Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 3580419, 14 pages
https://doi.org/10.1155/2020/3580419

Research Article

An Effective Computational Algorithm for the Global Solution of a Class of Linear Fractional Programming

XiaoLi Huang,[1,2] YueLin Gao ,[1,2] Bo Zhang,[3] and Xia Liu[3]

[1] *Ningxia Province Cooperative Innovation Center of Scientific Computing and Intelligent Information Processing, North Minzu University, Yinchuan 750021, China*
[2] *Ningxia Province Key Laboratory of Intelligent Information and Data Processing, North Minzu University, Yinchuan 750021, China*
[3] *School of Mathematics and Statistics, Ningxia University, Yinchuan 750021, China*

Correspondence should be addressed to YueLin Gao; gaoyuelin@263.net

Received 14 August 2020; Revised 21 September 2020; Accepted 22 October 2020; Published 10 November 2020

Academic Editor: S. A. Edalatpanah

For the minimization of the sum of linear fractions on polyhedra, it is likewise a class of linear fractional programming (LFP). In this paper, we mainly propose a new linear relaxation technique and combine the branch-and-bound algorithm framework to solve the LFP globally. It is worthwhile to mention that the branching operation of the algorithm occurs in the relatively small output space of the dimension rather than the space where the decision variable is located. When the number of linear fractions in the objective function is much lower than the dimension of the decision variable, the performance of the algorithm is better. After that, we also explain the effectiveness, feasibility, and other performances of the algorithm through numerical experiments.

1. Introduction

Consider the linear fractional programming (LFP):

$$\begin{aligned}(\mathrm{LFP}) \coloneqq\ & \min f(x) = \sum_{i=1}^{p} \frac{c_i^T x + d_i}{e_i^T x + f_i} \\ & \text{s.t. } x \in X = \{x \in \mathbb{R}^n \mid Ax \le b, x \ge 0\},\end{aligned} \tag{1}$$

where $c_i \in \mathbb{R}^n$, $d_i \in \mathbb{R}$, $e_i \in \mathbb{R}^n$, and $f_i \in \mathbb{R}$. X is a nonempty polyhedron-feasible set, with $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$, in a decision (or variable) space $\mathbb{R}^n$. Note that throughout the paper, T represents the transpose of vectors, such as the previous c_i^T representing the transpose of vector c_i. Furthermore, for each $i = 1, 2, \ldots, p$, we let $g_i(x) = c_i^T x + d_i$ and $h_i(x) = e_i^T x + f_i$, which are in favour of the following narrative. For the single linear fraction $g_i(x)/h_i(x)$ in the objective function, we assume that $g_i(x) \ge 0$ and $h_i(x) > 0$; at the same time, by [1], our hypothesis has not lost generality, that is, as long as the denominator $h_i(x)$ of $g_i(x)/h_i(x)$ is not 0, we can convert it into $g_i(x) \ge 0$ and $h_i(x) > 0$ by means of the method in [1].

The problem (LFP) has many important applications in laminated manufacturing [2, 3], material layout [4], MIMO networks [5], and economics [6], among others. From [7–9], it can be seen that the computational efficiency of the algorithm is very sensitive to the number p of linear fractions in the problem (LFP). When $p = 1$, Charnes and Cooper [10] indirectly solved the linear fractional programming by transforming the original linear fractional programming into an equivalent linear programming problem. Charnes and Novaes [11] proposed an updated objective function method to solve linear fractional programs by resolving a series of linear programs, while Dinkelbach [12] used a parametric approach to solving linear fractional programming problems. Through the transformation of the objective function and constraint, Das and Mandal [13] simplified the linear fractional programming into an equivalent linear program and then used the simplex method to solve this linear program. Indeed, when $p = 2$, Matsui [14] proved that the problem (LFP) is an NP-hard problem. Therefore, its solution method has caught the attention of numerous scholars. So far, many methods have been proposed and used to solve the LFP and its special forms, such as the image space method [15], outer approximation method [16], unifying monotone method [17], cutting plane method [18], branch-and-bound algorithm [7, 8, 19, 20], interval-split algorithm [21], internal point method [22], heuristic method [23], and concave minimization method [24]. Besides, based on [25], by introducing strategies such as bound-lift and cone-compression, Shen et al. [26] proposed a new branch-reduction-bound algorithm. According to Jiao and Liu [9], the original problem is transformed into a bilinear programming problem, and then the lower bound of the original problem is constructed by using the technique of linearization of convex envelope and concave envelope of the bivariate product function and then solved by the branch-and-bound algorithm. By utilising the linearization technique, Jiao et al. [27] constructed the linear relaxation lower bound of the original problem and then proposed a new algorithm by pruning strategy and gave the convergence of the algorithm. When p is a fixed number, an approximate solution algorithm is given in [1, 28, 29], and its complexity analysis shows that it is a complete polynomial-time approximation algorithm. Xia et al. [30] extended the conclusion in [28] to the problem of linear-ratio-sum with the linear matrix inequality. Moreover, relevant scholars have also studied linear fractional programming problems in uncertain environments, such as fuzzy linear fractional programming [31, 32], linear fractional programming with absolute value variables [33], and interval linear fractional programming [34]. Das et al. [31] proposed the concept of a simple sorting method between the fuzzy numbers of two triangles and gave an equivalent three-objective linear fractional programming problem to calculate the upper, middle, and lower bounds of the fuzzy linear fraction programming problem, thus numerically constructing and solving the optimal value. In [32], the authors used the Charnes–Cooper scheme and the multiobjective linear programming problem to obtain an effective algorithm for solving the fully fuzzy linear fractional programming problem, and in [33], they also proposed a new model of linear fractional programming problems with absolute value functions and then transformed the linear fractional programming problems into independent linear programming problems with some theorems. Then, popular algorithms (such as the simplex algorithm) are utilized to solve these problems. Recently, in [34], Abad et al. proposed two new approaches to interval linear fractional programming, and in each method, two submodels were used to obtain the range of the objective function.

In this paper, a branch-and-bound algorithm based on the branch of the p-dimensional output space is presented for solving the LFP. The literature studies [7–9] all report that the branching operation of the branch-and-bound algorithm occurring in the p-dimensional space may save more computation than in the $2p$-dimensional or n-dimensional space, especially in [8], which, by Theorem 5 and its corollary, shows that the branching operation occurring in the p-dimensional space, in the case of $p \ll n$, is the most advantageous. Therefore, the algorithm set forth in the present paper can greatly save the computational cost compared with the branching operation for the $2p$- or n-dimensional space. In addition, another advantage of our algorithm is the fact that we only need to solve linear programming in the iterative process of the algorithm, which is easier to solve than the general nonlinear programming algorithm.

This paper is organized as follows. In Section 2, we give the equivalence problem (EP) of the problem (LFP) as well as some preparatory work. Section 3 mainly discusses the relevant theories of the proposed algorithm based on the equivalence problem (EP), including the bounding, branching, pruning, detailed steps, and the convergence analysis of the proposed algorithm. In Section 4, numerical experiments are performed, which are utilized to illustrate the effectiveness and feasibility of the algorithm and other performances. Finally, the conclusion part mainly makes a simple summary and prospect of the algorithm in this paper.

2. Equivalent Problem of Problem (LFP)

To get the equivalent problem of the LFP, we first solve the following $4p$-linear programming problems using the existing linear programming method:

$$\begin{aligned}
\underline{y}_i^0 &= \min_{x\in X} g_i(x) = \min_{x\in X} c_i^T x + d_i, \quad i = 1, 2, \ldots, p,\\
\overline{y}_i^0 &= \max_{x\in X} g_i(x) = \max_{x\in X} c_i^T x + d_i, \quad i = 1, 2, \ldots, p,\\
\underline{z}_i^0 &= \min_{x\in X} h_i(x) = \min_{x\in X} e_i^T x + f_i, \quad i = 1, 2, \ldots, p,\\
\overline{z}_i^0 &= \max_{x\in X} h_i(x) = \max_{x\in X} e_i^T x + f_i, \quad i = 1, 2, \ldots, p.
\end{aligned} \tag{2}$$

Obviously, the optimal values of the above $4p$-linear programming problems determine the initial upper and lower bounds of the numerator and denominator in each linear fraction function, that is, $\underline{y}_i^0$, $\overline{y}_i^0$, $\underline{z}_i^0$, and $\overline{y}_i^0$, respectively.

Next, we define the initial rectangle

$$R := \left\{(y,z) \in \mathbb{R}^{2p}\colon \underline{y}_i^0 \le y_i \le \overline{y}_i^0, \underline{z}_i^0 \le z_i \le \overline{z}_i^0, \quad i = 1, 2, \ldots, p\right\} \tag{3}$$

and the subrectangle

$$R^k := \left\{(y,z) \in \mathbb{R}^{2p}\colon \underline{y}_i^0 \le y_i \le \overline{y}_i^0, \underline{z}_i^k \le z_i \le \overline{z}_i^k, \quad i = 1, 2, \ldots, p\right\} \subseteq R \tag{4}$$

at the current iteration $k (k \ge 1)$.

Consider the following equivalent problem (EP):

$$\text{(EP)} := \min \varphi(y,z) := \sum_{i=1}^{p} \frac{y_i}{z_i},$$
$$\text{s.t.} \begin{cases} g_i(x) - y_i = 0, & i = 1, 2, \ldots, p\\ h_i(x) - z_i = 0, & i = 1, 2, \ldots, p\\ x \in X, & (y,z) \in R. \end{cases} \tag{5}$$

The following Theorem 1 creates the equivalent relationship between the problem (EP) and the original problem (LFP).

