: *Input:*  $E, S, c^q$  for  $q = 1, 2, \dots, p, \lambda \in \mathcal{R}_{>}^p$  where  $\sum_{q=1}^p \lambda_q = 1$ .
  - 2: *Initialization:*  $\bar{E} = \emptyset, \bar{J} = \emptyset$  and  $C = \emptyset$ .
  - 3: **while**  $\bar{E} \neq E$  **do**
  - 4:   *Call Procedure 1 to obtain*  $\alpha_{j^*}$  *and*  $S_{j^*}$ .
  - 5:   **for**  $i = 1 \rightarrow n$  **do**
  - 6:     **if**  $e_i \in S_{j^*} \cap (E \setminus \bar{E})$  **then**
  - 7:        $p(e_i) = \alpha_{j^*}, \bar{J} = \bar{J} \cup j^*, \bar{J} \subseteq J$
  - 8:     **end if**
  - 9:   **end for**
  - 10: **end while**
  - 11: *Return:* Cover,  $C = \{S_j : j \in \bar{J}\}$ .

**Pseudo-code of the generic algorithm****Procedure 1** Vector cost effectiveness

- 
- 1: *Initialization:*  $\alpha_{j^*} = (\infty, \infty, \dots, \infty)$
  - 2: **for**  $j = 1 \rightarrow n$  **do**
  - 3:   **if**  $j \in J \setminus \bar{J}$  **then**
  - 4:      $\alpha_j = (\alpha_j^1, \alpha_j^2, \dots, \alpha_j^p)^T = \left( \lambda_1 \frac{c_j^1}{|S_j \cap (E \setminus \bar{E})|}, \lambda_2 \frac{c_j^2}{|S_j \cap (E \setminus \bar{E})|}, \dots, \lambda_p \frac{c_j^p}{|S_j \cap (E \setminus \bar{E})|} \right)^T$
  - 5:     **if**  $\max\{\alpha_j^1, \alpha_j^2, \dots, \alpha_j^p\} \leq \max\{\alpha_{j^*}^1, \alpha_{j^*}^2, \dots, \alpha_{j^*}^p\}$  **then**
  - 6:        $\alpha_{j^*} = \alpha_j$
  - 7:     **end if**
  - 8:   **end if**
  - 9: **end for**
  - 10: *Return:*  $\alpha_{j^*}$  *and*  $S_{j^*}$

**Pseudo-code of Procedure 1**

To evaluate the performance of Algorithm 1, we show that the cost of the cover associated with  $\bar{x}$ , the solution yield by Algorithm 1 with a weight  $\lambda$ , can be used to get a bound on the cost of a weak Pareto cover associated with the optimal objective function value of problem (2.2).

In the proofs presented in this section, we use the symbol  $\lceil a \rceil$  to denote the ceiling of  $a$  which is the smallest integer not less than  $a$ , where  $a$  is a real number. Given the data of the MOCSP, we define

$$\begin{aligned} c_{\min}^{k_1} &= \min\{c_1^{k_1}, c_2^{k_1}, \dots, c_n^{k_1}\} \\ c_{\max}^{k_2} &= \max\{c_1^{k_2}, c_2^{k_2}, \dots, c_n^{k_2}\}, \end{aligned} \quad (3.2)$$



for  $k_1, k_2 \in \{1, 2, \dots, p\}$ .

The main observation about Algorithm 1 is given in Lemma 1 in which we estimate the price  $p(e_k)$  of an item  $e_k$  assigned by Algorithm 1, where  $e_k$  is an item covered in the iteration  $k$ .

**Lemma 1** *Let  $x^* \in X$  be an optimal solution of problem (2.2) for  $\lambda \in \mathcal{R}_>^p$  and  $e_k$  be an item covered in the  $k^{\text{th}}$  iteration of Algorithm 1. Then*

$$p(e_k) \leq \frac{\delta}{m - k + 1} z(x^*)$$

where  $\delta = \max_{\substack{k_1, k_2=1, 2, \dots, p, \\ k_1 \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{k_2}^{\max}}{c_{k_1}^{\min}} \right] \right\}$ .

*Proof* By Proposition 1,  $x^* \in X_{wE}$ . Let  $S^*$  be the weak Pareto cover associated with  $x^*$ . Then  $S^* = \{S_1^*, S_2^*, \dots, S_{r^*}^*\}$  where  $\{S_1^*, S_2^*, \dots, S_{r^*}^*\}$  is a collection of sets from the family  $S$  and  $r^*$  is the number of sets in the weak Pareto cover. We know that  $E = \bigcup_{l=1}^{r^*} S_l^*$  and  $\bigcup_{l=1}^{r^*} S_l^*$  is a weak Pareto cover with the cost  $[\lambda_1 \sum_{l=1}^{r^*} c_l^1, \dots, \lambda_p \sum_{l=1}^{r^*} c_l^p]^T$ . At any iteration of Algorithm 1, the uncovered items are given by  $E \setminus \bar{E}$ . Since a weak Pareto cover covers all items, at any iteration,  $E \setminus \bar{E}$  can be expressed as  $E \setminus \bar{E} = \bigcup_{l=1}^{r^*} (S_l^* \cap (E \setminus \bar{E}))$ . Thus we have,

$$|E \setminus \bar{E}| \leq \sum_{l=1}^{r^*} |S_l^* \cap (E \setminus \bar{E})|. \tag{3.3}$$

