RAIRO-Oper. Res. 53 (2019) 1229–1244
https://doi.org/10.1051/ro/2018083

RAIRO Operations Research
www.rairo-ro.org

A COMPUTATIONALLY EFFICIENT ALGORITHM TO APPROXIMATE THE PARETO FRONT OF MULTI-OBJECTIVE LINEAR FRACTIONAL PROGRAMMING PROBLEM

Bogdana Stanojević[1,*] and Milan Stanojević[2]

Abstract. The main contribution of this paper is the procedure that constructs a good approximation to the non-dominated set of multiple objective linear fractional programming problem using the solutions to certain linear optimization problems. In our approach we propose a way to generate a discrete set of feasible solutions that are further used as starting points in any procedure for deriving efficient solutions. The efficient solutions are mapped into non-dominated points that form a 0th order approximation of the Pareto front. We report the computational results obtained by solving random generated instances, and show that the approximations obtained by running our procedure are better than those obtained by running other procedures suggested in the recent literature. We evaluated the quality of each approximation using classic metrics.

Mathematics Subject Classification. 90C29, 90C32.

Received July 13, 2017. Accepted September 19, 2018.

1. Introduction

In [14], Ruzika and Wiecek emphasized the need of finding an approximate efficient set or an approximate non-dominated set to multiple objective problems, especially when an exact description is not available, or the decision maker is not interested to obtain the complete solution set due to overflow of information. They also presented a survey of the approximation methods in multiple objective programming, proposed a classification scheme for reviewing and comparing diverse approaches, and included a discussion on the quality measures for approximations. In 2011, Hartikainen *et al.* [8] developed a theoretical approach to constructing a Pareto front approximation to computationally expensive multi-objective optimization problems based on the concept of inherent non-dominance. They underlined the usefulness of a Pareto front approximation for decision making.

In 2014, Kirlik and Sayin [9] introduced an algorithm based on ε-constraint method for generating all non-dominated solutions of multi-objective discrete optimization problems. They proposed a novel way of partitioning the search space in terms of rectangles. In 2013, Pereyra *et al.* [12] described an approach for constructing an equispaced Pareto front for constrained bi-objective optimization. In 2012, Ehrgott *et al.* [6] presented a dual

Keywords. Fractional programming, multiple objective programming, efficient solution, non-dominated point, 0th order approximation.

[1] Mathematical Institute of the Serbian Academy of Sciences and Arts, Serbia.
[2] Faculty of Organizational Sciences, University of Belgrade, Serbia.
*Corresponding author: bgdnpop@mi.sanu.ac.rs

Article published by EDP Sciences

variant of Benson's "outer approximation algorithm" for multiple objective linear programming, thus making the multiple objective optimization problems and the approaches for estimating the non-dominated sets of such problems actual again.

Shen *et al.* [15] emphasized that fractional programming is one of the most successful fields today in nonlinear optimization problems. The multiple-objective linear fractional programming (MOLFP) was widely studied and several surveys can be found in literature (see for instance [16]). Ruan and Gao [13] proposed a way to finding global solutions to fractional programming problem with ratio of non-convex functions. Addressing the bi-criteria linear fractional problem, Choo and Atkins [3] shown that the Pareto frontier of such problem is the image of a finite number of connected line segments of efficient solutions. They proposed a simple algorithm using a one-dimensional parametric linear programming techniques to evaluate the Pareto frontier.

Hamacher *et al.* [7] developed two box algorithms to compute a finite representative system for the non-dominated set of a discrete bi-criterion optimization problem. They applied the ϵ-constraint method to finding an approximation of a given accuracy to the optimization problem by inspecting the criterion space, and used the cardinality, accuracy, representation error and cluster density to evaluate the quality of their approximation.

Caballero and Hernández [1] introduced a method to estimate the weakly efficient set for the multi-objective linear fractional programming problem. Costa [4] proposed a technique to compute the maximum of a weighted sum of the objective functions in MOLFP, and Costa and Alves [5] introduced a reference point technique to compute non-dominated solutions to the same problem. Recently, in [17,18] two – apparently different – iterative procedures for finding efficient solutions to the MOLFP were presented. The purpose of Section 4 is to show that the two procedures are, in essence, the same; hence, similar results are to be expected no matter which of them is invoked.

The work presented in [19] is a continuation of [18] by the same authors. They improved their previous procedure by splitting the feasible set in convenient regions, and then generating a non-dominated point in each of them. The set of regions is dynamically changed in accordance to the currently obtained non-dominated point. The distance between each two generated non-dominated points is controlled to be greater than or equal to a given threshold. Their algorithm has elements of an ε-constraints method, since they define their special regions by adding constraints on the lower values of the objective functions. Three issues remained hidden in their algorithm: the sequence of the regions to be analyzed, the selection of the starting point inside each region, and the way to choose the weights in the objective function. Their method is able to control the cardinality of the resulting approximation set through a given upper bound parameter. Stopping the algorithm by reaching the upper bound on the number of efficient solutions generated might leave uncovered an important part of the Pareto frontier, depending on the sequence rule for a region to be analyzed. In their numerical example, the authors maximized the first objective function over the current region in order to obtain the starting point, and this particular rule automatically reduced the general algorithm to an ε-constraints algorithm. Reporting the results for a 3-objective linear fractional programming problem from the literature, we compare our approach to the approach introduced in [19].

The above mentioned approaches that aimed to compute Pareto front approximations of 0th order, first analyzed the criterion space, and then introduced some bounding constraints to the original feasible set according to the values of one objective function (see again [3,4,7,12,19]). On the other side, our method from the beginning inspects the feasible set, and then efficiently approaches the Pareto front.

Our solution approach is presented in Section 5 and constructs a 0th order approximation of the Pareto front of multi-objective linear fractional programming problems. Our algorithm generates in a special way the feasible solutions that are further used as starting points in any procedure for deriving efficient solutions, thus mapping them into non-dominated points that form a good approximation of the Pareto front. We call our new approach "the convex combination of efficient solutions" (CCES) due to the generation of new starting points from the convex combinations of p (that is the number of the objective functions) efficient solutions already generated.

Our computational results are presented in Section 6. A statistic of the results obtained for random generated instances of bi-objective problems, including the evaluations of the quality of the approximations based on the

metrics (hyper-area difference, Pareto spread, accuracy, number of distinct choices, and cluster) introduced in [20], is reported in Section 6.2. Section 7 concludes the paper with final remarks.

2. Problem's formulation

The MOLFP problem is defined as follows:

$$\text{“}\max_{x\in X}\text{”}\left\{\frac{N_1(x)}{D_1(x)},\ldots,\frac{N_p(x)}{D_p(x)}\right\} \tag{2.1}$$

where

(i) $X=\{x\in R^n|Ax\leq b, x\geq 0\}$ is a convex and bounded set,
(ii) A is an $m\times n$ constraint matrix. x is an n-dimensional vector of decision variables and $b\in R^m$,
(iii) $N_i(x)=c_i^Tx+\alpha_i, D_i(x)=d_i^Tx+\beta_i, \forall i=1,\ldots,p$,
(iv) $c_i,d_i\in R^n, \alpha_i,\beta_i\in R, \forall i=1,\ldots,p$,
(vi) $d_i^Tx+\beta_i>0, \forall i=1,\ldots,p, \forall x\in X$,
(vii) $p\geq 2$.