Theorem 1. *If (x^*, y^*, z^*) is a global optimal solution for problem (EP), then x^* is a global optimal solution for problem (LFP). Conversely, if x^* is an optimal solution for problem (LFP), then (x^*, y^*, z^*) is a global optimal solution for problem (EP), where $y^* := g(x^*)$ and $z^* := h(x^*)$.*

Proof. The conclusion of the theorem is obvious and omitted here.

Theorem 1 shows that the solution of problem (LFP) can be obtained indirectly by addressing problem (EP).

In addition, if the rectangle R in problem (EP) is replaced by its subrectangle R^k, this will generate subproblem (EP_{R^k}) of problem (EP) over the rectangle R^k. □

3. Branch-and-Bound Algorithm

In this section, we will consider the three components of the branch-and-bound algorithm, namely, branching, bounding, and pruning, respectively, by studying problem (EP).

3.1. Bounding. Note that the objective function of the problem (EP_{R^k})(or (EP)) is separable and still nonconvex. And then, we can propose a new underestimation method for $(y_i/z_i)\,(i = 1, 2, \ldots, p)$ and give it through the following Theorem 2.

Theorem 2. *Consider the rectangle $LR := \{(y, z) \in \mathbb{R} \times \mathbb{R}\colon\ \underline{y} \le y \le \overline{y}, \underline{z} \le z \le \overline{z}\}$, where $\underline{y}$, $\overline{y}$, $\underline{z}$, and $\overline{z}$ are all constants satisfying $0 \le \underline{y} \le y \le \overline{y}$ and $0 < \underline{z} \le z \le \overline{z}$. For any $(y, z) \in LR$, define the functions $\psi(y, z)$ and $\underline{\psi}(y, z)$ as follows:*

$$\begin{aligned}
\psi(y,z) &:= \frac{y}{z},\\
\phi^1(y,z) &:= -\frac{\underline{y}\,z}{\underline{z}\,\overline{z}} + \frac{y}{\overline{z}} + \frac{\underline{y}\left(2\overline{z} - \sqrt{\underline{z}\,\overline{z}}\right)}{\overline{z}\sqrt{\underline{z}\,\overline{z}}},\\
\phi^2(y,z) &:= -\frac{\overline{y}z}{\underline{z}\,\overline{z}} + \frac{\left(2\overline{z} - \sqrt{\underline{z}\,\overline{z}}\right)y}{\overline{z}\sqrt{\underline{z}\,\overline{z}}} + \frac{\overline{y}}{\overline{z}},\\
\underline{\psi}(y,z) &:= \max\left\{\phi^1(y,z), \phi^2(y,z)\right\}.
\end{aligned} \tag{6}$$

Then, the following conclusions hold:

(i) For any $(y, z) \in LR$, the functions $\psi(y, z)$ and $\underline{\psi}(y, z)$ satisfy

$$\psi(y,z) \ge \underline{\psi}(y,z). \tag{7}$$

(ii) Let $\nabla = ((2\overline{y}\sqrt{\overline{z}})/(\overline{z}\,\underline{z}(\sqrt{\overline{z}} + \sqrt{\underline{z}})))$ and $\Delta = \overline{z} - \underline{z}$; then, $\lim_{\Delta \longrightarrow 0}\psi(y,z) - \underline{\psi}(y,z) = 0$.

Proof. (i) For any $(y, z) \in \mathrm{LR}$, we have

$$\begin{aligned}
&\psi(y,z) - \underline{\psi}(y,z)\\
&= \psi(y,z) - \max\left\{-\frac{\underline{y}\,z}{\underline{z}\,\overline{z}} + \frac{y}{\overline{z}} + \frac{\underline{y}\left(2\overline{z} - \sqrt{\underline{z}\,\overline{z}}\right)}{\overline{z}\sqrt{\underline{z}\,\overline{z}}}, -\frac{\overline{y}z}{\underline{z}\,\overline{z}} + \frac{\left(2\overline{z} - \sqrt{\underline{z}\,\overline{z}}\right)y}{\overline{z}\sqrt{\underline{z}\,\overline{z}}} + \frac{\overline{y}}{\overline{z}}\right\},\\
&= \frac{y}{z} - \max\left\{\left[\frac{y}{\overline{z}} + \underline{y}\left(\frac{2}{\sqrt{\underline{z}\,\overline{z}}} - \frac{z}{\overline{z}\,\underline{z}}\right) - \frac{\underline{y}}{\overline{z}}\right], \left[-\frac{1}{\underline{z}\,\overline{z}}\left(\overline{y}z + y\,\underline{z} - \overline{y}\,\underline{z}\right) + \frac{2y}{\sqrt{\underline{z}\,\overline{z}}}\right]\right\},\\
&\ge \min\left\{\frac{y}{z} - \left(\frac{y}{\overline{z}} + \frac{\underline{y}}{z} - \frac{\underline{y}}{\overline{z}}\right), \frac{y}{z} - \left(-\frac{yz}{\underline{z}\,\overline{z}} + \frac{2y}{\sqrt{\underline{z}\,\overline{z}}}\right)\right\},\\
&= \min\left\{\left(y - \underline{y}\right)\left(\frac{1}{z} - \frac{1}{\overline{z}}\right), y\left[\frac{1}{z} - \left(\frac{2}{\sqrt{\underline{z}\,\overline{z}}} - \frac{z}{\underline{z}\,\overline{z}}\right)\right]\right\},\\
&\ge 0.
\end{aligned} \tag{8}$$

The establishment of the first inequality in formula (8) takes advantage of the convexity of the univariate function $1/z$ in the case of $z > 0$, i.e., $(1/z) \ge (2/\sqrt{\underline{z}\,\overline{z}}) - (z/\overline{z}\,\underline{z})$. Moreover, the establishment of the first inequality of formula (8) also takes advantage of the fact that $(y - \overline{y})(z - \underline{z}) \le 0$.

As a result, according to formula (8), it is easy to know $\psi(y,z) \geq \underline{\psi}(y,z)$. Conclusion (i) holds.

(ii) By the definition of $\psi(y,z)$ and $\underline{\psi}(y,z)$, we have

$$\begin{aligned}
\psi(y,z) - \phi^1(y,z) &= \left(\frac{y}{z} - \frac{y}{\overline{z}}\right) + \underline{y}\left(\frac{1}{\overline{z}} - \frac{2}{\sqrt{\underline{z}\,\overline{z}}} + \frac{z}{\overline{z}\,\underline{z}}\right),\\
&\leq \frac{\overline{y}(\overline{z} - \underline{z})}{\overline{z}\,\underline{z}} + \underline{y}\left(\frac{\underline{z}}{\underline{z}\,\overline{z}} - \frac{2}{\sqrt{\underline{z}\,\overline{z}}} + \frac{\overline{z}}{\overline{z}\,\underline{z}}\right),\\
&= \frac{\overline{y}\left(\overline{z} - \underline{z}\right)}{\overline{z}\,\underline{z}} + \frac{\underline{y}}{\underline{z}\,\overline{z}}\left(\sqrt{\overline{z}} - \sqrt{\underline{z}}\right)^2, \leq \frac{\overline{y}}{\overline{z}\,\underline{z}}\left(1 + \frac{\sqrt{\overline{z}} - \sqrt{\underline{z}}}{\sqrt{\overline{z}} + \sqrt{\underline{z}}}\right)(\overline{z} - \underline{z}), = \frac{2\overline{y}\sqrt{\overline{z}}(\overline{z} - \underline{z})}{\overline{z}\,\underline{z}\left(\sqrt{\overline{z}} + \sqrt{\underline{z}}\right)}, = \nabla\Delta,
\end{aligned}\tag{9}$$

$$\begin{aligned}
\psi(y,z) - \phi^2(y,z) &= y\left(\frac{1}{z} - \left(-\frac{z}{\underline{z}\,\overline{z}} + \frac{2}{\sqrt{\underline{z}\,\overline{z}}}\right)\right) + \frac{1}{\underline{z}\,\overline{z}}(\overline{y} - y)(z - \underline{z}),\\
&\leq y\left(\frac{1}{\sqrt{z}} - \frac{\sqrt{z}}{\sqrt{\underline{z}\,\overline{z}}}\right)^2 + \frac{1}{\underline{z}\,\overline{z}}\left(\overline{y} - \underline{y}\right)(\overline{z} - \underline{z}),\\
&\leq \frac{\overline{y}}{\underline{z}\,\overline{z}}\left(\sqrt{\overline{z}} - \sqrt{\underline{z}}\right)^2 + \frac{\overline{y}}{\underline{z}\,\overline{z}}(\overline{z} - \underline{z}),\\
&= \frac{\overline{y}}{\overline{z}\,\underline{z}}\left(1 + \frac{\sqrt{\overline{z}} - \sqrt{\underline{z}}}{\sqrt{\overline{z}} + \sqrt{\underline{z}}}\right)(\overline{z} - \underline{z}),\\
&= \nabla\Delta.
\end{aligned}\tag{10}$$

According to formulae (9) and (10) and conclusion (i), we have

$$0 \leq \lim_{\Delta \longrightarrow 0} \psi(y,z) - \underline{\psi}(y,z) \leq \lim_{\Delta \longrightarrow 0} \nabla\Delta. \tag{11}$$

Since ∇ is a bounded value, this implies that $\nabla\Delta \longrightarrow 0$ as $\Delta \longrightarrow 0$. Thus, combined with (11), we have $\lim_{\Delta \longrightarrow 0}\psi(y,z) - \underline{\psi}(y,z) = 0$, and the proof is completed.

