Mathematics

http://www.aimspress.com/journal/Math

AIMS Mathematics, 10(9): 21004–21024.
DOI:10.3934/math.2025938
Received: 30 June 2025
Revised: 28 August 2025
Accepted: 02 September 2025
Published: 12 September 2025

Research article

Efficient algorithm for addressing large-scale linear fractional program problems

Zhiguo Ge[1,*] **and Hongwei Jiao**[2,*]

[1] Innovation and Entrepreneurship College, Henan Medical University, Xinxiang 453003, China
[2] School of Mathematical Sciences, Henan Institute of Science and Technology, Xinxiang 453003, China

* **Correspondence:** Email: gezhiguo1979369@163.com, jiaohongwei@126.com.

Abstract: This paper presented an efficient algorithm for addressing large-scale linear fractional program problems, which are widely used in hospital management. First of all, we converted the initial problem into an equivalent problem by applying the Charnes-Cooper transformation technique. Next, by directly relaxing the nonlinear constraints, a mixed-integer linear relaxation problem was then constructed. Subsequently, by successively partitioning the initial output space rectangle and solving a series of mixed-integer linear relaxation problems, we proposed an efficient branch-relaxation-bound algorithm for globally addressing large-scale linear fractional program problems for the first time. Moreover, the computation complexity of the algorithm was analyzed, and the maximum number of iterations of the algorithm in the worst-case scenario was estimated. Furthermore, the experimental results demonstrated the high efficiency of the proposed algorithm in solving the investigated large-scale linear fractional program problem.

Keywords: large-scale linear fractional program; global optimization; branch-and-bound algorithm; computation complexity
Mathematics Subject Classification: 65K05, 90C26, 90C32

1. Introduction

As is known to all, the linear fractional program problem has a wide range of applications in hospital management [1, 12, 35], queueing location [20, 28], multi-stage transportation [28], chip layout and compaction [8], adaptive analysis [32], bond portfolio optimization [21], and image processing [33,34].

In recent years, numerous algorithms have been developed to solve linear fractional programming problems involving only continuous variables. Based on the construction characteristics and theoretical foundations of these algorithms, they can be classified as follows: the simplicial branch-and-dualitybound algorithm [3], image space analysis method [9], trapezoidal branch-and-bound algorithm [22], monotonic optimization approach [23], outer approximation algorithm [4], interior-point method [10], and rectangle branch-and-bound algorithms (for example: the rectangle branch-and-bound algorithms for nonlinear sum of ratios problems [2, 5], the rectangle branch-and-bound algorithms for the linear sum-of-ratios problem with lower dimension [6], the rectangle branch-and-bound algorithms for sum of affine fractional functions [15,31], the rectangle branch-and-bound algorithms for generalized linear fractional programming [16, 17]). More recently, to effectively solve generalized affine fractional programming problems, Jiao et al. [12] proposed an outer space approach using a branching method and a two-stage linearization technique.

Recently, Li et al. [36] proposed an outer space branch-and-bound algorithm for linear ratio problems based on the inverse denominator outer space partitioning search and direct relaxation bounding technique. Based on the entire fractional image space partitioning search and direct relaxation techniques, Li et al. [37] proposed an image space branch-and-bound algorithm for affine comparison equations. Based on the fractional image space partitioning search and two-stage relaxation technique, Hou and Liu [38] provided an image space branch-reduction-bound algorithm for generalized fractional programming problems. Based on the new spatial branching and relaxation bounding techniques, Hou and Liu [39] provided a spatial branching-pruning-bounding algorithm for generalized linear fractional problems. Hou and Liu [40] provided an accelerated outer space algorithm for generalized linear multiple-product programming problems based on the linear function image spatial branch search and region acceleration methods. Based on the standard space search and new relaxation techniques, Jiao et al. [41] proposed a standard space acceleration algorithm for the generalized multiple-product programming problems. For the generalized affine fractional programming problem, Jiao and Ma [42] proposed an efficient outer space algorithm based on the different relaxation processes or techniques. In addition, using the Charns-Cooper transform, a reduced outer space equivalence problem was constructed, and a direct relaxation technique was used to construct the relaxation problem. Jiao et al. [43] proposed a reduced outer space algorithm for the sum of linear fractional functions problems. The above algorithms are designed for linear fractional programming problems or generalized linear fractional programming problems with continuous variables. However, the computational efficiency is not high, making it difficult to solve large-scale linear fractional programming problem with mixed-integer variables.

The aforementioned method is employed to address the linear fractional program problem with only continuous variables. However, in numerous problems, quantities or logical relationships are typically represented using mixed-integer variables. Furthermore, mixed-integer programming proves to be exceedingly useful in resolving various issues, as demonstrated by Khalil [18], Wolsey [30], and Juan [29]. Recently, Chaiblaine and Mustapha [7] proposed an exact approach for solving the integer linear fractional program problem. However, the large-scale linear fractional program problems investigated in this paper have not received much attention in the current research. Actually, there is currently no method available to solve the mixed-integer linear fractional program problem. Therefore, it is necessary to develop an effective approach for solving the mixed-integer linear fractional programming problem.

The large-scale linear fractional program problem investigated in this article is defined as follows:

$$\text{(LFP):}\begin{cases} U = \min f(x) = \sum_{i=1}^{p} \frac{d_i^\top x + c_i}{e_i^\top x + g_i} \\ \quad \text{s.t. } x \in \chi = \{x \in \mathbb{R}^{n_a} \times \mathbb{Z}^{n_q} | Ax \le b\}, \end{cases}$$

where $b \in \mathbb{R}^m$; $A \in \mathbb{R}^{m \times n}$; $c_i, g_i \in \mathbb{R}$; $d_i, e_i \in \mathbb{R}^n$; $i \in \{1, 2, \ldots, p\}$; $n = n_a + n_q$; n_a and n_q respectively represent the number of continuous variables and integer variables; and n is a large positive integer. $\mathcal{X}$ is a non-empty bounded set, $c_i^\top x + f_i$ and $e_i^\top x + g_i$ are affine functions over $\mathcal{X}$, and $e_i^\top x + g_i > 0$, for all $x \in \mathcal{X}$.

Due to the non-convex objective function of problem (LFP), there may be multiple local optima that are not global optima, which brings enormous theoretical and computational complexity to solving problem (LFP).

In production and daily life, the hospital management problem [35] is a concrete example about the above large-scale linear fractional program problem. If a hospital wishes to increase its charges by a certain percentage from the last year, the mathematical modelling of the reward that the hospital receives is formulated as follows:

$$\begin{cases} \max\ F(x) = \sum_{i=1}^{\hat{d}} \left[\sum_{j=1}^{\hat{p}_i} \hat{o}_{ij} \left(1 + x_{ij}\right) + \hat{C}_i \frac{\sum_{j=1}^{\hat{p}_i} \hat{m}_{ij}(1+x_{ij})}{\sum_{j=1}^{\hat{p}_i} \hat{c}_{ij}(1+x_{ij})} \right], \\ \text{s.t. } \sum_{i=1}^{\hat{d}} \sum_{j=1}^{\hat{p}_i} \hat{c}_{ij} x_{ij} = \hat{q} \sum_{i=1}^{\hat{d}} \sum_{j=1}^{\hat{p}_i} \hat{c}_{ij}, \\ \quad \tilde{l}_{ij} \leqslant x_{ij} \leqslant \tilde{u}_{ij},\ i = 1, \ldots, \hat{d}, j = 1, \ldots, \hat{p}_i, x_{ij} \in \mathbb{R} \text{ or } \mathbb{Z}, \end{cases}$$

where $\hat{d}$ denotes the number of departments in a hospital, $\hat{p}_i$ represents the number of procedures in a department i, $i = 1, 2, \ldots, \hat{d}$; for each procedure j in a department i, $\hat{c}_{ij}, \hat{m}_{ij}$, and $\hat{o}_{ij}$, $i = 1, \ldots, \hat{d}, j = 1, \ldots, \hat{p}_i$, denotes the amount of total charges, Medicare outpatient charges, and other non-Medicare charges, respectively. $\hat{C}_i$, $i = 1, 2, \ldots, \hat{d}$, denotes the fixed Medicare outpatient cost for a department i; x_{ij} denotes the ratio of increase in charge for procedure j in department i, $i = 1, \ldots, \hat{d}, j = 1, \ldots, \hat{p}_i$; $\hat{q} \cdot 100$ percent denotes that the hospital manager wants an overall charge to increase the percentage; $\hat{c}_{ij} > 0$, $\hat{m}_{ij} \geqslant 0$, $\hat{o}_{ij} \geqslant 0$, $\tilde{l}_{ij} \leqslant \hat{q} \leqslant \tilde{u}_{ij}$, $i = 1, \ldots, \hat{d}, j = 1, \ldots, \hat{p}_i$. Due to the large number of departments i and procedures j in a hospital, the scale of variable x is often large, making the hospital cost problem a large-scale linear fractional program problem.

