Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 18, No. 1, pp. 177–194, 2022

GLOBAL OPTIMIZATION ALGORITHM FOR SOLVING SUM OF LINEAR RATIOS PROBLEMS[*]

Bingdi Huang and Peiping Shen[†]

Abstract: In this paper, we consider the sum of linear ratios problems (P), such problems often arise from numerous applications such as transportation problem. We first show problem (P) can be transformed into its equivalent problem (EP) via introducing auxiliary variables, and problem (EP) can be further reformulated as a two-layer problem (EP1). According to the special structure of problem (EP1), a novel convex relaxation technique is proposed to design a new branch and bound algorithm. Besides, the analysis of the convergence and computational complexity of the presented algorithm are given. Finally, preliminary experimental results illustrate that the developed algorithm is feasible and effective.

Key words: *sum-of-linear-ratios optimization, global optimization, branch and bound, computational complexity*

Mathematics Subject Classification: *90C30, 90C33, 90C15*

1 Introduction

Consider the following sum of linear ratios problems:

$$(\mathrm{P}): \begin{cases} v(\mathrm{P}) = & \min \quad \phi(x) = \sum\limits_{i=1}^{p} \dfrac{f_i(x)}{g_i(x)} \\ & \text{s.t.} \quad x \in \Omega = \{x \in R^n | Ax \le b, x \ge 0\}, \end{cases}$$

where $f_i(x) = c_i^\top x + c_{0i}, g_i(x) = d_i^\top x + d_{0i}, i = 1, \cdots, p$, with $c_i, d_i \in R^n, c_{0i}, d_{0i} \in R$, and Ω is a nonempty compact set with $b \in R^m$, $A \in R^{m\times n}$. As we have either $g_i(x) > 0$ or $g_i(x) < 0$ for all $x \in \Omega$, if there exists some $i \in \{1, \cdots, p\}$ such that $g_i(x) < 0$, we can replace $\frac{f_i(x)}{g_i(x)}$ with $\frac{-f_i(x)}{-g_i(x)}$, the prime problem keeps unaltered, so we can assume that $g_i(x) > 0$ for all $x \in \Omega$. What's more, if there exists some $i \in \{1, \cdots, p\}$ which makes $f_i(x) < 0$, we may substitute $\frac{f_i(x)}{g_i(x)}$ with $\frac{f_i(x)+Mg_i(x)}{g_i(x)}$, where M is a large enough number such that $f_i(x) + Mg_i(x) > 0$ for any $x \in \Omega$, the prime problem still remains unchanged, thus, we can also assume that $f_i(x) > 0$ for all $x \in \Omega$. In the following, without loss of generality, we suppose that $f_i(x) > 0$, $g_i(x) > 0$, $i = 1, \cdots, p$, for all $x \in \Omega$.

The sum of linear ratios problem (P) covers the classic transportation problem [2], and the production planning problems can also be cast as a special case of problem (P) [23]. In

[*]This work was supported by the National Natural Science Foundation of China (No. 12071133, 11871196).

[†]Corresponding author

© 2022 Yokohama Publishers

addition, problem (P) also appears in finance and investment [7], minimum ratio spanning tree problem [32], multi-objective bond portfolio optimization problem [18, 19], multi-stage stochastic shipping problem [1], MIMO networks [16], economics [27], geometric problem and other practical problems [6]. In theory, problem (P) is NP-hard [10], and it possesses many local optimal solutions, which increases the difficulty of finding the global solution of problem (P).

During the past several decades, effective global algorithms for sum of linear ratios problems have been reported in the literature. In the case of $p = 1$, problem (P) is called single ratio linear fractional programming problem, for this problem, Ozkok [23] presented a novel iterative algorithm based on the $(\varepsilon, \delta)-$definition of continuity. When $p = 2$, and the denominator of one of the linear fractions is 1, Xia et al. [35] developed an efficient branch-and-bound algorithm on the basis of improving the existing sawtooth-curve bounds to new wave-curve bounds. When $p > 2$, Benson [3] presented and validated a simplicial branch and bound duality bounds algorithm for globally solving the linear sum-of-ratios fractional program. Based on a linear relaxation of the objective function, Carlsson and Shi [5] introduced a global algorithm for solving the sum-of-linear-ratios problem with lower dimension. A branch and bound outer approximation algorithm was shown by Benson [4] for globally solving this class of problems. Moreover, Jiao et al. proposed an outcome space range reduction method [15], Shen et al. developed a regional division and reduction method [29] and a branch-reduction-bound algorithm [28], and a new branch and reduce approach was presented by Zhang and Wang [37] for solving generalized linear fractional programming. By means of classifying the coefficient symbols of all linear functions in the objective function, an output-space branch and bound algorithm based on solving the linear programming problem was proposed by Liu et al. [21]. In addition, based on different relaxation strategies, various branch and bound algorithms were proposed in [11, 13, 14, 30, 31, 33, 36, 38]. Different from the aforementioned global optimization approach, Xia et al. [34] developed a fully polynomial time approximation scheme for minimizing the sum of linear fractional functions over the cone of positive semidefinite matrices. Besides, two approximation algorithms for a class of linear ratios optimization problems were designed by Shen et al. [25], Depetrini and Locatelli [8], respectively. And Phuong and Tuy [24] introduced an uniform monotonic method for solving this class of problems. Apart from the approaches mentioned above, some practical algorithms have been utilized for solving the sum of linear ratios problems, such as parametric simplex method [20], image space method [9], heuristic method [17], and interior point algorithm [22].