The notation $z_i(x)=\frac{N_i(x)}{D_i(x)}$, $i=1,\ldots,p$ is also used through the paper. We follow [11] in assuming that $c_i^Tx+\alpha_i>0$, $\forall i=1,\ldots,p$, $\forall x\in X$.

The term “max” being used in problem (2.1) is for finding all efficient solutions in a maximization sense in terms of the following definition.

Definition 2.1. A feasible solution $x^*\in X$ is said to be efficient solution to problem (2.1) if and only if there is no $x\in X$ such that $\frac{N_i(x)}{D_i(x)}\geq\frac{N_i(x^*)}{D_i(x^*)}$, for all $i=1,\ldots,p$, and $\frac{N_{i_0}(x)}{D_{i_0}(x)}>\frac{N_{i_0}(x^*)}{D_{i_0}(x^*)}$ for at least an index i_0.

The set of all efficient solutions is called efficient set.

The image of the efficient set through the objective functions is called non-dominated set. The non-dominated set of the multiple objective optimization problem is called Pareto front.

The classic idea for solving the MOLFP problems is to weighting the objective functions and to solve a parametric single-objective optimization problem

$$\max_{x\in X}\sum_{i=1}^{p}w_i\frac{N_i(x)}{D_i(x)}, \tag{2.2}$$

where $w_i\in R_+$, $i=\overline{1,p}$. This method can be used either in an *a priori* context, *i.e.* one efficient solution is desired and w_i, $i=\overline{1,p}$ are chosen according to the relative importance of the criteria; or in an a posteriori context, *i.e.* all extreme points that are efficient solutions to problem (2.1) are desired and they are obtained as solutions to problem (2.2) for different values of the parameters w_i. The objective function in (2.2) is not linear fractional anymore, thus it is hard to solve problem (2.2) directly, even for fixed values of the parameters w_i, $i=\overline{1,p}$.

Several approaches for generating an efficient solution to the MOLFP problem can be found in literature. Two of them, introduced in [17] (let us call it “deviational variables approach” (DVA)) and [18] (let us call it “parametric approach” (PA)), are essential to our work. Both use a starting point that may be any feasible solution, and improve iteratively the values of the objective functions at the current solutions until an efficient solution is reached. We can use any of them as part of our new approach CCES. In Section 4 we will compare them from the theoretical point of view, and show that they are extremely similar.

Practically any approach that generates an efficient solution can be involved in a wider approach for approximating the non-dominated set, but it is important to obtain an approximation of a good quality. A draft overview of the metrics used for measuring the quality of the approximation of the non-dominated set is given

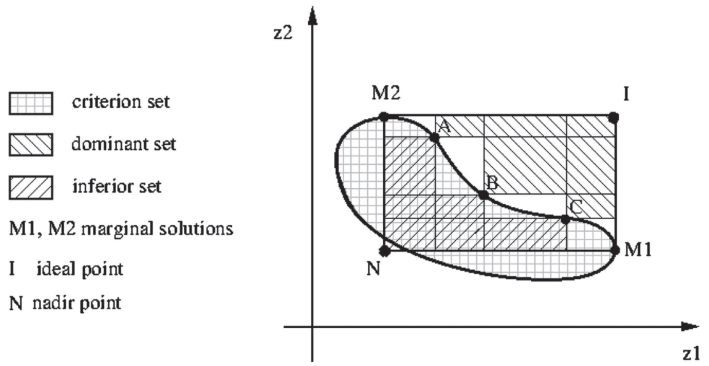

FIGURE 1. Overview of the specified terminology for a bi-objective maximization problem.

in Section 3. Both papers [17, 18] suggested an approach for approximating the efficient set of the MOLFP problems.

The approach mentioned in [17] – let us call it "the convex combination of the marginal solutions" (CCMS) – promoted the idea of using the feasible solutions that are convex combination of marginal solutions as starting points in DVA. The authors used the weights that represent the coefficients of the relative importance of the objective functions as weights in constructing the convex combination.

The approach mentioned in [18] – let us call it "the random generation" (RG) – recommended to partition the feasible set in a certain number of rectangles, and then, from each partition, to select randomly a starting point for PA. Then, the generated efficient solutions form the approximation.

In our experiments with computational results reported in Section 5, we compare the approximation obtained by running our new method CCES to the approximations obtained by running CCMS and RG.

3. Measures of approximation

In this section we briefly present the metrics introduced in [20] for measuring the quality of the approximation of the non-dominated set of MOLFP problems. In [20] the metrics were described for a problem where all objectives were minimized, and according to the so called "good point" and "bad point" that are estimations of the ideal point and max point of the problem. We restrict our attention to the MOLFP problems where the objectives have to be maximized. Also, we define all metrics according to the ideal point and nadir point. All definitions in this section are adapted from [20].

Let us denote by P_{gen} the set of the generated non-dominated points.

The surface inferior to a given point is the hyper-rectangle defined by the given point and the nadir point of the problem. The inferior set of a set of given points is the union of the inferior sets of the given points. Similar, the dominant set of a given point is the hyper-rectangle defined by the given point and the ideal point; and the dominant set of a set of given points is the union of the dominant sets of the given points.

For a graphical visualization of the specified terminology we refer the reader to Figure 1. In this figure the set of the generated efficient solutions is $P_{\text{gen}} = \{M1, M2, A, B, C\}$, and the inferior and dominant sets of P_{gen} are distinctly shaded.

3.1. The hyper-area difference

The hyper-area difference of P_{gen} (denoted by HD (P_{gen})) is the difference between the Lebesgue measure of the inferior set of the ideal point and the Lebesgue measure of the inferior set of P_{gen}. The Lebesgue measure is an extension of the classical notions of length and area to more complicated sets. The Lebesgue measure of

a p-dimensional hyper-rectangle is its p-volume. An approximation P^1_{gen} is considered better than another one P^2_{gen} if and only if $\text{HD}\left(P^1_{\text{gen}}\right) < \text{HD}\left(P^2_{\text{gen}}\right)$.

