ISSN: 1017-060X (Print)

ISSN: 1735-8515 (Online)

Special Issue of the

Bulletin of the Iranian Mathematical Society

Special Issue Dedicated to ORO2013 Conference

Vol. 42 (2016), No. 7, pp. 67–88

Title:

Solving multiobjective linear programming problems using ball center of polytopes

Author(s):

A. H. Dehmiry and M. A. Yaghoobi

Published by Iranian Mathematical Society
http://bims.ims.ir

Bull. Iranian Math. Soc.
Vol. 42 (2016), No. 7, pp. 67–88
Online ISSN: 1735-8515

SOLVING MULTIOBJECTIVE LINEAR PROGRAMMING PROBLEMS USING BALL CENTER OF POLYTOPES

A. H. DEHMIRY AND M. A. YAGHOOBI*

(Communicated by Dinh The Luc)

Abstract. Here, we aim to develop a new algorithm for solving a multiobjective linear programming problem. The algorithm is to obtain a solution which approximately meets the decision maker's preferences. It is proved that the proposed algorithm always converges to a weak efficient solution and at times converges to an efficient solution. Numerical examples and a simulation study are used to illustrate the performance of the proposed algorithm.

Keywords: Multiobjective linear programming, efficiency, polytope, ball center of a polytope, target value.

MSC(2010): Primary: 90C29; Secondary: 90C05.

1. Introduction

Multiobjective linear programming (MOLP) is a famous model used for multicriteria optimization [14]. Zopounidis et al. [38] used MOLP as a decision tool for the selection of stock portfolios. Xiao et al. [35] proposed a MOLP model on injection oil field recovery system. Ipsilandis [19] introduced a MOLP model for scheduling linear repetitive projects. Shao and Ehrgott [29] considered the beam intensity optimization problem of radiotherapy treatment planning and formulated it as a MOLP problem. For an overview of MOLP, we refer the reader to [14,31,37] and the references therein.

A MOLP problem deals with several linear objective functions subject to a set of linear constraints, for which we are to find a suitable efficient or weak efficient solution to satisfy the decision maker (DM). An efficient solution is an element of the feasible set for which an object value can not be improved without sacrificing the others. On the other hand, a weak efficient solution is a solution for which all the objective functions can not be improved simultaneously. Several methods have been developed to obtain efficient and/or weak

Article electronically published on December 18, 2016.
Received: 17 August 2013, Accepted: 3 May 2014.
*Corresponding author.

efficient solutions of a MOLP problem. Some only obtain a single efficient (or weak efficient) solution while others obtain all efficient (or weak efficient) solutions. The entire efficient set is of interest when the DM's preferences are not provided. Some simplex-based algorithms are available for generating the entire efficient set of a MOLP problem [6,20,36] and other algorithms provide the set of efficient solutions by maximal efficient faces [5, 13, 27]. However, finding all efficient solutions can be rather time-consuming and may not be possible for large-scale problems. On the other hand, since the number of efficient solutions can be quite large, it is not an easy task to select just one [17]. Therefore, algorithms that are able to find a suitable efficient (weak efficient) solution are of interest.

The well-known weighted sum, $\varepsilon-$constraint, minmax, and lexicographic methods are the most popular methods for obtaining a single efficient (weak efficient) solution of a MOLP problem [14, 31]. More comprehensive methods have been developed in the literature (e.g., see [8, 10, 12, 34]). These methods solve either a single optimization problem or a finite number of optimization problems interactively. The simplex method is commonly used, since the models are usually single objective linear programs. An important feature of the methods is modelling of the decision process and the type of information requested from the DM. We may ask the DM for his/her preferences in terms of the weights of the objectives, a reference or target point or the lower or upper bounds of the objective functions. For instance, if the DM can provide information in the form of target values and weights for the objective functions, then the well-known goal programming technique [21] can be used. Also, the reference point method [10, 22] allows the DM to offer reference points (or aspiration levels) for an objective. Moreover, one can change the reference point to improve the attained value of an objective function [23].

As mentioned, most MOLP methods are simplex-based. The simplex algorithm moves from one extreme point on the boundary of the feasible region to another one to attain an efficient solution. To accomplish this, some methods need efficiency tests in moving through the extreme points [20, 27, 31]. The number of extreme points, and also iterations, increases rapidly as the size of the problem increases. An alternative approach in such cases is moving through the interior of the feasible region instead of the boundary. In this context, applying interior point methods to solve a MOLP problem is quite popular [1, 3]. Some methods use the center of polytopes for moving towards a single efficient (or weak efficient) solution.

The center of a region can be defined in several ways, each leading to a different analytic and geometrical implication [4]. Trafalis and Alkahtani [33] used the analytic center of polytopes to construct an algorithm for MOLP problems. They computed the analytic centers of the restricted feasible regions obtained by inducing cuts in the main feasible region. Aghezzaf and Ouaderhman [2] constructed a sequence of smaller and smaller polytopes which shrank towards a compromise solution of a MOLP problem. Their algorithm moves from the center of a polytope to the center of the next polytope using a local optimization involving maximization of a linear function over an ellipsoid. However, both methods need to approximate the utility function of the DM. Tarek [32] suggested the method of centers algorithm for multiobjective programming problems which proceeded without such a utility function. Arbel and Vargas [4] addressed various issues associated with the derivation of the Euclidean center and illustrated an application to MOLP problems.

Recently, Murty and Oskoorouchi [24, 26] used the concept of the ball center of a polytope and applied it to solve a linear programming problem effectively. The current research aims to use the concept of ball center of a polytope to develop an algorithm for obtaining a suitable solution of a MOLP problem. In fact, the idea of the proposed algorithm is similar to Aghezzaf and Ouaderhman's [2] algorithm. In other words, it constructs a sequence of smaller and smaller polytopes, shrinking towards a solution. However, it does not need a utility function or its approximation. Indeed, it only asks the DM for target values and weights of the objectives. It is proved that the proposed algorithm always converges to a weak efficient solution. Furthermore, its convergency to an efficient solution in special cases is also proved. In fact, the aim of the proposed algorithm is not to determine all efficient (or weak efficient) solutions. It is intended to find a suitable solution which meets the DM's desires satisfactorily.

The rest of the paper is organized as follows. Section 2 introduces the MOLP problem and some preliminaries. Section 3 defines the ball center of a polytope and describes methods to approximate it. Section 4 presents an algorithm to obtain a solution for a MOLP problem. The main results are given in Section 5. The satisfactory performance of the proposed algorithm is illustrated by numerical examples as well as a simulation study in Section 6. Finally, Section 7 contains conclusions and suggestions for further research.

2. A survey of multiobjective linear programming

To compare vectors in MOLP, it is necessary to define an ordering. For $x^1, x^2 \in \mathbb{R}^n$, common orders are as follows [14]:

- $x^1 \geqq x^2 \Leftrightarrow x_i^1 \geqslant x_i^2,\ \forall\ i = 1, ..., n$, (weak componentwise order);
- $x^1 \geq x^2 \Leftrightarrow x^1 \geqq x^2$ and $x^1 \neq x^2$, (componentwise order);
- $x^1 > x^2 \Leftrightarrow x_i^1 > x_i^2,\ \forall\ i = 1, ..., n$, (strict componentwise order).

Throughout our work here, e_i $(i = 1, ..., n)$ will denote the ith coordinate vector in $\mathbb{R}^n$; i.e., a vector with zero components except for a 1 in the ith position.

A MOLP problem can be formulated as [14, 27]:

$$\begin{array}{rl} \max & Cx = (C_{1.}x, C_{2.}x, ..., C_{p.}x)^T \\ s.t. & x \in \mathcal{X} = \{x \in \mathbb{R}^n | Ax \leqq b\}, \end{array} \tag{2.1}$$

where C is a $p \times n$ matrix, whose rows are row vectors $C_{j.}$, $j = 1, ..., p$, $x \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$ are column vectors, A is an $m \times n$ matrix, and ^{T} stands for transpose.
In problem (2.1), every constraint in $\mathcal{X}$ defines a half space. In general, the set $\{x \in \mathbb{R}^n \mid \alpha' x \leqslant \beta\}$, where $0 \neq \alpha \in \mathbb{R}^n$ and $\beta \in \mathbb{R}$, is called a half space. The intersection of a finite number of half spaces is a polyhedron and a bounded polyhedron is a polytope. Therefore, in problem (2.1), $\mathcal{X}$ is a polyhedron, $\mathbb{R}^n$ and $\mathcal{X} \subseteq \mathbb{R}^n$ are repectively called the decision space and the feasible set. Moreover, let $\mathcal{Y} = C(\mathcal{X}) = \{y \in \mathbb{R}^p \mid y = Cx, x \in \mathcal{X}\}$, where $y = Cx$ is the criterion vector of $x \in \mathcal{X}$. Then, $\mathbb{R}^p$ and $\mathcal{Y} \subseteq \mathbb{R}^p$ are respectively called the criterion space and the image of the feasible set in the criterion space, respectively [14].