The main purpose of this paper is to present a reliable and effective algorithm for globally solving problem (P). To this end, we first show that problem (P) can be equivalently converted into problem (EP) by introducing auxiliary variables, and problem (EP) can be further represented as a two-layer problem (EP1). It can be proved that problems (P), (EP) and (EP1) have the same global optimal value. For purpose of acquiring the lower bound of the optimal value for problem (P), a novel convex relaxation problem (CRP) is derived based on problem (EP1). Then a new branch and bound method is designed to globally solve problem (P). The main computational effort of the presented algorithm lies in solving a series of linear programming problems. In addition, the convergence and computational complexity of the algorithm are given. The branch and bound algorithm proposed in our work is closely related to the method in [35], where the authors considered the problem of minimizing the sum of a convex-concave function and a convex function over a convex set (SFC). When $p = 2$, and the denominator of one of the linear fractions is 1, problem (P) reduces to a special case of problem (SFC). Similar to the approach in [35], we reformulate problem (P) as a box constrained minimization problem, where the objective function is evaluated by solving linear subproblems. The optimal Lagrangian multipliers of the linear subproblems are used to construct lower bounding functions for the objective of the box constrained minimization problem. However, as we shall see later, the method in [35] is tailored to the case that $p = 2$, while the presented algorithm in this work can deal with more general case such as $p > 2$. Besides, compared with the existing branch and bound method in [14], the main features of the introduced algorithm are listed as follows. To begin with, the computational complexity of the proposed algorithm is estimated, which is not available in [14]. Meanwhile, the number of linear subproblems need to be solved at each iteration in our algorithm is one less than that of the approach in [14], and the structure of the subproblem in our method is comparatively simple, concretely, the number of variable for the linear subproblem in our algorithm is $2p$ less than that of [14], and the number of constraint for the linear subproblem in our algorithm is $3p$ less than that of [14]. Finally, the experimental results for Examples 1-2 indicate that the introduced method can solve practical problems effectively. And the numerical results in Tables 2-4 show that the proposed approach requires much less CPU time than that of the algorithms in [14, 25, 35] for finding the global optimal solutions of the tested instances.

The contributions of this work are twofold. First, we consider the equivalent transformation of problem (P) on the basis of layering idea, and construct a novel convex relaxation which has a closed form solution to obtain the lower bound of the optimal value for problem (P). Based on the convex relaxation, we introduce a branch and bound algorithm to globally solve problem (P). Second, the computational complexity analysis of the proposed algorithm is presented, and the maximum iterations of the algorithm is estimated. Preliminary numerical results demonstrate that the developed algorithm outperforms the algorithms in [14, 25, 35] for all the tested instances.

The remainder of this paper is organized as follows. The equivalent problems (EP) and (EP1) of the problem (P) are constructed in Section 2. Section 3 presents an innovative convex relaxation technique. In Section 4, a new branch and bound algorithm for globally solving problem (P) is introduced, and the analysis of the convergence and computational complexity for the algorithm are given. In Section 5, we give two application examples of problem (P): the transportation problem [2] and the production planning problems [23]. And the numerical results for the tested instances are reported. Finally, some conclusions are gained in Section 6.

2 Equivalent Problem

In this section, for solving problem (P), we first establish a problem (EP) which is equivalent to problem (P), and then problem (EP) is represented as a two-layer problem (EP1), besides, it will be proved that problems (P), (EP) and (EP1) have the same optimal value.

In order to construct problem (EP), we introduce the additional variables $t_i,\ i = 1, \cdots, p$, and define the initial box T^0 as follows.

$$T^0 = [\underline{t}^0, \bar{t}^0] \triangleq \{t \in R^p |\ \underline{t}_i^0 \le t_i \le \bar{t}_i^0, i = 1, \cdots, p\},$$

where $\underline{t}_i^0 = \min\{g_i(x) \mid x \in \Omega\}, \bar{t}_i^0 = \max\{g_i(x) \mid x \in \Omega\}, i = 1, \cdots, p$. Then problem (P) can be equivalently converted to the following form:

$$(\text{EP}) : \begin{cases} v(\text{EP}) = & \min \quad \psi(x,t) = \sum\limits_{i=1}^{p} \dfrac{f_i(x)}{t_i} \\ & \text{s.t.} \quad g_i(x) \ge t_i,\ i = 1, \cdots, p, \\ & \qquad\ x \in \Omega,\ t \in T^0. \end{cases}$$

The equivalent theorem for problems (P) and (EP) is presented as follows.

Theorem 2.1. *If x^* is a global optimal solution of problem* (P)*, then (x^*, t^*) is a global optimal solution of problem* (EP) *with $t_i^* = g_i(x^*), i = 1, \ldots, p$. Conversely, if (x^*, t^*) is a global optimal solution of problem* (EP)*, then x^* is a global optimal solution of problem* (P)*. Additionally, the global optimal values of problems* (P) *and* (EP) *are equal.*

Proof. If x^* is a global optimal solution of problem (P), let $t_i^* = g_i(x^*), i = 1, \ldots p$, then it is clear that (x^*, t^*) is a feasible solution of problem (EP). Suppose that (x^*, t^*) is not an optimal solution of problem (EP), then there exists a feasible solution $(\hat{x}, \hat{t})$ of problem (EP) which satisfies

$$\psi(\hat{x}, \hat{t}) = \sum_{i=1}^{p} \frac{f_i(\hat{x})}{\hat{t}_i} < \sum_{i=1}^{p} \frac{f_i(x^*)}{t_i^*} = \psi(x^*, t^*). \tag{2.1}$$

Notice that $\hat{x} \in \Omega, g_i(\hat{x}) \geq \hat{t}_i, i = 1, \ldots p$, from (2.1) we can get

$$\phi(\hat{x}) = \sum_{i=1}^{p} \frac{f_i(\hat{x})}{g_i(\hat{x})} \leq \sum_{i=1}^{p} \frac{f_i(\hat{x})}{\hat{t}_i} < \sum_{i=1}^{p} \frac{f_i(x^*)}{t_i^*} = \sum_{i=1}^{p} \frac{f_i(x^*)}{g_i(x^*)} = \phi(x^*),$$

which contradicts the fact that x^* is a global optimal solution of problem (P). So (x^*, t^*) is a global optimal solution of problem (EP).