Note that at any iteration, the uncovered items in  $S_l^*$  are contained in  $S_l^* \cap (E \setminus \bar{E})$  for  $l = 1, 2, \dots, r^*$  and if all items in  $S_l^*$  are covered then  $S_l^* \cap (E \setminus \bar{E}) = \emptyset$ . Consider now the iteration in which an item  $e_k$  is covered. Let  $S_{j^*}$  be the set selected in this iteration to cover the item  $e_k$  and let  $\alpha_{j^*}$  be the vector cost effectiveness of this set. We obtain a bound for  $\alpha_{j^*}$  using the weak Pareto cover  $S^*$ . Let  $S_l^* \in S^*$  and  $\alpha_l = [\alpha_l^1, \dots, \alpha_l^p]^T$  be the cost effectiveness of  $S_l^*$ . We consider the following two cases.

**Case 1** Let  $l \in J \setminus \bar{J}$ , that is, some items in  $S_l^*$  are not covered by Algorithm 1 and thus the set  $S_l^*$  is a candidate for  $S_{j^*}$ . In this case, the uncovered items in  $S_l^*$  are given by  $S_l^* \cap (E \setminus \bar{E})$  and

$$S_l^* \cap (E \setminus \bar{E}) = S_j \cap (E \setminus \bar{E}) \text{ for some } j \in J \setminus \bar{J}.$$

We calculate  $\alpha_j$  vectors for all unselected sets based on formula (3.1) and select a best set using Definition 4. Thus, by Definition 4, we obtain

$$(\alpha_{j^*}^1, \dots, \alpha_{j^*}^p)^T \underset{mo}{\geq} \frac{1}{|S_l^* \cap (E \setminus \bar{E})|} (\lambda_1 c_l^1, \dots, \lambda_p c_l^p)^T \text{ for all } l \in J \setminus \bar{J}. \tag{3.4}$$



We define  $\delta_{k_2} = \max_{k=1,2,\dots,p, k \neq k_2} \left\{ \lambda_{k_2} \left[ \frac{c_{\max}^{k_2}}{c_{\min}^k} \right] \right\}$ . Then we get

$$\begin{pmatrix} \alpha_{j^*}^1 |S_l^* \cap (E \setminus \bar{E})| \\ \vdots \\ \alpha_{j^*}^{k_1} |S_l^* \cap (E \setminus \bar{E})| \\ \vdots \\ \alpha_{j^*}^{k_2} |S_l^* \cap (E \setminus \bar{E})| \\ \vdots \\ \alpha_{j^*}^p |S_l^* \cap (E \setminus \bar{E})| \end{pmatrix} \leq \begin{pmatrix} \lambda_{k_2} \left[ \frac{c_{\max}^{k_2}}{c_{\min}^1} \right] c_l^1 \\ \vdots \\ \lambda_{k_2} \left[ \frac{c_{\max}^{k_2}}{c_{\min}^{k_1}} \right] c_l^{k_1} \\ \vdots \\ \lambda_{k_2} c_l^{k_2} \\ \vdots \\ \lambda_{k_2} \left[ \frac{c_{\max}^{k_2}}{c_{\min}^p} \right] c_l^p \end{pmatrix} \leq \delta_{k_2} \begin{pmatrix} c_l^1 \\ \vdots \\ c_l^{k_1} \\ \vdots \\ c_l^{k_2} \\ \vdots \\ c_l^p \end{pmatrix} \text{ for } l = 1, 2, \dots, r^*. \tag{3.8}$$

Since there are  $p$  choices for  $k_2$  in inequality (3.8), we have  $\delta_{k_2}$  for  $k_2 = 1, 2, \dots, p$ . For  $k_2 = 1$ , inequality (3.8) can be written as:

$$|S_l^* \cap (E \setminus \bar{E})|(\alpha_{j^*}^1, \dots, \alpha_{j^*}^p)^T \leq \delta_1 (\lambda_1 c_l^1, \dots, \lambda_p c_l^p)^T \text{ for all } l = 1, 2, \dots, r^*, \tag{3.9}$$

where  $\delta_1 = \max_{k=1,2,\dots,p, k \neq 1} \left\{ \lambda_1 \left[ \frac{c_{\max}^1}{c_{\min}^k} \right] \right\}$ . For  $k_2 = 2$ , inequality (3.8) can be written as:

$$|S_l^* \cap (E \setminus \bar{E})|(\alpha_{j^*}^1, \dots, \alpha_{j^*}^p)^T \leq \delta_2 (\lambda_1 c_l^1, \dots, \lambda_p c_l^p)^T \text{ for all } l = 1, 2, \dots, r^*, \tag{3.10}$$

where  $\delta_2 = \max_{k=1,2,\dots,p, k \neq 2} \left\{ \lambda_2 \left[ \frac{c_{\max}^2}{c_{\min}^k} \right] \right\}$ . By continuing this process for  $k_2 = p$ , inequality (3.8) becomes

$$|S_l^* \cap (E \setminus \bar{E})|(\alpha_{j^*}^1, \dots, \alpha_{j^*}^p)^T \leq \delta_p (\lambda_1 c_l^1, \dots, \lambda_p c_l^p)^T \text{ for all } l = 1, 2, \dots, r^*, \tag{3.11}$$

where  $\delta_p = \max_{k=1,2,\dots,p, k \neq p} \left\{ \lambda_p \left[ \frac{c_{\max}^p}{c_{\min}^k} \right] \right\}$ . Let  $\delta = \max_{k_2=1,2,\dots,p} \left\{ \max_{\substack{k=1,2,\dots,p, \\ k \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{\max}^{k_2}}{c_{\min}^k} \right] \right\} \right\}$   
 $= \max_{\substack{k_1, k_2=1,2,\dots,p, \\ k_1 \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{\max}^{k_2}}{c_{\min}^{k_1}} \right] \right\}$ . Then inequality (3.11) becomes

$$|S_l^* \cap (E \setminus \bar{E})|(\alpha_{j^*}^1, \dots, \alpha_{j^*}^p)^T \leq \delta (\lambda_1 c_l^1, \dots, \lambda_p c_l^p)^T \text{ for all } l = 1, 2, \dots, r^*, \tag{3.12}$$

Summing over all sets in the weak Pareto cover, inequality (3.12) gives the following:

$$\sum_{l=1}^{r^*} |S_l^* \cap (E \setminus \bar{E})| (\alpha_{j^*}^1, \dots, \alpha_{j^*}^p)^T \leq \delta (z_1(x^*), \dots, z_p(x^*))^T \text{ for all } l = 1, 2, \dots, r^* \tag{3.13}$$

where  $(z_1(x^*), \dots, z_p(x^*))^T = \left( \sum_{l=1}^{r^*} c_l^1, \dots, \sum_{l=1}^{r^*} c_l^p \right)^T$ .

Using inequality (3.3), inequality (3.13) can be written as follows:

$$|(E \setminus \bar{E})| (\alpha_{j^*}^1, \dots, \alpha_{j^*}^p)^T \leq \delta (z_1(x^*), \dots, z_p(x^*))^T, \tag{3.14}$$

where  $\delta = \max_{\substack{k_1, k_2=1, 2, \dots, p, \\ k_1 \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{k_2}^{\max}}{c_{k_1}^{\min}} \right] \right\}$ . In the iteration in which the item  $e_k$  is covered,

the number of uncovered items is  $(m - k + 1)$ . That is,  $|(E \setminus \bar{E})| = m - k + 1$ . Thus, inequality (3.14) becomes:

$$(\alpha_{j^*}^1, \dots, \alpha_{j^*}^p)^T \leq \frac{\delta}{m - k + 1} (z_1(x^*), \dots, z_p(x^*))^T \text{ for all } l = 1, 2, \dots, r^*. \tag{3.15}$$

Since  $p(e_k) = \alpha_{j^*}$ , from inequality (3.15) we conclude

$$p(e_k) \leq \frac{\delta}{m - k + 1} z(x^*).$$

□

Corollary 1 implies that the cost of the cover associated with the solution yield by Algorithm 1 is equal to the cost of covering all items.

**Corollary 1** *If  $\bar{x}$  is yield by Algorithm 1 for  $\lambda \in \mathcal{R}_>^p$  then  $\bar{x}$  is a feasible solution of problem (2.2) for  $\lambda \in \mathcal{R}_>^p$  and*

$$[\lambda_1 z_1(\bar{x}), \dots, \lambda_p z_p(\bar{x})]^T = \sum_{k=1}^m p(e_k).$$

*Proof* Clearly  $\bar{x}$  is a feasible solution of (2.2) for  $\lambda \in \mathcal{R}_>^p$  since Algorithm 1 returns a feasible cover and  $\bar{x}$  is the solution associated with this cover. In the algorithm, the cost of each set selected in each iteration is distributed among the items covered in that iteration. Therefore, the cost of covering all items, at the termination of Algorithm 1, is equal to  $\sum_{k=1}^m p(e_k)$ . On the other hand, the cost of the selected cover is given by the objective vector of (2.2) for  $\lambda \in \mathcal{R}_>^p$ ,  $[\lambda_1 z_1(\bar{x}), \dots, \lambda_p z_p(\bar{x})]^T$ . By definition, a cover covers all items and thus, we have,  $[\lambda_1 z_1(\bar{x}), \dots, \lambda_p z_p(\bar{x})]^T = \sum_{k=1}^m p(e_k)$ . □

Based on the findings above we present Theorem 1 which is the main result of this work. We prove that Algorithm 1 yields a solution  $\bar{x}$  such that  $z(\bar{x}) \leq (1 + \epsilon(\lambda))z(x^*)$ , where  $x^*$  is an optimal solution of problem (2.2) for  $\lambda \in \mathcal{R}_>^p$  and  $\epsilon(\lambda)$  is the tolerance associated with the parameter  $\lambda$  and problem data.