In the following, by using Theorem 2, we will construct the lower bounding function of the objective function for problem (EP_{R^k})(or (EP)). Assume that $\widehat{R}$ denotes R or a subrectangle R^k of R that is generated by the branching process, where $\widehat{R} = \prod_{i=1}^{p} \widehat{R}_i$, with $\widehat{R}_i = [\underline{y}_i^0, \overline{y}_i^0] \times [\underline{z}_i, \overline{z}_i]$, $i = 1, 2, \ldots, p$. Obviously, $\underline{y}_i^0$, $\overline{y}_i^0$, $\underline{z}_i$, and $\overline{z}_i$ satisfy $0 \leq \underline{y}_i^0 \leq y_i \leq \overline{y}_i^0$ and $0 < \underline{z}_i \leq z_i \leq \overline{z}_i$. For each $i \in \{1, 2, \ldots, p\}$ and for each $(y_i, z_i) \in \widehat{R}_i$, define

$$\begin{aligned}
\psi_i(y_i,z_i) &:= \frac{y_i}{z_i},\\
\phi_i^1(y_i,z_i) &:= -\frac{\underline{y}_i^0 z_i}{\underline{z}_i\overline{z}_i} + \frac{y_i}{\overline{z}_i} + \frac{\underline{y}_i^0\left(2\overline{z}_i - \sqrt{\underline{z}_i\overline{z}_i}\right)}{\overline{z}_i\sqrt{\underline{z}_i\overline{z}_i}},\\
\phi_i^2(y_i,z_i) &:= -\frac{\overline{y}_i^0 z_i}{\underline{z}_i\overline{z}_i} + \frac{\left(2\overline{z}_i - \sqrt{\underline{z}_i\overline{z}_i}\right)y_i}{\overline{z}_i\sqrt{\underline{z}_i\overline{z}_i}} + \frac{\overline{y}_i^0}{\overline{z}_i},\\
\underline{\psi}_i(y_i,z_i) &:= \max\left\{\phi_i^1(y_i,z_i), \phi_i^2(y_i,z_i)\right\}.
\end{aligned}\tag{12}$$

Then, by Theorem 2, we have

$$\psi_i(y_i,z_i) \geq \underline{\psi}_i(y_i,z_i), \quad i \in \{1, 2, \ldots, p\}. \tag{13}$$

Thus, for any $(y,z) \in \widehat{R}$, let $\underline{\psi}(y,z) = \sum_{i=1}^{p} \underline{\psi}_i(y_i,z_i)$, and we have

$$\varphi(y,z) \geq \underline{\psi}(y,z), \quad \forall (y,z) \in \widehat{R}. \tag{14}$$

Furthermore, by using formulae (13) and (14), we can construct the following linear relaxation programming problem $(\mathrm{LRP}_{\widehat{R}})$ of $(\mathrm{EP}_{\widehat{R}})$:

$$\begin{aligned}
&\left(\mathrm{LRP}_{\widehat{R}}\right)\colon\ \underline{\psi}(y,z) = \min \sum_{i=1}^{p} r_i,\\
&\text{s.t.} \begin{cases}
\phi_i^1(y_i,z_i) \leq r_i, & i = 1,2,\ldots,p,\\
\phi_i^2(y_i,z_i) \leq r_i, & i = 1,2,\ldots,p,\\
g_i(x) - y_i = 0, & i = 1,2,\ldots,p,\\
h_i(x) - z_i = 0, & i = 1,2,\ldots,p,\\
x \in X, & (y,z) \in \widehat{R}.
\end{cases}
\end{aligned} \tag{15}$$

Based on the construction method of the linear relaxation programming problem $(\mathrm{LRP}_{\widehat{R}})$, obviously, the optimal value for problem $(\mathrm{NLRP}_{\widehat{R}})$ is less than or equal to that of $(\mathrm{EP}_{\widehat{R}})$, i.e., $v(\mathrm{NLRP}_{\widehat{R}}) \leq v(\mathrm{EP}_{\widehat{R}})$. Then, we have $v(\mathrm{LRP}_{\widehat{R}}) = v(\mathrm{NLRP}_{\widehat{R}}) \leq v(\mathrm{EP}_{\widehat{R}})$. Therefore, the linear relaxation programming problem $(\mathrm{LRP}_{\widehat{R}})$ provides a valid lower bound for the optimal value of $(\mathrm{EP}_{\widehat{R}})$.

It is noted that the solution $(\widehat{x},\widehat{y},\widehat{z})$ obtained from the above linear relaxation programming problem $(\mathrm{LRP}_{\widehat{R}})$, which is used to progressively update the upper bound of the optimal value of problem (EP), is a feasible solution to problem (EP). □

3.2. Branching. For the selected rectangle $\widehat{R}$, we use the dichotomization as a branching process, which subdivides a $2p$-dimensional rectangle

$$\widehat{R} = \prod_{i=1}^{p} \widehat{R}_i = \prod_{i=1}^{p} [\underline{y}_i^0, \overline{y}_i^0] \times [\underline{z}_i, \overline{z}_i] \subseteq R \tag{16}$$

into two $2p$-dimensional subrectangles $\widehat{R}^1$ and $\widehat{R}^2$ of the same volume along the midpoint of its longest edge. The specific rectangle-branch method is as follows:

(i) Let $j \in \arg\max\{\overline{z}_i - \underline{z}_i, i = 1,2,\ldots,p\}$.

(ii) $\widehat{R}_j$ is divided into $\widehat{R}_j^1$ and $\widehat{R}_j^2$, i.e.,

$$\widehat{R}_j^1 = \left[\underline{y}_j^0, \overline{y}_j^0\right] \times \left[\underline{z}_j, \frac{\overline{z}_j + \underline{z}_j}{2}\right], \widehat{R}_j^2 = \left[\underline{y}_j^0, \overline{y}_j^0\right] \times \left[\frac{\overline{z}_j + \underline{z}_j}{2}, \overline{z}_j\right]. \tag{17}$$

(iii) Let $\widehat{R}^1 = \prod_{i=1}^{j-1} \widehat{R}_i \times \widehat{R}_j^1 \times \prod_{i=j+1}^{p} \widehat{R}_i$ and $\widehat{R}^2 = \prod_{i=1}^{j-1} \widehat{R}_i \times \widehat{R}_j^2 \times \prod_{i=j+1}^{p} \widehat{R}_i$.

Be sure to note that the interval $[\underline{y}_i^0, \overline{y}_i^0]\,(i = 1,2,\ldots,p)$ is never subdivided during the branching process. Thus, the branching process occurs only in the p-dimensional space, which greatly saves this computational cost.

3.3. Pruning. Suppose $L(\widehat{R})$ is the lower bound of problem $(\mathrm{EP}_{\widehat{R}})$ over the rectangle $\widehat{R}$ and U represents the best upper bound that the algorithm has known so far. And let $\varepsilon \in (0,1)$ represent the tolerance of the algorithm. Then, the pruning process is the transversion of deleting the node information on the branch-and-bound tree corresponding to the rectangle $\widehat{R}$ that satisfies $L(\widehat{R}) > U - \varepsilon$.

3.4. Output-Space Branch-and-Bound Algorithm. Now, we present an output-space branch-and-bound algorithm for solving (LFP). For the k-th iteration, we give the following notation in advance: R^k is the rectangle to be subdivided, Q represents a set that produces a new solution after each iteration (note that the number of elements in the set Q does not exceed 2), and Ξ is a collection of rectangles left after pruning. (x^k, y^k, z^k) and $L(R^k)$ represent the optimal solution and optimal value of problem (LRP_{R^k}) over the rectangle R^k, respectively. L^k represents the current lower bound of the global optimal value of problem (EP), and U^k represents the current upper bound of the global optimal value of problem (EP). U' represents the best function value of all new feasible solutions to the resulting problem (EP) after each iteration is completed.

Combined with the above content, the proposed algorithm is as follows:

Step 0 (initialization):

$\underline{y}^0, \overline{y}^0, \underline{z}^0$, and $\overline{z}^0$ are obtained by (2), and then the initial rectangle $R^0 = R$ is constructed. Set tolerance $\varepsilon \in (0,1)$. Then, the initial linear relaxation programming problem (LRP_{R^0}) is solved, and its optimal solution (x^0, y^0, z^0) and optimal value $L(R^0)$ are obtained. And then, let

$$\begin{aligned}
U^1 &= \varphi\left(y^0, z^0\right),\\
L^1 &= L\left(R^0\right),\\
R^1 &= R^0,\\
\Xi &= R^1,\\
Q &= \varnothing,\ k = 1.
\end{aligned} \tag{18}$$

Step 1 (termination):

If $U^k - L^k \leq \varepsilon$, then terminate the algorithm. Then, (x^k, y^k, z^k) is the global ε-optimal solution of problem (EP), and the global optimal solution of problem (LFP) can be obtained according to Theorem 1 which is x^k. Otherwise, go to Step 2.

Step 2 (branching):

Let $R^k = \left\{\widehat{R} \in \Xi\colon L^k = L(\widehat{R})\right\}$ and $\Xi = \Xi \setminus R^k$. Then, using the rectangle-branching method in Section 3.2, R^k is divided into two subrectangles R^{k1} and R^{k2}.

Step 3 (pruning):

For each $s \in \{1,2\}$, solve $(\mathrm{LRP}_{R^{\mathrm{ks}}})$ to obtain $L(R^{\mathrm{ks}})$ and $(x^{\mathrm{ks}}, y^{\mathrm{ks}}, z^{\mathrm{ks}})$ for $R^{\mathrm{ks}} \subseteq R^k$. If $L(R^{\mathrm{ks}}) \geq U^k - \varepsilon$, then delete the rectangle R^{ks}. Otherwise, let $\Xi = \Xi \cup R^{\mathrm{ks}}$ and $Q = Q \cup (x^{\mathrm{ks}}, y^{\mathrm{ks}}, z^{\mathrm{ks}})$.