Definition 2.1. ([14]) A solution $\hat{x} \in \mathcal{X}$ is called (a) an (weak) efficient solution of problem (2.1) if there does not exist $x \in \mathcal{X}$ such that $Cx\ (>) \geq C\hat{x}$. $(\mathcal{X}_{WE})$ $\mathcal{X}_E$ denotes the set of all (weak) efficient solutions of problem (2.1).

Definition 2.2. ([14]) A solution $\hat{x} \in \mathcal{X}$ is called a strictly efficient solution of problem (2.1) if there does not exist $(\hat{x} \neq)$ $x \in \mathcal{X}$ such that $Cx \geqq C\hat{x}$ (or equivalently, $\hat{x} \in \mathcal{X}_E$ and $\{x \in \mathcal{X} \mid Cx = C\hat{x}\} = \{\hat{x}\}$). $\mathcal{X}_{SE}$ denotes the set of all strictly efficient solutions of problem (2.1).

Definition 2.3. ([14]) The points $y^I = (y_1^I, ..., y_p^I)^T$ and $y^N = (y_1^N, ..., y_p^N)^T$ are called the ideal point and the nadir point of problem (2.1), respectively, when $y_j^I = max_{x \in \mathcal{X}} C_{j.}x$ and $y_j^N = min_{x \in \mathcal{X}_E} C_{j.}x$, for $j = 1, ..., p$.

In fact, y^I and y^N are respectively tight lower and upper bounds on the efficient set. The ideal point can be found by solving p single objective linear programs. On the other hand, computation of y^N involves optimization over the efficient set which is a hard task. There is no exact method to determine the nadir point of a general multiobjective programming problem [14] and its components are usually estimated using a pay-off table. However, in general, this estimation is not good enough [14].

Definition 2.4. ([14]) A point $y^U = y^I + \varepsilon$, where $0 < \varepsilon \in \mathbb{R}^p$ has small positive components, is called a utopia point of problem (2.1).

Although different methods exist to generate efficient (or weak efficient) solutions, most of them use a common idea. Indeed, many approaches are based on converting the multiobjective programming problem to a single objective optimization problem. In this context, the use of scalarization techniques is very popular. Scalarization methods usually need auxiliary variables and/or parameters to be provided by the DM. Although the form of the scalarizing functions as well as the values of parameters depend on the assumptions made concerning the DM's preferences and behaviors, the DM is not often satisfied with the attained solution. Thus, the problem needs to be solved again with different parameters to meet the DM's preferences. On the other hand, these methods guarantee to achieve efficient (or weak efficient) solutions.

In the following, three well-known scalarization methods are mentioned briefly [14, 31]. Let w_j, $j = 1, ..., p$, be the weight assigned by the DM to the jth objective function. Then, associated with problem (2.1), the weighted sum method solves a linear program as follows:

$$\max_{x \in \mathcal{X}} \sum_{j=1}^{p} w_j C_{j.} x. \tag{2.2}$$

The ε-constraint method, as one of the best known techniques, optimizes only one of the objective functions, while the other functions are transformed to constraints. Hence, to solve problem (2.1), the ε-constraint method solves the following linear program:

$$\begin{array}{ll} \max & C_{i.} x \\ s.t. & C_{j.} x \geqslant \varepsilon_j \ \ j = 1, ..., p, j \neq i \\ & x \in \mathcal{X}, \end{array} \tag{2.3}$$

where $\varepsilon_j \in \mathbb{R}$ $(j = 1, ..., p, j \neq i)$ is the assigned lower bound to the jth objective function by the DM and $C_{i.} x$ is the selected objective function for optimization. The reference point method as another approach ask the DM to provide a reference (or target) point y^R, which is formed by the values that the DM wishes to achieve for the objective functions. Then, a reference point method aims to achieve solutions as close as possible to the reference point by the aid of an achievement function. For instance, a well-known achievement scalarizing function for solving problem (2.1) is $s(y^R, Cx, w)$ used in the following problem [22]:

$$\max_{x \in \mathcal{X}} \ s(y^R, Cx, w), \tag{2.4}$$

where $s(y^R, Cx, w) = \min_{j=1,\dots,p} (w_j(C_{j.} x - y_j^R)) + \rho \sum_{j=1}^{p} (C_{j.} x - y_j^R)$, the value $w_j > 0$ is the weight assigned to the jth objective function, and the parameter $\rho > 0$ is the so-called augmentation coefficient.
Solving problem (2.4) leads to an efficient solution which is presented to the DM. If the DM is not satisfied with the computed solution, the reference point method acts interactively by taking different values of w_j, and/or y_j^R, for $j = 1, ..., p$. Then, problem (2.4) is solved again to generate a new efficient solution.

As seen, scalarization methods use different scalarizing functions to achieve the DM's desires. Selecting a suitable scalarizing function, which can generate a good solution, is not an easy task. Our current work here is to introduce a new method to achieve the DM's targets without using a scalarizing or an achievement function.

3. Ball center of a polytope

The center of a polytope is an interesting concept in mathematics and has been used to solve optimization problems [4, 7]. Huard [18] applied the concept of center to solve a mathematical programming problem with nonlinear constraints. Huard's method was used by Boggs [9] to enhance interior-point methods using dual affine trajectories. Sonnevend [30] defined the "analytic center" and used it to develop a linear programming (LP) approach. Fagan and Falk [16] introduced a method based on Euclidean centers for solving single objective LP problems. Recently, the concept of ball center was used by Murty [24] to solve an LP problem.

Let $\|x\|_p$ $(1 \leqslant p < \infty)$ be the p-norm of $x \in \mathbb{R}^n$, where $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{\frac{1}{p}}$. Then, a closed ball of radius $\delta \geqslant 0$ centered at $x^0 \in \mathbb{R}^n$ is defined by $B(x^0, \delta) = \{x \in \mathcal{X} \mid \|x - x^0\|_p \leqslant \delta\}$.

Definition 3.1. ([26]) Let $\mathcal{X}$ be a polytope with a nonempty interior represented by $\mathcal{X} = \{x \in \mathbb{R}^n \mid Ax \leqq b\}$, where $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. $x^* \in \mathcal{X}$ is called ball center of $\mathcal{X}$ if $B(x^*, \delta^*) \subseteq \mathcal{X}$, where $\delta^* = \max\{\delta \mid B(x, \delta) \subseteq \mathcal{X}, x \in \mathcal{X}, \delta \geqslant 0\}$.

As a matter of fact, the ball center of a polytope is the center of a largest closed ball inside it. One strategy for finding such a point is maximizing the minimum of its distance from every hyperplane in the polytope [26]. To find a ball center of the set $\mathcal{X} = \{x \in \mathbb{R}^n \mid Ax \leqq b\}$, Murty and Oskoorouchi [26] suggested solving the following problem:

$$\max_{x\in\mathcal{X}} \min_{i=1,\dots,m} b_i - A_{i.}x, \tag{3.1}$$

where $A_{i.}$ is the ith row of A and moreover, $\|A_{i.}\|_2 = 1, i = 1, ..., m$. Problem (3.1) can be rewritten as

$$\begin{array}{ll} \max & \delta \\ s.t. & \delta + A_{i.}x \leqslant b_i, \;\; i = 1, ..., m \\ & \delta \geqslant 0. \end{array} \tag{3.2}$$

Theorem 3.2 below shows that the optimal solution of problem (3.2) is a ball center of the feasible set of problem (2.1).