**Theorem 1** *Let  $\bar{x} \in X$  be a feasible solution of problem (2.2) for  $\lambda \in \mathcal{R}_>^p$  yield by Algorithm 1 and  $x^*$  be an optimal solution of problem (2.2) for  $\lambda \in \mathcal{R}_>^p$ . Then*

$$z(\bar{x}) \leq \frac{(1 + \log m)\delta}{\lambda_{\min}} z(x^*),$$

where  $\delta = \max_{\substack{k_1, k_2=1, 2, \dots, p, \\ k_1 \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{k_2}^{\max}}{c_{k_1}^{\min}} \right] \right\}$  and  $\lambda_{\min} = \min\{\lambda_1, \dots, \lambda_p\}$ .

*Proof* Using Lemma 1, and summing over all items we get,

$$\sum_{k=1}^m p(e_k) \leq \sum_{k=1}^m \frac{\delta}{m - k - 1} z(x^*) = (1 + 1/2 + \dots + 1/m)\delta z(x^*),$$

where  $\delta = \max_{\substack{k_1, k_2=1, 2, \dots, p, \\ k_1 \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{k_2}^{\max}}{c_{k_1}^{\min}} \right] \right\}$ . As  $(1 + 1/2 + \dots + 1/m) \approx \log m$ , we have  $(1 + 1/2 + \dots + 1/m) \leq (1 + \log m)$  and obtain

$$\sum_{k=1}^m p(e_k) \leq (1 + \log m)\delta z(x^*) \tag{3.16}$$

and using Corollary 1

$$[\lambda_1 z_1(\bar{x}), \lambda_2 z_2(\bar{x}), \dots, \lambda_p z_p(\bar{x})]^T \leq (1 + \log m)\delta z(x^*). \tag{3.17}$$

Let  $\lambda_{\min} = \min\{\lambda_1, \dots, \lambda_p\}$ , then inequality (3.17) gives the following:

$$\lambda_{\min} z(\bar{x}) \leq [\lambda_1 z_1(\bar{x}), \lambda_2 z_2(\bar{x}), \dots, \lambda_p z_p(\bar{x})]^T \leq (1 + \log m)\delta z(x^*)$$

where  $\delta = \max_{\substack{k_1, k_2=1, 2, \dots, p, \\ k_1 \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{k_2}^{\max}}{c_{k_1}^{\min}} \right] \right\}$ . This completes the proof. □

Theorem 1 implies that Algorithm 1 produces a solution within a factor  $\frac{(1 + \log m)\delta}{\lambda_{\min}}$  of an optimal solution of problem (2.2) for  $\lambda \in \mathcal{R}_>^p$ . Given this factor, the tolerance of computing an optimal solution of (2.2) for  $\lambda \in \mathcal{R}_>^p$  is

$$\epsilon(\lambda) = \frac{(1 + \log m)\delta}{\lambda_{\min}} - 1 \text{ where } \delta = \max_{\substack{k_1, k_2=1, 2, \dots, p, \\ k_1 \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{k_2}^{\max}}{c_{k_1}^{\min}} \right] \right\}. \tag{3.18}$$

Note that  $\epsilon(\lambda) > 0$  since  $(1 + \log m) > 1$  and  $\delta/\lambda_{\min} > 1$ . It is also easy to show that the associated vector-valued function  $t(y) = (1 + \epsilon(\lambda))y$  satisfies Definition 3 and hence is indeed a tolerance function.

Additionally, considering the relationship between an optimal solution of problem (2.2) for  $\lambda \in \mathcal{R}_{>}^p$  and the weak efficient solutions of the MOSCP, we obtain the following corollary.

**Corollary 2** *For every  $x \in X_{wE}$  of the MOSCP, there exist a weight vector  $\lambda \in \mathcal{R}_{>}^p$  and a solution  $\bar{x} \in X$  yield by Algorithm 1 satisfying the following condition:*

$$z(\bar{x}) \leq t(z(x)) = (1 + \epsilon(\lambda))z(x), \quad (3.19)$$

where

$$(1 + \epsilon(\lambda)) = \frac{(1 + \log m)\delta}{\lambda_{\min}}, \quad \delta = \max_{\substack{k_1, k_2=1, 2, \dots, p, \\ k_1 \neq k_2}} \left\{ \lambda_{k_2} \left[ \frac{c_{\max}^{k_2}}{c_{\min}^{k_1}} \right] \right\},$$

and  $\lambda_{\min} \min\{\lambda_1, \dots, \lambda_p\}$ .