This paper exploits the structural characteristics of problem (LFP) by introducing auxiliary variables and applying the Charnes-Cooper transformation to derive an equivalent reformulation. A mixed-integer linear relaxation problem is then constructed by directly relaxing the nonlinear constraints, enabling the computation of a valid lower bound through a sequence of such relaxations. Furthermore, in the worst-case scenario, the maximum number of iterations of the algorithm is estimated. Numerical experiments demonstrate the effectiveness of the proposed method in solving large-scale problem (LFP).

The main contributions of this article are as follows: (1) For the first time, this article proposes an effective branch-and-bound algorithm for large-scale mixed-integer sum of linear ratios problems. (2) This article provides a complexity analysis of the algorithm, estimates the computational complexity of the algorithm in the worst-case scenario, and constructs a theoretical system for this type of algorithm.

(3) The numerical comparison results indicate that the algorithm has higher computational efficiency compared to the currently popular BARON solver [19].

The remainder of this paper is organized as follows: Section 2 reformulates the original problem (LFP) into an equivalent problem and constructs a mixed-integer linear relaxation using the proposed method. Section 3 introduces an output-space branch-relaxation-bound algorithm, along with complexity and convergence analysis. Section 4 presents numerical results on both deterministic instances and randomly generated examples from recent literature. Finally, Section 5 concludes the paper.

2. Equivalence transformation and its relaxation technique

For clarity of exposition, define the index set $\Theta = \{1, 2, \dots, p\}$. To obtain a global solution to problem (LFP), applying the Charnes-Cooper transformation, i.e., introducing variables $y_i = \frac{1}{e_i^\top x+g_i}$ and $s_i = y_i x, i = 1, 2, \dots, p$, we can obtain

$$\sum_{i=1}^{p} \frac{d_i^\top x + c_i}{e_i^\top x + g_i} = \sum_{i=1}^{p}(d_i^\top x + c_i)y_i = \sum_{i=1}^{p}(d_i^\top y_i x + c_i y_i) = \sum_{i=1}^{p}(d_i^\top s_i + c_i y_i),$$

$$1 = y_i(e_i^\top x + g_i) = (e_i^\top y_i x + g_i y_i) = e_i^\top s_i + g_i y_i.$$

Therefore, an equivalent problem of problem (LFP) can be derived as follows:

$$(\text{EP})\begin{cases} u = \min G(x, y, s) = \sum_{i=1}^{p}(d_i^\top s_i + c_i y_i) \\ \quad \text{s.t. } e_i^\top s_i + g_i y_i = 1,\ i \in \Theta, \\ \qquad s_i = y_i x,\ i \in \Theta, \\ \qquad y_i > 0,\ i \in \Theta, \\ \qquad Ax \le b, \\ \qquad x \in \mathbb{R}^{n_a} \times \mathbb{Z}^{n_q}, s \in \mathbb{R}^{p \times n}, y \in \mathbb{R}^p. \end{cases}$$

From the form of the equivalent problem above, it can be seen that by using the Charnes-Cooper transformation technique, we can transform the nonlinear objective function of problem (LFP) into a linear function, while the only nonlinear constraint function in the equivalent problem is the bilinear term. Therefore, for the equivalent problem (EP) after the Charnes-Cooper transformation, it is easier to construct its linear relaxation.

Next, the equivalence between problems (EP) and (LFP) is established.

Definition 2.1. *If there exists a point $x^* \in \chi$ such that $f(x^*) \le f(x)$ holds for any $x \in \chi$, then we say that x^* is an optimal solution of problem (LFP).*

Theorem 2.1. *x^* is a global optimum solution for problem (LFP) if and only if (x^*, y^*, s^*) is a global optimum solution for problem (EP) with $y_i^* = \frac{1}{e_i^\top x^* + g_i}$ and $s_i^* = y_i^* x^*$, $i \in \Theta$.*

Proof: If (x^*, y^*, s^*) is a global optimum solution for problem (EP), but x^* is not a global optimum solution for problem (LFP), then there always exists a feasible point $\bar{x}$ for problem (LFP) that satisfies $f(x^*) > f(\bar{x})$, that is,

$$\sum_{i=1}^{p} \frac{d_i^\top x^* + c_i}{e_i^\top x^* + g_i} = f(x^*) > f(\bar{x}) = \sum_{i=1}^{p} \frac{d_i^\top \bar{x} + c_i}{e_i^\top \bar{x} + g_i}. \tag{2.1}$$

Letting $\bar{y}_i = \frac{1}{e_i^\top \bar{x}+g_i}$ and $\bar{s}_i = \bar{y}_i\bar{x}, i \in \Theta$, $(\bar{x}, \bar{y}, \bar{s})$ can be obtained as a feasible point for problem (EP), and the following equation can be obtained:

$$\sum_{i=1}^{p} \frac{d_i^\top \bar{x} + c_i}{e_i^\top \bar{x} + g_i} = \sum_{i=1}^{p} d_i^\top \bar{s}_i + \bar{y}_i c_i. \tag{2.2}$$

By combining the inequalities (2.1) and (2.2), it can be deduced that

$$\sum_{i=1}^{p} \frac{d_i^\top x^* + c_i}{e_i^\top x^* + g_i} = f(x^*) > f(\overline{x}) = \sum_{i=1}^{p} \frac{d_i^\top \bar{x} + c_i}{e_i^\top \bar{x} + g_i} = \sum_{i=1}^{p} d_i^\top \bar{s}_i + \bar{y}_i c_i. \tag{2.3}$$

Furthermore, since (x^*, y^*, s^*) is a global optimum solution for problem (EP), it follows that

$$y_i^* = \frac{1}{e_i^\top x^* + g_i}, s_i^* = y_i^* x^*, i \in \Theta.$$

Hence, it follows that

$$\sum_{i=1}^{p} \frac{d_i^\top x^* + c_i}{e_i^\top x^* + g_i} = \sum_{i=1}^{p} d_i^\top s_i^* + c_i y_i^*. \tag{2.4}$$

Finally, by combining (2.3) and (2.4), we obtain

$$\sum_{i=1}^{p} d_i^\top \bar{s}_i + c_i \bar{y}_i < \sum_{i=1}^{p} d_i^\top s_i^* + c_i y_i^*.$$

The above inequality contradicts the assumption that (x^*, y^*, s^*) is a global optimum solution for problem (EP). Thus, the proof of the first half of the theorem has been established. Using a similar approach, we can also prove the converse case. □

In addition, the optimal values of problems (LFP) and (EP) are equal, which implies that solving problem (LFP) can be achieved by solving problem (EP).