Conversely, if (x^*, t^*) is a global optimal solution of problem (EP), then x^* is feasible to problem (P), assume that x^* is not an optimal solution of problem (P), then there exists $\hat{x} \in \Omega$ such that

$$\phi(\hat{x}) = \sum_{i=1}^{p} \frac{f_i(\hat{x})}{g_i(\hat{x})} < \sum_{i=1}^{p} \frac{f_i(x^*)}{g_i(x^*)} = \phi(x^*). \tag{2.2}$$

Let $\hat{t}_i = g_i(\hat{x}), i = 1, \ldots p$, then it is obvious that $(\hat{x}, \hat{t})$ is a feasible solution of problem (EP). Therefore, from (2.2) we have

$$\psi(\hat{x}, \hat{t}) = \sum_{i=1}^{p} \frac{f_i(\hat{x})}{\hat{t}_i} = \sum_{i=1}^{p} \frac{f_i(\hat{x})}{g_i(\hat{x})} < \sum_{i=1}^{p} \frac{f_i(x^*)}{g_i(x^*)} \leq \sum_{i=1}^{p} \frac{f_i(x^*)}{t_i^*} = \psi(x^*, t^*),$$

which contradicts the optimality of (x^*, t^*) for problem (EP). As a result, x^* is a global optimal solution of problem (P).

Moreover, it can be concluded that the global optimal solution (x^*, t^*) of problem (EP) satisfies $t_i^* = g_i(x^*),\ i = 1, \ldots p$. If not, there exists $g_j(x^*) > t_j^*$ for at least one $j \in \{1, \ldots p\}$, let $t^{'} = (t_i^{'})_{p\times 1}$ with $t_j^{'} = g_j(x^*) > t_j^*,\ t_i^{'} = t_i^*, i = 1, \cdots, p, i \neq j$, then $(x^*, t^{'})$ is feasible to problem (EP), and we have

$$\psi(x^*, t^{'}) = \sum_{i=1, i\neq j}^{p} \frac{f_i(x^*)}{t_i^{'}} + \frac{f_j(x^*)}{t_j^{'}} < \sum_{i=1, i\neq j}^{p} \frac{f_i(x^*)}{t_i^*} + \frac{f_j(x^*)}{t_j^*} = \psi(x^*, t^*),$$

which contradicts the optimality of (x^*, t^*) for problem (EP). Thus, it follows that

$$v(\text{EP}) = \sum_{i=1}^{p} \frac{f_i(x^*)}{t_i^*} = \sum_{i=1}^{p} \frac{f_i(x^*)}{g_i(x^*)} = \phi(x^*) = v(\text{P}).$$

The proof is completed. □

Next, to solve problem (EP), we first denote

$$X(t) = \{x \in R^n \mid x \in \Omega, g_i(x) \geq t_i, i = 1, \cdots, p\}. \tag{2.3}$$

Then problem (EP) can be transformed into a two-layer problem as follows:

$$(\mathrm{EP1}) : \begin{cases} h_{T^0} = & \min \quad h(t) \\ & \text{s.t.} \quad t \in T^0, \end{cases}$$

where $h(t)$ is the optimal value of the following optimization problem with parameter t:

$$(\mathrm{P_t}) : \begin{cases} h(t) = & \min \quad \psi(x,t) = \sum\limits_{i=1}^{p} \dfrac{f_i(x)}{t_i} \\ & \text{s.t.} \quad x \in X(t). \end{cases}$$

The key equivalence result for problems (EP) and (EP1) is given by the following theorem.

Theorem 2.2. *If t^* is a global optimal solution of problem* (EP1)*, then (x^*, t^*) is a global optimal solution of problem* (EP) *with $x^* = \operatorname{argmin}\{\psi(x, t^*) \mid x \in X(t^*)\}$. Conversely, if (x^*, t^*) is a global optimal solution of problem* (EP)*, then t^* is a global optimal solution of problem* (EP1)*. Moreover, problems* (EP) *and* (EP1) *have the same global optimal value.*

Proof. If t^* is a global optimal solution of problem (EP1), let

$$x^* = \operatorname{argmin}\{\psi(x, t^*) \mid x \in X(t^*)\},$$

then it is clear that (x^*, t^*) is a feasible solution of problem (EP). Suppose that (x^*, t^*) is not an optimal solution of problem (EP), then there exists a feasible solution $(\hat{x}, \hat{t})$ of problem (EP) which satisfies

$$\psi(\hat{x}, \hat{t}) < \psi(x^*, t^*). \tag{2.4}$$

Besides, it is obvious that $\hat{x} \in X(\hat{t})$, thus $\hat{x}$ is feasible to problem $(\mathrm{P}_{\hat{t}})$, from (2.4) we have

$$h(\hat{t}) = \min_{x \in X(\hat{t})} \psi(x, \hat{t}) \leq \psi(\hat{x}, \hat{t}) < \psi(x^*, t^*) = h(t^*). \tag{2.5}$$

Since $\hat{t}$ is feasible to problem (EP1), the formula (2.5) contradicts the fact that t^* is an optimal solution of problem (EP1), then it can be concluded that (x^*, t^*) is a global optimal solution of problem (EP).

Conversely, if (x^*, t^*) is a global optimal solution of problem (EP), then t^* is feasible to problem (EP1), and $x^* \in X(t^*)$ is a feasible solution of problem (P_{t^*}). Assume that t^* is not an optimal solution of problem (EP1), then there exists $\hat{t} \in T^0$ satisfying

$$h(\hat{t}) < h(t^*). \tag{2.6}$$

Denote

$$x(\hat{t}) = \operatorname{argmin}\{\psi(x, \hat{t}) \mid x \in X(\hat{t})\},$$

and apply (2.6), it holds that

$$\psi(x(\hat{t}), \hat{t}) = h(\hat{t}) < h(t^*) = \min_{x \in X(t^*)} \psi(x, t^*) \leq \psi(x^*, t^*). \tag{2.7}$$

Due to the feasibility of $(x(\hat{t}), \hat{t})$ for problem (EP), the formula (2.7) contradicts the optimality of (x^*, t^*) for problem (EP). As a result, t^* is a global optimal solution of problem (EP1).