Step 4 (bounding):

Step 4.1 (upper bounding):
If Q is not empty, let $U' = \min_{(x,y,z)\in Q}\varphi(y,z)$. If $U' < U^k$, set $U^k = U'$ and $(x^k, y^k, z^k) = \operatorname{argmin}_{(x,y,z)\in Q}\varphi(y,z)$.
If Q is empty, both U^k and (x^k, y^k, z^k) remain the same.
Step 4.2 (lower bounding):

$$L^k = \begin{cases} U^k, & \text{if } \Xi = \varnothing, \\ \min\left\{L(\widehat{R}) : \widehat{R} \in \Xi\right\}, & \text{else.} \end{cases} \tag{19}$$

Set $k = k + 1$ and $Q = \varnothing$, and go to Step 1.

3.5. Convergence of the Algorithm. The global convergence of the algorithm is reproduced as follows.

Theorem 3. *The above algorithm either terminates in a finite iteration and is accompanied by the generation of optimal values of problem (LFP) or produces an infinite sequence $\{x^k\}$ composed of feasible solutions and makes any convergence point of $\{x^k\}$ global optimal solutions for (LFP).*

Proof. If this algorithm is terminated at iteration $k\,(k \geq 1)$, then according to Step 1 of the algorithm, we have

$$U^k - L^k \leq \varepsilon. \tag{20}$$

So, if (x^*, y^*, z^*) is the global optimal solution of problem (EP), there is

$$U^k = \varphi\left(y^k, z^k\right) \geq \varphi\left(y^*, z^*\right) \geq L^k. \tag{21}$$

By integrating (20) and (21),

$$f\left(x^*\right) = \varphi\left(y^*, z^*\right) \leq f\left(x^k\right) = \varphi\left(y^k, z^k\right) \leq \varphi\left(y^*, z^*\right) + \varepsilon = f\left(x^*\right) + \varepsilon \tag{22}$$

can be obtained. Thus, when the algorithm terminates in the $k -$ th iteration, the corresponding solution $f(x^k)$ is a global optimal value of problem (LFP).

The algorithm produces an infinite sequence $\{x^k, y^k, z^k\}$ by solving a series of linear programs (LRP_{R^k}). And each point of this sequence is a feasible point for problem (EP), respectively. From the algorithm, there will be an infinite sequence $\{R^k\}_{k=1}^{\infty}$ of rectangles corresponding to this series of linear programs. By the branching process of Step 2, we have

$$\lim_{k\longrightarrow\infty} R^k = \prod_{i=1}^{p}\left[\underline{y}_i^0, \overline{y}_i^0\right] \times \lim_{k\longrightarrow\infty}\prod_{i=1}^{p}\left[\underline{z}_i^k, \overline{z}_i^k\right] = \prod_{i=1}^{p}\left[\underline{y}_i^0, \overline{y}_i^0\right] \times \prod_{i=1}^{p} z_i^*. \tag{23}$$

Therefore, we have

$$\lim_{k\longrightarrow\infty} \underline{z}_i^k = z_i^* = \lim_{k\longrightarrow\infty} \overline{z}_i^k, \quad i = 1, 2, \ldots, p. \tag{24}$$

Also, since $z_i^k = h_i(x^k)$, $i = 1, 2, \ldots, p$, we have

$$\underline{z}_i^k \leq z_i^k = h_i\left(x^k\right) \leq \overline{z}_i^k, \quad i = 1, 2, \ldots, p. \tag{25}$$

By (24) and (25), we have

$$h_i\left(x^*\right) = \lim_{k\longrightarrow\infty} h_i\left(x^k\right) = \lim_{k\longrightarrow\infty} z_i^k = z_i^*. \tag{26}$$

Thus, (x^*, y^*, z^*) is also a viable solution to problem (EP). Moreover, according to the properties of the branch-and-bound algorithm, the sequence $\{L^k\}$ is an increasing sequence bounded by $\varphi(y^*, z^*)$, and then we have

$$\varphi\left(y^*, z^*\right) \geq \lim_{k\longrightarrow\infty} L^k. \tag{27}$$

Through the update process of the lower bound in Step 4 of the algorithm, we have

$$\lim_{k\longrightarrow\infty} L^k = \lim_{k\longrightarrow\infty} \underline{\psi}\left(y^k, z^k\right). \tag{28}$$

Through Theorem 2 and the continuity of function $\psi(y, z)$, we have

$$\lim_{k\longrightarrow\infty} \underline{\psi}\left(y^k, z^k\right) = \lim_{k\longrightarrow\infty} \sum_{i=1}^{p} \underline{\psi}_i\left(y^k, z^k\right) = \varphi\left(y^*, z^*\right). \tag{29}$$

By using the previous formulae (27), (28), and (29), we have

$$\lim_{k\longrightarrow\infty} L^k = \varphi\left(y^*, z^*\right). \tag{30}$$

Ultimately, (x^*, y^*, z^*) is a global optimal solution to problem (EP), and then using equivalence Theorem 2, we can know that x^* is a global optimal solution to problem (LFP) immediately. The proof is completed. □

4. Numerical Experiments

In this section, several test problems are given to illustrate the performance of the algorithm. All of our testing procedures were performed via MATLAB (2012a) on computers with Intel(R) Core(TM)i5-2320 3.00 GHz power processor 4.00 GB memory and Microsoft Win7 operating system.

Problem 1 (see [1])

$$\min \frac{-3.333x_1 - 3.000x_2 - 1.000}{1.666x_1 + x_2 + 1.000} + \frac{-4.000x_1 - 3.000x_2 - 1.000}{x_1 + x_2 + 1.000}$$
$$\text{s.t.}\begin{cases} 5.000x_1 + 4.000x_2 \leq 10.000, -x_2 \leq -0.100 \\ -2.000x_1 - x_2 \leq -2.000, -2.000x_1 - x_2 \leq -2.000 \\ x_1, x_2 \geq 0.000. \end{cases} \tag{31}$$

For test Problem 1, we use the proposed algorithm to address the problem in detail, which can explain the feasibility of the algorithm. Then, the calculation results of the three test problems in other literature studies are also calculated and indicated in Table 1 together with the results of Problem 1. First, for convenience, we denote the feasible domain of Problem 1 by

Table 1: Comparison of results in Problems 1– 5.

Problem	Reference	Solution	Optimum	Iter	Time	ε
1	[1]	(0.1000, 2.3750)	−4.84151	2	0	10^{-2}
	Ours	(0.1000, 2.3750)	−4.84151	8	0.3187	10^{-2}
2	[8]	(1.5000, 1.5000)	4.9125	460	8.7944	10^{-4}
	[21]	(1.5, 1.5)	4.9125874	56	1.0870	10^{-3}
	Ours	(1.5, 1.5)	4.9125874	73	1.8091	10^{-4}
3	[8]	(0.0000, 1.66666667, 0.0000)	3.7109	169	4.2429	10^{-6}
	[21]	(0.0000, 1.66666667, 0.0000)	3.710919	8	0.1830	10^{-3}
	Ours	(0, 1.6667, 0)	3.7109	132	3.5214	10^{-6}
4	[8]	(0, 0.333333, 0)	−3.0029	2090	50.8226	10^{-6}
	[21]	(0, 0.333333, 0)	−3.0029	17	0.1290	10^{-3}
	Ours	(0, 0.333333, 0)	−3.0029	38	1.0237	10^{-6}
5	[9]	(6.24409, 20.0249, 3.79672, 5.93972, 0, 7.43852, 0, 23.2833, 0.515015, 40.9896, 0, 3.14363)	16.2619	927	—	10^{-3}
	[17]	(6.223689, 20.060317, 3.774684, 5.947841, 0, 7.456686, 0, 23.312579, 0.000204, 41.031824, 0, 3.171106)	16.077978	620	65.58	10^{-2}
	Ours	(6.22369, 20.0603, 3.7746, 5.94784,0, 7.45668, 0,23.31257, 0, 41.03182, 0, 3.17111)	16.077978	415	7.8815	10^{-3}

$$X := \begin{cases} 5.000x_1 + 4.000x_2 \le 10.000, -x_2 \le -0.100, \\ -2.000x_1 - x_2 \le -2.000, -2.000x_1 - x_2 \le -2.000, \\ x_1, x_2 \ge 0.000. \end{cases} \tag{32}$$

We also let

$$\begin{aligned} g_1(x) &= -3.333x_1 - 3.000x_2 - 1.000, \\ g_2(x) &= -4.000x_1 - 3.000x_2 - 1.000, \\ h_1(x) &= 1.666x_1 + x_2 + 1.000, \\ h_2(x) &= x_1 + x_2 + 1.000. \end{aligned} \tag{33}$$

It is easy to see that the molecules of Problem 1 are negative (i.e., $g_1(x) < 0$ and $g_2(x) < 0, \forall x \in X$), and we need to convert these two molecules into nonnegative functions. Next, we solve the following four linear programming problems:

$$\underline{y}_1^0 = \min_{x\in X} g_1(x), \underline{y}_2^0 = \min_{x\in X} g_2(x), \underline{z}_1^0 = \min_{x\in X} h_1(x), \underline{z}_2^0 = \min_{x\in X} h_2(x). \tag{34}$$

Among them, $\underline{y}_1^0 = -8.4583$, $\underline{y}_2^0 = -8.9800$, $\underline{z}_1^0 = 2.6827$, and $\underline{z}_2^0 = 2.0500$ are obtained by solving the above four linear programming problems. Apparently, $\underline{y}_1^0 \le g_1(x) < 0$, $\underline{y}_2^0 \le g_2(x) < 0$, $h_1 \;(x) \ge \underline{z}_1^0 > 0$, and $h_2(x) \ge \underline{z}_2^0 > 0$, and we let

$$M_1 = \frac{-\underline{y}_1^0}{\underline{z}_1^0} = 3.1529 > 0, M_2 = \frac{-\underline{y}_2^0}{\underline{z}_2^0} = 4.3804 > 0. \tag{35}$$