Theorem 3.2. *Let (x^*, δ^*) be the optimal solution of problem (3.2). Then, $B(x^*, \delta^*) = \{x \in \mathbb{R}^n \mid \|x - x^*\|_2 \leqslant \delta^*\} \subseteq \mathcal{X} = \{x \in \mathbb{R}^n \mid Ax \leqq b\}$ and x^* is a ball center of $\mathcal{X}$.*

Proof. Let $\hat{x} \in B(x^*, \delta^*)$. Then, $\|\hat{x} - x^*\|_2 \leqslant \delta^*$. Suppose that $\hat{x} \notin \mathcal{X}$. In other words, $\exists\ l \in \{1, ..., m\}$ such that $b_l - A_{l.}\hat{x} < 0$. Now, we have: $0 \leqslant \delta^* \leqslant b_l - A_{l.}x^* = b_l - A_{l.}\hat{x} + A_{l.}\hat{x} - A_{l.}x^* = b_l - A_{l.}\hat{x} + A_{l.}(\hat{x} - x^*) < A_{l.}(\hat{x} - x^*)$. Using the Cauchy-Schwartz inequality, we have

$$\delta^* < |A_{l.}(\hat{x} - x^*)| \leqslant \|A_{l.}\|_2 \|\hat{x} - x^*\|_2 = \|\hat{x} - x^*\|_2,$$

which is a contradiction.
To prove the second part, suppose that $B(\bar{x}, \bar{\delta}) = \{x \in \mathbb{R}^n \mid \|x - \bar{x}\|_2 \leqslant \bar{\delta}\}$ is a closed ball inside $\mathcal{X}$ such that $\bar{\delta} > \delta^*$. Thus, $B(\bar{x}, \bar{\delta}) \subseteq \mathcal{X}$ and the distance of $\bar{x}$ from every hyperplane in the polytope is greater than or equal to $\bar{\delta}$. Therefore,

$$b_i - A_{i.}\bar{x} = \frac{b_i - A_{i.}\bar{x}}{\|A_{i.}\|_2} \geqslant \bar{\delta}\ \forall\ i = 1, ..., m.$$

Consequently, $(\bar{x}, \bar{\delta})$ is a feasible solution of problem (3.2). Since (x^*, δ^*) is an optimal solution of problem (3.2), $\bar{\delta} \leqslant \delta^*$, which is a contradiction. Therefore, x^* is a ball center of $\mathcal{X}$ and the proof is complete. □

Lemma 3.3. *Let $\mathcal{X} = \{x \in \mathbb{R}^n \mid Ax \leqq b\}$ be bounded and $int(\mathcal{X}) \neq \emptyset$. Then, the ball center of $\mathcal{X}$ is an interior point.*

Proof. Problem (3.2) is feasible and has an optimal solution, since $\mathcal{X}$ is nonempty and bounded. On the other hand, $int(\mathcal{X}) \neq \emptyset$ implies that there exists $\hat{x} \in \mathcal{X}$ such that $A\hat{x} < b$. Thus, problem (3.2) has an optimal solution (x^*, δ^*) with $\delta^* > 0$. Now, by Theorem 3.2, x^* is a ball center of $\mathcal{X}$, which is also an interior point. □

3.1. **Approximating a ball center.** To compute a ball center exactly using problem (3.2), we need an LP algorithm. This may require tedious calculation. Murty and Oskoorouchi [26] have proposed two approaches called LSFN (Line Search steps in Facetal Normal directions) and LSCPD (Line Search steps using Computed Profitable Directions) to obtain a ball center of a polytope approximately. In this paper, only the LSFN strategy is used. LSFN moves along a profitable direction to obtain a new point such that the ball centered at it has a radius greater than that of the one centered at a starting point. Clearly, the radiuses of all balls are selected so that every ball is in the feasible set. The following theorem plays an important role in the LSFN approach and suggests a criterion for recognizing a profitable direction at a specific interior feasible solution.

Theorem 3.4. ([26]) *Consider $\mathcal{X} = \{x \in \mathbb{R}^n \mid Ax \leqq b\}$. A given direction $y \in \mathbb{R}^n$ is a profitable direction at the current interior feasible solution $\hat{x}$ if and only if $A_{i.}y < 0$, for all $i \in T(\hat{x})$, where $T(\hat{x}) = \{i \in \{1, ..., m\} \mid b_i - A_{i.}\hat{x} = \delta(\hat{x})\}$ and $\delta(\hat{x}) = \min_{i=1,...,m}(b_i - A_{i.}\hat{x})$. Also, $\hat{x}$ is a ball center of $\mathcal{X}$ if and only if the system $A_{i.}y < 0$, for every $i \in T(\hat{x})$, has no solution in $\mathbb{R}^n$.*

The LSFN approach uses directions normal to the facetal hyperplanes of $\mathcal{X}$ as the candidates for obtaining a profitable direction; i.e., directions y in $\Gamma = \{A_{i.}^T, -A_{i.}^T,\ i = 1, ..., m\}$. Let $\hat{x} \in \mathcal{X}$ be a feasible solution and $y \in \mathbb{R}^n$ be a profitable direction. It is easy to show that the minimum distance of $\hat{x}$ from all facetals is increased when it moves in direction y. Thus, to find a maximal length step at direction y, the following two variables LP problem should be solved [26]:

$$\begin{array}{rl} \max & \delta \\ s.t. & \delta + A_{i.}(\hat{x} + \alpha y) \leqslant b_i \ \ i = 1, ..., m \\ & \delta, \alpha \geqslant 0. \end{array} \tag{3.3}$$

The LSFN approach continues as long as profitable directions are found in Γ satisfying the criterion mentioned in Theorem 3.4, and terminates with the final point as an approximation to a required ball center of $\mathcal{X}$. To see more technical details, we refer to [24].

Remark 3.5. It is clear that to find a profitable direction all vectors in $\mathbb{R}^n$ should be checked. However, this increases the number of needed calculations. Hence, the LSFN approach only considers the vectors in Γ. For a better approximation, here we suggest using the vectors in $\bar{\Gamma} = \{A_{i.}^T, -A_{i.}^T,\ i = 1, ..., m,\ (A_{j.}^T + A_{(j+1).}^T),\ -(A_{j.}^T + A_{(j+1).}^T),\ j = 1, ..., m-1, (A_{m.}^T + A_{1.}^T), -(A_{m.}^T + A_{1.}^T)\}$.

4. Using ball centers to solve a multiobjective linear programming problem

Consider the MOLP problem (2.1). Suppose that $\mathcal{X}$ is bounded and $int(\mathcal{X}) \neq \emptyset$. Without loss of generality, assume that $Cx > 0, \forall x \in \mathcal{X}$. In fact, if this assumption is not valid, one can shift the origin of coordinates in the criterion space (i.e., $\mathcal{Y}$) to a suitable point. A candidate for such a point is a utopia point of $min_{x\in\mathcal{X}} Cx$. Let y^U be a utopia point of $min_{x\in\mathcal{X}} Cx$. Then, the MOLP problem $max_{x\in\mathcal{X}} \hat{C}x = Cx - y^U$ is equivalent to the MOLP problem (2.1) and $\hat{C}x > 0, \forall x \in \mathcal{X}$. Without loss of generality, we make the following assumptions in problem (2.1):

(1) $\|A_{i.}\|_2 = 1$, for $i = 1, ..., m$,
(2) $\|C_{j.}\|_2 = 1$, for $j = 1, ..., p$.

It is sufficient to divide both sides of the ith constraint by $\|A_{i.}\|_2, i = 1, ..., m$, and the jth objective function can be divided by $\|C_{j.}\|_2$, for $j = 1, ..., p$. The following lemma shows that this modification does not change the efficient (or weak efficient) set.

Lemma 4.1. *The efficient (or weak efficient) sets of problem (2.1) and problem*

$$\max\nolimits_{x\in\mathcal{X}} (a_1 C_{1.}x, a_2 C_{2.}x, ..., a_p C_{p.}x)', \tag{4.1}$$

where $a_j > 0,\ \forall\ j = 1, ..., p,$ are the same.

Proof. The proof is straightforward. □

To deal with MOLP problem (2.1), our proposed algorithm here (Algorithm 1) needs the desirable value (or target level), t_j, for the objective function $C_{j.}x$ ($j = 1, ..., p$) to be provided by the DM. It is assumed that more is preferred to less in each objective function for the DM. Moreover, since $Cx > 0, \forall x \in \mathcal{X}$, we suppose that $t = (t_1, ..., t_p)^T > 0$ is the target vector of the DM's desires. Furthermore, the DM may provide the vector $(w_1, ..., w_p)^T \geq 0$ as the weight vector, showing the relative importance of the objective functions. It is assumed that weights are normalized so that their sum is equal to 1. Furthermore, if the DM is not able to decide upon the weights, equal weights ($1/p$) are considered for the objective functions. Indeed, Algorithm 1 aims to solve the given MOLP problem by generating a sequence of ball centers. It first obtains the ball center of the feasible set. Then, new constraints are added to the problem to reduce the feasible set. Subsequently, the ball center of the new feasible set is obtained. This process is repeated until the radius of the last obtained ball is smaller than or equal to a given tolerance.