Based on the above results, we know that if t^* is a global optimal solution of problem (EP1), then (x^*, t^*) is a global optimal solution of problem (EP) with $x^* = \operatorname{argmin}\{\psi(x, t^*) \mid x \in X(t^*)\}$. Thus, we have

$$h(t^*) = \psi(x^*, t^*).$$

The proof is completed. □

Based on Theorems 2.1 and 2.2, it can be seen that problems (P), (EP) and (EP1) are equivalent to each other, and their optimal values satisfy:

$$v(\mathrm{P}) = v(\mathrm{EP}) = h_{T^0}. \tag{2.8}$$

In next section, a new convex relaxation problem for problem (EP1) will be shown, which can provide a lower bound for h_{T^0}.

3 Convex Relaxation Technique

In this section, for acquiring the lower bound of h_{T^0}, we need to establish the convex relaxation program for problem (EP1). With regards to this, the main approach is to construct the underestimated convex function for the objective function $h(t)$ of problem (EP1).

For the sake of discussion, assume that

$$T = [\underline{t}, \overline{t}] \triangleq \{t \in R^p \mid \underline{t}_i \le t_i \le \overline{t}_i, i = 1, \cdots, p\},$$

denotes T^0 or a subrectangle of T^0 which is generated by the branching process in the branch and bound algorithm to be presented. Therefore, the corresponding problem (EP1) on T can be rewritten as follows:

$$(\mathrm{EP1(T)}): \left\{ \begin{array}{lll} h_T = & \min & h(t) \\ & \text{s.t.} & t \in T. \end{array} \right.$$

Next, to construct a convex lower bounding function for $h(t)$ over the box $T = [\underline{t}, \overline{t}]$, we first fix $t = \underline{t}$ in problem $(\mathrm{P_t})$, then problem $(\mathrm{P}_{\underline{t}})$ is a linear program with decision variable x as follows:

$$(\mathrm{P}_{\underline{t}}): \left\{ \begin{array}{lll} h(\underline{t}) = & \min & \psi(x, \underline{t}) = \sum_{i=1}^{p} \dfrac{f_i(x)}{\underline{t}_i} \\ & \text{s.t.} & g_i(x) \ge \underline{t}_i, \; i = 1, \cdots, p, \\ & & x \in \Omega. \end{array} \right.$$

It is easy to get the dual problem $(\mathrm{DP}_{\underline{t}})$ for problem $(\mathrm{P}_{\underline{t}})$, given by

$$(\mathrm{DP}_{\underline{t}}): dh(\underline{t}) = \max_{\lambda \succeq 0} \min_{x \in \Omega} \sum_{i=1}^{p} (\frac{f_i(x)}{\underline{t}_i} + \lambda_i(-g_i(x) + \underline{t}_i)).$$

According to the strong duality for linear program, we have $h(\underline{t}) = dh(\underline{t})$. So it follows that

$$\begin{aligned} h(\underline{t}) &= \max_{\lambda \succeq 0} \min_{x \in \Omega} \sum_{i=1}^{p} (\frac{f_i(x)}{\underline{t}_i} + \lambda_i(-g_i(x) + \underline{t}_i)) \\ &= \min_{x \in \Omega} \sum_{i=1}^{p} (\frac{f_i(x)}{\underline{t}_i} + \lambda_i^*(-g_i(x) + \underline{t}_i)) \end{aligned} \tag{3.1}$$

where $\lambda^* \in R^p_+$ is the optimal Lagrange multiplier corresponding to the constraints $g_i(x) \geq \underline{t}_i,\ i = 1, \cdots, p$.

Similar to the above process, for any fixed $t \in T$, it holds that

$$\begin{aligned} h(t) &= \max_{\lambda \succeq 0} \min_{x \in \Omega} \sum_{i=1}^{p} (\frac{f_i(x)}{t_i} + \lambda_i(-g_i(x) + t_i)) \\ &\geq \min_{x \in \Omega} \sum_{i=1}^{p} (\frac{f_i(x)}{t_i} + \lambda_i^*(-g_i(x) + t_i)). \end{aligned} \tag{3.2}$$

It easy to see that

$$\sum_{i=1}^{p} (\frac{f_i(x)}{t_i} + \lambda_i^*(-g_i(x) + t_i)) = \sum_{i=1}^{p} (\frac{f_i(x)}{\underline{t}_i} + \lambda_i^*(-g_i(x) + \underline{t}_i) + f_i(x)(\frac{1}{t_i} - \frac{1}{\underline{t}_i}) + \lambda_i^*(t_i - \underline{t}_i)).$$

Then from (3.2), we have

$$h(t) \geq \min_{x \in \Omega} \sum_{i=1}^{p} (\frac{f_i(x)}{\underline{t}_i} + \lambda_i^*(-g_i(x) + \underline{t}_i) + f_i(x)(\frac{1}{t_i} - \frac{1}{\underline{t}_i}) + \lambda_i^*(t_i - \underline{t}_i)). \tag{3.3}$$

Next, in order to acquire the lower convex function of $h(t)$, let us denote

$$\bar{f}_i = \max_{x \in \Omega} f_i(x), i = 1, \cdots, p, \tag{3.4}$$

then apply (3.1) and (3.3), it holds that

$$\begin{aligned} h(t) &\geq \min_{x \in \Omega} \sum_{i=1}^{p} (\frac{f_i(x)}{\underline{t}_i} + \lambda_i^*(-g_i(x) + \underline{t}_i)) \\ &\quad + \sum_{i=1}^{p} (\bar{f}_i(\frac{1}{t_i} - \frac{1}{\underline{t}_i}) + \lambda_i^*(t_i - \underline{t}_i)) \\ &= h(\underline{t}) + \sum_{i=1}^{p} (\bar{f}_i(\frac{1}{t_i} - \frac{1}{\underline{t}_i}) + \lambda_i^*(t_i - \underline{t}_i)). \end{aligned}$$

Therefore, the lower bounding function of $h(t)$ over T can be defined as

$$\underline{h}(t) = h(\underline{t}) + \sum_{i=1}^{p} (\bar{f}_i(\frac{1}{t_i} - \frac{1}{\underline{t}_i}) + \lambda_i^*(t_i - \underline{t}_i)),$$

which is a simple convex function about the variable $t \in R^p$. Now, assume that

$$\delta = h(\underline{t}) - \sum_{i=1}^{p} (\frac{\bar{f}_i}{\underline{t}_i} + \lambda_i^* \underline{t}_i),$$

then $\underline{h}(t)$ can be rewritten as

$$\underline{h}(t) = \delta + \sum_{i=1}^{p} (\frac{\bar{f}_i}{t_i} + \lambda_i^* t_i). \tag{3.5}$$