3.2. The Pareto spread

The Pareto spread of an approximation refers to the ranges of the objective functions and ranges covered by the solutions in the approximation. The Pareto spread is described by two metrics: the overall Pareto spread and the Pareto spread with respect to each objective. The Pareto spread of P_{gen} with respect to the ith objective function, $i = 1, \ldots, p$, is defined as

$$\text{OS}_i\left(P_{\text{gen}}\right) = \frac{\overline{z}^i_{\max} - \overline{z}^i_{\min}}{z^i_{\max} - z^i_{\min}},$$

where

$$\overline{z}^i_{\max} = \max\left\{e_i | e \in P_{\text{gen}}\right\}, \quad \overline{z}^i_{\min} = \min\left\{e_i | e \in P_{\text{gen}}\right\},$$
$$z^i_{\max} = \max\left\{z_i\left(x\right) | x \in X\right\}, \; z^i_{\min} = \min\left\{z_i\left(x\right) | x \in X\right\}.$$

The overall Pareto spread of P_{gen} is defined as

$$\text{OS}\left(P_{\text{gen}}\right) = \text{OS}_1\left(P_{\text{gen}}\right) \cdot \ldots \cdot \text{OS}_p\left(P_{\text{gen}}\right).$$

An approximation P^1_{gen} is considered better than another one P^2_{gen} if and only if $\text{OS}\left(P^1_{\text{gen}}\right) > \text{OS}\left(P^2_{\text{gen}}\right)$. Similar, an approximation P^1_{gen} is considered better with respect to the ith objective than another one P^2_{gen} if and only if $\text{OS}_i\left(P^1_{\text{gen}}\right) > \text{OS}_i\left(P^2_{\text{gen}}\right)$. For those approximations that contain the efficient marginal solutions, the Pareto spread with respect to each objective is equal to 1, and consequently the overall Pareto spread is also equal to 1.

3.3. The accuracy

For measuring the accuracy of an approximation P_{gen} the so called frontier approximation of P_{gen} (denoted by $\text{AP}\left(P_{\text{gen}}\right)$) is involved. $\text{AP}\left(P_{\text{gen}}\right)$ is defined as the Lebesgue measure of the region bounded by the inferior and the dominant sets of P_{gen}. The entire Pareto front falls in the region described above, and the accuracy of an approximation is defined as

$$\text{AC}\left(P_{\text{gen}}\right) = \frac{1}{\text{AP}\left(P_{\text{gen}}\right)}.$$

The approximation P_{gen} is as accurate as $\text{AP}\left(P_{\text{gen}}\right)$ is closer to zero, and as $\text{AC}\left(P_{\text{gen}}\right)$ is closer to infinity.

3.4. The number of distinct choices

The number of distinct choices provided by an approximation is related to a parameter ν used for establishing whether two different non-dominated points are distinct or not from the point of view of the Decision Maker. The parameter ν represents the number of distinct choices. Partitioning the criterion space by a grid of size ν, a number of ν^p small hypercubes of dimension $\mu_1 \times \mu_2 \times \ldots \times \mu_p$ are obtained, where

$$\mu_i = \frac{z^i_{\max} - z^i_{\min}}{\nu}, i = 1, \ldots p. \tag{3.1}$$

Counting the hyper-rectangles of the grid that have non-empty intersection with P_{gen}, the number of distinct choices $\text{NDC}_\mu\left(P_{\text{gen}}\right)$ is obtained.

Comparing two approximations, the better is the one that has greater number of distinct choices.

The ratio between the cardinality of P_{gen} and the number of distinct choices defines another metric, the *cluster* CL_μ. Thus the cluster is defined as

$$\text{CL}_\mu\left(P_{\text{gen}}\right) = \frac{\left|P_{\text{gen}}\right|}{\text{NDC}_\mu\left(P_{\text{gen}}\right)}.$$

Comparing two approximations, the better is the one that is less clustered, *i.e.* it has smaller value for the metric cluster.

4. The comparison of DVA and PA

The aim of this section is to compare theoretically the approaches DVA and PA in order to involve the more convenient one in procedures for generating approximations to the Pareto front of the BOLFP. The conclusions will be drawn for the MOLFP problem (2.1).

4.1. The DVA procedure

The DVA (Procedure P_{sMOLF} in [17]) uses as inputs an instance of problem (2.1), and an arbitrary feasible starting point $x^1 \in X$. It consists of the following steps:

Algorithm 4.1. The DVA procedure.

Step 1. Set $x^* = x^1$ and $k = 1$.

Step 2. Solve problem (4.1)

$$\begin{aligned} \max \ & \sum_{i=1}^{p} \left(d_i^- + d_i^+\right) \\ \text{s.t.} \ & c_i^T x + \alpha_i - d_i^+ = n_i \theta_i, \quad i = 1, \ldots, p, \\ & d_i^T x + \beta_i + d_i^- = m_i \theta_i, \quad i = 1, \ldots, p, \\ & x \in X \\ & d_i^+, d_i^-, \theta_i \geq 0 \qquad\qquad i = 1, \ldots, p, \end{aligned} \tag{4.1}$$

where $n_i = c_i x^* + \alpha_i$, $m_i = d_i x^* + \beta_i$, for each $i = \overline{1,p}$. Let $\left(x^k, \left(\theta_i^k, d_i^{k+}, d_i^{k-}\right)_{i=\overline{1,p}}\right)$ be its optimal solution. Set $t^k = \sum_{i=1}^{p} \left(d_i^{k-} + d_i^{k+}\right)$.

Step 3. If $t^k = 0$, then STOP, x^k is an efficient solution to (2.1). Otherwise, set $x^* = x^k$, $k = k+1$ and go to Step 2.

4.2. The PA procedure

The PA (Algorithm 4.1 in [18]) uses as inputs an instance of problem (2.1), an arbitrary feasible starting point x^0, a set of weights $w_i > 0$, $i = \overline{1,p}$, $\sum_{i=1}^{p} w_i = 1$, and an admissible tolerance $\epsilon > 0$. It consists of the following steps:

Algorithm 4.2. The PA procedure.

Step 1. Set $\lambda^0 = \left(z_i\left(x^0\right)\right)_{i=\overline{1,p}}$, $k = 0$, $X^k = X \cap \left\{x \in R^n | z_i(x) \geq \lambda_i^k, i = \overline{1,p}\right\}$. Set $w = \frac{1}{\min\{w_1, \ldots, w_p\}}$.

Step 2. Solve problem (4.2)

$$\begin{aligned} \max \ & \sum_{i=1}^{p} w_i \left(N_i(x) - \lambda_i^k D_i(x)\right) \\ \text{s.t.} \ & x \in X^k \end{aligned} \tag{4.2}$$

and denote by x^{k+1} and v^{k+1} the optimal solution and the optimal value, respectively. Then, set $u^{k+1} = wv^{k+1}$.

Step 3. If $u^{k+1} < \epsilon$, then go to Step 4; else $k = k+1$, $\lambda^k = \left(z_1\left(x^k\right), \ldots, z_p\left(x^k\right)\right)$, $X^k = X \cap \left\{x \in R^n | z_i(x) \geq \lambda_i^k, i = 1, \ldots, p\right\}$, and return to Step 2.

Step 4. If $u^{k+1} = 0$, then stop, x^{k+1} is an efficient solution to (2.1); else STOP, x^{k+1} is an ε-efficient solution to (2.1) where $\varepsilon = (\epsilon, \ldots, \epsilon) \in R^p$. Also, every $x \in X^k$ is an ε-efficient solution to (2.1) which approximates an efficient solution.

