Journal of Computational Mathematics
Vol.xx, No.x, 2024, 1–26.

http://www.global-sci.org/jcm
doi:10.4208/jcm.2203-m2021-0085

# IMAGE SPACE BRANCH-REDUCTION-BOUND ALGORITHM FOR GLOBALLY SOLVING THE SUM OF AFFINE RATIOS PROBLEM*

Hongwei Jiao[1)]
*School of Mathematical Science, Henan Institute of Science and Technology, Xinxiang 453003, China*
*Email: jiaohongwei@126.com*
Youlin Shang
*School of Mathematics and Statistics, Henan University of Science and Technology, Luoyang 471023, China*
*Email: mathshang@sina.com*

**Abstract**

This article presents an image space branch-reduction-bound algorithm for globally solving the sum of affine ratios problem. The algorithm works by solving its equivalent problem, and by using convex hull and concave hull approximation of bilinear function, we can construct the affine relaxation problem of the equivalent problem, which can be used to compute the lower bounds during the branch-and-bound search. By subsequently refining the initial image space rectangle and solving a series of affine relaxation problems, the proposed algorithm is convergent to the global optima of the primal problem. For improving the convergence speed, an image space region reducing method is adopted for compressing the investigated image space rectangle. In addition, the global convergence of the algorithm is proved, and its computational complexity is analyzed. Finally, comparing with some existing methods, numerical results indicate that the algorithm has better computational performance.

*Mathematics subject classification:* 90C26, 90C32.
*Key words:* Sum of affine ratios, Global optimization, Affine relaxation problem, Branch-reduction-bound, Computational complexity.

## 1. Introduction

Consider the following sum of affine ratios problem defined by

$$\begin{aligned} \min G(x) &= \sum_{i=1}^{p} \frac{c_i^T x + f_i}{d_i^T x + g_i} \\ \text{s.t.} \quad & x \in D = \{x \in R^n \mid Ax \le b\}, \end{aligned} \tag{1.1}$$

where $p \ge 2$, $A$ is an $m \times n$ order real matrix, $b$ is an $m$ dimensional column vector, $D$ is a nonempty bounded polyhedron set, $c_i^T x + f_i$ and $d_i^T x + g_i$ are all affine functions defined over $D$, and for any $x \in D$, and for each ratio, the denominator $d_i^T x + g_i \ne 0$.

From 1980s, the problem (1.1) has attracted a growing attention of many practitioners and researchers. From the perspective of applications, the general form and special form of the problem (1.1) have a wide range of applications in information theory, optical processing of in-

* Received March 19, 2021 / Revised version received June 17, 2021 / Accepted March 21, 2022 / Published online January 08, 2024 /
1) Corresponding author

formation, macroeconomic planning, cluster analysis, loading problem, optimum transportation plan, game theory problem, optimal paths in graph theory, computer vision, location problem, and so on, see [10–13,17,22,34,35]. From a theoretical point of view, since the objective function of the problem (1.1) is neither quasiconvex nor quasiconcave, that is to say, it is a nonconvex global optimization problem, which usually possesses many local optimal solutions that are not globally optimal. Therefore, there are some important theoretical and computational difficulties in solving the problem (1.1).

Especially in the past 30 years, many algorithms have been proposed and developed for globally solving the problem (1.1). For example, Konno *et al.* [18] and Cambini *et al.* [2] propose separately a parametric simplex algorithm and a parametric linear programming algorithm for solving the problem (1.1) with only two affine ratios terms, and with that the numerator and denominator of each affine ratio are positive over the feasible region. By solving the corresponding equivalent concave minimization problem, Konno and Yamashita [19] propose an outer approximation algorithm for solving the problem (1.1) with the assumptions that all numerators must be nonnegative and all denominators must be positive over the feasible region. By iteratively searching the image space of affine ratios, Falk and Palocsay [4] first propose an image space analysis method for solving the sum of affine ratios problem. Based on the monotonic optimization theory, and by solving a parametric linear programming problem at each iteration, Phuong and Tuy [24] propose a unified monotonic optimization algorithm for globally solving the generalized affine fractional programming problem which includes the sum of affine ratios problem.

In addition, a large number of branch-and-bound algorithms have been also proposed for solving the problem (1.1). For example, Quesada and Grossman [25], and Konno and Fukaishi [16] propose two rectangular branch-and-bound algorithms for solving the problem (1.1) with the assumptions that each numerator must be nonnegative and each denominator must be positive over the feasible region. By using trapezoidal partition and concave envelope bound, Kuno [20,21] propose two trapezoidal branch-and-bound algorithms for globally solving the problem (1.1) with the assumptions that all numerators and denominators of affine ratios must be positive over the feasible region. By utilizing simplicial partition and Lagrangian duality bound technique, Benson [1] presents a simplicial branch-and-bound duality bound method for globally solving the problem (1.1). By replacing each denominator with the upper bound of its interval, to construct the affine relaxation of the original problem, Ji and Zhang [7] propose a branch-and-bound algorithm for solving the problem (1.1) with that all numerators and denominators of ratios must be positive over the feasible region. Recently, by utilizing the new two-level affine relaxation technique to construct the linear relaxation problem, Jiao *et al.* [14] present a rectangle branch-and-bound algorithm for globally solving the generalized affine multiplicative programming problem which includes the sum of affine ratios problem; by using region division and reduction techniques, Shen *et al.* [27,29,31] propose three different polynomial-time approximation algorithms for special cases of the problem (1.1) and the generalized affine fractional programming problem, respectively; by using the well-known concave envelope and convex underestimation method to derive the relaxation problem, Shen *et al.* [28] present a simplicial branch-and-bound algorithm for globally solving the sum of convex-convex ratios problem. For an excellent review of fractional programming algorithms, we can refer to Schaible and Shi [26] and Stancu-Minasian [36].

In this paper, we will present a practical image space branch-and-bound algorithm for globally solving the problem (1.1), where the branch-and-bound search takes place in the image

space $R^p$ of affine ratios. To this end, first of all, we transform the problem (1.1) into an equivalent problem (2.4), which has the same global optimum solution as the problem (1.1). Next, by utilizing convex hull and concave hull approximation of bilinear function, we construct the affine relaxation of nonlinear equality constraints in the problem (2.4), so that the affine relaxation problem of the problem (2.4) can be established, and which can be employed to compute the lower bound of the global minimum of the problem (2.4). In addition, a new image space region reducing method is proposed for improving the computational efficiency of the algorithm. In this light, an image space branch-reduction-bound algorithm is established by combining the affine relaxation problem and the branching rule with the image space region reducing method. The algorithm iteratively subdivides and successively searches the refined image space rectangles of $R^p$, and solves a series of affine relaxation problems, feasible solutions of the problem (1.1) can be found, and when the algorithm is infinite, any of accumulation points of the feasible solutions sequence is the global optimal solution of the problem (1.1).

Comparing with the algorithms reviewed above, the presented algorithm in this paper has the following several potential practical and computational superiorities. First of all, the presented algorithm does not impose any special sign restrictions on all numerators and denominators of affine ratios, the only assumption is that all denominators of affine ratios are nonzero over the feasible region, but some reviewed methods can solve only particular cases. Secondly, the branch-and-bound search occurs in the image space $R^p$ of affine ratios rather than the space $R^n$ of decision variable $x$, where $p$ is the number of affine ratios. Since the dimension number $n$ of decision variable $x$ usually far exceeds the number $p$ of affine ratios, by analyzing the computational complexity of the algorithm, it can be concluded that this will reduce the maximum number of iterations of the algorithm, so that this significantly shortens the computational time required to find a global optimum solution of the problem (1.1), it is to say, the image space branch-reduction-bound algorithm economizes the required computation. Thirdly, during the branch-and-bound searching process, all subproblems which need to be solved are linear programming problems, which can be solved by any effective linear programming solvers. Fourthly, the new image space region reducing technique is presented for enhancing the computational efficiency of the algorithm. Finally, numerical experimental results are given to validate that the presented algorithm can solve all test problems for finding their approximate global optimal solutions within the given tolerance with better computational performance.

The remaining sections of the paper are organized as follows. In Section 2, the problem (1.1) is transformed into the equivalent problems (2.3) and (2.4). Next, in Section 3, we derive the affine relaxation problem of the problem (2.4). In Section 4, the approximation of affine relaxation is derived by successive refinement of the image space region. In Section 5, an image space branch-reduction-bound algorithm is designed, the global convergence of the algorithm is proved, and the computational complexity of the algorithm is analyzed. In Section 6, numerical experimental results are given to verify the computational advantages of the presented algorithm. In Section 7, comparing the existing relaxation methods, the superiority of the algorithm is implied. Further improvement in the future are explored and discussed, and the improvement ideas are briefly described in Section 8. Finally, some conclusions are drawn in Section 9.

## 2. Preliminary Results

To find a global optimal solution of the problem (1.1), we need to transform the problem (1.1) into the equivalent problems (2.3) and (2.4). Next, the fundamental assignment is to

globally solve the problem (2.4). To this end, for each $i = 1, 2, \ldots, p$, we need to compute the lower bound

$$L_i^0 = \min_{x \in D} \frac{c_i^T x + f_i}{d_i^T x + g_i},$$

and the upper bound

$$U_i^0 = \max_{x \in D} \frac{c_i^T x + f_i}{d_i^T x + g_i}$$

of affine ratio $(c_i^T x + f_i)/(d_i^T x + g_i)$. Clearly, first of all, we consider solving the following linear fractional programming problems:

$$L_i^0 = \min_{x \in D} \frac{c_i^T x + f_i}{d_i^T x + g_i}, \quad i = 1, 2, \ldots, p. \tag{2.1}$$

Notice that each affine ratio $(c_i^T x + f_i)/(d_i^T x + g_i)$ is quasi-convex, so it can attain the minimum value at some vertex of $D$. Since $d_i^T x + g_i \neq 0$, and without loss of generality, we can assume that $d_i^T x + g_i > 0$. Therefore, in order to solve the problem (2.1), for each $i \in \{1, 2, \ldots, p\}$, by introducing new variables $t_i = 1/(d_i^T x + g_i)$ and $z = t_i x$, then the above problem (2.1) can be reformulated as the following linear programming problems:

$$\begin{aligned} \min \quad & c_i^T z + f_i t_i \\ \text{s.t.} \quad & d_i^T z + g_i t_i = 1, \\ & Az \leq b t_i, \quad t_i > 0, \quad i = 1, 2, \ldots, p. \end{aligned} \tag{2.2}$$

The key equivalence for the problems (2.1) and (2.2) is given by the following theorem.