From the above discussion, we know that $h(t) \geq \underline{h}(t)$ holds for any fixed $t \in T$. Thus, we can obtain the convex relaxation problem (CRP(T)) for problem (EP1(T)) as follows:

$$(\mathrm{CRP(T)}): \left\{ \begin{array}{lll} \underline{h}_T = & \min & \underline{h}(t) \\ & \text{s.t.} & t \in T. \end{array} \right.$$

Note that problem (CRP(T)) is separable, so the solution process of (CRP(T)) can be decomposed into solving p simple univariate convex subproblems, each subproblem has the form:

$$(\mathrm{CRP(T_i)}): \left\{ \begin{array}{lll} \underline{h}_{T_i} = & \min & \underline{h}^i(t) = \dfrac{\bar{f}_i}{t_i} + \lambda_i^* t_i, \\ & \text{s.t.} & t_i \in T_i, \end{array} \right.$$

where $T_i = [\underline{t}_i, \bar{t}_i]$. It is easy to see that the optimal value of problem (CRP(T$_\mathrm{i}$)) satisfies:

$$\underline{h}_{T_i} = \left\{ \begin{array}{ll} \underline{h}^i(\tilde{t}_i), & \text{if } \underline{t}_i < \tilde{t}_i < \bar{t}_i, \\ \min\{\underline{h}^i(\underline{t}_i), \underline{h}^i(\bar{t}_i)\}, & \text{otherwise,} \end{array} \right.$$

where $\tilde{t}_i = \sqrt{\frac{\bar{f}_i}{\lambda_i^*}}, i = 1, \cdots, p$. Therefore, the optimal value of problem (CRP(T)) can be obtained by

$$\underline{h}_T = \delta + \sum_{i=1}^{p} \underline{h}_{T_i}.$$

Based on the above results, it can be seen that the optimal value of problem (CRP(T)) provides a valid lower bound for the optimal value of problem (EP1(T)).

4 Algorithm and its Theoretical Analysis

In this section, on the basis of the former results, a new branch and bound algorithm for solving problem (P) is developed, and the analysis of the convergence and computational complexity of the proposed algorithm are then given.

4.1 Branch and bound algorithm

In each iterative step of the proposed branch and bound algorithm, we utilize the standard bisection rule which subdivides the rectangle T along the midpoint of its longest edge, and this branching rule can guarantee the convergence of the algorithm.

For each subrectangle $T = [\underline{t}, \bar{t}]$ generated by the branching process, solve linear problem (P$_{\underline{\mathrm{t}}}$) by using the linear programming solver "cplexlp" to obtain its optimal solution $\bar{x}$ and optimal value $h(\underline{t})$, and the corresponding optimal multiplier $\bar{\lambda}$ can be gained as an output parameter in "cplexlp". And then compute the optimal value $\underline{h}_T$ to (CRP(T)) to get a lower bound $LB(T)$ of h_T by letting $LB(T) = \underline{h}_T$. Additionally, notice that $\bar{x}$ is a feasible solution for problem (P), thus we can update the current upper bound UB of the optimal value v(P) to problem (P) via setting $UB = \phi(\bar{x})$. More precisely, the basic steps of the proposed branch and bound algorithm (denoted by BB for short) for globally solving problem (P) are summarized as follows.

BB Algorithm:

Step 0: Given $\varepsilon \geq 0$ and the initial box $T^0 = [\underline{t}^0, \bar{t}^0]$.

Step 1: Solve problem (P$_{\underline{\mathrm{t}}^0}$) to get its optimal solution $\bar{x}^0$, optimal multiplier $\bar{\lambda}^0$ and optimal value $h(\underline{t}^0)$, and then calculate the optimal solution $\tilde{t}^0$ and optimal value $\underline{h}_{T^0}$ to problem (CRP(T^0)). Let $LB_0 = \underline{h}_{T^0}$, $UB_0 = \phi(\bar{x}^0), \tilde{x}^0 = \bar{x}^0$. If $UB_0 - LB_0 \leq \varepsilon$, then the algorithm stops: $\tilde{x}^0$ is a global $\varepsilon-$optimal solution for problem (P). Otherwise, let $k = 1, T^1 = T^0, S = \{T^1\}$. Proceed to Step 2.

Step 2: Use the branching rule to subdivide T^k into two new sub-rectangles $T^{k1} = [\underline{t}^{k1}, \bar{t}^{k1}]$ and $T^{k2} = [\underline{t}^{k2}, \bar{t}^{k2}]$, set $\underline{t}^k = \underline{t}^{k2}, H = \{T^{k1}, T^{k2}\}$.

Step 3: Solve problem (P$_{\underline{t}^k}$) to obtain its optimal solution $\bar{x}^k$, optimal multiplier $\bar{\lambda}^k$ and optimal value $h(\underline{t}^k)$. For each rectangle $T^{ks}(s = 1, 2)$, compute the optimal solution $\tilde{t}^{ks}$ and optimal value $\underline{h}_{T^{ks}}$ to problem (CRP(T$^{\text{ks}}$)). Let $LB(T^{ks}) = \underline{h}_{T^{ks}}$, $\tilde{x}^k = \text{argmin}\{\phi(\tilde{x}^{k-1}), \phi(\bar{x}^k)\}, UB_k = \phi(\tilde{x}^k)$. If $LB(T^{ks}) > UB_k$, set $H = H \setminus T^{ks}$. Let $S = (S \setminus T^k) \cup H$, $LB_k = \min\{LB(T)|\ T \in S\}$.

Step 4: Set $S = S \setminus \{T|\ UB_k - LB(T) \leq \varepsilon, T \in S\}$. If $S = \emptyset$, terminate: $\tilde{x}^k$ is a global $\varepsilon-$optimal solution for problem (P). Otherwise, select T^{k+1} satisfying $T^{k+1} = \text{argmin}_{T \in S} LB(T)$, set $k = k + 1$, and return to Step 2.