Since $y_i = \frac{1}{e_i^\top x+g_i}$, its maximum and minimum values are obtained by computing the minimum and maximum values of $e_i^\top x + g_i$ over χ, respectively. Letting

$$t_i^0 = \frac{1}{\max_{x\in\chi} \left\{e_i^\top x + g_i\right\}} \text{ and } T_i^0 = \frac{1}{\min_{x\in\chi} \left\{e_i^\top x + g_i\right\}}, \quad i \in \Theta,$$

then an initial output space rectangle can be defined as follows:

$$H^0 = [t^0, T^0] = \left\{y \in \mathbb{R}^p \mid t_i^0 \le y_i \le T_i^0, i \in \Theta\right\}.$$

Subsequently, problem (EP) on the rectangle H^0 can be formulated as follows:

$$(\mathrm{EP}(H^0)) \begin{cases} u(H^0) = \min G(x, y, s) = \sum_{i=1}^{p} (d_i^\top s_i + c_i y_i) \\ \quad \text{s.t. } e_i^\top s_i + g_i y_i = 1, \ i \in \Theta, \\ \quad s_i = y_i x, \ i \in \Theta, \\ \quad y_i > 0, \ i \in \Theta, \\ \quad Ax \le b, \ x \in \mathbb{R}^{n_a} \times \mathbb{Z}^{n_q}, \ s \in \mathbb{R}^{p\times n}, \ y \in \mathbb{R}^p, \ y \in H^0. \end{cases}$$

Without loss of generality, we can define $H = [t, T] = \{y \in \mathbb{R}^p | t_i \leq y_i \leq T_i, i \in \Theta\} \subseteq H^0$, which results from the branching process. Then, problem (EP) corresponding to the domain H may be expressed as follows:

$$
(\text{EP}(H))\begin{cases}
u(H) = \min G(x, y, s) = \sum_{i=1}^{p} (d_i^\top s_i + c_i y_i) \\
\qquad \text{s.t. } e_i^\top s_i + g_i y_i = 1,\ i \in \Theta, \\
\qquad\quad s_i = y_i x,\ i \in \Theta, \\
\qquad\quad y_i > 0,\ i \in \Theta, \\
\qquad\quad Ax \leq b,\ x \in \mathbb{R}^{n_a} \times \mathbb{Z}^{n_q},\ s \in \mathbb{R}^{p \times n},\ y \in \mathbb{R}^p,\ y \in H.
\end{cases}
$$

To facilitate the solution of problem (EP(H)), the constraint $s_i = y_i x$ is relaxed by introducing bounds $t_i \leq y_i \leq T_i$ for all $i \in \Theta$, i.e., $s_i = y_i x$ is relaxed into $t_i x \leq s_i \leq T_i x$ for all $i \in \Theta$. Based on this relaxation technique, the mixed-integer linear relaxation problem (MILRP(H)) is formulated as follows:

$$
(\text{MILRP}(H))\begin{cases}
\underline{u}(H) = \min G(x, y, s) = \sum_{i=1}^{p} (d_i^\top s_i + c_i y_i) \\
\qquad \text{s.t. } e_i^\top s_i + g_i y_i = 1,\ i \in \Theta, \\
\qquad\quad t_i x \leq s_i \leq T_i x,\ i \in \Theta, \\
\qquad\quad y_i > 0,\ i \in \Theta, \\
\qquad\quad Ax \leq b,\ x \in \mathbb{R}^{n_a} \times \mathbb{Z}^{n_q},\ s \in \mathbb{R}^{p \times n},\ y \in \mathbb{R}^p,\ y \in H.
\end{cases}
$$

Based on the above relaxation technique, any feasible solution to problem (EP(H)) is also feasible to problem (MILRP(H)). At the same time, the objective functions of problems (EP(H)) and (MILRP(H)) are completely identical. Therefore, the optimal value of problem (MILRP(H)) provides a valid lower bound for that of problem (EP(H)), i.e., solving problem (MILRP(H)) yields a valid lower bound for the optimal value of problem (EP(H)).

3. Global algorithm and its theoretical analysis

In this section, an output space branch-relaxation-bound algorithm (OSBRBA) based upon the former relaxation method and the subsequent branching method is presented. In addition, theoretical analysis of the OSBRBA is also provided. Furthermore, the maximum number of iterations for the OSBRBA in the worst-case scenario is estimated.

3.1. Branching method

For any selected rectangle $\hat{H} = \{y \in \mathbb{R}^p |\ t_i \leq y_i \leq T_i, i = 1, 2, \ldots, p\} \subseteq H^0$, the proposed branching method is as follows:

(i) let $q = \arg\max\{T_i - t_i |\ i = 1, 2, \ldots, p\}$;

(ii) let $\lambda_q = (t_q + T_q)/2$;

(iii) let

$$\hat{H}^1 = \{y \in \mathbb{R}^p | t_i \leq y_i \leq T_i, i = 1, 2, \ldots, p, i \neq q;\ t_i \leq y_i \leq \lambda_i,\ i = q\},$$

and

$$\hat{H}^2 = \{y \in \mathbb{R}^p | t_i \leq y_i \leq T_i, i = 1, 2, \ldots, p, i \neq q;\ \lambda_i \leq y_i \leq T_i,\ i = q\}.$$

By the branching method, the rectangle $\hat{H}$ can be subdivided into two sub-rectangles $\hat{H}^1$ and $\hat{H}^2$.

3.2. Output space branch-relaxation-bound algorithm

The algorithm starts from the initial rectangle H^0, and gradually divides the investigated rectangle H into two equal sub-rectangles by bisecting its longest edge. Then, by solving problem (MILRP) about each sub-rectangle, we can compute the lower bounds of the optimal value for problem (EP) about each sub-rectangle. Meanwhile, we can also update the current best feasible solution for problem (EP), as well as the upper bounds. It should be noted that by calculating the objective function values at feasible points, we can update the upper bound. The smallest known lower bound is the lower bound of the original problem.

Definition 3.1. *Let v be the global optimum value for problem (EP), and let (x^k, y^k, s^k) be a feasible solution for problem (EP). For any given termination error $\epsilon > 0$, if $G(x^k, y^k, s^k) - v \leq \epsilon$, we call (x^k, y^k, s^k) an ϵ-global optimum solution for problem (EP).*

Steps for the OSBRBA:

Step 0. We are given that $\epsilon \in (0, 1)$.

Solve problem (MILRP(H^0)).

If problem (MILRP(H^0)) is infeasible, problem (LFP) is also infeasible. Otherwise, an optimal solution $(x_0, \hat{y}_0, \hat{s}_0)$ and optimal value $\underline{u}^0$ for problem MILRP(H_0) can be obtained. Let $y_i^0 = \frac{1}{e_i^\top x^0 + g_i}$, $s_i^0 = y_i^0 x^0, i = 1, \ldots, p$. Let the lower bound $LB_0 = \underline{u}^0$, and the upper bound $UB_0 = u^0 = G(x^0, y^0, s^0)$.

If $UB_0 - LB_0 \leq \epsilon$, then (x^0, y^0, s^0) is a globally ϵ-optimum solution for problem (EP), and the algorithm terminates.

Otherwise, let $k = 0$, denote $F = \{(x^0, y^0, s^0)\}$ as the set of feasible points, and denote $\Lambda_0 = \{H^0\}$ as the collection of all active nodes.

Step 1. By bisecting the longest edge of H^k, two new sub-rectangles H^{k_1} and H^{k_2} are generated, and the set $\Xi = \{H^{k_1}, H^{k_2}\}$ is formed.

Step 2. Solve problem (MILRP(H^{kj})) for each sub-rectangle $H^{kj} \in \Xi$, $j = 1, 2$.

If problem (MILRP(H^{kj})) is infeasible, problem (EP(H^{kj})) is also infeasible, and let $\Xi = \Xi \backslash \{H^{kj}\}$. Otherwise, obtain its optimal solution $\left(x^{H^{kj}}, \hat{y}^{H^{kj}}, \hat{s}^{H^{kj}}\right)$ and optimal value $\underline{u}\left(H^{kj}\right)$ of problem (MILRP(H^{kj})), and let $LB\left(H^{kj}\right) = \underline{u}\left(H^{kj}\right)$.

If $LB(H^{kj}) > UB_k$, let $\Xi = \Xi \backslash \{H^{kj}\}$. Otherwise, let

$$y_i^{H^{kj}} = \frac{1}{e^\top x^{H^{kj}} + g_i}, s_i^{H^{kj}} = y_i^{H^{kj}} x^{H^{kj}}, i = 1, \ldots, p,$$

and let

$$UB(H^{kj}) = G(x^{H^{kj}}, y^{H^{kj}}, s^{H^{kj}}).$$

Update the upper bound $UB_k = \min\{UB_k, UB(H^{kj})\}$, and update the feasible point set $F = F \bigcup \{(x^{H^{kj}}, y^{H^{kj}}, s^{H^{kj}})\}$. Meanwhile, refer to (x^k, y^k, s^k) as the current best feasible point for problem (EP), which satisfies $UB_k = G(x^k, y^k, s^k)$.