4.3. The comparison

We may replace problem (4.1) by (4.3)

$$\begin{array}{ll} \max & \sum_{i=1}^{p} w_i \left(\mu_i d_i^- + d_i^+\right) \\ \text{s.t.} & c_i^T x + \alpha_i - d_i^+ = n_i \theta_i, \quad i = 1, \ldots, p, \\ & d_i^T x + \beta_i + d_i^- = m_i \theta_i, \quad i = 1, \ldots, p, \\ & x \in X \\ & d_i^+, d_i^-, \theta_i \geq 0 \qquad\qquad i = 1, \ldots, p, \end{array} \tag{4.3}$$

in the DVA, with the same meaning for n_i and m_i. We may favor either the negative or positive deviational variables to become faster or slower equal to 0, as the corresponding μ_i is less or greater than 1; and/or certain objectives to reach higher values, as their weights w_i are relatively greater or less. Using μ_i, $i = 1, \ldots, p$, a normalization of numerators and denominators before aggregation may be achieved, but not a complete equivalence between linear fractional functions and the corresponding linear functions. Using w_i, $i = 1, \ldots, p$ a refined aggregation of the objectives may be achieved, but, unfortunately, in practice, when the objective functions describe quantities that have incomparable units of measuring, the values of these weights are rather irrelevant.

Choosing

$$\mu_i = \frac{N_i(x^*)}{D_i(x^*)}, \; i = 1, \ldots, p,$$

and putting the deviational variables as subjects in their constraints, we obtain

$$\begin{aligned} d_i^+ &= c_i^T x + \alpha_i - n_i \theta_i = N_i(x) - N_i(x^*)\,\theta_i, \qquad i = 1, \ldots, p, \\ d_i^- &= m_i \theta_i - \left(d_i^T x + \beta_i\right) = D_i(x^*)\,\theta_i - D_i(x), \quad i = 1, \ldots, p, \end{aligned}$$

and the following problem (4.4),

$$\begin{array}{lll} \max & \sum_{i=1}^{p} w_i \left(\dfrac{N_i(x^*)}{D_i(x^*)} \left(D_i(x^*)\,\theta_i - D_i(x)\right) + N_i(x) - N_i(x^*)\,\theta_i \right) & \\ \text{s.t.} & N_i(x) - N_i(x^*)\,\theta_i \geq 0, & i = 1, \ldots, p, \\ & D_i(x^*)\,\theta_i - D_i(x) \geq 0, & i = 1, \ldots, p, \\ & x \in X, & \\ & \theta_i \geq 0, & i = 1, \ldots, p, \end{array} \tag{4.4}$$

that is equivalent to problem (4.3).

Further, problem (4.4) is equivalent to problem (4.5)

$$\begin{array}{ll} \max & \sum_{i=1}^{p} w_i \left(N_i(x) - \dfrac{N_i(x^*)}{D_i(x^*)} D_i(x) \right) \\ \text{s.t.} & N_i(x) \geq N_i(x^*)\,\theta_i, \qquad i = 1, \ldots, p, \\ & D_i(x) \leq D_i(x^*)\,\theta_i, \qquad i = 1, \ldots, p, \\ & \theta_i \geq 0, \qquad i = 1, \ldots, p, \\ & x \in X. \end{array} \tag{4.5}$$

Noticing that the first three constraints together are equivalent to $z_i(x) \geq z_i(x^*)$, $i = 1, \ldots, p$ we derive the equivalence of problems (4.2) and (4.3). Therefore, the optimization problems solved in each iteration of both DVA and PA are slightly different due to the coefficients w_i and μ_i, but in essence they are the same.

The problem solved in PA has $m+p$ constraints and n variables while the DVA solves one with $m+2p$ constraints and $n+3p$ variables. Both procedures PA and DVA have the same complexity with respect to m and n. Note that p is generally a small number comparing to m and n.

The only nontrivial difference among PA and DVA is the use of the tolerance vector ε in PA. This tolerance vector is in fact an implementation detail, that is useful to avoid the possible numerical instability arising in some solvers, or to reduce the running time in the detriment of the approximation quality. The repeated process of solving the mathematical model in each of the procedures PA and DVA theoretically can assure the efficiency of the generated solutions, and no tolerance vector is needed to stop the algorithm. On the other side, replacing the DVA's stopping condition by $t^k < \varepsilon$, the ε-efficiency of the current solution in DVA is assured as well.

See the discussion in Example 6.1, Section 6.1 for a numerical comparison of the approaches DVA and PA.

5. The CCES method

Our algorithm is a recursive procedure that generates $p+\frac{p^k-1}{p-1}$ efficient solutions to problem (2.1), where k represents the depth of the recursion. Let us denote by E_{gen} the set of the generated efficient solutions that approximates the efficient set.

Our approach is based on "divide et impera" principle. In the beginning we insert the efficient marginal solutions M_i, $i=\overline{1,p}$ in E_{gen}. Then we use the centroid of the points $M_1, M_2, \ldots, M_p$ as the starting point in DVA, generate a new efficient solution e_1, and insert it in E_{gen}. In this way we split the initial problem of finding $p+\frac{p^k-1}{p-1}$ efficient solutions starting from M_i, $i=\overline{1,p}$ in p sub-problems. The ith sub-problem consists in generating $\frac{p^{k-1}-1}{p-1}$ efficient points starting from $M_1, \ldots, M_{i-1}, e_1, M_{i+1}, \ldots, M_p$. Each such sub-problem will consist in generating $\frac{p^{k-1}-1}{p-1}$ efficient points, since

$$\frac{p+\frac{p^k-1}{p-1}-p-1}{p}=\frac{\left(p^{k-1}+p^{k-2}+\ldots+1-1\right)}{p}=p^{k-2}+p^{k-3}+\ldots+1=\frac{p^{k-1}-1}{p-1}.$$

Further, we recursively break down each problem into p sub-problems of the same type, until k comes to 1, and the corresponding sub-problems are simply solved by adding one more efficient solution to E_{gen}. As in any classic "divide et impera" based algorithm, the solutions to the sub-problems are combined to give the solution to the original problem.

The formalized recursive procedure for generating the approximation of the efficient set is presented in Algorithm 5.1. The input values of the function called recursion () are: k – the depth of the recursion, and s_i, $i\in\{1,\ldots,p\}$ – the efficient solutions between which $\frac{p^k-1}{p-1}$ new efficient solutions have to be generated. The output L of the recursive function is the set of the generated efficient solutions. Inside the function, s_0 is the centroid of the set $\{M_1, M_2, \ldots, M_p\}$; e is the efficient solution generated by the DVA using the starting point s_0; and the list of points L_i is obtained by solving the ith sub-problem. In the main program, the final approximation E_{gen} will be obtained as $E_{\text{gen}}=\{M_1, M_2, \ldots, M_p\}\cup \text{recursion}\,(M_1, M_2, \ldots, M_p, k)$.

The steps of the recursive function for the CCES are given below.

Algorithm 5.1. L $\leftarrow$ recursion $(s_1 \ldots, s_p, k)$

Step 1. $s_0 \leftarrow \dfrac{1}{p}\displaystyle\sum_{i=1}^{p} s_i$;

Step 2. Generate the efficient solution $e \leftarrow \text{DVA}\,(s_0)$;

Step 3. If $k == 1$ then return $\{e\}$.

Step 4. For $i=1,\ldots,p$ do: $L_i \leftarrow \text{recursion}\,(s_1, \ldots, s_{i-1}, e, s_{i+1}, \ldots, s_p, k-1)$;

Step 5. Return $\left(\displaystyle\bigcup_{i=1}^{p} L_i\right)\cup\{e\}$.