**Theorem 2.1 ([27]).** *$x^* \in R^n$ is a global optimum solution of the problem* (2.1) *if and only if $(z^*, t_i^*) \in R^{n+1}$ is a global optimum solution of the problem* (2.2) *with $z^* = t_i^* x^*$. In addition, the problems* (2.1) *and* (2.2) *have the equal global optimal value.*

*Proof.* The conclusion of the theorem is easily deduced by the well-known Charnes-Cooper transformation [3], thus the proof is omitted. □

Theorem 2.1 shows that each $L_i^0$ can be obtained by solving a linear programming problem (2.2). Similarly, we can give the upper bound

$$U_i^0 = \max_{x \in D} \frac{c_i^T x + f_i}{d_i^T x + g_i}$$

of each affine ratio $(c_i^T x + f_i)/(d_i^T x + g_i)$. Without loss of generality, we can obtain the initial image space rectangle

$$\Lambda^0 = \left\{ s \in R^p \mid L_i^0 \leq s_i \leq U_i^0,\ i = 1, 2, \ldots, p \right\},$$

so that we can get the equivalent problem (2.3) of the problem (1.1) as follows:

$$\begin{aligned} \min \quad & F(x, s) = \sum_{i=1}^{p} s_i \\ \text{s.t.} \quad & s_i = \frac{c_i^T x + f_i}{d_i^T x + g_i}, \quad i = 1, 2, \ldots, p, \\ & x \in D, \quad s \in \Lambda^0. \end{aligned} \tag{2.3}$$

It is obvious that the feasible region

$$Z = \left\{ (x,s) \in R^{n+p} \,\middle|\, \frac{c_i^T x + f_i}{d_i^T x + g_i} - s_i = 0,\ i = 1,2,\ldots,p,\ x \in D,\ s \in \Lambda^0 \right\}$$

of the problem (2.3) is a nonempty compact convex set, and the feasible region $Z \neq \emptyset$ if and only if $D \neq \emptyset$. The important equivalence conclusions of the problems (1.1) and (2.3) are given by the following theorem.

**Theorem 2.2.** *Assume that $x^*$ is a global optimum solution for the problem* (1.1)*, then $(x^*, s^*)$ is a global optimum solution for the problem* (2.3)*, where*

$$s_i^* = \frac{c_i^T x^* + f_i}{d_i^T x^* + g_i}, \quad i = 1,2,\ldots,p.$$

*Conversely, assume that $(x^*, s^*)$ is a global optimum solution for the problem* (2.3) *with that*

$$s_i^* = \frac{c_i^T x^* + f_i}{d_i^T x^* + g_i}, \quad i = 1,2,\ldots,p,$$

*then $x^*$ is a global optimum solution for the problem* (1.1)*.*

*Proof.* The conclusions of the theorem can be easily obtained, so the proof is omitted. □

By the equivalence conclusions of Theorem 2.2, for globally solving the problem (1.1), we can substitute for solving its equivalent problem (2.3). In addition, the problems (1.1) and (2.3) have the same global minimum value.

By the denominator $d_i^T x + g_i \neq 0$ of each ratio, the problem (2.3) can be rewritten into the following equivalent problem (2.4), which has the same global optimal solution and optimal value as the problem (2.3):

$$\begin{aligned} \min \quad & F(x,s) = \sum_{i=1}^{p} s_i \\ \text{s.t.} \quad & s_i\big(d_i^T x + g_i\big) = c_i^T x + f_i, \quad i = 1,2,\ldots,p, \\ & x \in D, \quad s \in \Lambda^0. \end{aligned} \tag{2.4}$$

It is obvious that the feasible region

$$Z = \big\{ (x,s) \in R^{n+p} \,\big|\, s_i\big(d_i^T x + g_i\big) = c_i^T x + f_i,\ i = 1,2,\ldots,p,\ x \in D,\ s \in \Lambda^0 \big\}$$

of the problem (2.3) is a nonempty compact convex set, and the feasible region $Z \neq \emptyset$ if and only if $D \neq \emptyset$.

## 3. Deduction of Affine Relaxation

To globally solve the problem (2.4), we need to construct its affine relaxation problem, which can offer the reliable lower bounds in the branch-and-bound search. By extending the Reformulation-Linearization Technique of bilinear function proposed by Sherali and Adams [32], we give an affine relaxation approach for the problem (2.4). The detailed deriving process of the affine relaxation problem of the problem (2.4) is given as follows.

Without loss of generality, for each $i = 1, 2, \ldots, p$, solving the linear programming problems $\alpha_i = \min_{x\in D} d_i^T x + g_i$ and $\beta_i = \max_{x\in D} d_i^T x + g_i$, we can get the lower bound and the upper bound of the denominator of each ratio. First of all, for any $\Lambda = \{s \in R^p \mid L_i \le s_i \le U_i, i = 1, 2, \ldots, p\} \subseteq \Lambda^0$, we define the set

$$RC_i = \big\{(x, s_i) \in R^{n+1} \mid \alpha_i \le d_i^T x + g_i \le \beta_i,\ L_i \le s_i \le U_i\big\}, \quad i = 1, 2, \ldots, p.$$

For any $(x, s_i) \in RC_i$ with $s_i \neq 0$, by the definition of the set $RC_i$, we have

$$d_i^T x + g_i - \alpha_i \ge 0, \quad s_i - L_i \ge 0, \quad d_i^T x + g_i - \beta_i \le 0, \quad s_i - U_i \le 0.$$

Therefore, we can get that

$$\big(d_i^T x + g_i - \alpha_i\big)(s_i - L_i) \ge 0,$$
$$\big(d_i^T x + g_i - \beta_i\big)(s_i - U_i) \ge 0,$$

i.e.

$$s_i\big(d_i^T x + g_i\big) \ge \alpha_i s_i + L_i\big(d_i^T x + g_i\big) - \alpha_i L_i,$$
$$s_i\big(d_i^T x + g_i\big) \ge \beta_i s_i + U_i\big(d_i^T x + g_i\big) - \beta_i U_i.$$

Hence, we have

$$\max\big\{\alpha_i s_i + L_i\big(d_i^T x + g_i\big) - \alpha_i L_i, \beta_i s_i + U_i\big(d_i^T x + g_i\big) - \beta_i U_i\big\} \le s_i\big(d_i^T x + g_i\big).$$

Similarly, for any $(x, s_i) \in RC_i$ with $s_i \neq 0$, by the definition of the set $RC_i$, we have

$$d_i^T x + g_i - \alpha_i \ge 0, \quad s_i - U_i \le 0, \quad d_i^T x + g_i - \beta_i \le 0, \quad s_i - L_i \ge 0.$$

Therefore, we can get that

$$\big(d_i^T x + g_i - \alpha_i\big)(s_i - U_i) \le 0,$$
$$\big(d_i^T x + g_i - \beta_i\big)(s_i - L_i) \le 0,$$

i.e.

$$s_i\big(d_i^T x + g_i\big) \le \alpha_i s_i + U_i\big(d_i^T x + g_i\big) - \alpha_i U_i,$$
$$s_i\big(d_i^T x + g_i\big) \le \beta_i s_i + L_i\big(d_i^T x + g_i\big) - \beta_i L_i.$$

Hence, we have

$$\min\big\{\alpha_i s_i + U_i\big(d_i^T x + g_i\big) - \alpha_i U_i, \beta_i s_i + L_i\big(d_i^T x + g_i\big) - \beta_i L_i\big\} \ge s_i\big(d_i^T x + g_i\big).$$

Consequently, we can construct the affine relaxation problem (3.1) of the problem (2.4) over $\Lambda$ as follows, which is a linear programming problem:

$$\begin{aligned}
\min\ & F(x, s) = \sum_{i=1}^{p} s_i \\
\text{s.t.}\ & \alpha_i s_i + L_i\big(d_i^T x + g_i\big) - \alpha_i L_i \le c_i^T x + f_i, \quad i = 1, 2, \ldots, p, \\
& \beta_i s_i + U_i\big(d_i^T x + g_i\big) - \beta_i U_i \le c_i^T x + f_i, \quad i = 1, 2, \ldots, p, \\
& \alpha_i s_i + U_i\big(d_i^T x + g_i\big) - \alpha_i U_i \ge c_i^T x + f_i, \quad i = 1, 2, \ldots, p, \\
& \beta_i s_i + L_i\big(d_i^T x + g_i\big) - \beta_i L_i \ge c_i^T x + f_i, \quad i = 1, 2, \ldots, p, \\
& x \in D, \quad s \in \Lambda.
\end{aligned} \tag{3.1}$$

For any $\Lambda = \{s \in R^p \mid L_i \le s_i \le U_i, i = 1, 2, \ldots, p\} \subseteq \Lambda^0$, by the deriving method of the above affine relaxation problem, all feasible points of the problem (2.4) over sub-rectangle $\Lambda$ are all feasible to the problem (3.1), and the optimal value of the problem (3.1) is not more than that of the problem (2.4) over $\Lambda$. Therefore, the optimal value of the problem (3.1) can provide a valid lower bound for that of the problem (2.4) over $\Lambda$ during the image space branch-and-bound search.

## 4. Approximation of Affine Relaxation

In this section, we will derive the approximation of affine relaxation, which guarantees the global convergence of the image space branch-and-bound algorithm.

Without losing generality, for any $\Lambda = \{s \in R^p \mid L_i \le s_i \le U_i,\ i = 1, 2, \ldots, p\} \subseteq \Lambda^0$, for any $(x, s_i) \in RC_i = \{(x, s_i) \in R^{n+1} \mid \alpha_i \le d_i^T x + g_i \le \beta_i,\ L_i \le s_i \le U_i\}, i = 1, 2, \ldots, p$, define the following functions:

$$
\begin{aligned}
&\psi_i(x, s_i) = s_i\big(d_i^T x + g_i\big),\\
&\underline{\psi}_i(x, s_i) = \max\big\{H_i^1(x, s_i), H_i^2(x, s_i)\big\},\\
&\overline{\psi}_i(x, s_i) = \min\big\{H_i^3(x, s_i), H_i^4(x, s_i)\big\},\\
&H_i^1(x, s_i) = \alpha_i s_i + L_i\big(d_i^T x + g_i\big) - \alpha_i L_i,\\
&H_i^2(x, s_i) = \beta_i s_i + U_i\big(d_i^T x + g_i\big) - \beta_i U_i,\\
&H_i^3(x, s_i) = \alpha_i s_i + U_i\big(d_i^T x + g_i\big) - \alpha_i U_i,\\
&H_i^4(x, s_i) = \beta_i s_i + L_i\big(d_i^T x + g_i\big) - \beta_i L_i,
\end{aligned}
$$

then we get the following theorem.