Step 3. Set $\Lambda_k = (\Lambda_k \backslash H^k) \bigcup \Xi$, and by letting $LB_k = \min\{LB(H) \mid H \in \Lambda_k\}$, update the lower bound

LB_k.

Step 4. If $UB_k - LB_k < \epsilon$, the algorithm terminates, and a globally ε-optimal solution (x^k, y^k, s^k) to problem (EP) is obtained.

Otherwise, let $k = k + 1$, pick out a sub-rectangle H^k that satisfies $H^k = \arg\min_{H \in \Lambda_k} LB(H)$, and go back to Step 1.

3.3. Theoretical analysis

This subsection first presents Theorem 3.1, which shows that when the investigated subrectangle H^k satisfies certain conditions, the algorithm terminates and returns a globally ε-optimal solution to problem (EP). Theorem 3.2 then provides a theoretical guarantee by analyzing the algorithm's behavior and estimating its maximum number of iterations in the worst-case scenario. For clarity of exposition, we introduce the following definitions:

$$M_i = \max_{x \in \mathcal{X}} |x_i|,\ i \in \Theta,\ \ M = \max\{M_i \big|\ i \in \Theta\},\ \ \mu = \max\{|c_i| \big|\ i \in \Theta\},$$

$$\sigma = \max\{|d_{ij}| \big|\ i \in \Theta, j \in \{1, 2, \ldots, n\}\},\quad \lambda = nM\sigma + \mu.$$

Theorem 3.1. *Given termination error $\epsilon > 0$, after k iterations, if the investigated sub-rectangle $H^k = \prod_{i=1}^{p} \left[T_i^k, t_i^k\right]$ satisfies*

$$T_q^k - t_q^k \leq \frac{\epsilon}{p\lambda},$$

where $q = \arg\max\left\{T_i^k - t_i^k | i \in \Theta\right\}$, the OSBRBA will terminate, and a globally ϵ-optimum solution for the (EP) will be returned.

Proof. Without loss of generality, assume that $(x^k, \hat{y}^k, \hat{s}^k)$ is the optimal solution of problem (MILRP) over H^k. Let $y_i^k = \frac{1}{e_i^\top x^k + g_i}$ and $s_i^k = y_i^k x^k$, $i \in \Theta$. It is obvious that (x^k, y^k, s^k) is a feasible point for problem (EP(H^k)), and we can obtain that

$$LB_k = G(x^k, \hat{y}^k, \hat{s}^k) \leq UB_k \leq G(x^k, y^k, s^k).$$

Furthermore, when the algorithm reaches the k_{th} iteration, if $H^k = \prod_{i=1}^{p} \left[t_i^k, T_i^k\right]$ satisfies $T_q^k - t_q^k \leq \frac{\epsilon}{p\lambda}$, then we can get that

$$
\begin{aligned}
|UB_k - LB_k| &\le |G(x^k, y^k, s^k) - G(x^k, \hat{y}^k, \hat{s}^k)| \\
&= \left| \sum_{i=1}^{p} (d_i^\top s_i^k + c_i y_i^k) - \sum_{i=1}^{p} (d_i^\top \hat{s}_i^k + c_i \hat{y}_i^k) \right| \\
&= \left| \sum_{i=1}^{p} d_i^\top (s_i^k - \hat{s}_i^k) + \sum_{i=1}^{p} c_i (y_i^k - \hat{y}_i^k) \right| \\
&\le \left| \sum_{i=1}^{p} d_i^\top (s_i^k - \hat{s}_i^k) \right| + \left| \sum_{i=1}^{p} c_i (y_i^k - \hat{y}_i^k) \right| \\
&\le \sum_{i=1}^{p} |d_i^\top| \times |s_i^k - \hat{s}_i^k| + \sum_{i=1}^{p} |c_i| \times |y_i^k - \hat{y}_i^k)| \\
&\le \sum_{i=1}^{p} |d_i^\top| \times |T_i^k x^k - t_i^k x^k| + \sum_{i=1}^{p} |c_i| \times |y_i^k - \hat{y}_i^k)| \\
&\le \sum_{i=1}^{p} (T_i^k - t_i^k) \times |d_i^\top x^k| + \sum_{i=1}^{p} |c_i| \times (T_i^k - t_i^k) \\
&\le \sum_{i=1}^{p} (T_i^k - t_i^k) \times \left(\sum_{j=1}^{n} |d_{ij}| x_j^k \right) + \sum_{i=1}^{p} |c_i| \times (T_i^k - t_i^k)
\end{aligned}
$$

$$
\begin{aligned}
&\le \sum_{i=1}^{p} \sum_{j=1}^{n} M |d_{ij}| |T_q^k - t_q^k| + \sum_{i=1}^{p} |c_i| |T_q^k - t_q^k| \\
&\le \sum_{i=1}^{p} \left(\sum_{j=1}^{n} M |d_{ij}| + |c_i| \right) |T_q^k - t_q^k| \\
&\le p(nM\sigma + \mu) |T_q^k - t_q^k| \\
&= p\lambda |T_q^k - t_q^k| \\
&\le \epsilon.
\end{aligned}
$$

This implies that the proposed algorithm will terminate at Step 0 or Step 4. Meanwhile, denoting v^* as the optimal value of problem (LFP), from Steps 0, 3, and 4 of the OSBRBA, noting that $v^* \le UB_k = G(x^k, y^k, s^k)$, $LB_k \le v^*$, and the termination condition that $UB_k - LB_k \le \epsilon$, we can further get that $v^* \le G(x^k, y^k, s^k) \le \epsilon + LB_k \le \epsilon + v^*$. Then, the OSBRBA returns an ϵ-globally optimum solution of the (EP).

Remark 3.1. *Supposing that the OSBRBA does not end after k iterations, then in the previous iteration, the H^0 are divided into at most $k+1$ sub-rectangles. From Theorem 3.1, in the k-th iteration, we choose q and obtain $T_q^k - t_q^k \ge \frac{\epsilon}{p\lambda}$. The interval $\left[t_q^k, T_q^k\right]$ is subdivided into two subintervals at the midpoint $\frac{t_q^k + T_q^k}{2}$. The two generated subintervals satisfy $T_q^{k1} - t_q^{k1} = T_q^{k2} - t_q^{k2} > \frac{\epsilon}{2p\lambda}$.* This indicates that each rectangle $H = \prod_{i=1}^{p} [t_i, T_i]$ satisfies

$$
T_i - t_i \ge \min \left\{ \frac{\epsilon}{2p\lambda}, T_i^0 - t_i^0 \right\}, \quad i = 1, \ldots, p.
$$

Furthermore, if $T_i^0 - t_i^0 \leq \frac{\epsilon}{p\lambda}$, the i-th orientation of H^0 is never partitioned in these k iterations. Next, we will obtain the maximum number of iterations for the proposed algorithm through the following theoretical analysis.

Theorem 3.2. *Given the termination error $\epsilon > 0$, a globally ϵ-optimum solution for the (EP) may be found using the OSBRBA after at most*

$$\hat{N} = \left\lceil \prod_{i=1}^{p} \max\left\{\frac{2p\lambda}{\epsilon}(T_i^0 - t_i^0), 1\right\} \right\rceil$$

iterations, where $\lceil \cdot \rceil$ represents taking an integer downward.