Algorithm 1.
Input: An instance of MOLP problem (2.1) with the stated parameters and assumptions, $t = (t_1, ..., t_p)^T > 0$, $(w_1, ..., w_p)^T \geq 0$, and $0 < \epsilon \in \mathbb{R}$, as the given tolerance.

```
    Step 1. Set $k := 1$, $\hat{t} := \frac{(w_1 t_1, ..., w_p t_p)^T}{\|(w_1 t_1, ..., w_p t_p)^T\|_2}$, $\mathcal{X}^k := \mathcal{X}$, and $\theta_k := 0$.
    Step 2. Find $x^k$ as the ball center of the polytope $\mathcal{X}^k$. Let $\delta_k$ denote
    the radius of the largest closed ball inside $\mathcal{X}^k$ centered at $x^k$. If $\delta_k \leqslant \epsilon$
    then go to Step 4.
    Step 3. Set
    $x^{k_0} := x^k$;
    $x^{k_1} := x^k + \delta_k e_1, ..., x^{k_n} := x^k + \delta_k e_n$;
    $x^{k_{n+1}} := x^k - \delta_k e_1, ..., x^{k_{2n}} := x^k - \delta_k e_n$;
    $\theta^{k_i} := \min_{j \in \{j | j=1,...,p,\ \hat{t}_j > 0\}} \frac{C_{j.} x^{k_i}}{\hat{t}_j}, i = 0, 1, ..., 2n$;
    $\theta^{k_s} := \max_{i \in \{0,1,...,2n\}} \{\theta^{k_i}\}, \theta_{k+1} := \theta^{k_s}$;
    $\mathcal{X}^{k+1} := \{x \in \mathbb{R}^n \mid Cx \geqq \theta_{k+1}\hat{t}, Ax \leqq b\}$;
    $k := k + 1$ and go to Step 2.
    Step 4. Stop, $x^k$ is the desired solution.
```
Output: A solution to the given MOLP problem.

The main idea of Algorithm 1 is illustrated in Figure 1. Suppose that $\mathcal{X}^1$ is the feasible set of the given MOLP problem. In the first iteration of Algorithm 1, x^1 is obtained as the ball center of $\mathcal{X}^1$ (see Figure 1). At Step 2, x^1 is an interior point such that its distance from all the defining hyperplanes of $\mathcal{X}^1$ is at least δ_1. Then, in Step 3, first the points $x^{1_1}, ..., x^{1_4}$ (Figure 1) and then,

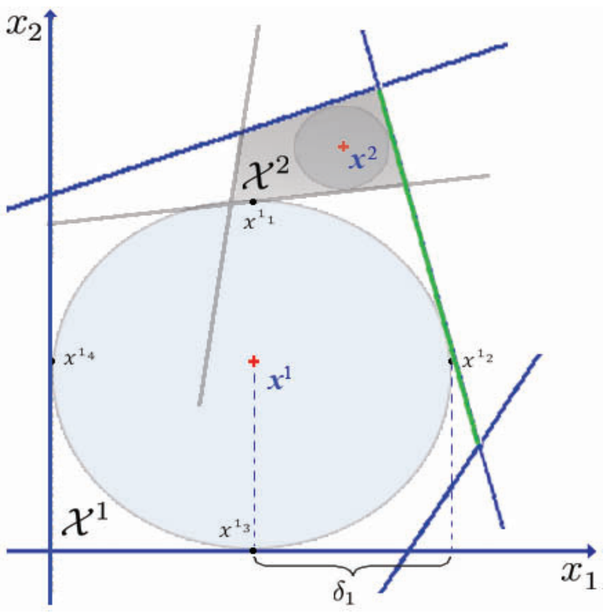

FIGURE 1. An iteration of Algorithm 1 on a MOLP problem with two variables and two objective functions.

the values of $\theta^{1_0}, \theta^{1_1}, ..., \theta^{1_4}$ are obtained. Indeed, θ^{1_i} is calculated such that x^{1_i} lies on the boundary of $\{x \in \mathbb{R}^n \mid Cx \geqq \theta^{1_i}\hat{t}\}$. Then, θ_1 is calculated as the maximum of $\theta^{1_0}, \theta^{1_1}, ..., \theta^{1_4}$ (say, e.g., θ^{1_1}) and $\mathcal{X}^2$ is set as the intersection of $\mathcal{X}$ and $\{x \in \mathbb{R}^n \mid Cx \geqq \theta_1\hat{t}\}$. The procedure guarantees that $\mathcal{X}^2 \neq \emptyset$, since x^{1_s} (here x^{1_1}) belongs to $\mathcal{X}^2$. Note that $\mathcal{X}^2$ is a new polytope which is a proper subset of $\mathcal{X}^1$. The same process is repeated for $\mathcal{X}^2$. The algorithm terminates when the radius of the largest closed ball centered at the ball center of the new polytope is less than or equal to the given tolerance ϵ.

5. Main results

In this section, some results are established to show the validity and properties of Algorithm 1. The first result states that the final solution of Algorithm 1 is a weak efficient solution. First, the following theorem is needed.

Theorem 5.1. ([11]) *Let $S \subseteq \mathbb{R}^n$ be a convex set with $int(S) \neq \emptyset$. Then, $cl(S) = cl(int(S))$, where $cl(S)$ denotes the closure of S.*

Theorem 5.2. *In Algorithm 1, let $\delta_{k+1} = 0$. Then, $x^{k+1} \in \mathcal{X}_{WE}$.*

Proof. If $x^{k+1} \notin \mathcal{X}_{WE}$, then $\exists\, \hat{x} \in \mathcal{X}$ so that $C\hat{x} > Cx^{k+1}$. Now, define $f(x) := Cx - Cx^{k+1}$. It is clear that f is continuous and $f(\hat{x}) > 0$. Therefore, $\exists\, r > 0$

so that $f(x) > 0$, for all $x \in \{x \in \mathbb{R}^n \mid \|x - \hat{x}\|_2 < r\}$. Since $\hat{x} \in \mathcal{X}$, by Theorem 5.1, $\hat{x} \in cl(int(\mathcal{X}))$. So, $\exists\, x^0 \in int(\mathcal{X})$ such that $x^0 \in \{x \in \mathbb{R}^n \mid \|x - \hat{x}\|_2 < r\}$ and thus $f(x^0) > 0$ (or $Cx^0 > Cx^{k+1}$). This implies that $x^0 \in int(\mathcal{X}^{k+1})$. Consequently, Lemma 3.3 yields $\delta_{k+1} > 0$, which is a contradiction. □

In the sequel, some interesting properties of Algorithm 1 are proved.

Lemma 5.3. *Suppose that Algorithm 1 does not stop at the kth iteration. Then, $\theta_{k+1} > \theta_k$, for $k \geqslant 1$, and $\theta_k > 0$, for $k \geqslant 2$.*

Proof. It is obvious that $\delta_k > 0$, since otherwise Algorithm 1 terminates at the kth iteration. Therefore, the ball center of $\mathcal{X}^k$ is an interior point of $\mathcal{X}^k$ and thus, $Cx^k > \theta_k \hat{t}$. Now, the definition of θ_{k+1} implies that $\theta_{k+1} \geqslant \theta^{k_0}$. Without loss of generality, suppose that $\theta^{k_0} = \frac{C_l.x^k}{\hat{t}_l}$. Hence,

$$\theta_{k+1} \geqslant \theta^{k_0} = \frac{C_l.x^k}{\hat{t}_l} > \frac{\theta_k \hat{t}_l}{\hat{t}_l} = \theta_k,$$

which completes the proof of the first part. For the second statement, note that if $k \geqslant 2$, then $\theta_k \geqslant \theta^{(k-1)_0}$. Without loss of generality, suppose that $\theta^{(k-1)_0} = \frac{C_q.x^{k-1}}{\hat{t}_q}$. Therefore,

$$\theta_k \geqslant \theta^{(k-1)_0} = \frac{C_q.x^{k-1}}{\hat{t}_q} > 0.$$

□

Corollary 5.4. *In Algorithm 1, we have $\mathcal{X}^{k+1} \subsetneqq \mathcal{X}^k$, for $k \geqslant 1$.*

Proof. By definition of $\mathcal{X}^{k+1}$ and Lemma 5.3, the proof is obvious. □

The following theorem implies that Algorithm 1 is convergent for any $\epsilon > 0$.