$\boxed{4.2}$ Convergence of BB Algorithm

In order to obtain the convergence result of BB Algorithm, we first give the following lemma.

Lemma 4.1. *Given $T = [\underline{t}, \bar{t}]$, assume that $\tilde{t}$ is the optimal solution of problem* (CRP(T))*, and $(\bar{x}, \bar{\lambda})$ is the KKT point for problem* (P$_{\underline{t}}$)*. Let $\sigma = \min_{1 \leq i \leq p} \underline{t}_i$, then*

$$\phi(\bar{x}) - \underline{h}(\tilde{t}) \leq \sum_{i=1}^{p} \frac{\bar{f}_i}{\sigma^2}(\bar{t}_i - \underline{t}_i).$$

Proof. Since $\bar{x}$ is the optimal solution of problem (P$_{\underline{t}}$), we have $\bar{x} \in \Omega$ and

$$g_i(\bar{x}) \geq \underline{t}_i, i = 1, \cdots, p. \tag{4.1}$$

Combining (4.1) and the definitions of $\phi(x)$ and $\underline{h}(t)$, it follows that

$$\begin{aligned}
\phi(\bar{x}) - \underline{h}(\tilde{t}) &= \sum_{i=1}^{p} \frac{f_i(\bar{x})}{g_i(\bar{x})} - h(\underline{t}) - \sum_{i=1}^{p} (\bar{f}_i(\frac{1}{\tilde{t}_i} - \frac{1}{\underline{t}_i}) + \bar{\lambda}_i(\tilde{t}_i - \underline{t}_i)) \\
&\leq \sum_{i=1}^{p} \frac{f_i(\bar{x})}{\underline{t}_i} - \sum_{i=1}^{p} \frac{f_i(\bar{x})}{\underline{t}_i} + \sum_{i=1}^{p} (\bar{f}_i(\frac{1}{\underline{t}_i} - \frac{1}{\tilde{t}_i}) + \bar{\lambda}_i(\underline{t}_i - \tilde{t}_i)) \\
&\leq \sum_{i=1}^{p} \bar{f}_i(\frac{1}{\underline{t}_i} - \frac{1}{\bar{t}_i}) \\
&= \sum_{i=1}^{p} \frac{\bar{f}_i}{\underline{t}_i \bar{t}_i}(\bar{t}_i - \underline{t}_i) \\
&\leq \sum_{i=1}^{p} \frac{\bar{f}_i}{\sigma^2}(\bar{t}_i - \underline{t}_i).
\end{aligned}$$

The proof is completed. □

Next, we will prove the convergence of BB Algorithm.

Theorem 4.2. *Given $\varepsilon \geq 0$, if BB Algorithm terminates finitely, then $\tilde{x}^k$ is a global $\varepsilon-$optimal solution to problem* (P) *in the sense that*

$$\phi(\tilde{x}^k) \leq v(\mathrm{P}) + \varepsilon,$$

otherwise, every accumulation point of the sequence $\{\tilde{x}^k\}$ is a global optimal solution for problem (P).

Proof. If BB Algorithm is finite, without loss of generality, suppose it terminates at kth iteration. According to the algorithm, it follows that

$$UB_k - LB_k \leq \varepsilon.$$

From Step 3 of the algorithm, we have

$$UB_k = \phi(\tilde{x}^k),$$

we can then obtain

$$\phi(\tilde{x}^k) - v(\mathrm{P}) = UB_k - v(\mathrm{P}) \leq UB_k - LB_k \leq \varepsilon.$$

Thus, $\tilde{x}^k$ is a global $\varepsilon-$optimal solution for problem (P).

If BB Algorithm is infinite, it must generate an infinitely nested sequence of rectangles $\{T^k\}$ such that

$$\lim_{k\to\infty} \underline{t}_i^k = \lim_{k\to\infty} \bar{t}_i^k = t_i^*, \ i = 1, \cdots, p. \tag{4.2}$$

Also, for each rectangle T^k, we can obtain the optimal solution sequence $\{\tilde{t}^k\}$ to problem $(\mathrm{CRP(T^k)})$ and the sequence $\{\bar{x}^k\}$ by solving problem $(\mathrm{P}_{\underline{\mathrm{t}}^{\mathrm{k}}})$. Besides, the algorithm generates the best known solution sequence $\{\tilde{x}^k\}$ (i.e. $UB_k = \phi(\tilde{x}^k)$), and the monotone bounded sequences $\{LB_k\}$ and $\{UB_k\}$. Since Ω and T^k are nonempty compact set, the sequences $\{\tilde{t}^k\}$, $\{\bar{x}^k\}$ and $\{\tilde{x}^k\}$ are all bounded infinite sequences.

Without loss of generality, suppose that the subsequence $\{T^{k_j}\}$ satisfies

$$LB_{k_j} = LB(T^{k_j}).$$

Then it is clear that

$$\lim_{j\to\infty} \underline{t}_i^{k_j} = \lim_{j\to\infty} \bar{t}_i^{k_j} = t_i^*, \ i = 1, \cdots, p. \tag{4.3}$$

And correspondingly, the subsequences $\{\tilde{t}^{k_j}\}$ and $\{\bar{x}^{k_j}\}$ are the optimal solution sequences for problems $(\mathrm{CRP(T^{k_j})})$ and $(\mathrm{P}_{\underline{\mathrm{t}}^{\mathrm{k_j}}})$, respectively. Then from Step 3 of BB Algorithm, we have

$$LB_{k_j} = LB(T^{k_j}) = \underline{h}(\tilde{t}^{k_j}). \tag{4.4}$$

Moreover, from Lemma 4.1, it holds that

$$\phi(\bar{x}^{k_j}) - \underline{h}(\tilde{t}^{k_j}) \leq \sum_{i=1}^{p} \frac{\bar{f}_i}{\sigma^2}(\bar{t}_i^{k_j} - \underline{t}_i^{k_j}). \tag{4.5}$$