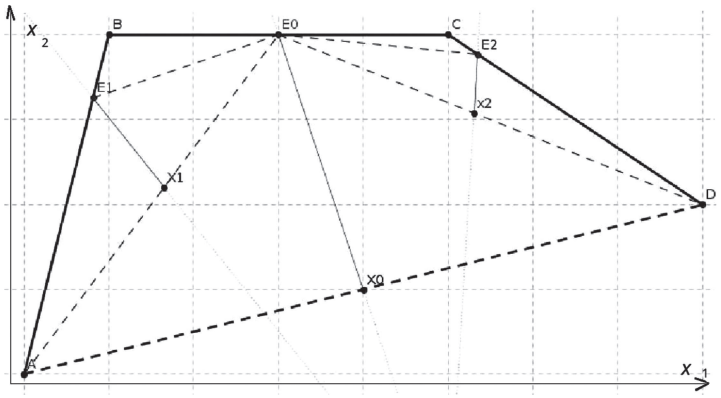

FIGURE 2. A simple example showing graphically, in the solution space, how CCES works.

In what follows we explain how the procedure CCES works on a very simple example. Let us consider that the efficient set of a bi-objective linear fractional programming problem is the line $ABCD$ shown in Figure 2. If the task is to find 5 efficient solutions, the procedure works as follows: the efficient marginal solutions – A and D – are automatically included in the generated efficient set; next efficient solution that will be included in the generated efficient set is $E0$, and it is obtained by applying the procedure DVA to the starting point $x0$ that is the midpoint (centroid) of the line segment AD; further, in the same way as $E0$ was obtained from A and D, the point $E1$ is obtained from A and $E0$ – it is generated by running DVA with the midpoint of $[AE0]$ as starting point. The point $E2$ is obtained by running DVA with the midpoint of $[E0D]$ as starting point. Finally, $E_{\text{gen}} = \{A, E1, E0, E2, D\}$.

Remark 5.2. In the case of choosing starting points on line segments $AE0$ and $E0D$ instead of choosing them on the line segment AD, the total number of iterations in DVA decreases. The total number of iterations decreases even more by considering the starting points on the line segments $[AE1]$, $[E1E0]$, $[E0E2]$ and $[E2D]$. Therefore, from the point of view of the total number of iteration in DVA, the new proposed method is more efficient than the method that uses the starting points that lie on the line segment formed by the marginal solutions (as proposed in [17]).

In addition, the particular cases of the sub-problems of finding efficient solutions between A and $E1$, and between $E2$ and D, are solved extremely fast since DVA has only to confirm, in one iteration, the efficiency of the corresponding midpoints.

6. Numerical results

For our examples and also for our experiments on the random generated instances, we used $\nu = 20$ in formula (3.1) to compute the sizes of the hyper-rectangles that form the grid required in the definition of the number of distinct choices. Experiments were performed on Intel® Core™ i3 at 1.80 GHz and 8 GB RAM using Scilab 5.5.2.

6.1. Examples

The first example is recalled from [17], and is used to show graphically how the new approach CCES works; to compare numerically the approaches DVA and PA; and to compare the approximations obtained by running RG, CCMS, and CCES – all combined with the procedure DVA.

Example 6.1 (Example 4 in [17])**.** Let us consider the following instance of the bi-objective linear fractional programming problem.

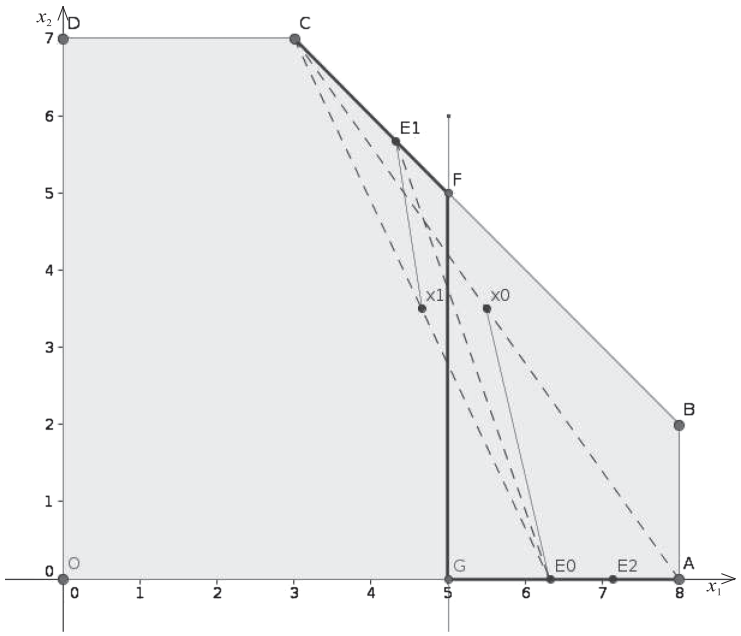

FIGURE 3. The feasible set, the efficient set and the first two steps in running CCES for solving problem (6.1).

$$\begin{aligned} \text{“max”} \quad & \left(z_1(x) = \frac{x_1 + 3x_2 + 1}{x_1 + 5x_2 + 5}, \;\; z_2(x) = \frac{-21x_1 - 19x_2 + 219}{-x_1 - x_2 + 11} \right) \\ \text{s.t.} \quad & x_1 + x_2 \leq 10, \\ & 0 \leq x_1 \leq 8, \\ & 0 \leq x_2 \leq 7. \end{aligned} \tag{6.1}$$

The feasible set of problem (6.1) – the pentagon $OABCD$ – is shown in Figure 3. The marginal solutions are A and C, and the efficient set is the line $CFGA$.

Figure 3 also shows the first two steps in running the procedure CCES for solving problem (6.1). For this instance, the efficient set contains interior points of the feasible set that are all together mapped in a single non-dominated point in the criterion space, thus making the example more complex than that one discussed in the end of Section 5. First, the efficient marginal solutions A and C are added to E_{gen}. The feasible solution $x0$ is the midpoint of [AC] and it generates the efficient solution $E0$. The midpoint of $CE0$ is x_1 and it generates $E1$. The midpoint of $AE0$ is $E2$ that is an efficient solution. Thus, $E_{\text{gen}} = \{C, E1, E0, E2, A\}$ is obtained by calling $recursion\,(C, A, \{C, A\}, 2)$.

Both DVA and PA behaved in the same way when were used for solving problem (6.1). About 300 distinct starting points were generated by running CCES, CCMS, and RG. Both DVA and PA generated the same efficient solution each time when they started from the same point. The weights w were all set to 1 in PA.

Running the three procedures RG, CCMS, and CCES we obtained the results reported in Table 1. These results show that CCES outperforms CCMS regarding hyper-area difference, accuracy and number of iterations; and they have the same values for the rest of the metrics. Both CCES and CCMS outperform RG regarding all metrics except the cluster.

For the rest of our experiments we used the DVA for generating efficient solutions.

Example 6.2 (Problem DEB in [12])**.** Let us consider the following instance of the bi-objective linear fractional programming problem.