Theorem 5.5. *Let $0 < \epsilon \in \mathbb{R}$ be any given tolerance in Algorithm 1. Then, there exists $k \geqslant 1$ such that $\delta_k \leqslant \epsilon$.*

Proof. If $\delta_1 \leqslant \epsilon$, then $k = 1$ and the proof is complete. For $\delta_1 > \epsilon$, consider the ith ($i \geqslant 2$) iteration of Algorithm 1 being an arbitrary iteration. Let x^i be the ball center of $\mathcal{X}^i = \{x \in \mathbb{R}^n \mid Cx \geqq \theta_i \hat{t}, Ax \leqq b\}$ with δ_i being the radius of the largest closed ball inside $\mathcal{X}^i$ centered at x^i. If $\delta_i = 0$, then the proof is evident. Otherwise, $\delta_i > 0$ and x^i is an interior point of $\mathcal{X}^i$. Thus, $Cx^i > \theta_i \hat{t}$. The definition of θ_{i+1} implies that $\theta_{i+1} \geqslant \theta^{i_0}$. Without loss of generality, suppose that $\theta^{i_0} = \frac{C_l.x^i}{\hat{t}_l}$. Hence, $\theta_{i+1}\hat{t}_l \geqslant C_l.x^i$. Moreover, since x^i is the ball center of $\mathcal{X}^i$, $\delta_i + \theta_i \hat{t}_j \leqslant C_j.x^i$, $\forall\, j \in \{1, ..., p\}$. Thus, $C_l.x^i \geqslant \delta_i + \theta_i \hat{t}_l$. On the other hand, the definition of $\hat{t}$ implies that $\|\hat{t}\|_2 = 1$, which yields $-1 \leqslant \hat{t}_j \leqslant 1$, $\forall\, j = 1, ..., p$. Also, $(t_1, ..., t_p)^T > 0$ and $(w_1, ..., w_p)^T \geq 0$ imply that $\hat{t} \geq 0$, and thus $0 < \hat{t}_l \leqslant 1$. This leads to the fact that $\delta_i \geqslant \delta_i \hat{t}_l$. Hence, $\theta_{i+1}\hat{t}_l \geqslant C_l.x^i \geqslant \delta_i + \theta_i \hat{t}_l \geqslant \delta_i \hat{t}_l + \theta_i \hat{t}_l = (\theta_i + \delta_i)\hat{t}_l$. Consequently,

$\theta_{i+1} \geqslant \theta_i+\delta_i$, $\forall\, i = 2, 3, \ldots$. Now, on the contrary suppose that $\delta_k > \epsilon$, $\forall\, k \geqslant 2$. Since $x^k \in \mathcal{X}^k$, $\forall\, k \geqslant 2$, then

$$Cx^k \geqq \theta_k\hat{t} \geqq (\theta_{k-1} + \delta_{k-1})\hat{t} > (\theta_{k-1} + \epsilon)\hat{t} \geqq \ldots > (\theta_2 + (k-2)\epsilon)\hat{t}.$$

Thus, $Cx^k > \theta_2\hat{t} + (k-2)\epsilon\hat{t}$, $\forall\, k = 2, 3, \ldots$. Therefore, $C_{j.}x^k \to +\infty$, $\forall\, j = 1, \ldots, p$, as $k \to +\infty$, which is a contradiction, since $C(\mathcal{X})$ is bounded. □

Corollary 5.6. *Algorithm 1 terminates after a finite number of iterations for any* $\epsilon > 0$.

Proof. By steps 2 and 4 of Algorithm 1 and Theorem 5.5, the proof is clear. □

The next result shows other important properties of Algorithm 1. Indeed, it will be proved that Algorithm 1 converges to an efficient or a strictly efficient solution.

Theorem 5.7. *Let $\hat{x} \in \mathcal{X}$ be an efficient solution of problem (2.1). In Algorithm 1, suppose that $\hat{t} = \gamma C\hat{x}$, where $0 < \gamma \in \mathbb{R}$, and $\delta_k \leqslant \epsilon$, for the given tolerance ϵ. Then, $\hat{x} \in \mathcal{X}^k$.*

Proof. Since $\hat{x}$ is an efficient solution of problem (2.1), $\{x \in \mathcal{X} \mid Cx \geq C\hat{x}\} = \emptyset$. On the other hand, $\mathcal{X}^k = \{x \in \mathbb{R}^n \mid Cx \geqq \theta_k\hat{t}, Ax \leqq b\} = \{x \in \mathcal{X} \mid Cx \geqq \theta_k\hat{t}\} = \{x \in \mathcal{X} \mid Cx \geqq \theta_k\gamma C\hat{x}\}$. Moreover, the ball center of $\mathcal{X}^k$, i.e., x^k, belongs to $\mathcal{X}^k$. It follows that $\theta_k\gamma \leqslant 1$. Otherwise, $x^k \in \{x \in \mathcal{X} \mid Cx \geq C\hat{x}\}$, which is a contradiction. Thus, $C\hat{x} \geqq \theta_k\gamma C\hat{x}$ and $\hat{x} \in \mathcal{X}^k$. □

The number of iterations in Algorithm 1 depends on the given tolerance ϵ. It increases as ϵ decreases to zero. In other words, $k \to +\infty$ as $\epsilon \to 0$. The next theorem expresses an interesting property of Algorithm 1.

Theorem 5.8. *Suppose that $\hat{x} \in \mathcal{X}_E$ and in Algorithm 1 $\hat{t} = \gamma C\hat{x}$, where $0 < \gamma \in \mathbb{R}$. Then, $\bigcap_{k=1}^{\infty} \mathcal{X}^k = \{x \in \mathcal{X} \mid Cx = C\hat{x}\}$ as $\epsilon \to 0$.*

Proof. First, we show that $\lim_{k\to\infty} \gamma\theta_k = 1$. Lemma 5.3 and $\gamma > 0$ imply that $\gamma\theta_{k+1} > \gamma\theta_k$ $\forall\, k = 1, 2, \ldots$. Also, arguments similar to those for Theorem 5.7 lead to $\gamma\theta_k \leqslant 1$, $\forall\, k = 1, 2, \ldots$. Thus, the sequence $\{\gamma\theta_k\}_{k=1}^{\infty}$ is a strictly increasing and bounded sequence. Therefore, it is convergent. Furthermore, $\lim_{k\to\infty} \gamma\theta_k \leqslant 1$. Now, suppose that $\lim_{k\to\infty} \gamma\theta_k = \alpha_0 < 1$. Then, $\bigcap_{k=1}^{\infty} \mathcal{X}^k = \{x \in \mathcal{X} \mid Cx \geqq \alpha_0 C\hat{x}\}$ and $\hat{x} \in \bigcap_{k=1}^{\infty} \mathcal{X}^k$, since $C\hat{x} > \alpha_0 C\hat{x}$. Consequently, $\hat{x} \in \mathcal{X}^k$, $\forall\, k = 1, 2, \ldots$. Define $f(x) := Cx - \alpha_0 C\hat{x}$. It is clear that f is continuous and $f(\hat{x}) > 0$. Therefore, $\exists\, r > 0$ such that $f(x) > 0$, for all $x \in \{x \in \mathbb{R}^n \mid \|x - \hat{x}\|_2 < r\}$. Since $\hat{x} \in \mathcal{X}$, by Theorem 5.1, $\hat{x} \in cl(int(\mathcal{X}))$. So,

$\exists\ x^0 \in int(\mathcal{X})$ such that $x^0 \in \{x \in \mathbb{R}^n \mid \|x - \hat{x}\|_2 < r\}$ and thus $f(x^0) > 0$ (or $Cx^0 > \alpha_0 C\hat{x}$). This implies that $x^0 \in int(\bigcap_{k=1}^{\infty} \mathcal{X}^k)$. Consequently, $\bigcap_{k=1}^{\infty} \mathcal{X}^k$ has a ball center with a positive radius, say δ_0. Furthermore, $x^0 \in int(\mathcal{X}^k),\ \forall\ k = 1, 2, \ldots$, since $\bigcap_{k=1}^{\infty} \mathcal{X}^k \subseteq \mathcal{X}^k,\ \forall\ k = 1, 2, \ldots$. Thus, $\delta_k \geqslant \delta_0 > 0,\ \forall\ k = 1, 2, \ldots$. This means that $\delta_k \nrightarrow 0$, which is a contradiction due to the fact that by Theorem 5.5, δ_k should converge to zero, as $\epsilon \to 0$. This proves the assertion that $\lim_{k\to\infty} \gamma\theta_k = \alpha_0 = 1$. Hence, $\bigcap_{k=1}^{\infty} \mathcal{X}^k = \{x \in \mathcal{X} \mid Cx \geqq C\hat{x}\}$. If $\exists\ x \in \mathcal{X}$ such that $Cx \geqq C\hat{x}$ and $Cx \neq C\hat{x}$, then $\hat{x} \notin \mathcal{X}_E$, which is a contradiction. This leads to $\bigcap_{k=1}^{\infty} \mathcal{X}^k = \{x \in \mathcal{X} \mid Cx = C\hat{x}\}$. □

Corollary 5.9. *Suppose that $\hat{x} \in \mathcal{X}_{SE}$ and in Algorithm 1, $\hat{t} = \gamma C\hat{x}$, where $0 < \gamma \in \mathbb{R}$. Then, $\bigcap_{k=1}^{\infty} \mathcal{X}^k = \{\hat{x}\}$, as $\epsilon \to 0$.*

Proof. Since $\hat{x} \in \mathcal{X}_{SE}$, $\hat{x} \in \mathcal{X}_E$ and $\{x \in \mathcal{X} \mid Cx = C\hat{x}\} = \{\hat{x}\}$. Now, the assertion follows immediately from Theorem 5.8. □

Remark 5.10. In fact, Theorem 5.8 and Corollary 5.9 show that Algorithm 1 converges to an efficient or a strictly efficient solution x, when the vector $\hat{t}$ is any point on the ray emanating from the origin in the direction of the criterion vector Cx. This fact can help us to obtain a subset of the efficient set. To do that, we can choose a set of N target vectors which are uniformly scattered on the set $\{(t_1, \ldots, t_p)^T \mid 0 < t_j, j = 1, \ldots, p \textit{ and } \sum_{j=1}^{p} t_j = 1\}$ and apply Algorithm 1 N times with each selected target vector.