Combining (4.3) and (4.5), we can derive that

$$\lim_{j\to\infty} \phi(\bar{x}^{k_j}) - \underline{h}(\tilde{t}^{k_j}) \leq 0. \tag{4.6}$$

Since we have

$$\phi(\bar{x}^{k_j}) - \underline{h}(\tilde{t}^{k_j}) \geq v(\mathrm{P}) - \underline{h}(\tilde{t}^{k_j}) = v(\mathrm{P}) - LB_{k_j} \geq 0, \tag{4.7}$$

it follows that

$$\lim_{j\to\infty} \phi(\bar{x}^{k_j}) - LB_{k_j} = \lim_{j\to\infty} \phi(\bar{x}^{k_j}) - \underline{h}(\tilde{t}^{k_j}) = 0. \tag{4.8}$$

Additionally, the sequence $\{\tilde{x}^k\}$ must have a convergent subsequence, without loss of generality, suppose that $\lim_{k\to\infty} \tilde{x}^k = \tilde{x}^*$. Since

$$LB_{k_j} \leq v(\mathrm{P}) \leq UB_k = \phi(\tilde{x}^k) \leq \phi(\bar{x}^{k_j}),$$

from (4.8), we can obtain that

$$\lim_{k\to\infty} LB_k = \lim_{j\to\infty} LB_{k_j} = v(\mathrm{P}) = \lim_{k\to\infty} UB_k = \phi(\tilde{x}^*).$$

Therefore, every accumulation point of the sequence $\{\tilde{x}^k\}$ is a global optimal solution for problem (P). The proof is completed. $\square$

$\boxed{4.3}$ Computational complexity of BB Algorithm

In this subsection, the computational complexity analysis of the BB Algorithm will be given. To this end, let us introduce some notations as follows:

$$\theta = \min_{i\in\{1,\cdots,p\}} \underline{t}_i^0,$$

$$U = \max_{i\in\{1,\cdots,p\}} \bar{f}_i,$$

where $\bar{f}_i$ is defined by (3.4).

Lemma 4.3. *Given $\varepsilon > 0$, if the selected rectangle $T^k = [\underline{t}^k, \bar{t}^k]$ in Step 3 of BB Algorithm at kth iteration satisfies*

$$\bar{t}_i^k - \underline{t}_i^k \leq \frac{\varepsilon\theta^2}{pU}, i = 1, \cdots, p,$$

then the algorithm terminates and returns a global $\varepsilon-$optimal solution $\tilde{x}^k$ for problem (P).

Proof. Assume that $\tilde{t}^k$ is the optimal solution of problem (CRP($\mathrm{T^k}$)), and $(\bar{x}^k, \bar{\lambda}^k)$ is the KKT point for problem ($\mathrm{P}_{\underline{\mathrm{t}}^{\mathrm{k}}}$). From Step 3 of the algorithm, it follows that

$$LB_k = \underline{h}(\tilde{t}^k).$$

If $\bar{t}_i^k - \underline{t}_i^k \leq \frac{\varepsilon\theta^2}{pU}, i = 1, \cdots, p$, from Lemma 4.1, we have

$$\phi(\bar{x}^k) - LB_k = \phi(\bar{x}^k) - \underline{h}(\tilde{t}^k) \leq \sum_{i=1}^{p} \frac{\bar{f}_i}{\sigma^2}(\bar{t}_i^k - \underline{t}_i^k) \leq \sum_{i=1}^{p} \frac{U}{\theta^2}(\bar{t}_i^k - \underline{t}_i^k) \leq \varepsilon. \tag{4.9}$$

Combining (4.9) and $UB_k \leq \phi(\bar{x}^k)$, it can be derived that

$$UB_k - LB_k \leq \varepsilon.$$

Thus, the algorithm terminates. Moreover, it holds that $UB_k = \phi(\tilde{x}^k)$ and $LB_k \geq UB_k - \varepsilon$, which implies that

$$\phi(\tilde{x}^k) - \varepsilon \leq LB_k \leq v(\mathrm{P}) \leq \phi(\tilde{x}^k),$$

thus, $\tilde{x}^k$ is a global $\varepsilon-$optimal solution for problem (P). The proof is completed. $\square$

Lemma 4.3 indicates that if each edge of the initial rectangle T^0 is subdivided into at most $\lceil\frac{(\bar{t}_i^0-\underline{t}_i^0)Up}{\varepsilon\theta^2}\rceil$ subintervals, the BB Algorithm will terminate and return a global $\varepsilon-$optimal solution for problem (P). Therefore, we have the following result.

Theorem 4.4. *The running time of BB Algorithm is bounded from above by*

$$T(n,m+p)\prod_{i=1}^{p}\lceil\frac{(\bar{t}_i^0-\underline{t}_i^0)Up}{\varepsilon\theta^2}\rceil$$

to obtain a global $\varepsilon-$optimal solution x^* to problem (P) *such that $\phi(x^*)\le\phi(x)+\varepsilon$ holds for all $x\in\Omega$, where $T(n,m+p)$ denotes the time taken to solve a linear program with n variables and $m+p$ constraints.*

5 Numerical Experiment

In this section, we report numerical comparisons for the proposed branch and bound algorithm and the algorithms introduced by [14, 25, 35]. All the numerical experiments were implemented in Matlab(2018a) and ran on a Intel(R) Core(TM) i5-3550S CPU 3.00GHz with 4G memory microcomputer. In our computational experiment, each linear programming problem was solved by using CPLEX [12], the tolerance parameter ε was set as 10^{-4}, and the maximum CPU time limit was set equal to 3600 seconds.

Some notations in the following Tables 1-4 have been used for column headers:

Opt.val: the average optimal value obtained by the relevant algorithm;

CPU: the average execution time in seconds;

Iter: the average number of the algorithm iterations;

"...": means that the method fails to find the optimal solution within 3600s at all cases.