TABLE 1. The values of the metrics that evaluate of quality of the approximations obtained by the three methods when solving problem (6.1).

	RG	CCMS	CCES
$\lvert P_{\text{gen}}\rvert\uparrow$	60	129	129
#it $\downarrow$	315	256	134
HD$\downarrow$	0.493526	0.475419	0.475051
OS$\uparrow$	0.836359	1	1
$\text{OS}_1\uparrow$	0.909419	1	1
$\text{OS}_2\uparrow$	0.919664	1	1
AC$\uparrow$	25.46681	189.4665	212.84288
NDC$\uparrow$	23	36	39
CL$\downarrow$	2.60869	3.58333	3.30769

$$\begin{aligned} &\text{“min”} \ \left(z_1(x) = x_1, \ z_2(x) = \tfrac{x_2+1}{x_1} \right) \\ &\text{s.t. } 9x_1 + x_2 \geq 16 \\ &\quad 9x_1 - x_2 \geq 1 \\ &\quad 0.1 \leq x_1 \leq 1 \\ &\quad 0 \leq x_2 \leq 5 \end{aligned} \tag{6.2}$$

Running the three procedures RG, CCMS, and CCES for solving problem (6.2) we obtained the approximations of the Pareto front shown in Figure 4. We refer the reader to this graphical representation to let him observe the performance of CCES comparing to RG and CCMS, but also to conclude about the accuracy of the approximation, since the exact representation of the Pareto front can be found in the literature ([12], Fig. 1).

Example 6.3. The subject of this example is one instance of the bi-objective linear fractional programming with $n = 10$. We used RG, CCMS and CCES combined with DVA to solve the instance, and compared their performance.

In Table 2, we report the values of the metrics obtained by running RG, CCMS, and CCES with 129 and 4097 distinct starting points. RG had a bad performance comparing to both CCMS and CCES, from the point of view of all metrics. The presence of the marginal solutions in the generated efficient set makes the values of the Pareto spread and Pareto spread with respect to each objective to be greater for CCMS and CCES than for RG. The values of the hyper-area difference are similar for CCMS and CCES. The main advantages of CCES over CCMS are related to the accuracy, number of distinct choices, and cluster.

The graphical representations of the non-dominated points generated by RG, CCMS, and CCES are shown in Figure 5. The approximation obtained by running CCES is more accurate and less clustered, comparing to those obtained by running CCMS and RG.

Comparing the results obtained with 4097 starting points to the results obtained with only 129 starting points one can see that there is no improvement considering the number of distinct choices for CCMS, and a very small improvement (7.89%) for CCES, when the number of starting points increased approximately 32 times. This is not the case for RG, where the number of distinct choices increased with 75%; but the reached value is still not significant, comparing to CCMS and CCES. The accuracy increased very little (less than 2.5 times) for RG and CCMS, and a lot (32.13 times) for CCES. The total number of iterations is also significantly smaller for CCES comparing to CCMS and RG.

Example 6.4. problem (6.3), was introduced in [10] in order to describe some difficulties that might appear when solving MOLFP problems. The example was recalled in [19] and used to illustrate the performance of the algorithm presented there.

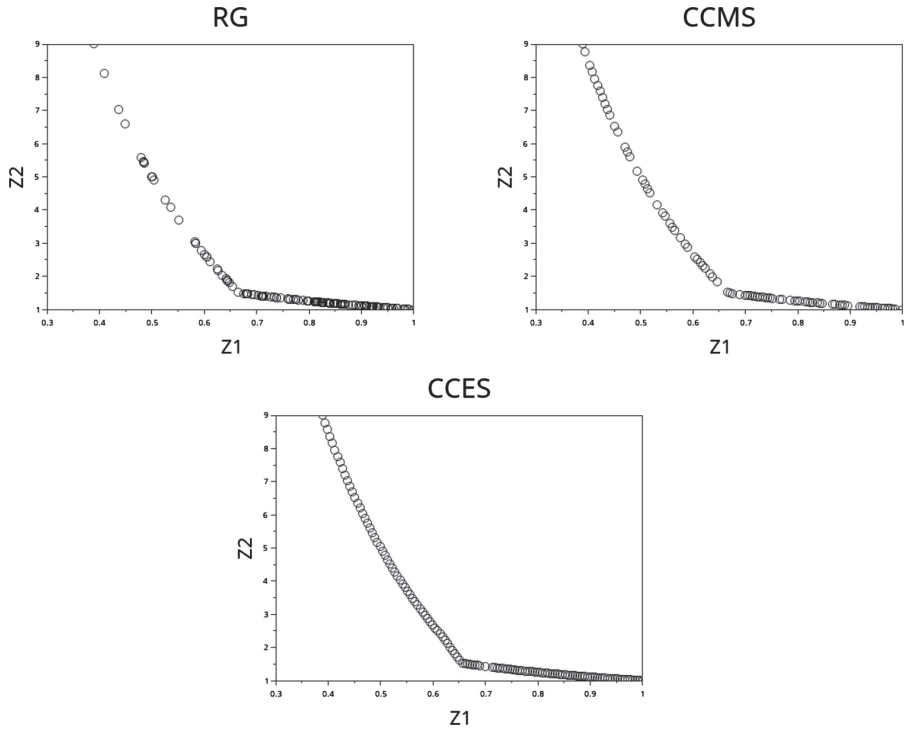

FIGURE 4. Graphical comparison of the three methods when using 129 starting points for solving problem (6.2).

TABLE 2. Computational results for Example 6.3 – the values of the metrics related to the approximations shown in Figure 5.

	129 starting points			4097 starting points		
	RG	CCMS	CCES	RG	CCMS	CCES
$\lvert P_{\text{gen}}\rvert \uparrow$	120	125	129	1435	3076	4097
#it $\downarrow$	434	360	190	13986	11484	4253
Time (s)	0.281	0.223	0.089	10.108	7.058	3.478
HD $\downarrow$	209.71998	207.24428	206.098	208.16195	207.19808	205.90125
OS $\uparrow$	0.0080573	1	1	0.0850226	1	1
$\text{OS}_1 \uparrow$	0.1722160	1	1	0.3233986	1	1
$\text{OS}_2 \uparrow$	0.0467863	1	1	0.2629035	1	1
AC $\uparrow$	0.0600281	0.2532716	2.3881194	0.1458783	0.2608864	76.460119
NDC $\uparrow$	4	20	38	7	20	41
CL $\downarrow$	30	6.25	3.3947	205	153.8	99.926829

$$\begin{aligned} &\text{“max”} \left(z_1(x) = -x_1 + x_2,\ z_2(x) = \frac{x_1 - 4}{-x_2 + 3},\ z_3(x) = \frac{-x_1 + 4}{x_2 + 1} \right) \\ &\text{s.t. } -x_1 + 4x_2 \leq 0 \\ &\quad x_1 - 0.5x_2 \leq 4 \\ &\quad x_1, x_2 \geq 0 \end{aligned} \tag{6.3}$$