Well, for any $x \in X$, we have

$$\begin{aligned} \overline{g}_i(x) &:= g_i(x) + M_i h_i(x) = g_i(x) + \frac{-\underline{y}_i^0}{\underline{z}_i^0} h_i(x) = \frac{\underline{z}_i^0 g_i(x) - \underline{y}_i^0 h_i(x)}{\underline{z}_i^0} = \underline{z}_i^0 g_i(x) - \underline{z}_i^0 \underline{y}_i^0 + \underline{z}_i^0 \frac{\underline{y}_i^0 - \underline{y}_i^0 h_i(x)}{\underline{z}_i^0} \\ &= \frac{\underline{z}_i^0\left(g_i(x) - \underline{y}_i^0\right) + \underline{y}_i^0\left(\underline{z}_i^0 - h_i(x)\right)}{\underline{z}_i^0} \ge 0, \quad i = 1, 2. \end{aligned} \tag{36}$$

According to the above data, we can get

$$\begin{aligned} \overline{g}_1(x) &= 1.9197x_1 + 0.1529x_2 + 2.1529, \\ \overline{g}_2(x) &= 0.3804x_1 + 1.3804x_2 + 3.3804. \end{aligned} \tag{37}$$

As a result, Problem 1 can be reconstructed as follows:

$$\begin{aligned} &\min \frac{\overline{g}_1(x)}{h_1(x)} + \frac{\overline{g}_2(x)}{h_2(x)} - (M_1 + M_2) \\ &\text{s.t. } x \in X. \end{aligned} \tag{38}$$

Therefore, the optimal solution of Problem 1 is the same as that of the following problem (P):

$$\begin{aligned}&(\mathrm{P})\text{:}\ \min \frac{\overline{g}_1(x)}{h_1(x)}+\frac{\overline{g}_2(x)}{h_2(x)},\\ &\text{s.t. } x\in X.\end{aligned}\tag{39}$$

Now, we convert problem (P) into the following initial equivalence problem (EP1):

$$\begin{aligned}&(\mathrm{EP1})\text{:}\ \min \frac{y_1}{z_1}+\frac{y_2}{z_2},\\ &\text{s.t.}\begin{cases}\overline{g}_i(x)-y_i=0, & i=1,2,\\ \overline{h}_i(x)-z_i=0, & i=1,2,\\ x\in X, & (y,z)\in R^1,\end{cases}\end{aligned}\tag{40}$$

where

$$R^1=\begin{bmatrix}2.6201 & 5.8541\\ 3.8800 & 6.6971\\ 2.6827 & 4.2987\\ 2.0500 & 3.4750\end{bmatrix}, \text{ 1 of flowchart 1 represents rectangle } R^1.\tag{41}$$

By solving the linear relaxation programming problem (LPR_{R^1}) constructed over the rectangle R^1, we can obtain the initial lower bound 2.5843, the initial upper bound 2.6918 of problem (EP1), and the corresponding optimal solution $(x^1, y^1, z^1) = [0.1000; 2.2357; 2.7080; 6.6972; 3.5416; 3.4750]$; then, we select the rectangle R^1 corresponding to the initial lower bound and then generate the subrectangle

$$\begin{aligned}R^{11}&=\begin{bmatrix}2.6201 & 5.8541\\ 3.8800 & 6.6971\\ 2.6827 & 3.4907\\ 2.0500 & 3.4750\end{bmatrix},\\ R^{12}&=\begin{bmatrix}2.6201 & 5.8541\\ 3.8800 & 6.6971\\ 3.4907 & 4.2987\\ 2.0500 & 3.4750\end{bmatrix}\end{aligned}\tag{42}$$

by the branching operation. 2 of flowchart 1 represents rectangle R^{11}; 3 of flowchart 1 represents rectangle R^{12}; through solving linear programming problem ($\mathrm{LPR}_{R^{11}}$), the corresponding optimal solution and the optimal value are $(x^{11}, y^{11}, z^{11}) = [0.1000; 2.3241; 2.7003; 6.6269; 3.4907; 3.4241]$ and 2.5916, respectively, and the objective function value of problem (EP1) is $2.7086 > 2.6918$, so the current optimal solution and optimal value of problem (EP1) are not updated. Then, we continue to solve linear programming problem ($\mathrm{LPR}_{R^{12}}$) and obtain the corresponding optimal solution and optimal value which are $(x^{12}, y^{12}, z^{12}) = [0.1000; 2.3257; 2.7080; 6.6972; 3.5416; 3.4750]$ and 2.5844, respectively, and the objective function value of problem (EP1) is 2.6919; then, the current optimal solution and optimal value of the problem will not be updated. As a result, the lower bound of the current problem (EP1) is updated to 2.5844. Next, we select the rectangle R^{12} corresponding to the contemporary lower boundary to divide and then obtain two subrectangles:

$$\begin{aligned}R^{21}&=\begin{bmatrix}2.6201 & 5.8541\\ 3.8800 & 6.6971\\ 3.4907 & 4.2987\\ 2.0500 & 2.7625\end{bmatrix},\\ R^{22}&=\begin{bmatrix}2.6201 & 5.8541\\ 3.8800 & 6.6971\\ 3.4907 & 4.2987\\ 2.7625 & 3.4750\end{bmatrix}.\end{aligned}\tag{43}$$

4 of flowchart 1 represents rectangle R^{21}; 5 of flowchart 1 represents rectangle R^{22}; through solving linear programming problem ($\mathrm{LPR}_{R^{21}}$), the corresponding optimal value and the optimal solution are $2.8388 > 2.6918$ and $(x^{21}, y^{21}, z^{21}) = [1.6625; 0.1000; 5.3598; 4.1511; 3.8697; 2.7625]$, respectively. So, this node is deleted. Then, we continue to solve linear programming problem ($\mathrm{LPR}_{R^{22}}$) and obtain the corresponding optimal solution and optimal value which are $(x^{22}, y^{22}, z^{22}) = [0.1000; 2.3750; 2.7080; 6.6972; 3.5416; 3.4750]$ and 2.6655, respectively, and the objective function value of problem (EP1) is 2.6918; then, the current optimal solution and optimal value of the problem will not be updated. As a consequence, the lower bound of the current problem (EP1) is updated to 2.5916. We continue to select the rectangle R^{11} corresponding to the current lower bound and divide it to obtain two subrectangles:

$$\begin{aligned}R^{31}&=\begin{bmatrix}2.6201 & 5.8541\\ 3.8800 & 6.6971\\ 2.6827 & 3.4907\\ 2.0500 & 2.7625\end{bmatrix},\\ R^{32}&=\begin{bmatrix}2.6201 & 5.8541\\ 3.8800 & 6.6971\\ 2.6827 & 3.4907\\ 2.7625 & 3.4750\end{bmatrix}.\end{aligned}\tag{44}$$

6 of flowchart 1 represents rectangle R^{31}; 7 of flowchart 1 represents rectangle R^{32}; through solving linear programming problem ($\mathrm{LPR}_{R^{31}}$), the corresponding optimal value and the optimal solution are $2.9048 > 2.6918$ and $(x^{31}, y^{31}, z^{31}) = [1.0934; 0.6691; 4.3543; 4.7202; 3.4907; 2.76250]$, respectively. So, this node is deleted. Then, we continue to solve linear programming problem ($\mathrm{LPR}_{R^{32}}$) and obtain the corresponding optimal solution and optimal value which are $(x^{32}, y^{32}, z^{32}) = [0.1000; 2.3241; 2.7003; 6.6269; 3.4907; 3.4241]$ and 2.6697, respectively, and the objective function value of problem (EP1) is $2.7089 > 2.6918$; then, the current optimal solution and optimal value of the problem will not be updated. As a result, the lower bound of the current problem (EP1) is updated to 2.6655.

Continue to select and section the rectangle R^{22} corresponding to the current lower bound, resulting in two subrectangles:

$$R^{41} = \begin{bmatrix} 2.6201 & 5.8541 \\ 3.8800 & 6.6971 \\ 3.4907 & 3.8947 \\ 2.7625 & 3.4750 \end{bmatrix}, \quad R^{42} = \begin{bmatrix} 2.6201 & 5.8541 \\ 3.8800 & 6.6971 \\ 3.8947 & 4.2987 \\ 2.7625 & 3.4750 \end{bmatrix}. \tag{45}$$

8 of flowchart 1 represents rectangle R^{41}; 9 of flowchart 1 represents rectangle R^{42}; through solving linear programming problem $(\mathrm{LPR}_{R^{41}})$, the corresponding optimal value and the optimal solution are 2.6719 and $(x^{41}, y^{41}, z^{41}) = [0.1000; 0.3750; 2.7080; 6.6972; 3.5416; 3.4750]$, and the objective function value of problem (EP1) is 2.6918, respectively. Then, we continue to solve linear programming problem $(\mathrm{LPR}_{R^{42}})$ and get the corresponding optimal solution and value which are $(x^{42}, y^{42}, z^{42}) = [0.9488; 1.3139; 4.1753; 5.5554; 3.8947; 3.2628]$ and $2.7375 > 2.6918$, respectively. So, this node is deleted; then, the current optimal solution and optimal value of the problem will not be updated. As a result, the lower bound of the current problem (EP1) is updated to 2.6697. Continue to select and section the rectangle R^{32} corresponding to the current lower bound, resulting in two subrectangles:

$$R^{51} = \begin{bmatrix} 2.6201 & 5.8541 \\ 3.8800 & 6.6971 \\ 2.6827 & 3.0867 \\ 2.7625 & 3.4750 \end{bmatrix}, \quad R^{52} = \begin{bmatrix} 2.6201 & 5.8541 \\ 3.8800 & 6.6971 \\ 3.0867 & 3.4907 \\ 2.7625 & 3.4750 \end{bmatrix}. \tag{46}$$