Proof: By Remark 3.1, after k iterations, the initial rectangle H^0 can be partitioned into at most $k + 1$ sub-rectangles by the branching process, and each sub-rectangle $H = \prod_{i=1}^{p} [t_i, T_i]$ satisfies

$$T_i - t_i \geq \min\left\{\frac{\epsilon}{2p\lambda}, T_i^0 - t_i^0\right\}, \quad i = 1, \ldots, p.$$

Thus, if we denote the volume of all these sub-rectangles as V, then we can obtain the following inequality:

$$V \geq (k+1) \prod_{i=1}^{p} \min\left\{\frac{\epsilon}{2p\lambda}, T_i^0 - t_i^0\right\}. \tag{3.1}$$

Besides,

$$V = \prod_{i=1}^{p} (T_i^0 - t_i^0) \tag{3.2}$$

is apparent. Thus, from (3.1) and (3.2), we can follow that

$$\prod_{i=1}^{p} (T_i^0 - t_i^0) \geq (k+1) \prod_{i=1}^{p} \min\left\{\frac{\epsilon}{2p\lambda}, T_i^0 - t_i^0\right\}.$$

This indicates that

$$k \leq \prod_{i=1}^{p} \max\left\{\frac{2p\lambda}{\epsilon}(T_i^0 - t_i^0), 1\right\}.$$

That is to say, this algorithm iterates at most

$$\hat{N} = \left\lceil \prod_{i=1}^{p} \max\left\{\frac{2p|nM\sigma + \mu|}{\epsilon}(T_i^0 - t_i^0), 1\right\} \right\rceil$$

times, and will be terminated by Theorem 3.1. Otherwise, if $k \geq \hat{N}$, then we can get the following inequality:

$$(k+1) \prod_{i=1}^{p} \min\left\{\frac{\epsilon}{2p\lambda}, T_i^0 - t_i^0\right\} > \prod_{i=1}^{p} \max\left\{\frac{2p\lambda}{\epsilon}(T_i^0 - t_i^0), 1\right\} \cdot \min\left\{\frac{\epsilon}{2p\lambda}, T_i^0 - t_i^0\right\}. \tag{3.3}$$

Subsequently, we take into account the following two situations:

(i) If $\frac{\epsilon}{2p\lambda} < T_i^0 - t_i^0$, we have

$$\max\left\{\frac{2p\lambda}{\epsilon}(T_i^0 - t_i^0), 1\right\} \times \min\left\{\frac{\epsilon}{2p\lambda}, T_i^0 - t_i^0\right\} = \frac{2p\lambda}{\epsilon}(T_i^0 - t_i^0) \times \frac{\epsilon}{2p\lambda} = T_i^0 - t_i^0. \tag{3.4}$$

(ii) If $\frac{\epsilon}{2p\lambda} > T_i^0 - t_i^0$, we have

$$\max\left\{\frac{2p\lambda}{\epsilon}(T_i^0 - t_i^0), 1\right\} \times \min\left\{\frac{\epsilon}{2p\lambda}, T_i^0 - t_i^0\right\} = 1 \times (T_i^0 - t_i^0) = T_i^0 - t_i^0. \tag{3.5}$$

From (3.1)–(3.5), it can be concluded that

$$\prod_{i=1}^{p}(T_i^0 - t_i^0) = V > \prod_{i=1}^{p}(T_i^0 - t_i^0),$$

which is obviously contradictory. Therefore, a globally ϵ-optimum solution for problem (EP) may be found using the OSBRBA after at most

$$\hat{N} = \lceil \prod_{i=1}^{p} \max\{\frac{2p\lambda}{\epsilon}(T_i^0 - t_i^0), 1\}\rceil$$

iterations, and the proof is complete.

Remark 3.2. *According to the conclusion of Theorem 3.1, when the maximum side length of all examined sub-rectangles satisfies the condition $T_q^k - t_q^k \le \frac{\epsilon}{p\lambda}$, where $q = \arg\max\left\{T_i^k - t_i^k | i \in \Theta\right\}$, the algorithm proposed in this paper terminates and obtains a globally ϵ-optimal solution to the problem. From the exhaustive nature of the rectangular maximum edge splitting rule used in this paper and the conclusion of Theorem 3.2, it can be concluded that after $\hat{N} = \lceil \prod_{i=1}^{p} \max\{\frac{2p\lambda}{\epsilon}(T_i^0 - t_i^0), 1\}\rceil$ iterations, the maximum edge length of all examined sub-rectangles will satisfy the condition $T_q^k - t_q^k \le \frac{\epsilon}{p\lambda}$. Therefore, the algorithm proposed in this paper converges to a globally ϵ-optimal solution of problem (EP).*

4. Numerical experiments

In this section, to assess the computing efficiency of the OSBRBA, we conducted a range of numerical experiments and implemented the code of the OSBRBA using MATLAB (2023a). All numerical computations were completed on a computer with an AMD Ryzen 5 5500 CPU 2.1GHz processor and 16GB RAM memory. Each mixed-integer linear relaxation problem was solved by the Intlinprog solver in the MATLAB Optimization Toolbox. In all experiments, the convergence error ϵ is set to 10^{-6}.

First, to demonstrate the superiority of the OSBRBA compared to the Baron solver, we tested ten deterministic examples and listed their numerical results in Table 1. Some symbols used in Table 1 are listed as follows: Time: CPU time(s), Iter: number of iterations, Opt. val.: optimum value.

Subsequently, to assess the robustness and computational effectiveness of the OSBRBA, we set $\epsilon = 10^{-6}$, and solved a randomly generated test problem (Example 11) with large-scale variables. Numerical comparisons between the OSBRBA and the Baron solver [19] (version 25.3.19) are reported in Table 2 and Figure 1, where the Baron solver is regarded as one of the best global optimization solvers available for commercial use.

Table 1. Comparisons of numerical results between the OSBRBA and BARON on test Examples 1–10.

Example	Algorithm	Opt. val.	Optimal solution	Iter.	Time
1	OSBRBA	-2.47143	(1.0000, 0.0000, 0.0000)	0	0.013
	Baron	-2.47140	(1.0000, 0.0000, 0.0000)	1	0.03
2	OSBRBA	-1.90000	(0.0000, 3.3333, 0.0000)	69	0.70
	Baron	-1.90000	(0.0000, 3.3333, 0.0000)	1	0.03
3	OSBRBA	1.62320	(0.0000, 0.2875)	502	3.56
	Baron	1.62320	(0.0000, 0.2839)	1	0.02
4	OSBRBA	2.86190	(5.0000, 0.0000, 0.0000)	1	0.016
	Baron	2.86190	(5.0000, 0.0000, 0.0000)	1	0.03
5	OSBRBA	-4.09070	(1.1111, 0.0000, 0.0000)	45	0.43
	Baron	-4.09070	(1.1111, 0.0000, 0.0000)	1	0.03
6	OSBRBA	3.71092	(0.0000, 1.6667, 0.0000)	99	1.02
	Baron	3.71090	(0.0000, 1.6667, 0.0000)	1	0.03
7	OSBRBA	-3.00292	(0.0000, 3.3333, 0.0000)	287	2.60
	Baron	-3.00290	(0.0000, 3.3333, 0.0000)	7	0.03
8	OSBRBA	-4.09070	(1.1111, 0.0000, 0.0000)	44	0.41
	Baron	-4.09070	(1.1109, 0.0000, 0.0005)	1	0.02
9	OSBRBA	3.29167	(3.0000, 4.0000)	0	0.005
	Baron	3.29170	(3.0000, 4.0000)	1	0.01
10	OSBRBA	4.42857	(5.0000, 0.0000, 0.0000)	0	0.005
	Baron	4.42860	(5.000, 0.0000, 0.0000)	1	0.02

Table 2. Computational comparisons between the OSBRBA and BARON on Example 11 with $p = 2$ and $m = 100$.