(3.20)

*Proof* Let  $x \in X_{wE}$ . Then, by Proposition 1, there exists  $\lambda \in \mathcal{R}_{>}^p$  such that  $x$  is an optimal solution of (2.2) for  $\lambda \in \mathcal{R}_{>}^p$ . We show that there exists a solution  $\bar{x} \in X$  such that (3.19) and (3.20) hold. Suppose that Algorithm 1 is executed for  $\lambda$  and returns a cover associated with the solution  $\bar{x}$ . Then by Theorem 1, we have  $z(\bar{x}) \leq \frac{(1 + \log m)\delta}{\lambda_{\min}}z(x)$ , and (3.19) and (3.20) hold.  $\square$

Since Algorithm 1, by construction, is run for a fixed  $\lambda \in \mathcal{R}_{>}^p$ , we prove that its running time is polynomial.

**Theorem 2** *Algorithm 1 is a polynomial-time algorithm.*

*Proof* In the generic algorithm the loop in the main step iterates for  $O(m)$  time, where  $|E| = m$ . In Procedure 1, the maximum component of each cost effectiveness vector  $\alpha_j$  for  $j = 1, 2, \dots, n$  can be found in  $O(\log p)$  time using a priority heap, where  $p$  is the number of objectives [20]. Then we get the minimum of these maximum components in constant time since we update the minimum value as we calculate each maximum. Therefore the total running time of Algorithm 1 is  $O(n \log p)$  and thus Algorithm 1 is a polynomial-time algorithm.  $\square$

To assess the actual quality of  $(1 + \epsilon(\lambda))$ -approximation of the weak Pareto point produced by Algorithm 1, it is desirable to compare the point computed by this algorithm with the corresponding true Pareto point. To accomplish this, we gauge the quality of the approximation with a measure that is defined to resemble the meaning of the theoretical factor given in Theorem 1.

Let Algorithm 1 run for  $\lambda \in \mathcal{R}_{>}^p$  and return a feasible solution  $\bar{x}$  to the MOSCP. Let  $x^*$  be a weakly efficient solution of the MOSCP resulting from solving problem (2.2) with the weight  $\lambda$ . Based on Theorem 1, the theoretical factor  $(1 + \epsilon(\lambda))$  of Algorithm 1 satisfies

$$\frac{z_q(\bar{x})}{z_q(x^*)} \leq (1 + \epsilon(\lambda)) = \frac{(1 + \log m)\delta}{\lambda_{\min}} \quad (3.21)$$

for  $q = 1, 2, \dots, p$ , and so it also satisfies

$$\max_{q=1,2,\dots,p} \left\{ \frac{z_q(\bar{x})}{z_q(x^*)} \right\} \leq (1 + \epsilon(\lambda)) = \frac{(1 + \log m)\delta}{\lambda_{\min}}. \tag{3.22}$$

We calculate the experimental factor of Algorithm 1, reflecting the actual quality of error of the approximation, as follows:

$$(1 + \epsilon_c(\lambda)) = \max_{q=1,2,\dots,p} \left\{ \frac{z_q(\bar{x})}{z_q(x^*)} \right\}, \tag{3.23}$$

where  $\epsilon_c(\lambda)$  denotes the experimental tolerance of computing an optimal solution to problem (2.2) for  $\lambda \in \mathcal{R}_>^p$ . Both factors are reported and the effectiveness of Algorithm 1 is practically evaluated in the next section.

### 4 Computational experiments

In this section we present the computational results we obtained with Algorithm 1. We analyzed the performance of the algorithm using 44 benchmark BOSCP instances that are available in the library invOptLib [11]. The algorithm was implemented using MATLAB R2017b interface and experiments were conducted using a computer with an I-7 processor and 8 GB RAM. The exact weak Pareto points  $z(x^*)$  of the instances were computed by solving problem (2.2) exactly using the MATLAB intlinprog function.

Consider the instance 2scp11A that has  $m = 10$  items and  $n = 100$  sets. For  $\lambda = [0.98, 0.02]$ , the weak Pareto outcome and the outcome yield by Algorithm 1 are  $z(x^*) = [89, 531]^T$  and  $z(\bar{x}) = [108, 600]^T$ , respectively. Applying (3.21) and noting that  $c_{\max}^1 = 100$ ,  $c_{\min}^1 = 2$ ,  $c_{\max}^2 = 100$ ,  $c_{\min}^2 = 3$ , we obtain  $(1 + \epsilon(\lambda)) = \frac{(1+\log 10) \cdot \max\{0.98 \cdot \lceil 100/3 \rceil, 0.02 \cdot \lceil 100/2 \rceil\}}{0.02} = 3332$ . Applying (3.23), we obtain  $(1 + \epsilon_c(\lambda)) = \max\{z_1(\bar{x})/z_1(x^*), z_2(\bar{x})/z_2(x^*)\}^T = 1.21$ . It is clear that due to the involvement of the weight vectors in the theoretical factor, that factor is very large while the experimental factor is significantly smaller. The vector-valued tolerance function associated with the experimental factor can be given by  $t(y) = 1.21y$ .