**Theorem 4.1.** *For any $i \in \{1, 2, \ldots, p\}$, let $\psi_i(x, s_i)$, $\underline{\psi}_i(x, s_i)$, $\overline{\psi}_i(x, s_i)$, $RC_i$, $H_i^1(x, s_i)$, $H_i^2(x, s_i), H_i^3(x, s_i)$, and $H_i^4(x, s_i)$ be defined in the former, and let $\Delta s_i = U_i - L_i$. Then, we have*

$$
\begin{aligned}
\big|\psi_i(x, s_i) - \underline{\psi}_i(x, s_i)\big| &\to 0, \quad as \quad \Delta s_i \to 0,\\
\big|\overline{\psi}_i(x, s_i) - \psi_i(x, s_i)\big| &\to 0, \quad as \quad \Delta s_i \to 0.
\end{aligned}
$$

*Proof.* By the definitions of the functions $H_i^1(x, s_i), H_i^2(x, s_i), \underline{\psi}_i(x, s_i)$ and $\psi_i(x, s_i)$, for any $(x, s_i) \in RC_i$, we get that

$$
\begin{aligned}
\big|\psi_i(x, s_i) - H_i^1(x, s_i)\big| &= \big|s_i\big(d_i^T x + g_i\big) - \big[\alpha_i s_i + L_i\big(d_i^T x + g_i\big) - \alpha_i L_i\big]\big|\\
&= \big|\big[s_i\big(d_i^T x + g_i\big) - s_i\alpha_i\big] - \big[L_i\big(d_i^T x + g_i\big) - \alpha_i L_i\big]\big|\\
&= \big|s_i\big[\big(d_i^T x + g_i\big) - \alpha_i\big] - L_i\big[\big(d_i^T x + g_i\big) - \alpha_i\big]\big|\\
&= \big|\big[\big(d_i^T x + g_i\big) - \alpha_i\big] \times [s_i - L_i]\big|\\
&\le \big|[\beta_i - \alpha_i] \times [U_i - L_i]\big|\\
&= (\beta_i - \alpha_i)\Delta s_i,
\end{aligned}
$$

this implies that

$$
\big|\psi_i(x, s_i) - H_i^1(x, s_i)\big| \to 0, \quad \text{as} \quad \Delta s_i \to 0. \tag{4.1}
$$

Similarly, we also have

$$\begin{aligned}\big|\psi_i(x,s_i)-H_i^2(x,s_i)\big| &= \big|s_i\big(d_i^Tx+g_i\big)-\big[\beta_is_i+U_i\big(d_i^Tx+g_i\big)-\beta_iU_i\big]\big|\\ &= \big|\big[s_i\big(d_i^Tx+g_i\big)-s_i\beta_i\big]-\big[U_i\big(d_i^Tx+g_i\big)-\beta_iU_i\big]\big|\\ &= \big|s_i\big[\big(d_i^Tx+g_i\big)-\beta_i\big]-U_i\big[\big(d_i^Tx+g_i\big)-\beta_i\big]\big|\\ &= \big|\big[\big(d_i^Tx+g_i\big)-\beta_i\big]\times[s_i-U_i]\big|\\ &\le \big|[\alpha_i-\beta_i]\times[L_i-U_i]\big|\\ &= (\beta_i-\alpha_i)\Delta s_i,\end{aligned}$$

this indicates that

$$\big|\psi_i(x,s_i)-H_i^2(x,s_i)\big| \;\to\; 0, \quad \text{as} \quad \Delta s_i \;\to\; 0. \tag{4.2}$$

Hence, by (4.1) and (4.2), we can get that

$$\big|\psi_i(x,s_i)-\underline{\psi}_i(x,s_i)\big| \;\to\; 0, \quad \text{as} \quad \Delta s_i \;\to\; 0.$$

By the definitions of the functions $H_i^3(x,s_i)$, $H_i^4(x,s_i)$, $\overline{\psi}_i(x,s_i)$ and $\psi_i(x,s_i)$, for any $(x,s_i)\in RC_i$, we can get that

$$\begin{aligned}\big|H_i^3(x,s_i)-\psi_i(x,s_i)\big| &= \big|\alpha_is_i+U_i\big(d_i^Tx+g_i\big)-\alpha_iU_i-s_i\big(d_i^Tx+g_i\big)\big|\\ &= \big|\alpha_is_i-\alpha_iU_i+U_i\big(d_i^Tx+g_i\big)-s_i\big(d_i^Tx+g_i\big)\big|\\ &= \big|\alpha_i(s_i-U_i)+\big(d_i^Tx+g_i\big)(U_i-s_i)\big|\\ &= \big|(U_i-s_i)\big[\big(d_i^Tx+g_i\big)-\alpha_i\big]\big|\\ &\le \big|[\beta_i-\alpha_i]\times[U_i-L_i]\big|\\ &= (\beta_i-\alpha_i)\Delta s_i,\end{aligned}$$

this implies that

$$\big|H_i^3(x,s_i)-\psi_i(x,s_i)\big| \;\to\; 0, \quad \text{as} \quad \Delta s_i \;\to\; 0. \tag{4.3}$$

Similarly, we also have

$$\begin{aligned}\big|H_i^4(x,s_i)-\psi_i(x,s_i)\big| &= \big|\beta_is_i+L_i\big(d_i^Tx+g_i\big)-\beta_iL_i-s_i\big(d_i^Tx+g_i\big)\big|\\ &= \big|\beta_is_i-\beta_iL_i+L_i\big(d_i^Tx+g_i\big)-s_i\big(d_i^Tx+g_i\big)\big|\\ &= \big|\beta_i(s_i-L_i)-\big(d_i^Tx+g_i\big)(s_i-L_i)\big|\\ &= \big|(s_i-L_i)\big[\beta_i-\big(d_i^Tx+g_i\big)\big]\big|\\ &\le \big|[\beta_i-\alpha_i]\times[U_i-L_i]\big|\\ &= (\beta_i-\alpha_i)\Delta s_i,\end{aligned}$$

this indicates that

$$\big|H_i^4(x,s_i)-\psi_i(x,s_i)\big| \;\to\; 0, \quad \text{as} \quad \Delta s_i \;\to\; 0. \tag{4.4}$$

Hence, by (4.3) and (4.4), we can get that

$$\big|\overline{\psi}_i(x,s_i)-\psi_i(x,s_i)\big| \;\to\; 0, \quad \text{as} \quad \Delta s_i \;\to\; 0.$$

The proof is complete. □

Therefore, by Theorem 4.1, the affine underestimating function $\underline{\psi}_i(x,s_i)$ and the affine overestimating function $\overline{\psi}_i(x,s_i)$ will approximate infinitely the function $\psi_i(x,s_i)$ as $\|U-L\|\to 0$, which ensures the global convergence of the proposed image space branch-and-bound algorithm.

## 5. Algorithm and Its Computational Complexity

In this section, first of all, we present a basic branching method. Next, based on the characteristics of objective function of the problem (2.4) and the currently known upper bound of the image space branch-and-bound algorithm, an image space region reducing method is constructed. By combining the branching method and the constructed affine relaxation problem with the image space region reducing method, a new image space branch-reduction-bound algorithm is proposed for globally solving the problem (1.1).

### 5.1. Branching method

In the algorithm, the partitioning process takes place in the image space $R^p$ of ratios. Without losing generality, suppose that $\Lambda = \{s \in R^p | L_i \leq s_i \leq U_i, i = 1, 2, \ldots, p\}$ is $\Lambda^0$ or a sub-rectangle of $\Lambda^0$, which will be partitioned, the selected branching method is described as follows. Let

$$q = \arg\max\{U_i - L_i,\ i = 1, 2, \ldots, p\},$$

by using the maximum edge binding method of rectangles, subdivide the investigated image space rectangle $\Lambda$ into two new sub-rectangles

$$\hat{\Lambda}^1 = \left\{ s \in R^p \,\middle|\, L_i \leq s_i \leq \frac{L_i + U_i}{2},\ i = q;\ L_i \leq s_i \leq U_i,\ i = 1, 2, \ldots, p,\ i \neq q \right\},$$
$$\hat{\Lambda}^2 = \left\{ s \in R^p \,\middle|\, \frac{L_i + U_i}{2} \leq s_i \leq U_i,\ i = q;\ L_i \leq s_i \leq U_i,\ i = 1, 2, \ldots, p,\ i \neq q \right\}.$$

From Horst and Tuy [6], we have that the branching process is exhaustive, so that there exists a nested image space rectangular subsequence $\{\Lambda^k\}$, which be generated by the proposed image space branch-and-bound algorithm, and which satisfies that $\lim_{k\to\infty} \bigcap_k \Lambda^k = \{s^*\}$.

### 5.2. Image space region reducing method

For improving the convergence speed of the algorithm, for any image space rectangle $\Lambda^k = [L^k, U^k] \subseteq \Lambda^0$ formed by the branching process, and which still needs to be examined, without losing any global optimal solution of the problem (2.4), the image space region reducing method aims at compressing the investigated image space rectangle $\Lambda^k$ into a smaller rectangle $\bar{\Lambda}^k$ or deleting the whole rectangle $\Lambda^k$, so that we need to check whether $\Lambda^k$ contains the global optimal solution of the problem (2.4). Therefore, for this purpose, we let

$$\hat{\varphi}^k = \sum_{i=1}^{p} L_i^k,$$

the derived process of the smaller rectangle $\bar{\Lambda}^k$ with $\bar{\Lambda}^k \subseteq \Lambda^0$ can be given by the following theorem.

**Theorem 5.1.** *Assume that $UB$ is the best currently known upper bound of the global optimal value of the problem* (2.4)*, then for any rectangle $\Lambda^k = [L^k, U^k] \subseteq \Lambda^0$, we have the following conclusions:*

*(i) If $\hat{\varphi}^k > UB$, then the rectangle $\Lambda^k$ contains no global optimal solution of the problem* (2.4)*.*

*(ii) If* $\hat{\varphi}^k \le UB$ *and* $L_\rho^k \le \tau_\rho^k \le U_\rho^k$ *for some* $\rho \in \{1, 2, \ldots, p\}$*, then the rectangle* $\hat{\Lambda}^k$ *contains no global optimal solution of the problem* (2.4)*, where*

$$\hat{\Lambda}^k = \left\{ s \in R^p \,|\, \tau_\rho^k < s_\rho \le U_\rho^k,\ L_i^k \le s_i \le U_i^k,\ i = 1, 2, \ldots, p,\ i \ne \rho \right\}$$

*with*

$$\tau_\rho^k = UB - \hat{\varphi}^k + L_\rho^k.$$

*Proof.* For any rectangle $\Lambda^k = [L^k, U^k] \subseteq \Lambda^0$, we consider the following two kinds of cases: (i) and (ii).

(i) If $\hat{\varphi}^k > UB$, then for any feasible solution $(\check{x}, \check{s})$ of the problem (2.4) over $\Lambda^k$, its corresponding objective function value $F(\check{x}, \check{s})$ satisfies that

$$F(\check{x}, \check{s}) = \sum_{i=1}^{p} \check{s}_i \ge \sum_{i=1}^{p} L_i^k = \hat{\varphi}^k > UB.$$

Therefore, the rectangle $\Lambda^k$ contains no global optimal solution of the problem (2.4).

(ii) If $\hat{\varphi}^k \le UB$ and $L_\rho^k \le \tau_\rho^k \le U_\rho^k$ for some $\rho \in \{1, 2, \ldots, p\}$, then for any feasible solution $(\check{x}, \check{s})$ of the problem (2.4) over $\hat{\Lambda}^k$, it follows that

$$\begin{aligned} F(\check{x}, \check{s}) = \sum_{i=1}^{p} \check{s}_i &> \sum_{i=1, i\ne\rho}^{p} L_i^k + \tau_\rho^k \\ &= \hat{\varphi}^k - L_\rho^k + \tau_\rho^k = \hat{\varphi}^k - L_\rho^k + UB - \hat{\varphi}^k + L_\rho^k = UB. \end{aligned}$$

Thus, the rectangle $\hat{\Lambda}^k$ contains no global optimal solution of the problem (2.4). □

From Theorem 5.1, the presented image space region reducing method can provide a possibility for deleting the whole or a large part of the currently investigated image space rectangle $\Lambda^k$ in which there exists no any global optimal solution of the problem (2.4).

### 5.3. Image space branch-reduction-bound algorithm

In this algorithm, for any $\Lambda \subseteq \Lambda^0$, let $\omega(\Lambda)$ be the optimal solution of the problem (3.1), and let $v_\Lambda$ be the optimal value of the problem (2.4) over $\Lambda$. The steps of the image space branch-reduction-bound algorithm are described as follows.