n	Algorithm	Iteration			Time		
		min	ave	max	min	ave	max
600	BARON	1	1.4	5	21.41	29.51	37.50
	Ours	12	48.1	89	10.85	26.16	48.75
700	BARON	1	1.8	5	28.70	37.39	57.55
	Ours	6	45.8	106	13.20	29.60	55.67
800	BARON	1	1.4	5	39.16	56.889	77.84
	Ours	15	44.7	93	17.11	44.95	132.28
900	BARON	1	3	5	62.05	84.95	128.45
	Ours	22	53.9	103	24.84	59.17	96.64
1000	BARON	1	1	1	68.11	91.01	151.70
	Ours	9	65.4	106	15.43	86.90	178.92
1500	BARON	1	1	1	223.05	284.07	364.07
	Ours	24	96.5	324	140.27	276.23	853.38
2000	BARON	1	1	1	425.38	594.04	831.70
	Ours	46	80.4	140	182.03	377.90	716.73
2500	BARON	1	1.4	5	780.41	1040.48	1479.05
	Ours	40	69.5	124	234.83	453.63	827.45
3000	BARON	1	2.6	5	1042.59	1561.41	1888.05
	Ours	56	96.2	148	477.18	836.55	1676.83
3500	BARON	1	1.4	5	1568.23	2295.708	2979.62
	Ours	36	70.8	122	448.70	831.18	1427.87
4000	BARON	1	3.4	5	2146.97	3209.87	3866.33
	Ours	50	82.3	123	804.15	1277.88	1814.45
4500	BARON	1	2.2	5	3017.02	3558.13	3851.44
	Ours	51	66	116	672.39	1295.32	2216.27
5000	BARON	1	2.2	5	3596.69	3868.91	4119.98
	Ours	65	96.5	178	1464.30	2237.79	4118.10

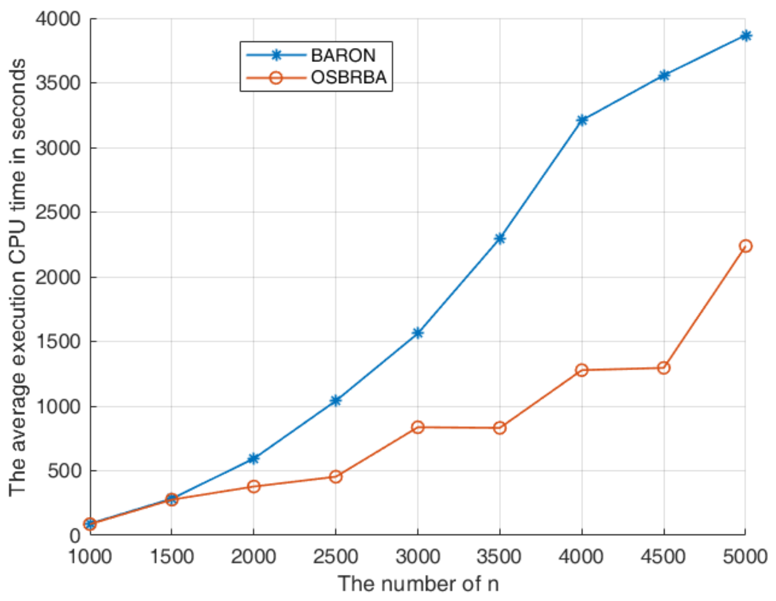

Figure 1. Numerical comparisons between the OSBRBA and BARON solver on the average time in Example 11 with $p = 2$ and $m = 100$.

For the random generated test Example 11, letting $p = 2$, $m = 100$, and $n \geq 600$, setting $\epsilon = 10^{-6}$, taking the first 50 components of variable x as integer variables, and setting other components of variable x as continuous variables, we solved 10 randomly generated test problems and recorded their computational results in Table 2. In Table 2, "ave" denotes the average value of the results.

These ten deterministic examples and one random example with their computational results are listed as below.

Example 1. (Phuong & Tuy [23], Shen et al. [26])

$$\begin{cases} \max & \frac{3x_1+x_2-2x_3+0.8}{2x_1-x_2+x_3} + \frac{4x_1-2x_2+x_3}{7x_1+3x_2-x_3} \\ \text{s.t.} & -6x_1 + x_2 + x_3 \leq -4.1, \\ & 12x_1 + 12x_2 + 7x_3 \leq 29.1, \\ & x_1 + x_2 - x_3 \leq 1, \\ & 12x_1 + 5x_2 + 12x_3 \leq 34.8, \\ & -x_1 + x_2 - x_3 \leq -1, \end{cases}$$

where x_1, x_2, and x_3 are integers.

Example 2. (Shen et al. [26], Shen & Wang [24])

$$\begin{cases} \max & -\frac{4x_1+3x_2+3x_3+50}{3x_2+3x_3+50} - \frac{x_1+2x_2+4x_3+50}{5x_2+4x_3+50} - \frac{3x_1+5x_2+3x_3+50}{5x_1+5x_2+4x_3+50} + \frac{3x_1+4x_2+50}{3x_1+5x_2+4x_3+50} \\ \text{s.t.} & 6x_1 + 3x_2 + 3x_3 \leq 10, \\ & 10x_1 + 3x_2 + 8x_3 \leq 10, \\ & x_1, x_2, x_3 \geq 0, \end{cases}$$

where x_1 and x_3 are integers.

Example 3. (Shen et al. [26])

$$\begin{cases} \min & \frac{4x_1-3x_2+4}{-2x_1+x_2+3} + \frac{-x_1+2x_2+2}{3x_1-4x_2+5} \\ \text{s.t.} & x_1 + x_2 \leq 1.5, \\ & x_1 \leq x_2, \\ & 0 \leq x_1, x_2 \leq 1, \end{cases}$$

where x_1 is an integer.
Example 4. (Shen & Lu [25], Gao & Jin [11])

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

where x_1, x_2, and x_3 are integers.
Example 5. (Shen & Lu [25])

$$\begin{cases} \max & \frac{x_1+2x_2+4x_3+50}{5x_2+4x_3+50} + \frac{x_1+2x_2+5x_3+50}{x_1+5x_2+5x_3+50} + \frac{3x_1+4x_3+50}{4x_1+4x_2+5x_3+50} + \frac{4x_1+3x_2+3x_3+50}{3x_2+3x_3+50} \\ \text{s.t.} & 9x_1 + 7x_2 + 3x_3 \leq 10, \\ & x_1 + 6x_2 + 3x_3 \leq 10, \\ & 2x_1 + x_2 + 5x_3 \leq 10, \\ & 5x_1 + 9x_2 + 2x_3 \leq 10, \\ & x_1, x_2, x_3 \geq 0, \end{cases}$$

where x_2 and x_3 are integers.
Example 6. (Shen & Lu [25], Gao & Jin [11])

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

where x_1 and x_3 are integers.
Example 7. (Shen & Lu [25])

$$\begin{cases} \max & \frac{4x_1+2x_2+4x_3+50}{5x_1+4x_2+3x_3+50} + \frac{3x_1+5x_2+3x_3+50}{3x_1+4x_2+5x_3+50} + \frac{3x_1+4x_2+50}{4x_1+3x_2+2x_3+50} \\ \text{s.t.} & 6x_1 + 3x_2 + 3x_3 \leq 10, \\ & 10x_1 + 3x_2 + 8x_3 \leq 10, \\ & x_1, x_2, x_3 \geq 0, \end{cases}$$

where x_1 and x_3 are integers.

Example 8. (Jiao et al. [14], Jiao & Liu [13], Shen & Wang [24])

$$\begin{cases} \max & \frac{4x_1+3x_2+3x_3+50}{3x_2+2x_3+50} + \frac{x_1+2x_2+5x_3+50}{x_1+5x_2+5x_3+50} + \frac{x_1+2x_2+4x_3+50}{5x_2+4x_3+50} + \frac{3x_1+4x_2+50}{4x_1+4x_2+5x_3+50} \\ \text{s.t.} & 9x_1 + 7x_2 + 3x_3 \leq 10, \\ & 5x_1 + 9x_2 + 2x_3 \leq 10, \\ & 2x_1 + x_2 + 5x_3 \leq 10, \\ & x_1 + 6x_2 + 3x_3 \leq 10, \\ & x_1, x_2, x_3 \geq 0, \end{cases}$$

where x_2 and x_3 are integers.
Example 9. (Shen & Wang [24], Shi [27])

$$\begin{cases} \max & \frac{37x_1+73x_2+13}{13x_1+13x_2+13} + \frac{13x_1+13x_2+13}{63x_1-18x_2+39} + \frac{13x_1+26x_2+13}{-37x_2-73x_3-13} + \frac{63x_1-18x_2+39}{-13x_1-26x_2-13} \\ \text{s.t.} & 1.5 \leq x_1 \leq 3, \\ & 5x_1 - 3x_2 = 3, \end{cases}$$

where x_1 and x_2 are integers.
Example 10. (Shi [27])

$$\begin{cases} \max & \frac{x_1+2x_2+4x_3+50}{5x_2+4x_3+50} + \frac{x_1+2x_2+5x_3+50}{x_1+5x_2+5x_3+50} + \frac{3x_1+4x_3+50}{4x_1+4x_2+5x_3+50} + \frac{4x_1+3x_2+3x_3+50}{3x_2+3x_3+50} \\ \text{s.t.} & 9x_1 + 7x_2 + 3x_3 \geq 10, \\ & x_1 + 6x_2 + 2x_3 \leq 10, \\ & 2x_1 + x_2 + 5x_3 \leq 10, \\ & x_1, x_2, x_3 \geq 0, \end{cases}$$

where x_1, x_2, and x_3 are integers.
Example 11.