In a similar way, we solved the same instance for 99 weight vectors  $\lambda \in \mathcal{R}_>^2$  from a set  $\Lambda = \{[0.01, 0.99], [0.02, 0.98], \dots, [0.99, 0.01]\}$  with a 0.01 increment in which, without loss of generality, each vector was normalized so that  $\sum_{q=1,2} \lambda_q = 1$ . For each  $\lambda \in \Lambda$ , we ran Algorithm 1 and also solved the weighted-max-ordering problem (2.2). For all 99 cases, the experimental factors are contained in the interval (1.00, 1.44) and so the experimental tolerance  $\epsilon_c(\lambda)$  is in the interval (0, 0.44). The vector-valued tolerance function associated with the experimental factor for the set of weight vectors in  $\Lambda$  can be given by  $t(y) = 1.44y$ .

The computational experiment, which we performed for the instance 2scp11A and described above, we also conducted on 41 other test instances. For the 2scp201C and 2scp201D instances we were not able to solve 99 weighted-max-ordering problems (2.2) in a reasonable time. Therefore for these two problems we solved (2.2) for only

five weight vectors from the set  $\Lambda' = \{[0.01, 0.99], [0.21, 0.79], [0.41, 0.59], [0.61, 0.39], [0.81, 0.19]\}$  with a 0.2 increment, which took more than 6 h.

The computational results for all 44 instances are given in Table 1, in which columns 1 and 5 list the instance name, columns 2 and 6 list the number of items  $m$  and the number of sets  $n$  in the instance, and columns 3 and 7 show the ranges of the experimental factor  $\epsilon_c(\lambda)$  values for each test instance obtained for all  $\lambda \in \Lambda$ . Finally, columns 4 and 8 give the run (CPU) times in seconds of Algorithm 1. Because the experimental factors were calculated and the CPU time was recorded for the instances 2scp201C and 2scp201D for  $\lambda \in \Lambda'$ , these problems are denoted in Table 1 respectively by 2scp201C\* and 2scp201D\*.

According to Table 1, the maximum experimental tolerance  $\epsilon_c(\lambda)$  is less than 1 for all test instances except for 2scp61C and 2scp81C, for which  $\epsilon_c(\lambda)$  lies in the interval (0.11, 1.03) and (0.10, 1.87), respectively. The theoretical factors associated with this experiment achieve magnitudes bigger than 100 since  $\lambda_{\min} = 0.01$  and  $\delta(\log(m) + 1) > 1$ .

Based on Table 1, we observe the following: (i) for all test instances, the experimental factors not only agree with but are far better than the theoretical factors; (ii) the computational factors do not increase with the size of the instance and stay less than 1 for almost all test problems; (iii) the large magnitude of the theoretical tolerance results from the use of  $\lambda_{\min}$  which, in the algorithm, serves as a tool to reach different weak Pareto points; (iv) the CPU times are very reasonable and smaller than 10.5 s for all test instances including 2scp201A and 2scp201B.

We also note that the magnitude of theoretical factor goes down for the MOSCPs with specially selected cost coefficients. Consider coefficients  $c_j^q \in (k, \Delta k)$  such that

$$0 < \Delta k \leq k \text{ for } q = 1, 2, \dots, p \text{ and } j \in J. \text{ We get } \max_{\substack{k_1, k_2=1, 2, \dots, p, \\ k_1 \neq k_2}} \left[ \frac{c_{\max}^{k_2}}{c_{\min}^{k_1}} \right] = 2, \text{ and}$$

the value of  $\delta$  given in Theorem 1 is then always less than 2 for any  $\lambda \in \mathcal{R}_{>}^p$  such that  $\sum_{q=1}^p \lambda_q = 1$ . In Table 2, we give the theoretical factor  $(1 + \epsilon(\lambda))$  we calculated using (3.21) for different  $m$  and  $\lambda_{\min}$  values for the case  $\delta < 2$ , that is, for the MOSCPs with cost coefficients that are at most doubled.

The experiments and observations lead us to conclude that the derived theoretical factor  $\epsilon(\lambda)$  in (3.21) is very conservative and its practical value is very low. On the other hand, the proposed Algorithm 1, which obeys the rationale for this factor, produces approximate weak Pareto points of the BOSCP with a much smaller experimental factor  $\epsilon_c(\lambda)$  in (3.23).

## 5 Discussion

We have developed an algorithm for approximating a weak Pareto point for the MOSCP and derived the associated tolerance function that depends on the problem data and the weight vector used for computing the point. The approximate point is computed in polynomial time.