**Algorithm 5.1:** Image Space Branch-Reduction-Bound Algorithm.

**Step 1.** Let $\Lambda^1 = \Lambda^0$, solve the problem $(\mathrm{LP}_{\Lambda^1})$. If the problem $(\mathrm{LP}_{\Lambda^1})$ is not feasible, then the problem (2.4) is also not feasible. Otherwise, let $LB_1 = LB(\Lambda^1)$ be the lower bound of $v_{\Lambda^1}$, let $(\bar{x}^1, \bar{s}^1)$ be the optimal solution of the problem $(\mathrm{LP}_{\Lambda^1})$, let $\omega(\Lambda^1) = (x^1, s^1)$, where $x^1 = \bar{x}^1$ and $s_i^1 = (c_i^T x^1 + f_i)/(d_i^T x^1 + g_i), i = 1, \ldots, p$, let $Z(\Lambda^1) = \{\omega(\Lambda^1)\}$, and let $v^1 = \sum_{i=1}^{p} s_i^1$. Let $\Omega_1 = \{\Lambda^1\},\ k = 1$.

**Step 2.** Let $\Xi_k = \{\Lambda \in \Omega_k \mid LB(\Lambda) < v^k\}$, and let $\Omega_{k+1} = \{\Lambda \in \Omega_k \mid \Lambda \in \Xi_k\}$. If $v^k - LB_k \le \epsilon$, then the presented algorithm terminates, and $(x^k, s^k)$ and $x^k$ are the global optimal solutions to the problems (2.4) and (1.1), respectively.

**Step 3.** Let $LB_k = \min\{LB(\Lambda) \mid \Lambda \in \Xi_k\}$, and let $\Lambda^k$ be a selected rectangle, which satisfies that $\Lambda^k \in \operatorname{argmin}\{LB(\Lambda) \mid \Lambda \in \Xi_k\}$, and which will be partitioned, subdivide $\Lambda^k$ into two new sub-rectangles $\Lambda^{2k}$ and $\Lambda^{2k+1}$.

**Step 4.** Let $\Omega_{k+1} = (\Omega_{k+1} \setminus \{\Lambda^k\}) \bigcup \{\Lambda^{2k}, \Lambda^{2k+1}\}$. For each $\Lambda \in \{\Lambda^{2k}, \Lambda^{2k+1}\}$, use the image space region reducing method to compress its range, and still denote the remaining sub-rectangle as $\Lambda$, solve the problem (3.1).

If the problem (3.1) is not feasible, then the problem (2.4) over $\Lambda$ is also not feasible, and delete the sub-rectangle $\Lambda$ from $\Omega_{k+1}$.

Otherwise, if $LB(\Lambda) > v^k$, then delete the sub-rectangle $\Lambda$ from $\Omega_{k+1}$, i.e. let $\Omega_{k+1} = \Omega_{k+1} \setminus \{\Lambda\}$, else we can get a lower bound $LB(\Lambda)$ of $v_\Lambda$, and let $\Xi_k = \Xi_k \bigcup \Lambda$ and $Z(\Lambda) = Z(\Lambda) \bigcup \{\omega(\Lambda)\}$, where $\omega(\Lambda) = (x(\Lambda), s(\Lambda))$ is the obtained feasible point by solving the problem (3.1), and let $LB_k = \min\{LB(\Lambda) | \Lambda \in \Xi_k\}$.

**Step 5.** Let $k = k + 1$, and let $(x^k, s^k)$ be the best currently known feasible point of the problem (2.4) among $Z(\Lambda)$, where $s_i^k = (c_i^T x^k + f_i)/(d_i^T x^k + g_i), i = 1, \ldots, p$, and let $v^k = \sum_{i=1}^p s_i^k$, and return to Step 2.

### 5.4. Convergence analysis

In the subsection, without loss of generality, we assume that $v^*$ is the global optimal value of the problem (1.1), the global convergence of the algorithm is analyzed as follows.

**Definition 5.1.** *Let $x^k$ be the known feasible solution of the problem* (1.1)*, and let $v^*$ be the global minimum value of the problem* (1.1)*. If $G(x^k) - v^* \leq \epsilon$, then $x^k$ is a global $\epsilon$-optimum solution of the problem* (1.1)*.*

**Theorem 5.2.** *The presented algorithm either terminates finitely with getting a global optimum solution of the problem* (1.1)*, or generates an infinite sequence of feasible solutions $\{x^k\}$ such that any of its accumulation points is a global optimum solution of the problem* (1.1)*.*

*Proof.* If the presented algorithm terminates finitely after $k$ iterations, so with the termination of the algorithm, we can obtain a better feasible solution $x^k$ of the problem (FP), and we also obtain a better feasible solution $(x^k, s^k)$ of the problem (2.3) with that

$$s_i^k = \frac{c_i^T x^k + f_i}{d_i^T x^k + g_i}, \quad i = 1, 2, \ldots, p.$$

From the terminating conditions of the algorithm, the updating methods of the lower bound, the updating methods of the upper bound, and the structure of the branch-and-bound algorithm, we can get the following results:

$$LB_k \leq v^*, \quad v^* \leq F(x^k, s^k), \quad G(x^k) = F(x^k, s^k) = v^k, \quad v^k - \epsilon \leq LB_k.$$

Combining the above formulas together, we get

$$G(x^k) - \epsilon = F(x^k, s^k) - \epsilon \leq LB_k \leq v^* \leq F(x^k, s^k) = G(x^k).$$

Thus, $x^k$ is a global $\epsilon$-optimum solution of the problem (FP).

If the presented algorithm generates an infinite sequence of feasible solutions $\{x^k\}$ of the problem (1.1) and an infinite sequence of feasible solutions $\{(x^k, s^k)\}$ of the problem (2.3) with that

$$s_i^k = \frac{c_i^T x^k + f_i}{d_i^T x^k + g_i}, \quad i = 1, 2, \ldots, p,$$

respectively, and let $x^*$ be an accumulation point of $\{x^k\}$, then, without loss of generality, we have that

$$\lim_{k\to\infty} x^k = x^*.$$

From the continuity of the function $(c_i^T x + f_i)/(d_i^T x + g_i)$,

$$\frac{c_i^T x^k + f_i}{d_i^T x^k + g_i} = s_i^k \in \left[L_i^k, U_i^k\right], \quad i = 1, 2, \ldots, p,$$

and the exhaustiveness of the branching method, we have that

$$\frac{c_i^T x^* + f_i}{d_i^T x^* + g_i} = \lim_{k\to\infty} \frac{c_i^T x^k + f_i}{d_i^T x^k + g_i} = \lim_{k\to\infty} s_i^k = \lim_{k\to\infty} [L_i^k, U_i^k] = \lim_{k\to\infty} \bigcap_k \left[L_i^k, U_i^k\right] = s_i^*.$$

Therefore, $(x^*, s^*)$ is a feasible solution to the problem (2.3). And because $\{LB_k\}$ is an increasing sequence of the lower bound such that $LB_k \le v^*$, we can get that

$$F(x^*, s^*) \ge v^* \ge \lim_{k\to\infty} LB_k = \lim_{k\to\infty} F(x^k, s^k) = F(x^*, s^*). \tag{5.1}$$

Thus, from the updating method of the upper bound and the continuity of the function $G(x)$, we can draw the following conclusions:

$$\lim_{k\to\infty} v^k = \lim_{k\to\infty} \sum_{i=1}^{p} s_i^k = \lim_{k\to\infty} F(x^k, s^k) = F(x^*, s^*) = G(x^*) = \lim_{k\to\infty} G(x^k). \tag{5.2}$$

By the above inequalities (5.1) and (5.2), we have that

$$\lim_{k\to\infty} v^k = v^* = G(x^*) = \lim_{k\to\infty} G(x^k) = F(x^*, s^*) = \lim_{k\to\infty} LB_k.$$

Thus, any of accumulation points $x^*$ for the infinite sequence of solutions $\{x^k\}$ is a global optimum solution to the problem (1.1), and the proof is complete. □

### 5.5. Computational complexity of the algorithm

**Definition 5.2.** *Assume that $\Lambda_p = [L_1, U_1] \times \ldots \times [L_p, U_p] \subset R^p$ be a compact hyper-rectangle, the diameter of the hyper-rectangle $\Lambda_p \subset R^p$ is defined by*

$$\delta(\Lambda_p) = \max\left\{\|\alpha - \alpha'\|_2 : \alpha, \alpha' \in \Lambda_p\right\} = \sqrt{(U_1 - L_1)^2 + \cdots + (U_p - L_p)^2}.$$

**Theorem 5.3.** *For the proposed algorithm, for any given hyper-rectangle $\Lambda_p$, assume that there exist some a fixed positive constant $C_p$ and an accuracy error $\epsilon$, and assume that the branching operation will eventually partition the hyper-rectangle $\Lambda_p$ into $\varrho = 2^p$ smaller sub-hyper-rectangles. Then, by subdividing the hyper-rectangle $\Lambda_p$, the number of iterations of the proposed algorithm in the worst case can be expressed as follows:*

$$\sum_{t=0}^{r_p} 2^{p.t}, \quad \textit{where} \quad r_p = \left\lceil \log_2 \frac{C_p \cdot \delta(\Lambda_p)}{\epsilon} \right\rceil, \quad \delta(\Lambda_p) = \max\left\{\delta\left(\Lambda_p^l\right) : l \in \{1, 2, \ldots, \varrho\}\right\}. \tag{5.3}$$

*We call $\mathcal{O}(p) = \sum_{t=0}^{r_p} 2^{p.t}$ be the convergence rate of the algorithm by subdividing space $R^p$.*

*Proof.* The proof is similar to [23, Theorem 5], thus it is omitted here. □

Similar as Theorem 5.3, if we subdivide the $n$-dimensional hyper-rectangle $\Lambda_n$ by rectangular dichotomy, the number of iteration of the algorithm in the worst case can be expressed as follows:

$$\sum_{t=0}^{r_n} 2^{n.t}, \quad \text{where} \quad r_n = \left\lceil \log_2 \frac{C_n \cdot \delta(\Lambda_n)}{\epsilon} \right\rceil, \quad \delta(\Lambda_n) = \max\left\{\delta\big(\Lambda_n^l\big) : l \in \{1, 2, \ldots, 2^n\}\right\}, \quad (5.4)$$

where $n, C_n, r_n$ and $\Lambda_n$ are corresponding to $p, C_p, r_p$ and $\Lambda_p$ in (5.3). We also call $\mathcal{O}(n) = \sum_{t=0}^{r_n} 2^{n.t}$ as the convergence rate of the algorithm by subdividing the $n$-dimensional hyper-rectangle $\Lambda_n$.

By (5.3) and (5.4), when $p \ll n$, we have the following conclusions:

(i) If $r_p \le r_n$, then $\sum_{t=0}^{r_p} 2^{p.t} \le \sum_{t=0}^{r_n} 2^{p.t} \ll \sum_{t=0}^{r_n} 2^{n.t}$.