$$\begin{cases} \min & \sum_{i=1}^{p} \frac{\hat{c}_i^\top x + \hat{f}_i}{\hat{e}_i^\top x + \hat{g}_i} \\ \text{s.t.} & Ax \leq b, \\ & x \geq 0, \end{cases}$$

where $\hat{f}_i, \hat{g}_i \in \mathbb{R}$, $i \in \Theta$; $\hat{e}_i, \hat{c}_i \in \mathbb{R}^n$, $i \in \Theta$; $b \in \mathbb{R}^m$; $A \in \mathbb{R}^{m\times n}$; each element of $\hat{c}_i$ and $\hat{e}_i$ is randomly generated in $[-0.1, 0.1]$; every element in the vector b is 10; every elementary substance of A is randomly created from $[0.01, 1]$; and $\hat{f}_i$ and $\hat{g}_i$ satisfy $\hat{c}_i^\top x + \hat{f}_i > 0$ and $\hat{e}_i^\top x + \hat{g}_i > 0, i \in \Theta$.

For all deterministic Examples 1–10, from Table 1, compared with the known global optimization solver BARON, with almost the same computational efficiency, the OSBRBA can obtain almost the same optimum solutions.

For the large-scale randomly generated Example 11, from Table 2 and Figure 1, compared with the known global optimization solver BARON, the OSBRBA clearly has high computational efficiency.

Furthermore, to demonstrate the efficiency of the algorithm proposed in this paper, for the random generated test Example 11, letting $p = 2$, $m = 100$, and $n = 500$, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000, setting $\epsilon = 10^{-4}$, and setting all variables x as continuous variables, we used the proposed algorithm in this paper and the algorithm of Li et al. [37] to solve 10 randomly generated test problems, and recorded their numerical comparison results in Table 3. Here, it should be noted that the selection and symbol representation of experimental parameters in Table 3 are the same as those in Table 2.

Table 3. Numerical comparisons between our algorithm and the algorithms of Li et al. [37] on Problem 1 with $p = 2$ and $m = 100$, and the given termination error $\epsilon = 10^{-4}$.

n	Algorithms	Optimum	Iter.			Time		
			min.	ave.	max.	min.	ave.	max.
500	Algorithm of [37]	1.9540	57	58.2	62	3.44	3.66	3.73
	Ours	1.9540	12	14	17	1.64	1.78	1.94
1000	Algorithm of [37]	1.9535	52	58.4	62	8.92	9.76	10.51
	Ours	1.9535	9	13.4	15	3.45	6.30	8.10
1500	Algorithm of [37]	1.9499	56	58.5	62	19.87	20.68	22.13
	Ours	1.9499	12	13.8	17	11.27	13.88	17.85
2000	Algorithm of [37]	1.9484	55	57.9	60	32.56	34.16	36.73
	Ours	1.9484	12	13.9	15	18.95	23.17	27.55
2500	Algorithm of [37]	1.9494	58	60	63	50.60	53.01	57.07
	Ours	1.9494	14	15	16	35.44	40.31	46.93
3000	Algorithm of [37]	1.9474	58	59.2	61	70.56	72.45	76.32
	Ours	1.9474	14	14.5	15	47.36	50.75	54.97
3500	Algorithm of [37]	1.9464	58	60.4	69	100.87	104.07	109.83
	Ours	1.9464	14	15	19	69.07	79.13	110.59
4000	Algorithm of [37]	1.9459	58	59.6	64	119.79	126.87	132.54
	Ours	1.9459	14	14.7	17	82.87	94.22	117.07
4500	Algorithm of [37]	1.9451	59	61.8	68	157.68	166.02	175.37
	Ours	1.9451	14	16.1	19	122.03	142.75	180.38
5000	Algorithm of [37]	1.9450	56	61.4	69	180.96	195.27	218.74
	Ours	1.9450	13	16.2	20	115.29	168.60	227.45

From the numerical results in Table 3, compared with the algorithm of Li et al. [37], the algorithm proposed in this paper not only obtains the same optimal value and solution, but also has higher computational efficiency.

5. Conclusions

This paper studies the large-scale linear fractional program problem and presents an effective solving algorithm. In this algorithm, by applying the Charnes-Cooper transformation technique, the original problem (LFP) is first equivalently reformulated as problem (EP). Subsequently, by applying the novel linearizing method, the mixed-integer linear relaxation problem of problem (EP) is established. In addition, the theory of the proposed OSBRBA is analyzed. The OSBRBA can achieve a globally ϵ-optimum solution after at most finite iterations. Numerical experimental results show that the OSBRBA is superior to the BARON solver in globally finding the optimum solution of problem (LFP) with large-scale variables. However, in the worst-case scenario, the OSBRBA exhibits an exponential increase in the maximum number of iterations as the number of fractional terms p increases, which also indicates that the OSBRBA may not be efficient for problem (LFP) with many fractional terms. In future work, a meaningful direction is to develop a more effective algorithm for addressing the mixed-integer nonlinear fractional program problems using other branch techniques and bound schemes.

Author contributions

Zhiguo Ge: formal analysis, investigation, resources, methodology, writing–original draft, validation, data curation, and funding acquisition; Hongwei Jiao: writing review and editing, software, conceptualization, supervision, project administration, and formal funding acquisition. All authors have read and agreed to the published version of the manuscript.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by the Key Scientific and Technological Research Projects in Henan Province (232102211085).

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. E. B. Bajalinov, *Linear-fractional programming theory, methods, applications and software*, New York: Springer, 2003. http://doi.org/10.1007/978-1-4419-9174-4
2. H. P. Benson, Global optimization algorithm for the nonlinear sum of ratios problem, *J. Optim. Theory Appl.*, **112** (2002), 1–29. https://doi.org/10.1023/A:1013072027218
3. H. P. Benson, A simplicial branch and bound duality-bounds algorithm for the linear sum-of-ratios problem, *Eur. J. Oper. Res.*, **182** (2007), 597–611. https://doi.org/10.1016/j.ejor.2006.08.036
4. H. P. Benson, Branch-and-bound outer approximation algorithm for sum-of-ratios fractional programs, *J. Optim. Theory Appl.*, **146** (2010), 1–18. https://doi.org/10.1007/s10957-010-9647-8
5. H. P. Benson, Using concave envelopes to globally solve the nonlinear sum of ratios problem, *J. Glob. Optim.*, **22** (2002), 343–364. https://doi.org/10.1023/A:1013869015288
6. J. G. Carlsson, J. M. Shi, A linear relaxation algorithm for solving the sum-of-linear-ratios problem with lower dimension, *Oper. Res. Lett.*, **41** (2013), 381–389. https://doi.org/10.1016/j.orl.2013.04.005