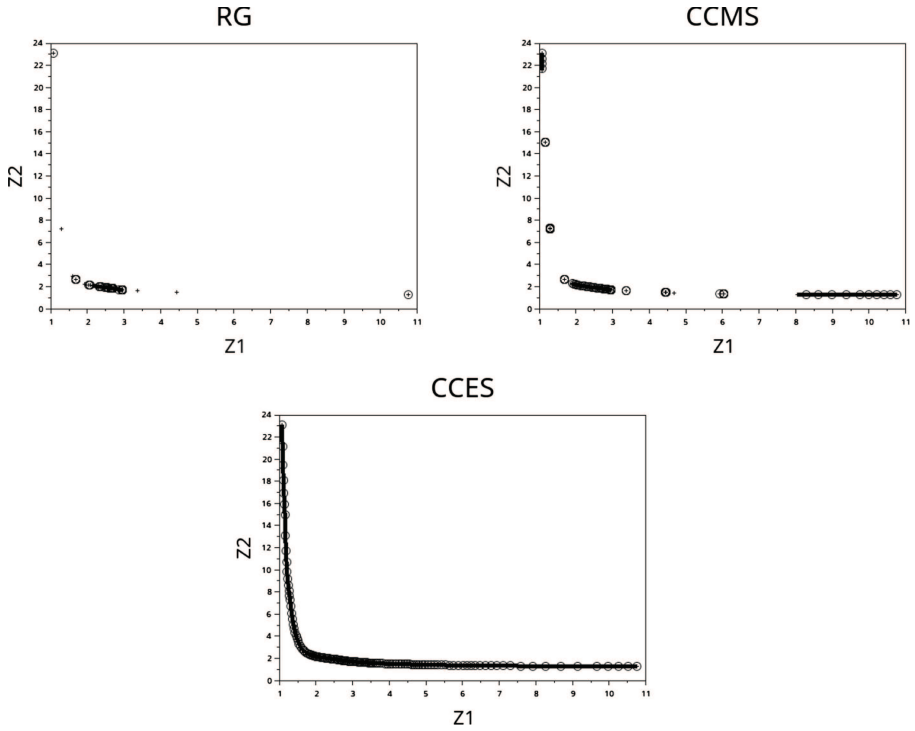

FIGURE 5. Graphical comparison of the three methods for solving the problem considered in Example 6.3. Symbols ○ and + are used for representing the points generated from 129 and 4097 starting points respectively.

TABLE 3. Information regarding the CCES when solving problem (6.3).

	Algorithm [19]	CCES
No. of solved LPs	58	49
No. generated points	15	42

This problem is degenerated in sense of the number of distinct efficient marginal solutions, since the first and third objectives reach their optimal values at the same feasible solution. Due to this particularity, CCES does not behave well, *i.e.* it can determine only two starting points instead of three. In order to overcome this situation, we randomly generated a feasible solution and used it as the third starting point, instead of the missing marginal solution. See Table 3 to compare our results to the results reported in [19].

Figure 6 presents the resulting approximations of the non-dominated set of problem (6.3) obtained by running our algorithm, and the algorithm presented in [19]. Our procedure had to solve only 49 linear problems, and generated 42 non-dominated points, comparing to 58 linear problems and 15 generated non-dominated points reported in [19]. In our opinion, visually, the spread of the generated non-dominated points over the non-dominated set are very good for both algorithms.

Example 6.5. This example was introduced in [2] in order to analyze the connectedness of the efficient solutions for MOLFP problems.

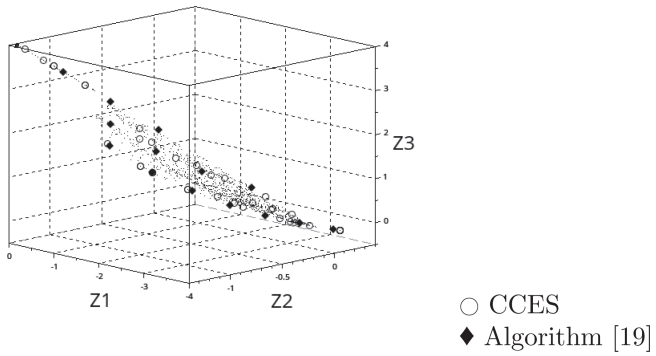

FIGURE 6. Graphical comparison of the resulting approximations of the non-dominated set problem (6.3) generated by the CCES and the algorithm presented in [19].

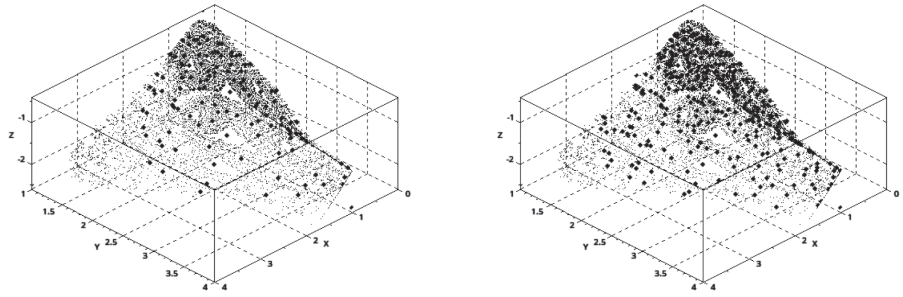

FIGURE 7. Two approximations of the non-dominated set of problem (6.4) containing 360 and 1000 non-dominated points respectively.

$$\begin{aligned} &\text{“max”} \left(z_1(x) = \frac{x_1}{x_2},\, z_2(x) = x_3,\, z_3(x) = \frac{-x_1 - x_3}{x_2} \right) \\ &\text{s.t.} \\ &\quad 0 \le x_1, x_2, x_3 \le 4 \end{aligned} \tag{6.4}$$

Using our procedure to solving problem (6.4), we obtained the results shown in Figure 7. On the left side 360 non-dominated points (obtained in 0.13 s) are presented, and on the right side 1000 (obtained in 0.28 s). Presenting Figure 7 we aim to give a clue about how the approximation of the non-dominated set varies with respect to the depth of the recursion.

6.2. Experiments on random generated instances

The experiments presented in this section were performed on three groups of random generated bi-objective instances with $n = 10$, $n = 20$, and $n = 50$, respectively, where n is the number of the decision variables. For each group (*i.e.* for each value of n), 10 random instances were generated in the same way as in [17]: for each instance, the feasible set was defined by the box constraints $0 \le x_i \le 10$, $i = 1, \ldots, n$, and 10 additional constraints. Each additional constraint was defined as the hyperplane that passes through n randomly chosen points that fulfill all previously defined constraints. The coefficients c_i and d_i, $i = 1, 2$ were randomly generated from $[-4, 3]^n$ and the values α_i (β_i), $i = 1, 2$ were randomly generated from an interval of length 3 with the left bound greater than the modulus of the minimal value of the corresponding nominator (denominator) over the feasible set, in order to make the objective functions strictly positive over the feasible set.

TABLE 4. The statistical results for 3 groups of 10 random generated instances each, with $n = 10$, $n = 20$, and $n = 50$ respectively.