(ii) If $r_p \ge r_n$, then there must exist a positive number $N \ge \lfloor r_p p/r_n + 1 \rfloor$ such that $p < r_p p/r_n < N$, this means that when $N \ll n$, we have $p < r_p p/r_n < N \ll n$ and $pr_p \ll nr_n$, so that

$$\begin{aligned}
\sum_{t=0}^{r_n} 2^{n.t} - \sum_{t=0}^{r_p} 2^{p.t} &= \frac{2^{n(r_n+1)} - 1}{2^n - 1} - \frac{2^{p(r_p+1)} - 1}{2^p - 1} \\
&= \frac{(2^{n(r_n+1)} - 1)(2^p - 1) - (2^{p(r_p+1)} - 1)(2^n - 1)}{(2^n - 1)(2^p - 1)} \\
&= \frac{2^{nr_n}2^{n+p} - 2^{nr_n}2^n - 2^p + 1 - 2^{pr_p}2^{n+p} + 2^{pr_p}2^p + 2^n - 1}{(2^n - 1)(2^p - 1)} \\
&= \frac{2^{n+p}(2^{nr_n} - 2^{pr_p}) - 2^{nr_n}2^n + 2^{pr_p}2^p + 2^n - 2^p}{(2^n - 1)(2^p - 1)} \\
&= \frac{\big(2^{n+p}(2^{nr_n} - 2^{pr_p}) - 2^{nr_n}2^n + 2^{pr_p}2^p + 2^n - 2^p\big)/(2^{n+p}2^{nr_n})}{(2^n - 1)(2^p - 1)/(2^{n+p}2^{nr_n})} \\
&= \frac{1 - 2^{pr_p}/2^{nr_n} - 1/2^p + 2^{pr_p}/2^n 2^{nr_n} + 1/2^p 2^{nr_n} - 1/2^n 2^{nr_n}}{(2^n - 1)(2^p - 1)/(2^{n+p}2^{nr_n})} \gg 0.
\end{aligned}$$

The last inequality above holds, it is since

$$\frac{2^{pr_p}}{2^{nr_n}} \to 0, \quad \frac{1}{2^p} \to 0, \quad \frac{2^{pr_p}}{2^n 2^{nr_n}} \to 0, \quad \frac{1}{2^p 2^{nr_n}} \to 0, \quad \frac{1}{2^n 2^{nr_n}} \to 0,$$

as $p \ll n$ and $pr_p \ll nr_n$, so that we have

$$\left(1 - \frac{2^{pr_p}}{2^{nr_n}} - \frac{1}{2^p} + \frac{2^{pr_p}}{2^n 2^{nr_n}} + \frac{1}{2^p 2^{nr_n}} - \frac{1}{2^n 2^{nr_n}}\right) \to 1,$$

as $p \ll n$ and $pr_p \ll nr_n$. Similarly, we have

$$0 < \frac{(2^n - 1)(2^p - 1)}{2^{n+p}2^{nr_n}} \to 0,$$

as $p \ll n$ and $pr_p \ll nr_n$. To sum up, we have that the last inequality above holds.

**Remark 5.1.** From the above discussions, we know that $\mathcal{O}(n)$ and $\mathcal{O}(p)$ are both functions of exponential growth, but when $p \ll n$, the branch-and-bound search of the proposed algorithm takes place in space $R^p$ which typically has a much smaller dimension than space $R^n$ of the decision variable $x$ for the problem (1.1), the characteristic of the algorithm is expected to considerably shorten the length of the branch-and-bound search, i.e. $\mathcal{O}(p) \ll \mathcal{O}(n)$ .

## 6. Numerical Experiment

In this section, we numerically compare our image space branch-reduction-bound algorithm with the software BARON [15] and some known existing algorithms. All numerical tests are implemented in MATLAB R2014a and run on a microcomputer with Intel(R) Core(TM) i5-7200U CPU @2.50 GHz processor and 16 GB RAM. The maximum CPU time limit of all algorithms is set at 4800 s. For all test problems, we presented statistics of the numerical results.

First of all, some small size certainty examples in Appendix A were tested with the proposed algorithm for comparison with the known existing algorithms [1, 5, 8, 9, 27, 29, 30, 33], and the corresponding numerical results are reported in Table 6.1 with the given convergence tolerance, where some notations have been used for column headers: Opt. val.: global optimal value; Iter.: number of iterations of the algorithm; Time: CPU execution time of the algorithm in seconds.

From the numerical results in Table 6.1, for all examples A.1-A.12, we can follow that our algorithm can obtain better global optimal solutions and optimal values than the existing algorithms in [1, 5, 8, 9, 27, 29, 30, 33] with higher computational efficiency, in terms of test examples A.1-A.12, so that our algorithm highly outperforms the existing algorithms in [1, 5, 8, 9, 27, 29, 30, 33].

Next, we chose some large-scale stochastic test problems generated randomly to verify the proposed algorithm further, see test Problems 1 and 2 for details.

For test Problem 1 with the large-size number of variables, with the given convergence tolerance $\epsilon = 10^{-2}$, numerical comparisons among algorithms of Shen *et al.* [27], Jiao and Liu [8], BARON, and our algorithm are reported in Table 6.2, respectively. In addition, for test Problem 1 with $\epsilon = 10^{-6}$, numerical comparisons between our algorithm and BARON are reported in Table 6.3. For each random test Problem 1, we solved ten independently generated test instances and recorded the best results, the worst results, and the average results among these ten tests, and we highlight in bold the winner of average results in numerical comparisons.

For test Problem 2 with the large-size number of ratios $p$, with the given convergence tolerance $\epsilon = 10^{-3}$, numerical comparisons between our algorithm and BARON are reported in Table 6.4, respectively. In addition, for test Problem 2 with $\epsilon = 10^{-6}$, numerical comparisons between our algorithm and BARON are reported in Table 6.5. For each random test Problem 2, we also solve ten independently generated test instances and record the best results, the worst results, and the average results among these ten test instances, and we also highlight in bold the winner of average results in numerical comparisons.

Some notations have been used for column headers in Tables 6.2-6.5 #iter stands for the number of iterations of the algorithm, time (s) stand for the CPU time of the algorithm in seconds, and "$*$" stands for the situation that the algorithm failed to terminate in 4800 s.

From the numerical results in Table 6.2, for test Problem 1 with the large-size number of variables $n$, with $\epsilon = 10^{-2}$, we firstly get the observation that the software BARON is more time-consuming than our algorithm proposed in this paper, though the number of iterations for the software BARON is smaller. Secondly, our algorithm obviously outperforms the algorithms proposed in Shen *et al.* [27] and Jiao and Liu [8]. The number of iterations of our algorithm is much less than that of the proposed algorithms in Shen *et al.* [27] and Jiao and Liu [8]. Especially, when $p = 2$ and $n = 8000, 10000$ or $20000$, BARON and the algorithm of Shen *et al.* [27] failed to terminate in 4800 s; and $p = 3$ and $n = 8000$, BARON and the algorithms of Shen *et al.* [27] and Jiao and Liu [8] also failed to terminate in 4800 s, but our algorithm in this paper can obtain the global optimal solution of test Problem 1 with higher computational efficiency.

From the numerical results in Table 6.3, for test Problem 1 with $\epsilon = 10^{-6}$, we can observe that, when $p = 2$ and $n \geq 4000$, the software BARON is more time-consuming than our algorithm; when $p = 2$ and $n \geq 5000$, the software BARON failed to terminate in 4800 s, but our algorithm can find the global optimal solution of test Problem 1.

From the numerical results in Table 6.4, for test Problem 2 with the large-size number of sum $p$, with $\epsilon = 10^{-3}$, we can observe that, when $p = 10$ and $n = 500, p = 15$ and $n = 500$, $p = 20$ and $n = 400$, BARON failed to terminate in 4800 s for any ten independently generated instances, but our algorithm can successfully find the global optimal solution of test Problem 2.

Table 6.1: Numerical comparisons between some existing algorithms and our algorithm on test examples A.1-A.12.

| No. | Algorithms | Opt. val. | Optimal solution | Iter. | Time | $\epsilon$ |
|---|---|---|---|---|---|---|
| 1 | Our algorithm | −4.84151 | (0.1000, 2.3750) | 2 | 0.012 | $10^{-2}$ |
| | Jiao & Liu [8] | −4.84151 | (0.1000, 2.3750) | 200 | 4.257 | $10^{-2}$ |
| | Benson [1] | −4.84151 | (0.1000, 2.3750) | 4 | 0.190 | $10^{-2}$ |
| 2 | Our algorithm | −2.47143 | (1.0000, 0.0000, 0.0000) | 1 | 0.013 | $10^{-2}$ |
| | Jiao & Liu [8] | −2.47124 | (1.0001, 0.0000, 0.0001) | 54 | 1.135 | $10^{-2}$ |
| | Shen et al. [27] | −2.47143 | (1.0000, 0.0000, 0.0000) | 2 | 0.015 | $10^{-2}$ |
| 3 | Our algorithm | −1.90000 | (0.0000, 3.3333, 0.0000) | 1 | 0.011 | $10^{-6}$ |
| | Shen & Wang [30] | −1.90000 | (0.0000, 3.3333, 0.0000) | 8 | 0.926 | $10^{-6}$ |
| 4 | Our algorithm | 1.62319 | (0.0000, 0.2861) | 47 | 1.294 | $10^{-2}$ |
| | Jiao & Liu [8] | 1.62319 | (0.0000, 0.2861) | 93 | 2.485 | $10^{-2}$ |
| 5 | Our algorithm | 2.86190 | (5.0000, 0.0000, 0.0000) | 4 | 0.064 | $10^{-3}$ |
| | Shen & Lu [29] | 2.86191 | (5.0000, 0.0000, 0.0000) | 16 | 0.125 | $10^{-3}$ |
| | Gao & Jin [5] | 2.86190 | (5.0000, 0.0000, 0.0000) | 12 | 28.29 | $10^{-3}$ |
| | Jiao & Liu [8] | 2.86241 | (4.8302, 0.0000, 0.0666) | 4008 | 128.0 | $10^{-3}$ |
| 6 | Our algorithm | −4.09070 | (1.1111, 0.0000, 0.0000) | 1 | 0.044 | $10^{-2}$ |
| | Jiao & Liu [8] | −4.09062 | (1.1106, 0.0000, 0.0015) | 619 | 16.62 | $10^{-2}$ |
| | Shen & Lu [29] | −4.08741 | (1.0715, 0.0000, 0.0000) | 17 | 3.251 | $10^{-2}$ |
| 7 | Our algorithm | 3.71092 | (0.0000, 1.6667, 0.0000) | 1 | 0.009 | $10^{-4}$ |
| | Jiao & Liu [8] | 3.71093 | (0.0000, 1.6667, 0.0000) | 2747 | 94.64 | $10^{-4}$ |
| | Gao & Jin [5] | 3.7087 | (0.0000, 1.6667, 0.0000) | 5 | 4.190 | $10^{-4}$ |
| 8 | Our algorithm | −3.00292 | (0.0000, 3.3333, 0.0000) | 23 | 0.798 | $10^{-6}$ |
| | Jiao & Liu [8] | −3.00292 | (0.0000, 3.3333, 0.0000) | 1072 | 31.746 | $10^{-2}$ |
| 9 | Our algorithm | 4.91259 | (1.5000, 1.5000) | 16 | 0.237 | $10^{-3}$ |
| | Shen & Lu [29] | 4.91259 | (1.5000, 1.5000) | 56 | 1.087 | $10^{-3}$ |
| 10 | Our algorithm | −4.09070 | (1.1111, 0.0000, 0.0000) | 2 | 0.071 | $10^{-6}$ |
| | Jiao & Liu [8] | −4.09065 | (1.1109, 0.0000, 0.0005) | 977 | 32.41 | $10^{-6}$ |
| | Jiao et al. [9] | −4.09070 | (1.1111, 0.0000, 0.0000) | 2 | 0.008 | $10^{-6}$ |
| 11 | Our algorithm | 3.29167 | (3.0000, 4.0000) | 7 | 0.114 | $10^{-6}$ |
| | Shen & Wang [30] | 3.29167 | (3.0000, 4.0000) | 9 | 0.489 | $10^{-6}$ |
| 12 | Our algorithm | 4.42857 | (5.0000, 0.0000, 0.0000) | 1 | 0.056 | $10^{-4}$ |
| | Jiao & Liu [8] | 4.42794 | (4.9930, 0.0000, 0.0000) | 128 | 4.213 | $10^{-4}$ |
| | Shi [33] | 4.42857 | (5.0000, 0.0000, 0.0000) | 58 | 2.968 | $10^{-4}$ |