Based on Corollary 2, we can make the following theoretical observation. If Algorithm 1 was run for all  $\lambda \in \mathcal{R}_{>}^p$ , for every weak Pareto point of the MOSCP there



**Table 1** Characteristics of 44 test instances and the computational tolerance (error)

Instance	$(m, n)$	$\epsilon_c(\lambda)$	CPU sec	Instance	$(m, n)$	$\epsilon_c(\lambda)$	CPU sec
2scp11A	(10,100)	[0.00, 0.44]	0.26	2scp62A	(60,600)	(0.08, 0.55)	2.16
2scp11B		[0.00, 0.34]	0.43	2scp62B		(0.06, 0.52)	2.14
2scp11C		[0.00, 0.73]	0.11	2scp62C		(0.03, 0.17)	1.36
2scp11D		[0.00, 0.93]	0.17	2scp62D		[0.00, 0.50]	1.85
2scp41A	(40,200)	(0.08, 0.36)	1.43	2scp81A	(80,800)	(0.03, 0.32)	7.65
2scp41B		(0.04, 0.28)	1.33	2scp81B		(0.07, 0.37)	7.76
2scp41C		(0.08, 0.45)	1.02	2scp81C		(0.10, 1.87)	6.38
2scp41D		(0.12, 0.32)	0.92	2scp81D		(0.10, 0.64)	5.75
2scp42A	(40,200)	(0.07, 0.43)	2.30	2scp82A	(80,800)	(0.06, 0.58)	3.59
2scp42B		(0.04, 0.23)	2.47	2scp82B		(0.05, 0.51)	3.24
2scp42C		(0.08, 0.41)	1.76	2scp82C		[0.00, 0.50]	2.17
2scp42D		(0.02, 0.21)	1.44	2scp82D		(0.08, 0.47)	2.34
2scp43A	(40,200)	(0.08, 0.59)	0.70	2scp101A	(100,1000)	(0.04, 0.41)	4.64
2scp43B		(0.01, 0.34)	0.77	2scp101B		(0.02, 0.39)	4.92
2scp43C		(0.05, 0.56)	0.65	2scp101C		(0.06, 0.35)	3.57
2scp43D		(0.07, 0.43)	0.71	2scp101D		(0.06, 0.38)	3.20
2scp61A	(60,600)	(0.11, 0.53)	5.94	2scp102A	(100,1000)	[0.00, 0.72]	3.08
2scp61B		(0.08, 0.45)	4.39	2scp102B		[0.00, 0.53]	3.35
2scp61C		(0.11, 1.03)	3.85	2scp102C		[0.00, 0.79]	2.45
2scp61D		(0.08, 0.54)	3.53	2scp102D		(0.07, 0.83)	2.18
2scp201A	(200,1000)	(0.07, 0.47)	10.43	2scp201C*	(200,1000)	(0.07, 0.47)	0.41
2scp201B		(0.07, 0.47)	10.31	2scp201D*		(0.12, 0.47)	0.43

**Table 2** Theoretical factor calculated for the MOSCPs with special cost coefficients

$m$	$\lambda_{\min}$	$(1 + \epsilon(\lambda))$
10	0.1	40
	0.01	400
100	0.1	60
	0.01	600
1000	0.1	80
	0.01	800

would be an approximate solution in the set of solutions returned by Algorithm 1 satisfying condition (3.19). This observation immediately takes us further. First, the approximation relies on the existence of the weight vectors however, it has not been addressed how these vectors can be found. Second, given the tolerance function, we can define a  $t$ -representation of the Pareto set for the MOSCP, following Vanderpooten et al. [25].

**Definition 5** Let  $Y$  be a set in  $\mathcal{R}^p$  and  $t$  be a tolerance function. A subset  $R$  in  $Y$  is called a  $t$ -representation of  $z(X_E)$  if for every  $y \in z(X_E)$  there exists  $r \in R$  such that  $r \leq t(y)$ .

In view of Definition 5, and assuming construction of appropriate weight vectors can be successfully integrated with Algorithm 1, the new algorithm would return a representation of the weak Pareto set of the MOSCP with an associated tolerance function. According to Vanderpooten et al. [25], this representation would be referred to as a *cover* of the weak Pareto set of the MOSCP. (We emphasize that their use of the word “cover” is different from the specific meaning of “cover” in the MOSCP, which we defined in Sect. 2.) Due to this far reaching potential consequences, an obvious and important avenue of a future study is to investigate how suitable weight vectors can be constructed.

Based on a computational experiment conducted on a collection of BOSCPs of various sizes, we observed that the derived theoretical tolerance has only theoretical value. However, we defined a closely related experimental tolerance providing evidence that the developed algorithm is effective because for almost all test instances it computes approximate weak Pareto outcomes with an error smaller than 1.