10 of flowchart 1 represents rectangle R^{51}; 11 of flowchart 1 represents rectangle R^{52}; through solving linear programming problem $(\mathrm{LPR}_{R^{51}})$, the corresponding optimal value and the optimal solution are $2.8417 > 2.6918$ and $(x^{51}, y^{51}, z^{51}) = [0.1000; 1.9201; 2.6385; 6.0692; 3.0867; 3.0201]$, respectively. So, this node is deleted. Then, we continue to solve linear programming problem $(\mathrm{LPR}_{R^{52}})$ and obtain the corresponding optimal solution and optimal value which are $(x^{52}, y^{52}, z^{52}) = [0.0999; 2.3241; 2.7002; 6.6269; 3.4907; 3.4241]$ and 2.6816, and the objective function value of problem (EP1) is $2.7089 > 2.6918$, respectively; then, the current optimal solution and optimal value of the problem will not be updated. As a result, the lower bound of the current problem (EP1) is updated to 2.6719. Continue to select and section the rectangle R^{41} corresponding to the current lower bound, resulting in two subrectangles:

$$R^{61} = \begin{bmatrix} 2.6201 & 5.8541 \\ 3.8800 & 6.6971 \\ 3.4907 & 3.8947 \\ 2.7625 & 3.1188 \end{bmatrix}, \quad R^{62} = \begin{bmatrix} 2.6201 & 5.8541 \\ 3.8800 & 6.6971 \\ 3.4907 & 3.8947 \\ 3.1188 & 3.4750 \end{bmatrix}. \tag{47}$$

12 of flowchart 1 represents rectangle R^{61}; 13 of flowchart 1 represents rectangle R^{62}; through solving linear programming problem $(\mathrm{LPR}_{R^{61}})$, the corresponding optimal value and the optimal solution are $2.8039 > 2.6918$ and $(x^{61}, y^{61}, z^{61}) = [0.7855; 1.3332; 3.8649; 5.5198; 3.6420; 3.1188]$, respectively. So, this node is removed. Then, we continue to solve linear programming problem $(\mathrm{LPR}_{R^{62}})$ and obtain the corresponding optimal solution and optimal value which are $(x^{62}, y^{62}, z^{62}) = [0.1000; 2.3750; 2.7080; 6.6972; 3.5416; 3.4750]$ and 2.6850, and the objective function value of problem (EP1) is 2.6918, respectively; then, the current optimal solution and optimal value of problem (EP1) will not be updated. As a result, the lower bound of the current problem (EP1) is updated to 2.6816. Continue to select and section the rectangle R^{52} corresponding to the current lower bound, resulting in two subrectangles:

$$R^{71} = \begin{bmatrix} 2.6201 & 5.8541 \\ 3.8800 & 6.6971 \\ 3.0867 & 3.4907 \\ 2.7625 & 3.1188 \end{bmatrix}, \quad R^{72} = \begin{bmatrix} 2.6201 & 5.8541 \\ 3.8800 & 6.6971 \\ 3.0867 & 3.4907 \\ 3.1188 & 3.4750 \end{bmatrix}. \tag{48}$$

14 of flowchart 1 represents rectangle R^{71}; 15 of flowchart 1 represents rectangle R^{72}; through solving linear programming problem $(\mathrm{LPR}_{R^{71}})$, the corresponding optimal value and the optimal solution are $2.8162 > 2.6918$ and $(x^{71}, y^{71}, z^{71}) = [0.1000; 2.0188; 2.6536; 6.2054; 3.1854; 3.1188]$, respectively. So, this node is removed. Then, we continue to solve linear programming problem $(\mathrm{LPR}_{R^{72}})$ and obtain the corresponding optimal solution and optimal value which are $(x^{72}, y^{72}, z^{72}) = [0.1000; 2.3241; 2.7003; 6.6269; 3.4907; 3.4241]$ and $2.7008 > 2.6918$, respectively. So, this node is deleted; then, the current optimal solution and optimal value of problem (EP1) will not be updated. So far, there is one node 13 left on the branch-and-bound tree, and the current best optimal value and best solution for problem (EP1) are 2.6918 and [0.1000; 2.3750; 2.7080; 6.6972; 3.5416; 3.4750], respectively. As a result, the lower bound of the current problem (EP1) is updated to 2.6850. At this point, in the current iteration, the difference between the upper and lower bounds of problem (EP1) satisfies $2.6918 - 2.6850 = 0.0068 < 0.01$, so the algorithm iteration stops. Then, the optimal solution of Problem 1 (or (P)) is [0.1000; 2.3750], and the optimal value is $2.6918 - (M_1 + M_2) = -4.84151$. The specific steps are shown in Figure 1.

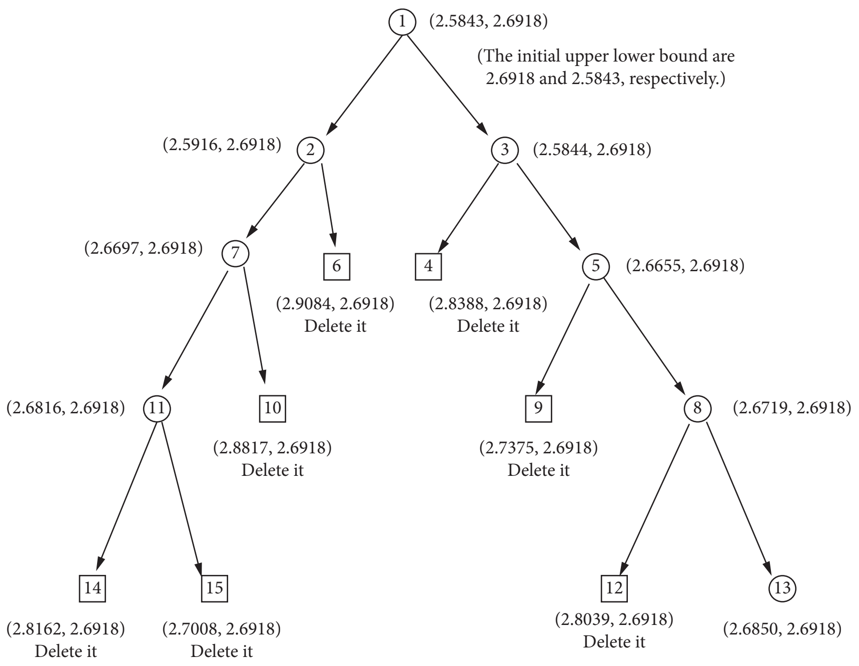

Figure 1: Algorithm flowchart 1. *Note.* (a, b), where "a" represents the object value of the relaxation problem and "b" represents the object value of the original problem. Spherical arrows represent the termination nodes. Node box represents a corresponding reduced rectangle, and it is finally removed.

Problem 2 (see [8, 21])

$$\min \frac{37x_1 + 73x_2 + 13}{13x_1 + 13x_2 + 13} + \frac{63x_1 - 18x_2 + 39}{13x_1 + 26x_2 + 13}$$
$$\text{s.t.} \quad 5x_1 - 3x_2 = 3, 1.5 \le x_1 \le 3. \tag{49}$$

Problem 3 see [1, 8, 21])

$$\min \frac{4x_1 + 3x_2 + 3x_3 + 50}{3x_2 + 3x_3 + 50} + \frac{3x_1 + 4x_3 + 50}{4x_1 + 4x_2 + 5x_3 + 50}$$
$$+ \frac{x_1 + 2x_2 + 4x_3 + 50}{x_1 + 5x_2 + 5x_3 + 50} + \frac{x_1 + 2x_2 + 4x_3 + 50}{5x_2 + 4x_3 + 50} \tag{50}$$
$$\text{s.t.} \quad \begin{cases} 2x_1 + x_2 + 5x_3 \le 10, x_1 + 6x_2 + 2x_3 \le 10 \\ 9x_1 + 7x_2 + 3x_3 \ge 10, x_1, x_2, x_3 \ge 0. \end{cases}$$

Problem 4 (see [1, 8, 21])

$$\min -\frac{3x_1 + 5x_2 + 3x_3 + 50}{3x_1 + 4x_2 + 5x_3 + 50} - \frac{3x_1 + 4x_2 + 50}{4x_1 + 3x_2 + 2x_3 + 50}$$
$$- \frac{4x_1 + 2x_2 + 4x_3 + 50}{5x_1 + 4x_2 + 3x_3 + 50}$$
$$\text{s.t.} \quad 6x_1 + 3x_2 + 3x_3 \le 10, 10x_1 + 3x_2 + 8x_3 \le 10, x_1, x_2, x_3 \ge 0. \tag{51}$$