Further results related to Algorithm 1 are as follows.

Theorem 5.11. *In Algorithm 1, $\mathcal{X}^k \cap \mathcal{X}_E \neq \emptyset$, for $k \geqslant 1$.*

Proof. For an arbitrary k, let $x^0 \in \mathcal{X}^k$. It is clear that $X(x^0) \cap \mathcal{X}_E \neq \emptyset$, where $X(x^0) = \{x \in \mathbb{R}^n \mid Ax \leqq b,\ Cx \geqq Cx^0\}$. Since $X(x^0) \subseteq \mathcal{X}^k$, $\mathcal{X}^k \cap \mathcal{X}_E \neq \emptyset$. □

Corollary 5.12. *In Algorithm 1, $(\bigcap_{k=1}^{\infty} \mathcal{X}^k) \cap \mathcal{X}_E \neq \emptyset$.*

Proof. Since $\mathcal{X}^{k+1} \subseteq \mathcal{X}^k$, the conclusion is clear. □

6. **Numerical examples and a simulation study**

In this section, three numerical examples are presented. They are solved by Algorithm 1, which is coded in MATLAB software environment. To see the performance of Algorithm 1, a simulation study is carried out in which different MOLP problems are generated randomly. Moreover, some comparisons are given.

Example 6.1. Consider the following MOLP problem:

$$
\begin{array}{rcl}
\max\ Cx & = & \begin{bmatrix} 19 & 2 \\ 30 & 31 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\
s.t. & & \\
x_1 + 3x_2 & \leqslant & 150, \\
5x_1 - x_2 & \leqslant & 170, \\
2x_1 + x_2 & \leqslant & 90, \\
-27x_1 + 2x_2 & \leqslant & -30, \\
x_2 & \geqslant & 0.
\end{array}
\tag{6.1}
$$

Let $w = (0.6, 0.4)^T$. Although Algorithm 1 asks the DM one target vector, Table 1 presents the output of Algorithm 1 for different target vectors. It helps us to see the output of Algorithm 1 for the different DM's preferences. Note that the given data in Table 1 are provided under two scenarios: exact ball center and approximate ball center.

Table 1. Outputs of Algorithm 1 for the MOLP problem (6.1) for different target vectors.

	t	k	x^k	Cx^k
	$(339,2034)^T$	7	$(23.999, 42.000)^T$	$(539.99, 2021.98)^T$
	$(368,2024)^T$	13	$(24.483, 41.034)^T$	$(547.24, 2006.55)^T$
Exact	$(400,2000)^T$	10	$(26.707, 36.585)^T$	$(580.61, 1935.37)^T$
ball	$(402,1809)^T$	8	$(29.221, 31.558)^T$	$(618.31, 1854.93)^T$
center	$(430,1720)^T$	10	$(32.083, 25.833)^T$	$(661.25, 1763.33)^T$
	$(476,1666)^T$	8	$(35.373, 19.254)^T$	$(710.60, 1658.06)^T$
	$(501,1503)^T$	5	$(37.143, 15.714)^T$	$(737.14, 1601.43)^T$
	$(339,2034)^T$	8	$(24.999, 42.000)^T$	$(539.97, 2021.98)^T$
	$(368,2024)^T$	11	$(24.483, 41.034)^T$	$(547.24, 2006.55)^T$
Approximate	$(400,2000)^T$	9	$(26.708, 36.583)^T$	$(580.62, 1935.33)^T$
ball	$(402,1809)^T$	9	$(29.221, 31.558)^T$	$(618.31, 1854.93)^T$
center	$(430,1720)^T$	8	$(32.083, 25.833)^T$	$(661.25, 1763.32)^T$
	$(476,1666)^T$	6	$(35.373, 19.253)^T$	$(710.59, 1658.05)^T$
	$(501,1503)^T$	5	$(37.142, 15.711)^T$	$(737.14, 1601.43)^T$

It can be seen from Table 1 that the final solution obtained by Algorithm 1

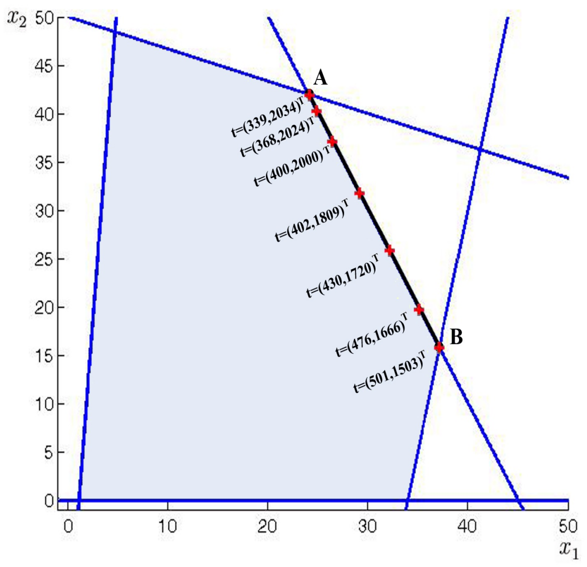

FIGURE 2. Feasible region of MOLP problem (6.1) and outputs of Algorithm 1 for different target vectors.

varies as the target vector varies. Moreover, the value of the first (second) objective function increases (decreases) as the first (second) component of the target vector increases (decreases). This means that the solutions obtained by Algorithm 1 meets the DM's preferences. Furthermore, note that the results of the two cases, exact ball centers and approximate ball centers, are very close to each other. The feasible region of MOLP problem (6.1) and the outputs of Algorithm 1 given in Table 1 are depicted in Figure 2. Note that the line segment between the points A and B (AB) includes all efficient solutions of problem (6.1).

Example 6.2. This simple example shows that $\delta_k = 0$ may occur in Algorithm 1. Consider a MOLP problem as follows:

$$\begin{aligned} \max\ Cx &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ s.t. & \\ -2x_1 + 3x_2 &\leqslant 50, \\ x_1 &\leqslant 15, \\ x_1, x_2 &\geqslant 0. \end{aligned} \tag{6.2}$$

Let $t = (1, 1)^T$ and $w = (0.6, 0.4)^T$. All obtained solutions of Algorithm 1 are given in Table 2.

TABLE 2. All obtained solutions of Algorithm 1 for problem (6.2).

k	x^k	Cx^k	δ_k
1	$(7.50, 10.1051)^T$	$(7.50, 10.1051)^T$	7.5
2	$(15.0, 19.4354)^T$	$(15.0, 19.4354)^T$	0

It can be concluded from Table 2 and Theorem 5.2 that $(x_1, x_2)^T = (15.0, 19.4354)^T$ is a weak efficient solution of problem (6.2).

Example 6.3. Example 6.1 revealed that Algorithm 1 obtains different solutions for different target vectors. This example shows the performance of Algorithm 1 for different weight vectors. Consider the following MOLP problem:

$$\begin{aligned}
\max\ Cx &= \begin{bmatrix} 9 & 2 \\ -3 & 13 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\
s.t. & \\
4x_1 + x_2 &\leqslant 120, \\
x_1 + 3x_2 &\leqslant 90, \\
-x_1 + 3x_2 &\leqslant 70, \\
-6x_1 + x_2 &\leqslant 0, \\
x_2 &\geqslant 0.
\end{aligned} \tag{6.3}$$

Let $t = (270, 135)^T$. Table 3 gives the outputs of Algorithm 1 for different weight vectors.
It can be seen from Table 3 that the value of the first (second) objective increases (decreases) as its weight increases (decreases). The feasible region of MOLP problem (6.3) and the outputs of Algorithm 1 given in Table 3 are depicted in Figure 3. Note that efficient solutions of problem (6.3) lie on the line segments AB and BC.