	$n = 10$			$n = 20$			$n = 50$		
	RG	CCMS	CCES	RG	CCMS	CCES	RG	CCMS	CCES
$\lvert P_{\text{gen}}\rvert \uparrow$	114	125.4	129	124.3	128.9	129	127.8	128.4	129
#it $\downarrow$	396.2	317.1	168.2	433.5	332.5	193.1	502.7	387.5	235.3
Time (s)	0.299	0.199	0.188	0.478	0.245	0.193	2.018	1.482	0.7923
HD $\downarrow$	354.7	346.1	343.9	6029.8	6217.1	6209.4	50417.4	49449.4	49441.1
OS $\uparrow$	0.109	1	1	0.016	1	1	0.0001467	1	1
$OS_1 \uparrow$	0.246	1	1	0.169	1	1	0.0090807	1	1
$OS_2 \uparrow$	0.325	1	1	0.132	1	1	0.0159513	1	1
AC $\uparrow$	0.051	0.281	1.427	0.005	0.124	0.413	0.0012530	0.046	0.132
NDC $\uparrow$	6.449	25.2	35.9	4.069	23	34.1	1.2374603	24.9	30.6
CL $\downarrow$	30.674	5.405	3.642	40.946	5.796	3.852	112.44508	5.322	4.261

Table 4 reports the averages of the values of the metrics that describe the approximations obtained by running RG, CCMS, and CCES on random generated instances, for each group separately. As seen in this table, CCES outperforms both RG and CCMS regarding all metrics, except the hyper-area difference for $n = 20$ (where RG is better than both CCMS and CCES). The overall Pareto spread and the Pareto spread with respect to both objectives are the same for CCES and CCMS due to the inclusion of the efficient marginal solutions in both generated non-dominated sets. For the rest of the metrics, all values are significantly better for CCES comparing to CCMS.

7. FINAL REMARKS

In this paper we introduced a new approach – CCES – for constructing a 0th order approximation of the non-dominated set for the MOLFP problems. We combined a special way of generating relevant feasible solutions with a procedure for mapping them into non-dominated points. We showed that the obtained non-dominated points form a good approximation for the Pareto front of the problem, much better than other approximations obtained by applying the approaches suggested in the literature. We used classic metrics for evaluating the quality of the approximations.

In the recent literature we found two different procedures for deriving efficient solutions from given feasible solutions, DVA and PA. In order to decide which of them to use in CCES for mapping a feasible solution to a non-dominated point, we compared them theoretically in the third section of the paper, and found them much the same. We picked the older one, DVA, and used it in our experiments.

In the paper we reported the computational results obtained by solving both simple instances (with two-three decision variables and two-three objective functions) from the literature – to compare our results to known results, – and random generated instances – to give statistical significance to our conjectural conclusions. We analyzed generated random instances with 10, 20 and 50 decision variables and two objective functions. The new proposed procedure CCES outperformed CCMS and RG when these big size instances were solved. For small size instances, CCES and CCMS were much the same regarding some metrics, but CCES was better regarding the other over viewed metrics.

We also compared our results to the results presented in the most recent reference found in the literature [19]. Our algorithm outperformed the algorithm introduced in [19] on a 3-objective optimization problem in terms of number of generated points, running time, and number of linear problems solved.

The well-known approaches found in the literature – that aimed to compute Pareto front approximations of 0th order for multi-objective optimization problems – analyzed the criterion space, and then introduced

some bounding constraints to the original feasible set according to the values of one objective function (see [3, 4, 7, 12, 19]). On the other side, the method proposed in this paper inspected the feasible set and then efficiently approached the Pareto front.

Acknowledgements. This research was partially supported by the Ministry of Education and Science, Republic of Serbia, Project numbers TR36006 and TR32013.

References

[1] R. Caballero and M. Hernández, The controlled estimation method in the multiobjective linear fractional problem. *Comput. Oper. Res.* **31** (2004) 1821–1832.
[2] E. Choo, *Multicriteria linear fractional programming*. Ph.D. thesis, University of British Columbia (1980).
[3] E. Choo and D. Atkins, Bicriteria linear fractional programming. *J. Optim. Theory App.* **36** (1982) 203–220.
[4] J.P. Costa, Computing non-dominated solutions in MOLFP. *Eur. J. Oper. Res.* **181** (2007) 1464–1475.
[5] J.P. Costa and M.J. Alves, A reference point technique to compute non-dominated solutions in MOLFP. *J. Math. Sci.* **161** (2009) 820–830.
[6] M. Ehrgott, A. Löhne and L. Shao, A dual variant of Benson's "outer approximation algorithm" for multiple objective linear programming. *J. Global Optim.* **57** (2012) 757–778.
[7] H. Hamacher, C. Pedersen and S. Ruzika, Finding representative systems for discrete bicriterion optimization problems. *Oper. Res. Lett.* **35** (2007) 336–344.
[8] M. Hartikainen, K. Miettinen and M. Wiecek, Constructing a pareto front approximation for decision making. *Math. Methods Oper. Res.* **73** (2011) 209–234.
[9] G. Kirlik and S. Sayin. A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems. *Eur. J. Oper. Res.* **232** (2014) 479–488.
[10] J.S.H. Kornbluth and R.E. Steuer, Multiple objective linear fractional programming. *Manage. Sci.* **27** (1981) 1024–1039.
[11] F.H. Lotfi, A.A. Noora, G.R. Jahanshahloo, M. Khodabakhshi and A. Payan, A linear programming approach to test efficiency in multi-objective linear fractional programming problems. *App. Math. Model.* **34** (2010) 4179–4183.
[12] V. Pereyra, M. Saunders and J. Castillo, Equispaced pareto front construction for constrained bi-objective optimization. *Math. Comput. Model.* **57** (2013) 2122–2131.
[13] N. Ruan and D. Gao, Global solutions to fractional programming problem with ratio of nonconvex functions. *Appl. Math. Comput.* **255** (2015) 66–72.
[14] S. Ruzika and M.M. Wiecek, Approximation methods in multiobjective programming. *J. Optim. Theory App.* **126** (2005) 473–501.
[15] P. Shen, W. Li, and X. Bai, Maximizing for the sum of ratios of two convex functions over a convex set. *Comput. Oper. Res.* **40** (2013) 2301–2307.
[16] I.M. Stancu-Minasian, Fractional Programming: Theory, Methods and Applications. Kluwer Academic Publishers, Alphen aan den Rijn (1997).
[17] B. Stanojević and M. Stanojević, On the efficiency test in multi-objective linear fractional programming problems by Lotfi et al. 2010. *Appl. Math. Model.* **37** (2013) 7086–7093.
[18] E. Valipour, M.A. Yaghoobi and M. Mashinchi, An iterative approach to solve multiobjective linear fractional programming problems. *Appl. Math. Model.* **38** (2014) 38–49.
[19] E. Valipour, M.A. Yaghoobi and M. Mashinchi, An approximation to the nondominated set of a multiobjective linear fractional programming problem. *Optimization* **65** (2016) 1539–1552.
[20] J. Wu and S. Azarm, Metrics for quality assessment of a multiobjective design optimization solution set. *J. Mech. Des.* **123** (2001) 18–25.