Problem 5 (see [9, 17])

$$\max \sum_{i=1}^{5} \frac{c_i^T x + d_i}{e_i^T x + f_i}$$
$$\text{s.t. } Ax \le b, x \ge 0, \tag{52}$$

where

$$
\begin{aligned}
c_1 &= (0.0, -0.1, -0.3, 0.3, 0.5, 0.5, -0.8, 0.4, -0.4, 0.2, 0.2, -0.1)^T,\\
d_1 &= 14.6,\\
e_1 &= (-0.3, -0.1, -0.1, -0.1, 0.1, 0.4, 0.2, -0.2, 0.4, 0.2, -0.4, 0.3)^T,\\
f_1 &= 14.2,\\
c_2 &= (0.2, 0.5, 0.0, 0.4, 0.1, -0.6, -0.1, -0.2, -0.2, 0.1, 0.2, 0.3)^T,\\
d_2 &= 7.1,\\
e_2 &= (0.0, 0.1, -0.1, 0.3, 0.3, -0.2, 0.3, 0.0, -0.4, 0.5, -0.3, 0.1)^T,\\
f_2 &= 1.7,\\
c_3 &= (-0.1, 0.3, 0.0, 0.1, -0.1, 0.0, 0.3, -0.2, 0.0, 0.3, 0.5, 0.3)^T,\\
d_3 &= 1.7,\\
e_3 &= (0.8, -0.4, 0.7, -0.4, -0.4, 0.5, -0.2, -0.8, 0.5, 0.6, -0.2, 0.6)^T,\\
f_3 &= 8.1,\\
c_4 &= (-0.1, 0.5, 0.1, 0.1, -0.2, -0.5, 0.6, 0.7, 0.5, 0.7, -0.1, 0.1)^T,\\
d_4 &= 4.0,\\
e_4 &= (0.0, 0.6, -0.3, 0.3, 0.0, 0.2, 0.3, -0.6, -0.2, -0.5, 0.8, -0.5)^T,\\
f_4 &= 26.9,\\
c_5 &= (0.7, -0.5, 0.1, 0.2, -0.1, -0.3, 0.0, -0.1, -0.2, 0.6, 0.5, -0.2)^T,\\
d_5 &= 6.8,\\
e_5 &= (0.4, 0.2, -0.2, 0.9, 0.5, -0.1, 0.3, -0.8, -0.2, 0.6, -0.2, -0.4)^T,\\
f_5 &= 3.7,\\
A &= \begin{bmatrix}
-1.8 & -2.2 & 0.8 & 4.1 & 3.8 & -2.3 & -0.8 & 2.5 & -1.6 & 0.2 & -4.5 & -1.8\\
4.6 & -2.0 & 1.4 & 3.2 & -4.2 & -3.3 & 1.9 & 0.7 & 0.8 & -4.4 & 4.4 & 2.0\\
3.7 & -2.8 & -3.2 & -2.0 & -3.7 & -3.3 & 3.5 & -0.7 & 1.5 & -3.1 & 4.5 & -1.1\\
-0.6 & -0.6 & -2.5 & 4.1 & 0.6 & 3.3 & 2.8 & -0.1 & 4.1 & -3.2 & -1.2 & -4.3\\
1.8 & -1.6 & -4.5 & -1.3 & 4.6 & 3.3 & 4.2 & -1.2 & 1.9 & 2.4 & 3.4 & -2.9\\
-0.5 & -4.1 & 1.7 & 3.9 & -0.1 & -3.9 & -1.5 & 1.6 & 2.3 & -2.3 & -3.2 & 3.9\\
0.3 & 1.7 & 1.3 & 4.7 & 0.9 & 3.9 & -0.5 & -1.2 & 3.8 & 0.6 & -0.2 & -1.5\\
0.5 & -4.2 & 3.6 & -0.6 & -4.8 & 1.5 & -0.3 & 0.6 & -3.6 & 0.2 & 3.8 & -2.8\\
0.1 & 3.3 & -4.3 & 2.4 & 4.1 & 1.7 & 1.0 & -3.3 & 4.4 & -3.7 & -1.1 & -1.4\\
-0.6 & 2.2 & 2.5 & 1.3 & -4.3 & -2.9 & -4.1 & 2.7 & -0.8 & -2.9 & 3.5 & 1.2\\
4.3 & 1.9 & -4.0 & -2.6 & 1.8 & 2.5 & 0.6 & 1.3 & -4.3 & -2.3 & 4.1 & -1.1\\
0.0 & 0.4 & -4.5 & -4.4 & 1.2 & -3.8 & -1.9 & 1.2 & 3.0 & -1.1 & -0.2 & 2.5\\
-0.1 & -1.7 & 2.9 & 1.5 & 4.7 & -0.3 & 4.2 & -4.4 & -3.9 & 4.4 & 4.7 & -1.0\\
-3.8 & 1.4 & -4.7 & 1.9 & 3.8 & 3.5 & 1.5 & 2.3 & -3.7 & -4.2 & 2.7 & -0.1\\
0.2 & -0.1 & 4.9 & -0.9 & 0.1 & 4.3 & 1.6 & 2.6 & 1.5 & -1.0 & 0.8 & 1.6
\end{bmatrix},\\
b &= (15.7, 31.8, -36.4, 38.5, 40.3, 10.0, 89.8, 5.8, 2.7, -16.3, -14.6, -72.7, 57.7, -34.5, 69.1)^T.
\end{aligned}
\tag{53}
$$

Table 2: Comparison of numerical results by using Problem 6.

$(p,\ m,\ n)$	The algorithm of the [1]		Our algorithm in this paper	
	Avg (Std).Iter	Avg (Std).Time	Avg (Std).Iter	Avg (Std).Time
(2,100,100)	48.3571 (17.2386)	3.9248 (1.7489)	16.8667(5.4267)	2.2959 (0.7955)
(2,100,500)	67 (27.1833)	20.0863 (10.6748)	17.3333 (4.7703)	8.9810 (2.6774)
(2,100,800)	78 (7.1614)	42.3103 (19.5719)	16.4000 (4.1091)	14.3999 (4.2656)
(2,100,1000)	75.2308 (24.3145)	57.3531 (35.5371)	14.9333 (2.5157)	16.1681 (3.2219)
(2,100,1500)	86.0769 (5.7076)	107.0487 (37.6392)	15.2667 (2.4891)	27.3314 (5.2243)
(2,100,2000)	88.5333 (7.7724)	152.7871 (62.4697)	15 (1.1596)	37.5513 (4.4475)
(2,100,2500)	92.8462 (5.9979)	184.3362 (66.3833)	16 (3.0984)	54.4253 (12.8335)
(2,100,3000)	91.9000 (7.7810)	296.6591 (32.0870)	15.0667 (2.5157)	63.8862 (13.5362)
(3,50,50)	230.5217 (74.0483)	5.2437 (1.6432)	64.9333 (36.3858)	3.1009 (1.5003)
(3,50,100)	317.6522 (69.9140)	10.0783 (2.5300)	79.8000 (42.6555)	4.8213 (2.4819)
(3,50,200)	329 (85.8534)	15.6736 (4.0080)	99.6667 (55.4997)	9.3447 (3.8551)
(3,50,500)	367.9412 (85.5055)	34.1162 (8.4457)	89.7333 (37.3050)	18.4939 (8.0485)
(3,50,1000)	387.6667 (80.9100)	111.7268 (56.1365)	86.8667 (27.7101)	35.3195 (11.5188)
(3,50,2000)	462.6667 (112.0655)	242.6109 (117.1050)	86.2667 (22.8195)	80.6322 (21.9889)
(3,100,50)	236.0714 (74.0088)	11.4581 (7.0665)	78.6000 (65.1356)	8.1895 (6.8602)
(3,100,100)	252.8182 (74.0362)	20.5326 (8.1031)	89 (57.1116)	12.8639 (8.5089)
(3,100,200)	290.8824 (54.0600)	44.2078 (20.3639)	78.8000 (33.2189)	18.2206 (7.6801)
(3,100,300)	313.9167 (79.1483)	62.4293 (29.0958)	79.6667 (33.3500)	26.5176 (12.0702)
(3,100,400)	326.5333 (72.9421)	80.5609 (28.5460)	83.0667 (23.9429)	36.1708 (11.4116)
(3,100,500)	338.5833 (63.1278)	91.9672 (36.7591)	91.4667 (40.4242)	50.8475 (24.0507)

Problem 6 (see [1])

$$\begin{aligned} \min \sum_{i=1}^{p} \frac{\sum_{j=1}^{n} a_j^i x_j + b_i}{\sum_{j=1}^{n} c_j^i x_j + d_i} \\ \text{s.t.} \sum_{j=1}^{n} Q_j^l x_j \le q_l, \quad l = 1, 2, \ldots, m, x \ge 0, \end{aligned} \tag{54}$$

where both real numbers a_j^i, c_j^i and Q_j^l are randomly generated in the interval [0, 10], $b_i = d_i = 10$, and $q_l = 10$, which are the same as the data generation method in [1].

Besides, as can be seen from Table 1, the algorithm in this paper can effectively solve the five test problems known in the literature. Through the description of the solving process of Problem 1, it can be seen that the proposed algorithm finds the optimal solution in the initial iteration, but the branch-and-bound algorithm needs to run again 7 times, which can make the upper and lower bounds meet the precision requirements. The results are not as good as the methods in [1]. However, Problem 1 is a small-scale example. In the relatively large-scale random experiments to Problem 6 in the following, the numerical results show that the proposed algorithm is better than that of [1]. At least, the algorithm is better than the algorithm in [8] when solving Problems 2– 4. And then, for Problem 5, we find that the solution obtained by the algorithm in [9] is not feasible, but the solution solved by the algorithm in [17] is feasible, and the results obtained by the proposed algorithm are the same as those obtained in [17], and the number of iterations used is less than the previous two.

Next, we also perform random tests on Problem 6 to further explore the performance of the algorithm. We set the convergence accuracy of the algorithm to 0.01. For each set of the fixed parameter (p, m, n), we run the algorithm 15 times to compare with the algorithm in [1] and give the numerical results in Table 2. In addition, we have done a series of large-scale numerical experiments, and the numerical results are also indicated in Table 3. In Tables 2 and 3, Avg.Time and Avg.Iter represent the average CPU run time and average number of iterations applied by the algorithm to run 15 times, and Std.Time and Std.Iter represent the standard deviation of the CPU run time and number of iterations used by the algorithm to run 15 times, respectively.

Table 3: Numerical results for Problem 6 with large-scale variables.