As discussed in Section 1, there are several algorithms to solve MOLP problems. Most classical methods usually use some extra information to build a single objective function which is a combination of different criteria (e.g., weighted sum method) or a special objective (e.g., ε-constraint method), and then solve the single objective optimization problem. Although the attained solutions may be efficient (or weak efficient), the DM is not often satisfied with them, since these methods suffer from some weaknesses. For instance, the solution of the weighted sum method may not change when the weights of the objectives change. On the other hand, it may change extremely when the weights have slight alterations [31]. Furthermore, sometimes the solution is completely against the weighting scheme [31]. This is because the weighted

TABLE 3. Outputs of Algorithm 1 for the MOLP problem (6.3) for different weight vectors.

	t	k	x^k	Cx^k
Exact ball center	$(0.1,0.9)^T$	5	$(10.000, 26.667)^T$	$(143.33, 316.67)^T$
	$(0.2,0.8)^T$	6	$(11.250, 26.250)^T$	$(153.75, 307.50)^T$
	$(0.3,0.7)^T$	10	$(18.761, 23.745)^T$	$(216.34, 252.40)^T$
	$(0.4,0.6)^T$	9	$(24.753, 20.986)^T$	$(264.75, 198.56)^T$
	$(0.5,0.5)^T$	5	$(25.946, 16.216)^T$	$(265.95, 132.97)^T$
	$(0.6,0.4)^T$	4	$(26.746, 13.012)^T$	$(266.74, 88.920)^T$
	$(0.7,0.3)^T$	4	$(27.322, 10.712)^T$	$(267.32, 57.284)^T$
	$(0.8,0.2)^T$	4	$(27.755, 8.9797)^T$	$(267.75, 33.471)^T$
	$(0.9,0.1)^T$	3	$(28.086, 7.6325)^T$	$(268.04, 14.965)^T$
Approximate ball center	$(0.1,0.9)^T$	4	$(10.004, 26.665)^T$	$(143.36, 316.64)^T$
	$(0.2,0.8)^T$	7	$(11.251, 26.250)^T$	$(153.75, 307.49)^T$
	$(0.3,0.7)^T$	10	$(18.758, 23.747)^T$	$(216.31, 252.44)^T$
	$(0.4,0.6)^T$	8	$(24.754, 20.985)^T$	$(264.75, 198.54)^T$
	$(0.5,0.5)^T$	5	$(25.946, 16.216)^T$	$(265.95, 132.97)^T$
	$(0.6,0.4)^T$	5	$(26.747, 13.013)^T$	$(266.75, 88.924)^T$
	$(0.7,0.3)^T$	4	$(27.321, 10.713)^T$	$(267.32, 57.310)^T$
	$(0.8,0.2)^T$	4	$(27.755, 8.9796)^T$	$(267.75, 33.470)^T$
	$(0.9,0.1)^T$	4	$(28.092, 7.6325)^T$	$(268.09, 14.948)^T$

sum method merely moves from one extreme point to another. However, Example 6.3 showed that Algorithm 1 is sensitive to changes in the weights of the objectives. Moreover, the output of Algorithm 1 is consistent with the weights given by the DM. Note that in Algorithm 1 the given MOLP is not converted to any single objective optimization problem.

In the ε-constraint method, the DM is permitted to assign appropriate bounds to the objectives. If the bounds are not specified appropriately, model (2.3) may be infeasible. Indeed, the ε-constraint method suffers from lack of inflexibility of the constraints [15]. Ehrgott and Ruzika [15] suggested the improved ε-constraint method to overcome the weaknesses of the model (2.3). They showed that the new formulation is more flexible in terms of finding efficient solutions, which are close to the bounds of the objectives. In both the ε-constraint method and the Ehrgott and Ruzika's method one objective is selected and the others are considered as bound constraints. However, Algorithm 1 deals with all the objectives in the same manner. Moreover, it tries to achieve efficient (weak efficient) solutions which satisfy the given bounds as much as possible.

Both Algorithm 1 and the reference point method take weights for the objectives and a target or a reference vector from the DM. However, in the reference point method, usually for technical reasons, the reference point should lie between the ideal and the nadir points of the given MOLP problem [28]. Finding the nadir point is a very difficult task [14]. Furthermore, the reference point

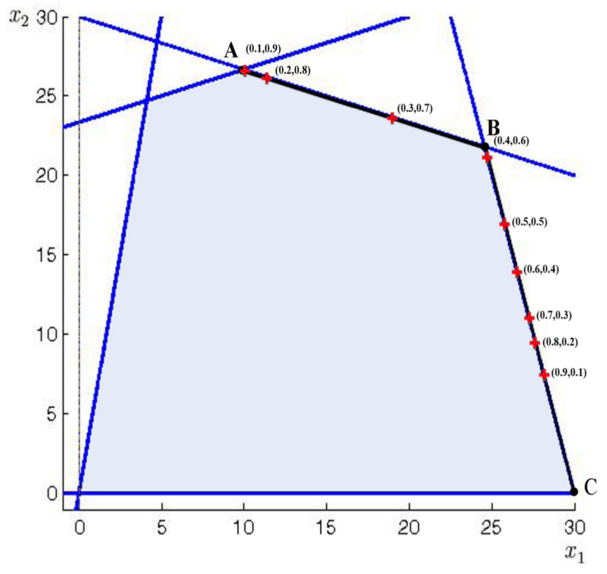

FIGURE 3. Feasible set of MOLP problem (6.3) and outputs of Algorithm 1 for different weight vectors.

method needs an achievement function to convert the MOLP problem to a single optimization problem. Different kinds of achievement functions exist, having different performances [22]. However, Algorithm 1 does not need the nadir point or any achievement function.

6.1. **A simulation study.** Here, the performance of Algorithm 1 on some randomly generated MOLP problems is shown. MATLAB 7.9.0.529 routines were used to implement Algorithm 1 and test its performance on the generated problems. We tested Algorithm 1 on random problems with large number of variables and constraints (up to 3000) and reasonable number of objectives (up to 10). MATLAB function "*random*" was used to generate uniformly distributed random numbers in (-1,1) for all entries of the coefficient matrices A and C. To ensure feasibility of the generated MOLP problems, all variables were considered to be nonnegative and $b_i \in (0,10)$ was a uniformly distributed random number. Moreover, to ensure boundedness, the constraint $\sum_{i=1}^{n} x_i \leqslant max_{i=1,\ldots,m} b_i$ was added to the problems. We ran test problems on a PC with AMD Phenom$^{\text{TM}}$II $\times$ 4 840 processor 3.21 GHz and 4 GB of RAM. To access reliable data, each measure was obtained through the average of 20 runs, ignoring the five worst and five best values. Table 4 shows the results. The last column of Table 4 shows the standard deviation (Std). It can be seen from Table 4 that the performance of Algorithm 1 on these randomly generated MOLP problems is acceptable.

Table 4. CPU times for different randomly generated MOLP problems.

Constraints m	Variables n	Objectives p	Average CPU times (sec.)	Std (sec.)
50		2	0.4681	0.0676
100	300	5	0.6918	0.9393
200		10	2.3370	3.1129
	50	2	0.2797	0.4909
300	100	5	0.5656	2.3118
	200	10	2.1268	2.3118
50		2	0.8371	1.2590
100	500	5	1.2439	2.0585
200		10	2.8959	2.3400
	50	2	0.5063	1.0616
500	100	5	0.8410	1.5129
	200	10	2.5906	3.8390
50		2	2.6452	3.9409
100	1000	5	3.8052	0.3668
200		10	7.2859	1.0016
	50	2	1.1444	0.2981
1000	100	5	2.1116	1.1613
	200	10	4.9125	2.4555
100		2	7.1871	0.2950
200	1500	5	11.2565	0.6581
300		10	17.8184	1.2489
	100	2	2.0642	0.6234
1500	200	5	4.1991	1.3668
	300	10	9.4366	1.9361
100		2	13.0512	0.2855
200	2000	5	19.9915	1.2272
300		10	32.2033	1.7519
	100	2	2.1407	0.8103
2000	200	5	6.3292	2.4675
	300	10	10.8480	2.6244
100		2	29.7989	0.4448
200	3000	5	44.6041	1.3670
300		10	46.3087	2.1528
	100	2	1.2172	0.1621
3000	200	5	3.3515	1.2415
	300	10	6.7225	3.1145

7. Conclusion

We have proposed an algorithm based on finding ball center of some polytopes obtained from the feasible region to solve a multiobjective linear programming (MOLP) problem. The main feature of the algorithm is that it converges to a solution of the MOLP problem through the interior of the feasible region. In addition, it solves a MOLP problem without the aid of an auxiliary objective function such as scalarizing, achievement, utility, or other objective functions. It was proved that the algorithm always converged to a weak efficient solution of the MOLP problem. Moreover, it converges to an efficient solution in some special cases. Furthermore, algorithm taken as input the decision maker's target vector and weights of the objectives in order to find a solution meeting the preferences. Furthermore, the algorithm attains different solutions when the target and/or weight vectors vary. Some randomly generated MOLP test problems were generated and solved to demonstrate the performance of the algorithm. Applying the algorithm in an interactive way to deal with a MOLP problem is a good direction for further research. Also, extending the idea similar to the one used in our algorithm to solve other multiobjective optimization problems such as network flows and nonlinear problems can be investigated for future researches.