Table 6.2: Numerical comparisons among our algorithm, BARON, and the algorithms of Jiao & Liu [8] and Shen *et al.* [27] for Problem 1 with the given convergence tolerance $\epsilon = 10^{-2}$.

| $(p,m,n)$ | Algorithms | #iter | | | time(s) | | |
|---|---|---|---|---|---|---|---|
| | | min | average | max | min | average | max |
| (2,100,2000) | Jiao & Liu [8] | 28 | 108.7 | 222 | 51.92 | 205.71 | 441.71 |
| | Shen *et al.* [27] | 1 | 207.3 | 317 | 0.91 | 281.57 | 793.46 |
| | BARON | **1** | **1.2** | **3** | 77.42 | 279.01 | 478.45 |
| | Our algorithm | 24 | 30.1 | 40 | **68.60** | **96.96** | **140.93** |
| (2,100,3000) | Jiao & Liu [8] | 46 | 82.7 | 153 | 136.07 | 239.74 | 459.27 |
| | Shen *et al.* [27] | 1 | 249.1 | 510 | 3.20 | 688.35 | 1900.8 |
| | BARON | **1** | **1.4** | **5** | 214.25 | 587.91 | 1198.08 |
| | Our algorithm | 23 | 27.1 | 31 | **120.76** | **144.33** | **187.26** |
| (2,100,5000) | Jiao & Liu [8] | 40 | 104.8 | 244 | 186.21 | 530.14 | 1244.53 |
| | Shen *et al.* [27] | * | * | * | * | * | * |
| | BARON | **1** | **1.2** | **3** | 920.05 | 1083.93 | 1408.27 |
| | Our algorithm | 6 | 19.4 | 28 | **38.52** | **118.06** | **176.30** |
| (2,100,7000) | Jiao & Liu [8] | 31 | 81.7 | 184 | 217.49 | 615.68 | 1290.42 |
| | Shen *et al.* [27] | * | * | * | * | * | * |
| | BARON | **1** | **1** | **1** | 2253.22 | 2778.35 | 3727.55 |
| | Our algorithm | 20 | 27.5 | 34 | **301.85** | **415.01** | **541.82** |
| (2,100,8000) | Jiao & Liu [8] | 32 | 84.9 | 139 | 276.25 | 802.90 | 1323.32 |
| | Shen *et al.* [27] | * | * | * | * | * | * |
| | BARON | * | * | * | * | * | * |
| | Our algorithm | **23** | **27.4** | **35** | **415.32** | **525.70** | **650.89** |
| (2,100,10000) | Jiao & Liu [8] | 35 | 76.6 | 112 | 405.80 | 933.54 | 1414.22 |
| | Shen *et al.* [27] | * | * | * | * | * | * |
| | BARON | * | * | * | * | * | * |
| | Our algorithm | **22** | **25.5** | **29** | **560.4** | **704.2** | **864.4** |
| (2,100,20000) | Jiao & Liu [8] | 41 | 69.4 | 105 | 1239.04 | 2216.69 | 3495.84 |
| | Shen *et al.* [27] | * | * | * | * | * | * |
| | BARON | * | * | * | * | * | * |
| | Our algorithm | **21** | **25.5** | **34** | **1389.3** | **1763.6** | **2495.1** |
| (3,100,5000) | Jiao & Liu [8] | * | * | * | * | * | * |
| | Shen *et al.* [27] | * | * | * | * | * | * |
| | BARON | **3** | **9.8** | **31** | 1320.47 | 2310.83 | 3113.8 |
| | Our algorithm | 90 | 182.4 | 330 | **780.5** | **1851.1** | **3134.0** |
| (3,100,8000) | Jiao & Liu [8] | * | * | * | * | * | * |
| | Shen *et al.* [27] | * | * | * | * | * | * |
| | BARON | * | * | * | * | * | * |
| | Our algorithm | **88** | **134.1** | **183** | **1580.3** | **2384.0** | **3660.5** |

In addition, for Problem 2, when $p$ is larger, the algorithms of Shen *et al.* [27] and Jiao and Liu [8] failed to solve each of ten tests in 4800 s, so that we only report the numerical comparisons between BARON and our algorithm in Table 6.4, this implies the robustness and stability of our algorithm in this paper.

Table 6.3: Numerical comparisons between our algorithm and BARON for Problem 1 with the given convergence tolerance $\epsilon = 10^{-6}$.

| $(p, m, n)$ | Algorithms | #iter | | | time(s) | | |
|---|---|---|---|---|---|---|---|
| | | min | average | max | min | average | max |
| (2,100,1000) | BARON | **3** | **25.4** | **79** | **39.11** | **69.11** | **112.23** |
| | Our algorithm | 70 | 168.1 | 343 | 55.02 | 219.72 | 503.13 |
| (2,100,2000) | BARON | **5** | **28.6** | **61** | **259.97** | **362.59** | **656.88** |
| | Our algorithm | 96 | 203.1 | 455 | 107.93 | 421.01 | 737.92 |
| (2,100,3000) | BARON | **5** | **21.2** | **37** | **463.86** | **806.30** | **1263.56** |
| | Our algorithm | 129 | 261.5 | 532 | 454.40 | 1273.07 | 3794.77 |
| (2,100,4000) | BARON | **7** | **25.8** | **49** | 1283.69 | 1753.03 | 2700.50 |
| | Our algorithm | 55 | 154.3 | 318 | **539.63** | **1311.99** | **3052.07** |
| (2,100,5000) | BARON | * | * | * | * | * | * |
| | Our algorithm | **58** | **230.2** | **435** | **510.12** | **2620.04** | **4568.96** |
| (2,100,6000) | BARON | * | * | * | * | * | * |
| | Our algorithm | **104** | **241.2** | **348** | **1101.25** | **3185.66** | **4771.08** |

From the numerical results in Table 6.5, for test Problem 2 with $\epsilon = 10^{-6}$, we can observe that, when $p = 6, n = 400$ and 500; $p = 7$ and $n = 500, p = 8, n = 400$ and 500, BARON failed to terminate in 4800 s for any ten independently generated instances, but our algorithm can successfully find the global optimal solution of test Problem 2. In addition, for test Problem 2, when $n \geq 400$, our algorithm is obviously better than BARON in computational efficiency.

**Test Problem 1.**

$$\begin{aligned} \min \quad & \sum_{i=1}^{p} \frac{c_i^T x + f_i}{d_i^T x + g_i} \\ \text{s.t.} \quad & Ax \leq b, \\ & x \geq 0, \end{aligned}$$

where $c_i \in R^n, d_i \in R^n, A \in R^{m\times n}, b \in R^m, f_i \in R, g_i \in R$, $i = 1, 2, \ldots, p$; each element of $c_i, d_i$, and $A$ is randomly generated from the interval $[0, 10]$; each element of $b$ is equal to 10, $f_i$ and $g_i, i = 1, 2, \ldots, p$, are all randomly generated from the interval $[0, 1]$.

**Test Problem 2.**

$$\begin{aligned} \min \quad & \sum_{i=1}^{p} \frac{\gamma_i^T x + \xi_i}{\delta_i^T x + \eta_i} \\ \text{s.t.} \quad & Ax \leq b, \\ & x \geq 0, \end{aligned}$$

where $\gamma_i \in R^n, \delta_i \in R^n, A \in R^{m\times n}, b \in R^m, \xi_i \in R, \eta_i \in R, i = 1, 2, \ldots, p$; each element of $\gamma_i$ and $\delta_i$ is randomly generated from the interval $[-0.1, 0.1]$; each element of $A$ is randomly generated from the interval $[0.01, 1]$; each element of $b$ is equal to $10, \xi_i$ and $\eta_i$ satisfy $\gamma_i^T x + \xi_i > 0$ and $\delta_i^T x + \eta_i > 0, i = 1, 2, \ldots, p$.

Table 6.4: Numerical comparisons between BARON software and our algorithm for Problem 2 with the given convergence tolerance $\epsilon = 10^{-3}$.

| $(p, m, n)$ | Algorithms | #iter | | | time(s) | | |
|---|---|---|---|---|---|---|---|
| | | min | average | max | min | average | max |
| (10,100,300) | BARON | **3** | **9.2** | **13** | **8.28** | **12.66** | **17.64** |
| | Our algorithm | 12 | 17.5 | 25 | 9.87 | 15.33 | 22.73 |
| (10,100,400) | BARON | 9 | 35.8 | 93 | 22.28 | 30.86 | 42.33 |
| | Our algorithm | **14** | **20.4** | **30** | **14.24** | **22.24** | **33.04** |
| (10,100,500) | BARON | * | * | * | * | * | * |
| | Our algorithm | **16** | **22.3** | **29** | **19.27** | **28.03** | **37.71** |
| (15,100,300) | BARON | **5** | **10.4** | **17** | **15.66** | **24.24** | **38.45** |
| | Our algorithm | 42 | 87.3 | 155 | 42.81 | 106.7 | 201.77 |
| (15,100,400) | BARON | **11** | **34** | **157** | **36.14** | **47.92** | **79.81** |
| | Our algorithm | 44 | 81.4 | 128 | 63.37 | 125.56 | 200.27 |
| (15,100,500) | BARON | * | * | * | * | * | * |
| | Our algorithm | **36** | **91.5** | **162** | **67.15** | **170.12** | **297.63** |
| (20,100,200) | BARON | **15** | **17** | **19** | **17.84** | **20.50** | **26.39** |
| | Our algorithm | 80 | 241.5 | 559 | 86.25 | 267.01 | 651.5 |
| (20,100,300) | BARON | **5** | **14** | **17** | **22.53** | **36.14** | **51.11** |
| | Our algorithm | 172 | 344.9 | 834 | 233.06 | 461.53 | 1135.6 |
| (20,100,400) | BARON | * | * | * | * | * | * |
| | Our algorithm | **135** | **450.6** | **1302** | **225.7** | **765.52** | **2246.21** |

Table 6.5: Numerical comparisons between BARON software and our algorithm for Problem 2 with the given convergence tolerance $\epsilon = 10^{-6}$.

| $(p, m, n)$ | Algorithms | #iter | | | time(s) | | |
|---|---|---|---|---|---|---|---|
| | | min | average | max | min | average | max |
| (6,100,300) | BARON | **13** | **23.2** | **35** | 29.55 | 43.54 | 63.08 |
| | Our algorithm | 151 | 335.3 | 605 | **14.58** | **33.13** | **57.80** |
| (6,100,400) | BARON | * | * | * | * | * | * |
| | Our algorithm | **195** | **799.4** | **2556** | **27.49** | **115.20** | **360.45** |
| (6,100,500) | BARON | * | * | * | * | * | * |
| | Our algorithm | **280** | **740.4** | 1674 | **55.73** | **147.70** | **332.51** |
| (7,100,300) | BARON | **11** | **28.8** | **57** | **37.75** | **61.62** | **97.45** |
| | Our algorithm | 141 | 1379.4 | 6020 | 14.44 | 166.98 | 700.08 |
| (7,100,400) | BARON | **11** | **754.8** | **2947** | 45.03 | 229.69 | 593.89 |
| | Our algorithm | 359 | 1111.3 | 1809 | **56.80** | **173.11** | **272.82** |
| (7,100,500) | BARON | * | * | * | * | * | * |
| | Our algorithm | **385** | **1236.2** | **2189** | **152.08** | **300.82** | **471.26** |
| (8,100,300) | BARON | **15** | **26.2** | **43** | **40.05** | **54.91** | **66.81** |
| | Our algorithm | 811 | 4428.1 | 17507 | 83.01 | 530.48 | 2234.82 |
| (8,100,400) | BARON | * | * | * | * | * | * |
| | Our algorithm | **456** | **2960.7** | **10441** | **86.69** | **519.21** | **1982.62** |
| (8,100,500) | BARON | * | * | * | * | * | * |
| | Our algorithm | **596** | **1873.8** | **3573** | **133.07** | **434.52** | **847.17** |

From the numerical results of Tables 6.1-6.5, it is seen that the presented algorithm in this paper can be used to globally solve the large scale general sum of affine ratios problems with the extreme robustness and effectiveness.