(p, m, n)	Avg (Std).Iter	Avg (Std).Time
(2, 100, 4000)	15.4000 (3.3025)	96.5791 (25.2020)
(2, 100, 5000)	15.5333 (2.2171)	133.1045 (24.0255)
(3, 100, 1000)	80.3333 (21.2781)	93.9706 (25.2906)
(3, 100, 2000)	87.0667 (29.1330)	243.9302 (87.2392)
(3, 100, 3000)	74.2667 (15.0309)	350.9187 (75.8487)
(3, 100, 4000)	69.0667 (14.7036)	484.1551 (110.7565)
(3, 100, 5000)	65.0667 (15.8513)	606.5856 (154.7327)
(4, 100, 1000)	459.9333 (239.5927)	577.2682 (322.2345)
(4, 100, 2000)	508.5333 (239.1791)	1462.4441 (692.1418)
(4, 100, 3000)	403.8667 (161.4930)	1997.0592 (825.9073)
(5, 100, 1000)	1908.4000 (1298.0444)	1795.8044 (769.7409)
(6, 100, 1000)	4874.2000 (2968.7632)	4714.3262 (936.0817)

Table 2 shows that the proposed algorithm is better than the one in [1] in terms of average CPU running time and average number of iterations. In addition, from the standard deviation of the number of iterations and CPU running time, it can be seen that our algorithm is more stable than the algorithm in [1]. Moreover, it can be seen from the data in Tables 2 and 3 that, under the premise of fixed parameter (p, m), the CPU running time required by the algorithm is gradually increasing with the scale of Problem 6. On the premise of fixed parameter (m, n), however, the CPU running time and iteration times of the algorithm are increasing with the increase of the number p of the linear fraction in the objective function of problem (LFP). Interestingly, the numerical results in Table 3 show that our algorithm has more advantages in solving large-scale (LFP) problems when $p \ll n$, which is in agreement with the conclusions described in [8].

5. Concluding Remarks

In this paper, we mainly propose an algorithm that can address problem (LFP) effectively. Based on the branching operation of the p-dimensional output space, a unique construction method of the linear lower-bound relaxation subproblem is given. A branch-and-bound algorithm is therefore proposed to find the global optimal solution of problem (LFP) by combining the pruning operation. Numerical experiments show that their algorithm is effective and feasible, its calculation effect is better than that in [1], and it is more suitable for solving large-scale (LFP) problems in the case of $p \ll n$. Moreover, linear fractional programming problems in uncertain environments (such as [31–34]) have also been gradually studied by relevant scholars. In future studies, we will also try to gradually study the linear fractional programming problems in uncertain environments.

Data Availability

All data and models generated or used during the study are described in the numerical experiments section (Section 4) of the submitted manuscript.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (Grant no. 11961001), the Construction Project of First-Class Subjects in Ningxia Higher Education (Grant no. NXYLXK2017B09), and the Major Proprietary Funded Project of North Minzu University (Grant no. ZDZX201901).

References

[1] P. Shen, B. Huang, and L. Wang, "Range division and linearization algorithm for a class of linear ratios optimization problems," *Journal of Computational and Applied Mathematics*, vol. 350, pp. 324–342, 2019.

[2] J. Majhi, R. Janardan, J. Schwerdt et al., "Minimizing support structures and trapped area in two-dimensional layered manufacturing," *Computational Geometry*, vol. 12, no. 3-4, pp. 241–267, 1999.

[3] J. Majhi, R. Janardan, M. Smid et al., "On some geometric optimization problems in layered manufacturing (extended abstract)," *Computational Geometry*, vol. 12, no. 3-4, pp. 219–239, 1999.

[4] E. M. Arkin, Y.-J. Chiang, M. Held et al., "On minimum-area hulls," *Algorithmica*, vol. 21, no. 1, pp. 119–136, 1998.

[5] A. A. Khan, R. S. Adve, and W. Yu, "Optimizing downlink resource allocation in multiuser mimo networks via fractional programming and the Hungarian algorithm," *IEEE Transactions on Wireless Communications*, vol. 19, no. 8, pp. 5162–5175, 2020.

[6] S. Schaible, *Fractional Programming*, Springer, Berlin, Germany, pp. 495–608, 1995.

[7] H. Jiao, S. Liu, J. Yin, and Y. Zhao, "Outcome space range reduction method for global optimization of sum of affine ratios problem," *Open Mathematics*, vol. 14, no. 1, pp. 736–746, 2016.

[8] X. Liu, Y. L. Gao, B. Zhang, and F. P. Tian, "A new global optimization algorithm for a class of linear fractional programming," *Mathematics*, vol. 7, no. 9, p. 867, 2019.

[9] H.-W. Jiao and S.-Y. Liu, "A practicable branch and bound algorithm for sum of linear ratios problem," *European Journal of Operational Research*, vol. 243, no. 3, pp. 723–730, 2015.

[10] A. Charnes and W. W. Cooper, "Programming with linear fractional functionals," *Naval Research Logistics Quarterly*, vol. 9, no. 3-4, pp. 181–186, 1962.

[11] G. Charnes and A. Novaes, "Linear programming with a fractional objective function," *Operation Research*, vol. 21, no. 1, pp. 22–29, 1973.

[12] W. Dinkelbach, "On nonlinear fractional programming," *Management Science*, vol. 13, no. 7, pp. 492–498, 1967.

[13] S. Das and T. Mandal, "A single stage single constraints linear fractional programming problem: an approach," *Operations Research and Applications: An International Journal*, vol. 2, no. 1, 2015.

[14] T. Matsui, "Np-hardness of linear multiplicative programming and related problems," *Journal of Global Optimization*, vol. 9, no. 2, pp. 113–119, 1996.

[15] J. E. Falk and S. W. Palocsay, "Image space analysis of generalized fractional programs," *Journal of Global Optimization*, vol. 4, no. 1, pp. 63–88, 1994.

[16] H. P. Benson, "Branch-and-Bound outer approximation algorithm for sum-of-ratios fractional programs," *Journal of Optimization Theory and Applications*, vol. 146, no. 1, 2010.

[17] N. T. H. Phuong and H. Tuy, "A unified monotonic approach to generalized linear fractional programming," *Journal of Global Optimization*, vol. 26, no. 3, pp. 229–259, 2003.

[18] A. M. Ashtiani and P. A. V. Ferreira, "A branch-and-cut algorithm for a class of sum-of-ratios problems," *Applied Mathematics and Computation*, vol. 268, pp. 596–608, 2015.

[19] H. Jiao, Z. Wang, and Y. Chen, "Global optimization algorithm for sum of generalized polynomial ratios problem," *Applied Mathematical Modelling*, vol. 37, no. 1-2, pp. 187–197, 2013.

[20] P. Shen and B. Huang, "Global algorithm for solving linear multiplicative programming problems," *Optimization Letters*, vol. 14, no. 3, pp. 693–710, 2019.

[21] P.-P. Shen and T. Lu, "Regional division and reduction algorithm for minimizing the sum of linear fractional functions," *Journal of Inequalities and Applications*, vol. 2018, no. 1, p. 63, 2018.

[22] Y. E. Nesterov and A. S. Nemirovskii, "An interior-point method for generalized linear-fractional programming," *Mathematical Programming*, vol. 69, no. 1-3, pp. 177–204, 1995.

[23] H. Konno and N. Abe, "Minimization of the sum of three linear fractional functions," *Journal of Global Optimization*, vol. 15, no. 4, pp. 419–432, 1999.

[24] H. P. Benson, "On the global optimization of sums of linear fractional functions over a convex set," *Journal of Optimization Theory and Applications*, vol. 121, no. 1, pp. 19–39, 2004.

[25] T. Kuno, "A revision of the trapezoidal branch-and-bound algorithm for linear sum-of-ratios problems," *Journal of Global Optimization*, vol. 33, no. 2, pp. 215–234, 2005.

[26] P. Shen, W. Li, and Y. Liang, "Branch-reduction-bound algorithm for linear sum-of-ratios fractional programs," *Pacific Journal of Optimization*, vol. 11, no. 1, pp. 79–99, 2015.

[27] H. Jiao, Q. Feng, P. Shen, and Y. Guo, "Global optimization for sum of linear ratios problem using new pruning technique," *Mathematical Problems in Engineering*, vol. 2008, no. 3, 13 pages, Article ID 646205, 2008.

[28] D. Depetrini and M. Locatelli, "Approximation of linear fractional-multiplicative problems," *Mathematical Programming*, vol. 128, no. 1-2, pp. 437–443, 2011.

[29] P. Shen, T. Zhang, and C. Wang, "Solving a class of generalized fractional programming problems using the feasibility of linear programs," *Journal of Inequalities and Applications*, vol. 2017, no. 1, p. 147, 2017.

[30] Y. Xia, L. Wang, and S. Wang, "Minimizing the sum of linear fractional functions over the cone of positive semidefinite matrices: approximation and applications," *Operations Research Letters*, vol. 46, no. 1, pp. 76–80, 2018.

[31] S. K. Das, T. Mandal, and S. A. Edalatpanah, "A new approach for solving fully fuzzy linear fractional programming problems using the multi-objective linear programming," *RAIRO-Operations Research*, vol. 51, no. 1, pp. 285–297, 2017.

[32] S. K. Das, S. A. Edalatpanah, and T. Mandal, "A proposed model for solving fuzzy linear fractional programming problem: numerical point of view," *Journal of Computational Science*, vol. 25, pp. 367–375, 2018.

[33] S. K. Das, S. A. Edalatpanah, and T. Mandal, "A new method for solving linear fractional programming problem with absolute value functions," *International Journal of Operational Research*, vol. 36, no. 4, pp. 455–466, 2019.

[34] F. Abad, M. Allahdadi, and H. Nehi, "Interval linear fractional programming: optimal value range of the objective function," *Computational and Applied Mathematics*, vol. 39, no. 4, pp. 1–17, 2020.