## References

1. Ahuja, R.K., Ergun, Ö., Orlin, J.B., Punnen, A.P.: A survey of very large-scale neighborhood search techniques. *Discrete Appl. Math.* **123**(1), 75–102 (2002)
2. Angel, E., Bampis, E., Gourvés, L.: Approximating the Pareto curve with local search for the bicriteria TSP (1, 2) problem. *Theor. Comput. Sci.* **310**(1–3), 135–146 (2004)
3. Caprara, A., Fischetti, M., Toth, P.: A heuristic method for the set covering problem. *Oper. Res.* **47**(5), 730–743 (1999)
4. Chvatal, V.: A greedy heuristic for the set-covering problem. *Math. Oper. Res.* **4**(3), 233–235 (1979)
5. Daskin, M.S., Stern, E.H.: A hierarchical objective set covering model for emergency medical service vehicle deployment. *Transp. Sci.* **15**(2), 137–152 (1981)
6. Ehrgott, M.: Approximation algorithms for combinatorial multicriteria optimization problems. *Int. Trans. Oper. Res.* **7**(1), 5–31 (2000)
7. Ehrgott, M.: *Multicriteria Optimization*. Springer, Berlin (2006)
8. Ehrgott, M., Gandibleux, X.: A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spectr.* **22**(4), 425–460 (2000)
9. Erlebach, T., Kellerer, H., Pferschy, U.: Approximating multiobjective Knapsack problems. *Manag. Sci.* **48**(12), 1603–1612 (2002)
10. Florios, K., Mavrotas, G.: Generation of the exact Pareto set in multi-objective traveling salesman and set covering problems. *Appl. Math. Comput.* **237**, 1–19 (2014)
11. <https://github.com/vOptSolver/vOptLib/tree/master/SCP>. Accessed 30 Apr 2018
12. Jaszkiwicz, A.: Do multiple-objective metaheuristics deliver on their promises? A computational experiment on the set-covering problem. *IEEE Trans. Evolut. Comput.* **7**(2), 133–143 (2003)
13. Jaszkiwicz, A.: A comparative study of multiple-objective metaheuristics on the bi-objective set covering problem and the Pareto memetic algorithm. *Ann. Oper. Res.* **131**(1), 135–158 (2004)
14. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W., Bohlinger, J.D. (eds.) *Complexity of Computer Computations*, pp. 85–103. Springer, Berlin (1972)
15. Liu, Y.-H.: A heuristic algorithm for the multi-criteria set-covering problems. *Appl. Math. Lett.* **6**(5), 21–23 (1993)
16. Lust, T., Teghem, J., Tuytens, D.: Very large-scale neighborhood search for solving multiobjective combinatorial optimization problems. In: Takahashi, R.H.C., Deb, K., Wanner, E.F., Greco, S. (eds.) *EMO*, pp. 254–268. Springer, Berlin (2011)

17. Lust, T., Tuytens, D.: Variable and large neighborhood search to solve the multiobjective set covering problem. *J. Heurist.* **20**(2), 165–188 (2014)
18. McDonnell, M.D., Possingham, H.P., Ball, I.R., Cousins, E.A.: Mathematical methods for spatially cohesive reserve design. *Environ. Model. Assess.* **7**(2), 107–114 (2002)
19. Musliu, N.: Local search algorithm for unicost set covering problem. In: Dapoigny, R., Ali, M. (eds.) IEA/AIE, pp. 302–311. Springer, Berlin (2006)
20. Orlin, J.B., Ahuja, R.K., Magnanti, T.L.: *Network flows: theory, algorithms, and applications*. Prentice Hall, New Jersey (1993)
21. Papadimitriou, C.H., Yannakakis, M.: On the approximability of trade-offs and optimal access of web sources. In: *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pp. 86–92. IEEE (2000)
22. Prins, C., Prodhon, C., Calvo, R.W.: Two-phase method and Lagrangian relaxation to solve the bi-objective set covering problem. *Ann. Oper. Res.* **147**(1), 23–41 (2006)
23. Saxena, R.R., Arora, S.R.: Linearization approach to multi-objective quadratic set covering problem. *Optimization* **43**(2), 145–156 (1998)
24. Ulungu, E.L., Teghem, J.: Multi-objective combinatorial optimization problems: a survey. *J. Multi-Criteria Decis. Anal.* **3**(2), 83–104 (1994)
25. Vanderpooten, D., Weerasena, L., Wiecek, M.M.: Covers and approximations in multiobjective optimization. *J. Glob. Optim.* **67**(3), 601–619 (2017)
26. Vazirani, V.V.: *Approximation Algorithms*. Springer, Berlin (2013)
27. Weerasena, L., Shier, D., Tonkyn, D.: A hierarchical approach to designing compact ecological reserve systems. *Environ. Model. Assess.* **19**(5), 437–449 (2014)
28. Weerasena, L., Wiecek, M.M., Soylu, B.: An algorithm for approximating the Pareto set of the multi-objective set covering problem. *Ann. Oper. Res.* **248**(1–2), 493–514 (2017)