7. Y. Chaiblaine, M. Moulaï, An exact method for solving the integer sum of linear ratios problem, *Optimization*, **73** (2024), 461–479. https://doi.org/10.1080/02331934.2022.2112190
8. M. C. Dorneich, N. V. Sahinidis, Global optimization algorithms for chip layout and compaction, *Eng. Optimiz.*, **25** (1995), 131–154. https://doi.org/10.1080/03052159508941259
9. J. E. Falk, S. W. Palocsay, Image space analysis of generalized fractional programs, *J. Glob. Optim.*, **4** (1994), 63–88. https://doi.org/10.1007/BF01096535
10. R. W. Freund, F. Jarre, Solving the sum-of-ratios problem by an interior-point method, *J. Glob. Optim.*, **19** (2001), 83–102. https://doi.org/10.1023/A:1008316327038
11. Y. L. Gao, S. Q. Jin, A global optimization algorithm for sum of linear ratios problem, *J. Appl. Math.*, **2013** (2013), 276245. https://doi.org/10.1155/2013/276245
12. H. W. Jiao, B. B. Li, Y. L. Shang, An outer space approach to tackle generalized affine fractional program problems, *J. Optim. Theory Appl.*, **201** (2024), 1–35. https://doi.org/10.1007/s10957-023-02368-0
13. H. W. Jiao, S.-Y. Liu, A practicable branch and bound algorithm for sum of linear ratios problem, *Eur. J. Oper. Res.*, **243** (2015), 723–730. https://doi.org/10.1016/j.ejor.2015.01.039
14. H. W. Jiao, S. Y. Liu, J. B. Yin, Y. F. Zhao, Outcome space range reduction method for global optimization of sum of affine ratios problem, *Open Math.*, **14** (2016), 736–746. https://doi.org/10.1515/math-2016-0058
15. H. W. Jiao, Y. L. Shang, R. J. Chen, A potential practical algorithm for minimizing the sum of affine fractional functions, *Optimization*, **72** (2023), 1577–1607. https://doi.org/10.1080/02331934.2022.2032051
16. H. W. Jiao, Y. L. Shang, Two-level linear relaxation method for generalized linear fractional programming, *J. Oper. Res. Soc. China*, **11** (2023), 569–594. https://doi.org/10.1007/s40305-021-00375-4
17. H. W. Jiao, Y. L. Shang, W. J. Wang Solving generalized polynomial problem by using new affine relaxed technique, *Int. J. Comput. Math.*, **99** (2022), 309–331. https://doi.org/10.1080/00207160.2021.1909727
18. E. Khalil, P. Le Bodic, L. Song, G. Nemhauser, B. Dilkina, Learning to branch in mixed integer programming, *Proceedings of the AAAI conference on artificial intelligence*, **30** (2016). https://doi.org/10.1609/aaai.v30i1.10080
19. A. Khajavirad, N. V. Sahinidis, A hybrid LP/NLP paradigm for global optimization relaxations, *Math. Prog. Comp.*, **10** (2018), 383–421. https://doi.org/10.1007/s12532-018-0138-5
20. A. A. Khan, R. S. Adve, W. Yu, Optimizing downlink resource allocation in multiuser MIMO networks via fractional programming and the hungarian algorithm, *IEEE T. Wirel. Commun.*, **19** (2020), 5162–5175. https://doi.org/10.1109/TWC.2020.2990176
21. H. Konno, H. Watanabe, Bond portfolio optimization problems and their applications to index tracking: A partial optimization approach, *J. Oper. Res. Soc. Jpn.*, **39** (1996), 295–306. https://doi.org/10.15807/jorsj.39.295
22. T. Kuno, A branch-and-bound algorithm for maximizing the sum of several linear ratios, *J. Glob. Optim.*, **22** (2002), 155–174. https://doi.org/10.1023/A:1013807129844

23. N. T. H. Phuong, H. Tuy, A unified monotonic approach to generalized linear fractional programming, *J. Glob. Optim.*, **26** (2003), 229–259. https://doi.org/10.1023/A:1023274721632

24. P. P. Shen, C. F. Wang, Global optimization for sum of linear ratios problem with coefficients, *Appl. Math. Comput.*, **176** (2006), 219–229. https://doi.org/10.1016/j.amc.2005.09.047

25. P. P. Shen, T. Lu, Regional division and reduction algorithm for minimizing the sum of linear fractional functions, *J. Inequal. Appl.*, **2018** (2018), 63. https://doi.org/10.1186/s13660-018-1651-9

26. P. P. Shen, B. D. Huang, L. F. Wang, Range division and linearization algorithm for a class of linear ratios optimization problems, *J. Comput. Appl. Math.*, **350** (2019), 324–342. https://doi.org/10.1016/j.cam.2018.10.038

27. Y. H. Shi, Global optimization for sum-of-ratios problems, Master Thesis, Henan Normal University, 2011.

28. I. M. Stancu-Minasian, *Fractional programming: theory, methods and applications*, Dordrecht: Springer, 1997. https://doi.org/10.1007/978-94-009-0035-6

29. J. P. Vielma, S. Ahmed, G. Nemhauser, Mixed-integer models for nonseparable piecewise-linear optimization: unifying framework and extensions, *Oper. Res.*, **58** (2009), 303–315. https://doi.org/10.1287/opre.1090.0721

30. L. A. Wolsey, Mixed integer programming, In: *Wiley encyclopedia of computer science and engineering*, New York: Wiley, 2008. https://doi.org/10.1002/9780470050118.ecse244

31. B. Zhang, Y. L. Gao, X. Liu, X. L. Huang, A new deterministic global computing algorithm for solving a kind of linear fractional programming, *Optimization*, **72** (2023), 1485–1513. https://doi.org/10.1080/02331934.2022.2027940

32. X. F. Zhang, R. Liu, J. X. Ren, Q. L. Gui, Adaptive fractional image enhancement algorithm based on rough set and particle swarm optimization, *Fractal Fract.*, **6** (2022), 100. https://doi.org/10.3390/fractalfract6020100

33. J.-X. Zhang, G.-H. Yang, Low-complexity tracking control of strict-feedback systems with unknown control directions, *IEEE T. Automat. Contr.*, **64** (2019), 5175–5182. https://doi.org/10.1109/TAC.2019.2910738

34. J.-X. Zhang, Y.-Q. Liu, T. Y. Chai, Singularity-free low-complexity fault-tolerant prescribed performance control for spacecraft attitude stabilization, *IEEE T. Autom. Sci. Eng.*, **22** (2025), 15408–15419. https://doi.org/10.1109/TASE.2025.3569566

35. F. H. Mathis, L. J. Mathis, A nonlinear programming algorithm for hospital management, *SIAM Rev.*, **37** (1995), 230–234. https://doi.org/10.1137/1037046

36. H. W. Li, L. F. Wang, Y. F. Zhao, Global optimization algorithm for a class of linear ratios *AIMS Mathematics*, **9** (2024), 16376–16391. https://doi.org/10.3934/math.2024793

37. H. W. Li, Y. F. Feng, H. W. Jiao, Y. L. Shang, A novel algorithm for solving sum of several affine fractional functions, *AIMS Mathematics*, **8** (2023), 9247–9264. https://doi.org/10.3934/math.2023464

38. Z. S. Hou, S. Y. Liu, An efficient image space branch-reduction-bound algorithm to globally solve generalized fractional programming problems for *J. Comput. Appl. Math.*, **451** (2024), 116070. https://doi.org/10.1016/j.cam.2024.116070
39. Z. S. Hou, S. Y. Liu, A spatial branch-reduction-bound algorithm for solving generalized linear fractional problems globally, *Chaos Soliton. Fract.*, **176** (2023), 114144. https://doi.org/10.1016/j.chaos.2023.114144
40. Z. S. Hou, S. Y. Liu, An accelerating outer space algorithm for globally solving generalized linear multiplicative problems, *Numer. Algor.*, **94** (2023), 877–904. https://doi.org/10.1007/s11075-023-01523-y
41. H. W. Jiao, B. B. Li, W. Q. Yang, A criterion-space branch-reduction-bound algorithm for solving generalized multiplicative problems, *J. Glob. Optim.*, **89** (2024), 597–632. https://doi.org/10.1007/s10898-023-01358-w
42. H. W. Jiao, J. Q. Ma, Optimizing generalized linear fractional program using the image space branch-reduction-bound scheme, *Optimization*, **74** (2025), 1–32. https://doi.org/10.1080/02331934.2023.2253816
43. H. W. Jiao, J. Q. Ma, Y. L. Shang, Reduced outer space algorithm for globally computing affine sum-of-ratios problems, *Asia Pac. J. Oper. Res.*, **42** (2025), 2450015. https://doi.org/10.1142/S0217595924500155