Acknowledgment

The authors express their gratitude to the two anonymous referees for their comments and suggestions on the first version of the manuscript.

References

[1] A. Abel and P. Korhonen, Using aspiration levels in an interactive interior multiobjective linear programming algorithm, *European J. Oper. Res.* **89** (1996), no. 1, 193–201.

[2] B. Aghezzaf and T. Ouaderhman, An interactive interior point algorithm for multiobjective linear programming problems, *Oper. Res. Lett.* **29** (2001), no. 4, 163–170.

[3] A. Arbel, An interior multiobjective linear programming algorithm, *Comput. Oper. Res.* **20** (1993), no. 7, 723–735.

[4] A. Arbel and L. G. Vargas, Euclidean centers: computation, properties and a MOLP application, *Math. Comput. Model.* **48** (2008), no. 1-2, 197–205.

[5] P. Armand, Finding all maximal efficient faces in multiobjective linear programming, *Math. Program.* **61** (1993), no. 3, 357–375.

[6] P. Armand and C. Malivert, Determination of the efficient set in multiobjective linear programming, *J. Optim. Theory Appl.* **70** (1991), no. 3, 467–489.

[7] D. S. Atkinson and P. M. Vaidya, A scaling technique for finding the weighted analytic center of a polytope, *Math. Program.* **57** (1992), no. 2, 163–192.

[8] R. Benayoun, J. D. Montgolfier, J. Tergny and O. Laritchev, Linear programming with multiple objective functions: Step method (STEM), *Math. Program.* **1** (1971), no. 1, 366–375.

[9] P. T. Boggs, P. D. Domich, J. R. Donaldson and C. Witzgall, Algorithmic enhancements to the method of centers for linear programming problems, *INFORMS J. Comput.* **1** (1989), no. 3, 159–171.

[10] R. Caballero, M. Luque, J. Molina and F. Ruiz, MOPEN: A computational package for linear multiobjective and goal programming problems, *Decis. Support Syst.* **41** (2005) 160–175.

[11] A. Cambini and L. Martein, Generalized Convexity and Optimization, Lecture Notes in Econom. and Math. Systems, 616, Springer-Verlag, Berlin, 2009.

[12] H. K. Chen and H. W. Chou, Solving multiobjective linear programming problems- a generic approach, *Fuzzy Sets and Systems* **82** (1996), no. 1, 35–38.

[13] J. G. Ecker, N. S. Hegner and I. A. Kouada, Generating all maximal efficient faces for multiple objective linear programs, *J. Optim. Theory Appl.* **30** (1980), no. 3, 353–381.

[14] M. Ehrgott, Multicriteria Optimization, Springer-Verlag, Berlin, 2005.

[15] M. Ehrgott and S. Ruzika, Improved ε-constraint method for multiobjective programming, *J. Optim. Theory Appl.* **138** (2008), no. 3, 375–396.

[16] J. T. Fagan and J. E. Falk, A method of Euclidean centers, *Comput. Oper. Res.* **23** (1996), no. 1, 13–25.

[17] S. I. Gass and P. G. Roy, The compromise hypersphere for multiobjective linear programming, *European J. Oper. Res.* **144** (2003), no. 3, 459–479.

[18] P. Huard, Resolution of mathematical programming with nonlinear constraints by the method of centers, in: J. Abadie (ed.), Nonlinear Programming, pp. 207–219, North-Holland, Amsterdam, 1964.

[19] P. G. Ipsilandis, Multiobjective linear programming model for scheduling linear repetitive projects, *J. Constr. Eng. Manage.* **133** (2007), no. 6, 417–424.

[20] H. Isermann, The enumeration of the set of all efficient solutions for a linear multiple objective program, *Oper. Res. Q.* **28** (1977), no. 1, 711-725.

[21] D. F. Jones and M. Tamiz, Practical Goal Programming, Springer, Berlin, 2010.

[22] M. Luque, K. Miettinen, A. B. Ruiz and F. Ruiz, A two-slope achievement scalarizing function for interactive multiobjective optimization, *Comput. Oper. Res.* **39** (2012), no. 7, 1673–1681.

[23] B. Metev and I. Yourdanov, Use of reference points for MOLP problems analysis, *European J. Oper. Res.* **68** (1993), no. 3, 374–378.

[24] K. G. Murty, New Sphere Methods for LP, *Tutorials in OR, INFORMS J. Comput.* **2** (2009), no. 1, 62–80.

[25] K. G. Murty and M. R. Oskoorouchi, Note on implementing the new sphere method for LP using matrix inversions sparingly, *Optim. Lett.* **3** (2009), no. 1, 137–160.

[26] K. G. Murty and M. R. Oskoorouchi, Sphere Methods for LP, *Algorithmic Oper. Res.* **5** (2010), no. 1, 21–33.

[27] L. Pourkarimi, M. A. Yaghoobi and M. Mashinchi, Determining maximal efficient faces in multiobjective linear programming problem, *J. Math. Anal. Appl.* **354** (2009), no. 1, 234–248.

[28] F. Ruiz, M. Luque and J. Caballero, A classification of the weighting schemes in reference point procedures for multiobjective programming, *J. Oper. Res. Soc.* **60** (2008), no. 4, 544–553.

[29] L. Shao and M. Ehrgott, Approximating the nondominated set of an MOLP by approximately solving its dual problem, *Math. Methods Oper. Res.* **68** (2008), no. 2, 257–276.

[30] G. Sonnevend, An “analytical centre” for polyhedrons and new classes of global algorithms for linear (smooth, convex) programming, in: A. Prekopa, J. Szelezsaan and B. Strazicky (eds.), System Modelling and Optimization, pp. 866–875, Lect. Notes Cont. Inf. Sci. 84, Springer, 1986.

[31] R. E. Steuer, Multiple Criteria Optimization: Theory, Computation, and Application, Robert E. Krieger Publishing, Malabar, 1989.

[32] E. Tarek, Method of centers algorithm for multi-objective programming problems, *Acta Math. Sci. Ser. B Engl. Ed.* **29** (2009), no. 5, 1128–1142.

[33] T. B. Trafalis and R. M. Alkahtani, An interactive analytic center trade-off cutting plane algorithm for multiobjective linear programming, *Comput. Ind. Eng.* **37** (1999), no. 3, 649–669.

[34] H. M. Winkels and M. Meika, An integration of efficiency projections into the Geoffrion approach for multiobjective linear programming, *European J. Oper. Res.* **16** (1984), no. 1, 113–127.

[35] W. Xiao, Z. Liu, M. Jiang and Y. Shi, Multiobjective linear programming model on injection oilfield recovery system, *Comput. Math. Appl.* **36** (1998), no. 5, 127–135.

[36] P. L. Yu and M. Zeleny, The set of all nondominated solutions in linear cases and a multicriteria simplex method, *J. Math. Anal. Appl.* **49** (1975), no. 2, 430–468.

[37] M. Zeleny, Linear Multiobjective Programming, Lecture Notes in Econom. and Math. Systems, 95, Springer-Verlag, Berlin-New York, 1974.

[38] C. Zopounidis, D. K. Despotis and I. Kamaratou, Portfolio selection using the ADELAIS multiobjective linear programming system, *Comput. Econ.* **11** (1998), no. 3, 189–204.

(Alireza Hosseini Dehmiry) Department of Applied Mathematics, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran.
E-mail address: `dehmiry@mail.vru.ac.ir`

(Mohammad Ali Yaghoobi) Department of Applied Mathematics, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran.
E-mail address: `yaghoobi@uk.ac.ir`