## 7. Comparing with Some Existing Relaxation Algorithms

To present a clear distinction of our work from some existing relaxation algorithms, and demonstrate the innovative contribution of our algorithm, we give some detailed comparisons with several existing relaxation algorithms (Ji *et al.* [7], Jiao and Liu [8] and Jiao *et al.* [9]).

Firstly, from the perspective of model, Ji *et al.* [7] present a rectangle branch and bound algorithm for the problem (1.1) with that each denominator is greater than 0; Jiao and Liu [8] also present a branch and bound algorithm for solving the problem (1.1) with the assumption that each numerator must be nonnegative and each denominator must be positive over feasible region; Jiao *et al.* [9] propose a branch and bound algorithm for solving the problem (1.1) with the assumption that all numerators must be nonnegative and all denominators must be positive over feasible region. However, in this paper, we investigate the problem (1.1) with only assumption that all denominators are not equal to 0. Therefore, the investigated mathematical modeling in this paper is more general than that of Ji *et al.* [7], Jiao and Liu [8] and Jiao *et al.* [9].

Secondly, from the perspective of linear relaxation structure, by using the upper and lower bounds of the denominator to directly replace the denominator of ratio, a linearizing technique is proposed in the algorithm of Ji *et al.* [7], and by utilizing this technique, the linear relaxation problems of the problem (1.1) and its subproblem are constructed; the algorithm of Jiao and Liu [8] works by globally solving an equivalent bilinear programming problem, by directly using convex hull and concave hull of bilinear function over a rectangle, the authors construct linear relaxation of objective function, and by directly relaxing constrained functions, so that a linear relaxation programming problem of the equivalent problem is constructed; the algorithm of Jiao *et al.* [9] also works by globally solving an equivalent problem, by replacing each denominator with its upper bound or lower bound, the authors construct linear relaxation of objective function of the equivalent problem, and by directly relaxing constrained functions, so that a linear relaxation programming problem is constructed. It should be noted that, the relaxation methods of Jiao and Liu [8] and Jiao *et al.* [9] not only relax the objective function but also relax the feasible region of the problem. However, in this paper, first of all, we transform the problem (1.1) into the equivalent problem (2.4). Next, by utilizing the characteristics of bilinear function, we only construct linear relaxation of constrained function of the problem (2.4), a linear relaxation problem of the problem (2.4) is constructed.

Thirdly, from the perspective of searching space, the branch and bound search in Ji *et al.* [7] takes place in space $R^n$ rather than $R^p$, where $n$ is the number of decision variable $x$ and $p$ is the number of ratios; the algorithm of Jiao and Liu [8] economizes the required computations by conducting the branch-and-bound search in the outer space $R^p$ of the reciprocals of denominators of ratios; the algorithm of Jiao *et al.* [9] also economizes the required computations by conducting the branch-and-bound search in the outer space $R^p$ of the denominators of ratios. However, in this paper, the branch and bound search takes place in the image space $R^p$ of ratio functions. Therefore, the algorithms of Jiao and Liu [8], Jiao *et al.* [9], and our algorithm all economize the required computational efforts, and which are all convergent to the global optimal solution through the successive refinement partition of the outer space region and solving

a sequence of linear relaxation problems. As everyone knows, in many practical applications of the problem (1.1), $p$ is significantly smaller than $n$, so that the characteristic of the algorithm is expected to considerably shorten the length of the branch-and-bound search. Moreover, from the conclusions of computational complexity in Section 5.5, we can follow that, when $p$ is far less than $n$, the complexity of the algorithm proposed in this paper is far less than that of Ji *et al.* [7]. Therefore, when $p$ is much less than $n$, the proposed algorithm in this paper has higher computational efficiency than the algorithm of Ji *et al.* [7].

Finally, from the perspective of computational efficiency, in order to compare the efficiency of these algorithms, using our algorithm and the algorithm of Jiao and Liu [8] to solve the random test Problem 1 with the same fixed parameter $(p, m, n)$. From the numerical results in Table 6.2, it is also seen that our algorithm has higher computational efficiency than that of Jiao and Liu [8]. In terms of computational performance, when the scale of the variable is larger, since the algorithms of Ji *et al.* [7] and Jiao *et al.* [9] failed to terminate in 4800 s for any ten independently generated large scale test Problem 1, so that we only list numerical comparisons between our algorithm and the algorithm of Jiao and Liu [8] in Table 6.2.

## 8. Discussions on Further Improvement

An efficient global optimization algorithm for solving the problem (1.1) with $p = 2$ is proposed in [37]. Following the presentation there, using Charnes-Cooper transformation [3], it is sufficient to partition in a reduced image space $R^{p-1}$, which can be regarded as a further improvement of this paper, and which is given as follows.

By introducing new variables $\hat{t} = 1/(d_p^T x + g_p)$ and $\hat{z} = \hat{t}x$, the problem (1.1) can be rewritten into the following equivalent form:

$$\begin{aligned} \min \quad & \Phi(\hat{z}, \hat{t}) = \sum_{i=1}^{p-1} \frac{c_i^T \hat{z} + f_i \hat{t}}{d_i^T \hat{z} + g_i \hat{t}} + c_p^T \hat{z} + f_p \hat{t}, \\ \text{s.t.} \quad & d_p^T \hat{z} + g_p \hat{t} = 1, \\ & A\hat{z} - b\hat{t} \le 0, \\ & \hat{z} \ge 0,\ \hat{t} > 0. \end{aligned} \tag{8.1}$$

Let

$$W = \big\{(\hat{z}, \hat{t}) \in R^{n+1} \mid d_p^T \hat{z} + g_p \hat{t} = 1,\ A\hat{z} - b\hat{t} \le 0,\ \hat{z} \ge 0,\ \hat{t} > 0\big\},$$

and it is obvious that $W$ is a nonempty compact convex set, and that $W \neq \emptyset$ if and only if $D \neq \emptyset$. Without losing generality, for each $i = 1, 2, \ldots, p-1$, let

$$\begin{aligned} \hat{\alpha}_i &= \min_{(\hat{z},\hat{t}) \in W} d_i^T \hat{z} + g_i \hat{t}, \quad \hat{\beta}_i = \max_{(\hat{z},\hat{t}) \in W} d_i^T \hat{z} + g_i \hat{t}, \\ \hat{L}_i^0 &= \min_{(\hat{z},\hat{t}) \in W} \frac{c_i^T \hat{z} + f_i \hat{t}}{d_i^T \hat{z} + g_i \hat{t}}, \quad \hat{U}_i^0 = \max_{(\hat{z},\hat{t}) \in W} \frac{c_i^T \hat{z} + f_i \hat{t}}{d_i^T \hat{z} + g_i \hat{t}}, \end{aligned}$$

since $d_i^T \hat{z} + g_i \hat{t}, i = 1, \ldots, p-1$, are all bounded affine functions over $W$, the values of $\hat{\alpha}_i$ and $\hat{\beta}_i$ can be easily computed by solving the corresponding linear programming problems. By Charnes-Cooper transformation [3], $\hat{L}_i^0$ and $\hat{U}_i^0, i = 1, 2, \ldots, p-1$, can also be computed by solving the corresponding linear programming problems, see Section 2 for details.

By introducing new variables

$$\hat{s}_i = \frac{c_i^T \hat{z} + f_i \hat{t}}{d_i^T \hat{z} + g_i \hat{t}}, \quad i = 1, 2, \ldots, p-1,$$

let the rectangle $\hat{\Lambda}^0 = \{\hat{s} \in R^{p-1} \mid \hat{L}_i^0 \le \hat{s}_i \le \hat{U}_i^0, i = 1, 2, \ldots, p-1\}$, we can establish the equivalent problem (8.2) of the problem (8.1) as follows:

$$\begin{aligned} \min \ & \lambda(\hat{s}, \hat{z}, \hat{t}) = \sum_{i=1}^{p-1} \hat{s}_i + c_p^T \hat{z} + f_p \hat{t}, \\ \text{s.t.} \ & \hat{s}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) = c_i^T \hat{z} + f_i \hat{t}, \quad i = 1, 2, \ldots, p-1, \\ & (\hat{z}, \hat{t}) \in W, \ \hat{s} \in \hat{\Lambda}^0. \end{aligned} \tag{8.2}$$

Next, for any $(\hat{z}, \hat{t}) \in W$, and for any

$$\hat{s} \in \Lambda = \big\{\hat{s} \in R^{p-1} \mid \hat{L}_i \le \hat{s}_i \le \hat{U}_i, \ i = 1, 2, \ldots, p-1\big\} \subseteq \hat{\Lambda}^0,$$

it follows that $\hat{\alpha}_i \le d_i^T \hat{z} + g_i \hat{t} \le \hat{\beta}_i$, $\hat{L}_i \le \hat{s}_i \le \hat{U}_i$, $i = 1, 2, \ldots, p-1$, and using the previous relaxation method presented in Section 3, we can obtain that

$$\max\big\{\hat{\alpha}_i \hat{s}_i + \hat{L}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) - \hat{\alpha}_i \hat{L}_i, \ \hat{\beta}_i \hat{s}_i + \hat{U}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) - \hat{\beta}_i \hat{U}_i\big\} \le \hat{s}_i \big(d_i^T \hat{z} + g_i \hat{t}\big),$$

$$\hat{s}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) \le \min\big\{\hat{\alpha}_i \hat{s}_i + \hat{U}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) - \hat{\alpha}_i \hat{U}_i, \ \hat{\beta}_i \hat{s}_i + \hat{L}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) - \hat{\beta}_i \hat{L}_i\big\}.$$

Based on the above discussions, we can construct the affine relaxation problem (8.3) of the problem (8.2) over $\hat{\Lambda}$ as follows:

$$\begin{aligned} \min \ & \sum_{i=1}^{p-1} \hat{s}_i + c_p^T \hat{z} + f_p \hat{t}, \\ \text{s.t.} \ & \hat{\alpha}_i \hat{s}_i + \hat{L}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) - \hat{\alpha}_i \hat{L}_i \le c_i^T \hat{z} + f_i \hat{t}, \quad i = 1, 2, \ldots, p-1, \\ & \hat{\beta}_i \hat{s}_i + \hat{U}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) - \hat{\beta}_i \hat{U}_i \le c_i^T \hat{z} + f_i \hat{t}, \quad i = 1, 2, \ldots, p-1, \\ & \hat{\alpha}_i \hat{s}_i + \hat{U}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) - \hat{\alpha}_i \hat{U}_i \ge c_i^T \hat{z} + f_i \hat{t}, \quad i = 1, 2, \ldots, p-1, \\ & \hat{\beta}_i \hat{s}_i + \hat{L}_i \big(d_i^T \hat{z} + g_i \hat{t}\big) - \hat{\beta}_i \hat{L}_i \ge c_i^T \hat{z} + f_i \hat{t}, \quad i = 1, 2, \ldots, p-1, \\ & d_p^T \hat{z} + g_p \hat{t} = 1, \quad A\hat{z} - b\hat{t} \le 0, \quad \hat{z} \ge 0, \quad \hat{t} > 0, \quad \hat{s} \in \Lambda. \end{aligned} \tag{8.3}$$

Based on the partition search of the image space $R^{p-1}$ of the fractional function in the objective function of the problem (8.1), and based on the affine relaxation problem (8.3) to determine the lower bound of the optimal value of the problem (8.1), we can construct a new branch-and-bound algorithm, which is similar to that in Section 5.3. Through the improvement, the branching process of the algorithm only occurs in $(p-1)$-dimensional image space $R^{p-1}$, instead of acting on $p, 2p$ or $n$-dimensional space, which will greatly mitigates the required computational efforts of the algorithm. Due to the length limitation of the article, we will not give the numerical experiment of the improved algorithm here. In the future research, we will give the details of the improved algorithm.

## 9. Concluding Remarks

We study the problem (1.1). By exploiting equivalent transformation and convex and concave envelope approximation of bilinear function, the initial problem (1.1) can be transformed into a series of affine relaxation problems. Using the image space search, the affine relaxation problem, and the image space region reduction technique, we propose an image space branch-reduction-bound algorithm to solve the problem (1.1). In contrast to the existing branch-and-bound algorithms, the computational superiorities of the algorithm are as follows:

(i) The branch-and-bound search takes place in the image space $R^p$, which will shorten the time required to find a global optimum solution of the problem (1.1).

(ii) By utilizing the characteristics of objective function of equivalent problem and the currently known upper bound of the algorithm, we construct an image space region reducing method, which provides a possibility for deleting the whole or a large part of the currently investigated image space region where there exists no any global optimal solution of the problem (2.4).

(iii) Our algorithm can find an $\epsilon$-approximate global optimal solution in at most $\sum_{t=0}^{r_p} 2^{p.t}$ iterations, where

$$r_p = \left\lceil \log_2 \frac{C_p \cdot \delta(\Lambda_p)}{\epsilon} \right\rceil,$$
$$\delta(\Lambda_p) = \max\left\{\delta\left(\Lambda_p^l\right) : l \in \{1, 2, \ldots, \varrho\}\right\}.$$

Numerical results indicate that the presented algorithm has higher computational efficiency.

The future work is to give a further improvement of the algorithm and extend our new algorithm to solve the min-max affine fractional programming problem.

## Appendix A

**A.1** (cf. Benson [1]).

$$\begin{aligned}
\min \quad & f(x) = \frac{-3.333x_1 - 3x_2 - 1}{1.666x_1 + x_2 + 1} + \frac{-4x_1 - 3x_2 - 1}{x_1 + x_2 + 1} \\
\text{s.t.} \quad & 5x_1 + 4x_2 \le 10, \\
& -x_1 \le -0.1, \\
& -x_2 \le -0.1, \\
& -2x_1 - x_2 \le -2, \\
& x_1, x_2 \ge 0.
\end{aligned}$$

Example A.1 has a local minimum point at $x = (1.92, 0.1)$ with $f(x) = -4.7642$ and a global minimum point at $x^* = (0.1, 2.3750)$ with $f(x^*) = -4.8415$.

**A.2** (cf. Phuong and Tuy [24] and Shen *et al.* [27]).

$$\max \quad \frac{3x_1 + x_2 - 2x_3 + 0.8}{2x_1 - x_2 + x_3} + \frac{4x_1 - 2x_2 + x_3}{7x_1 + 3x_2 - x_3}$$

$$\begin{aligned}
\text{s.t.}\quad & x_1 + x_2 - x_3 \le 1,\\
& -x_1 + x_2 - x_3 \le -1,\\
& 12x_1 + 5x_2 + 12x_3 \le 34.8,\\
& 12x_1 + 12x_2 + 7x_3 \le 29.1,\\
& -6x_1 + x_2 + x_3 \le -4.1.
\end{aligned}$$

**A.3** (cf. Shen *et al.* [27] and Shen and Wang [30]).

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

**A.4** (cf. Shen *et al.* [27]).

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

**A.5** (cf. Shen and Lu [29] and Gao and Jin [5]).

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

**A.6** (cf. Ji *et al.* [7] and Shen and Lu [29]).

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

**A.7** (cf. Shen and Lu [29] and Gao and Jin [5]).

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

**A.8** (cf. Ji *et al.* [7]) and Shen and Lu [29].

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

**A.9** (cf. Shen and Lu [29] and Gao and Jin [5]).

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

**A.10** (cf. Jiao and Liu [8], Shen and Wang [30] and Jiao *et al.* [9]).

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

**A.11** (cf. Shen and Wang [30] and Shi [33]).

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

**A.12** (cf. Shi [33]).

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

**Acknowledgments.** The authors are very grateful to the responsible editors and the anonymous referees for their valuable comments and suggestions, which have greatly improved the earlier version of this paper.

This work is supported by the National Natural Science Foundation of China (Grant Nos. 11871196, 12071133, 12071112), by the Key Scientific and Technological Research Projects of Henan Province (Grant Nos. 232102211085, 202102210147), by the China Postdoctoral Science Foundation (Grant No. 2017M622340), and by the Science and Technology Climbing Program of Henan Institute of Science and Technology (Grant No. 2018JY01).

## References

[1] 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.

[2] A. Cambini, L. Martein, and S. Schaible, On maximizing a sum of ratios, *J. Inform. Optim. Sci.*, **10** (1989), 65–79.

[3] A. Charnes and W.W. Cooper, Programming with linear fractional functionals, *Nav. Res. Logist. Q.*, **9** (1962), 181–186.

[4] J.E. Falk and S.W. Palocsay, Image space analysis of generalized fractional programs, *J. Global Optim.*, **4** (1994), 63–88.

[5] Y. Gao and S. Jin, A global optimization algorithm for sum of linear ratios problem, *J. Appl. Math.*, **2013** (2013), 276245.

[6] R. Horst and H. Tuy, *Global Optimization: Deterministic Approaches*, Springer, 1993.

[7] Y. Ji, K. Zhang, and S. Qu, A deterministic global optimization algorithm, *Appl. Math. Comput.*, **185** (2007), 382–387.

[8] H. Jiao and S. Liu, A practicable branch and bound algorithm for sum of linear ratios problem, *Eur. J. Oper. Res.*, **243** (2015), 723–730.

[9] H. Jiao, S. Liu, J. Yin, and Y. Zhao, Outcome space range reduction method for global optimization of sum of affine ratios problem, *Open Math.*, **14**:1 (2016), 736–746.

[10] H. Jiao, J. Ma, and Y. Shang, Image space branch-and-bound algorithm for globally solving minimax linear fractional programming problem, *Pac. J. Optim.*, **18**:1 (2022), 195–212.

[11] H.W. Jiao and Y.L. Shang, Two-level linear relaxation method for generalized linear fractional programming, *J. Oper. Res. Soc. China*, **11** (2023), 569–594.

[12] H. Jiao, Y. Shang, and R. Chen, A potential practical algorithm for minimizing the sum of affine fractional functions, *Optimization*, **72** (2023), 1577–1607.

[13] H. Jiao, Y. Shang, and W. Wang, Solving generalized polynomial problem by using new affine relaxed technique, *Int. J. Comput. Math.*, **99**:2 (2022), 309–331.

[14] H. Jiao, Z. Wang, and Y. Chen, Global optimization algorithm for sum of generalized polynomial ratios problem, *Appl. Math. Model.*, **37** (2013), 187–197.
[15] A. Khajavirad and N.V. Sahinidis, A hybrid LP/NLP paradigm for global optimization relaxations, *Math. Program. Comput.*, **10** (2018), 383–421.
[16] H. Konno and K. Fukaishi, A branch-and-bound algorithm for solving low-rank linear multiplicative and fractional programming problems, *J. Global Optim.*, **18** (2000), 283–299.
[17] H. Konno and 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.
[18] H. Konno, Y. Yajima, and T. Matsui, Parametric simplex algorithms for solving a special class of nonconvex minimization problems, *J. Glob. Optim.*, **1** (1991), 65–81.
[19] H. Konno and H. Yamashita, Minimizing sums and products of linear fractional functions over a polytope, *Nav. Res. Logist.*, **46** (1999), 583–596.
[20] T. Kuno, A branch-and-bound algorithm for maximizing the sum of several linear ratios, *J. Global Optim.*, **22** (2002), 155–174.
[21] T. Kuno, A revision of the trapezoidal branch-and-bound algorithm for linear sum-of-ratios problems, *J. Global Optim.*, **33** (2005), 215–234.
[22] T. Kuno and T. Masaki, A practical but rigorous approach to sum-of-ratios optimization in geometric applications, *Comput. Optim. Appl.*, **54** (2013), 93–109.
[23] X. Liu, Y. Gao, B. Zhang, and F. Tian, A new global optimization algorithm for a class of linear fractional programming, *Mathematics*, **7** (2019), 867.
[24] N.T.H. Phuong and H. Tuy, A unified monotonic approach to generalized linear fractional programming, *J. Global Optim.*, **26** (2003), 229–259.
[25] J. Quesada and I. Grossmann, A global optimization algorithm for linear fractional and bilinear programs, *J. Global Optim.*, **6** (1995), 39–76.
[26] S. Schaible and J. Shi, Fractional programming: The sum-of-ratios case, *Optim. Method Softw.*, **18** (2003), 219–229.
[27] P. Shen, B. Huang, and L. Wang, Range division and linearization algorithm for a class of linear ratios optimization problems, *J. Comput. Appl. Math.*, **350** (2019), 324–342.
[28] P. Shen, W. Li, and X. Bai, Maximizing for the sum of ratios of two convex functions over a convex set, *Comput. Oper. Res.*, **40**:10 (2013), 2301–2307.
[29] P.P. Shen and T. Lu, Regional division and reduction algorithm for minimizing the sum of linear fractional functions, *J. Inequal. Appl.*, **2018** (2018), 63.
[30] P. Shen and C. Wang, Global optimization for sum of linear ratios problem with coefficients, *Appl. Math. Comput.*, **176** (2006), 219–229.
[31] P. Shen and C. Wang, Linear decomposition approach for a class of nonconvex programming problems, *J. Inequal. Appl.*, **2017** (2017), 74.
[32] H.D. Sherali and W.P. Adams, *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*, in: *Nonconvex Optimization and Its Applications*, Kluwer Academic Publishers, **31**, 1999.
[33] Y. Shi, *Global Optimization for Sum of Ratios Problems*, MA Thesis, Henan Normal University, 2011. (in Chinese)
[34] I.M. Stancu-Minasian, *Fractional Programming: Theory, Methods and Applications*, Kluwer Academic Publishers, 1997.
[35] I.M. Stancu-Minasian, A eighth bibliography of fractional programming, *Optimization*, **66**:3 (2017), 439–470.
[36] I.M. Stancu-Minasian, A ninth bibliography of fractional programming, *Optimization*, **68**:11, (2019), 2123–2167.
[37] Y. Xia and L. Wang, A new global optimization algorithm for minimizing the sum of two linear ratios, *J. Henan Normal University (Natural Science Edition)*, **46**:1 (2018), 9–15. (in Chinese)