5.1 A Transportation Problem

We consider a transportation problem [2]: A company has three electric power plants that supply the power needs of four cities. The variable x_{ij} for each possible path of electricity is defined to denote the unknown quantity of kwh of electricity sent from ith plant to jth city ($i=1,2,3, j=1,2,3,4$). Using the relevant data provided by [2], the transportation problem can be formulated as:

Example 1.

$$\begin{array}{ll}\max & \dfrac{f(x)}{g(x)}\\ \text{s.t.} & x_{11}+x_{12}+x_{13}+x_{14}\le 35,\\ & x_{21}+x_{22}+x_{23}+x_{24}\le 50,\\ & x_{31}+x_{32}+x_{33}+x_{34}\le 40,\\ & x_{11}+x_{21}+x_{31}\ge 45,\\ & x_{12}+x_{22}+x_{32}\ge 20,\\ & x_{13}+x_{23}+x_{33}\ge 30,\\ & x_{14}+x_{24}+x_{34}\ge 30,\\ & x_{ij}\ge 0,\quad i=1,2,3,j=1,2,3,4,\end{array}$$

where

$$\begin{aligned}f(x)=&\,5x_{11}+4x_{12}+4x_{13}+3x_{14}+16x_{21}+2x_{22}+3x_{23}+4x_{24}+\\ &+10x_{31}+5x_{32}+6x_{33}+2x_{34},\end{aligned}$$

$$g(x) = 8x_{11} + 6x_{12} + 10x_{13} + 9x_{14} + 9x_{21} + 12x_{22} + 13x_{23} + 7x_{24} + \\ + 14x_{31} + 9x_{32} + 16x_{33} + 5x_{34}.$$

By solving Example 1 through BB Algorithm, after 1 iteration we get the optimal solution $x^* = (0, 20, 15, 0, 5, 0, 15, 30, 40, 0, 0, 0)^\top$ and the optimal value 0.5742 with 0.004 seconds CPU time.

$\boxed{5.2}$ The production planning problems

We next consider the randomly generated instances with the production planning problem structure in [23]. The mathematical model for this problem can be constructed as follows:

Example 2.

$$\begin{array}{ll} \max & \dfrac{\sum_{j=1}^n c_j x_j + c_0}{\sum_{j=1}^n d_j x_j + d_0} \\ \text{s.t.} & Ax \le b, x \ge 0, \end{array}$$

where the elements of A, b, d, d_0 are generated as random integer numbers in $[0, 10]$, and the elements of c, c_0 are generated as random integer numbers in $[-10, 0]$. For each problem instance, we ran BB Algorithm for five times, and summarized the average numerical results in Table 1. From Table 1 we can see that BB Algorithm can solve Example 2 efficiently.

Table 1: Numerical results for Example 2

(m, n)	CPU	Iter	(m, n)	CPU	Iter
(10,10)	0.0282	11.2	(1000,1000)	0.3030	1.0
(100,100)	0.0186	3.4	(1500,1500)	0.7436	1.0
(250,250)	0.0190	1.0	(2000,2000)	1.3120	1.0
(500,500)	0.0748	1.0	(3000,3000)	3.0774	1.0
(750,750)	0.1408	1.0	(5000,5000)	8.8282	1.0

$\boxed{5.3}$ Randomly generated examples

In this subsection, we test several randomly generated examples to verify the performance of the proposed BB Algorithm. We first test the BB Algorithm and the algorithm in [35] on Example 3, and Table 2 demonstrated the average performance of these two algorithms for five instances of Example 3.

Example 3.

$$\begin{array}{ll} \min & \dfrac{a_1^\top x + b_1}{a_2^\top x + b_2} + a_0^\top x \\ \text{s.t.} & Bx = c, x \in [0, 2]^n, \end{array}$$

where $b_1 = b_2 = 5$, the parameters a_1, a_0 are randomly generated in $[0.1, 1]$, and the elements of a_2 and B are randomly generated in $[1, 2]$ and $[0, 20]$, respectively. To guarantee the feasible region $\Omega = \{x \in R^n | Bx = c, x \in [0, 2]^n\}$ is nonempty, we first randomly generate an $x^0 \in [0, 2]^n$ and then set $c = Bx^0$.

Table 2: Computational results for Example 3

(m,n)	BB Algorithm			Algorithm in [35]		
	Opt.val	CPU	Iter	Opt.val	CPU	Iter
(40,200)	70.1808	0.0442	11.4	70.1808	0.0472	10.2
(100,500)	185.3839	0.4642	10.8	185.3839	0.5568	10.6
(200,1000)	385.7250	3.9894	11.0	385.7251	4.7102	10.6
(240,1200)	454.0594	6.9336	10.8	454.0594	8.3450	10.6
(300,1500)	577.4159	15.6244	11.4	577.4160	18.1758	10.6
(400,2000)	769.8532	43.2348	10.8	769.8532	51.2278	10.4
(440,2200)	834.5629	62.3506	11.0	834.5630	73.2158	10.2
(500,2500)	951.8400	99.9148	10.6	951.8400	122.5110	10.8
(560,2800)	1059.8792	151.2046	11.0	1059.8792	179.1894	10.8
(750,3000)	1185.5338	93.5688	11.0	1185.5338	131.7112	10.8
(800,3200)	1264.5978	116.7984	11.4	1264.5979	154.8932	10.2
(900,3600)	1427.7086	170.3298	10.6	1427.7086	252.8228	10.8
(1000,4000)	1572.0285	241.2394	11.0	1572.0285	355.4106	10.4
(1100,4400)	1727.3257	327.0276	11.4	1727.3258	454.6804	10.4
(1250,5000)	1975.8170	595.0798	13.4	1975.8170	788.8456	10.8

From Table 2 it can be observed that, for each set of fixed (m,n), to obtain the optimal values, the algorithm in [35] is more time-consuming than BB Algorithm, though the number of iterations is smaller for most cases. The main reason is that the approach in [35] uses the function value information of two endpoints of the current interval in each iteration, while the BB Algorithm only utilizes the function value information of one endpoint. This indicates that BB Algorithm may require less CPU time but more iterations to get the global optimal values than that of the method in [35]. Besides, the algorithm in [35] can only deal with the case that $p = 2$, and the denominator of one of the linear fractions is 1, while the BB Algorithm is capable of solving more general cases, as we will see later, the case when $p > 2$.

In order to further evaluate the performance of BB Algorithm, we finally test the BB Algorithm and two algorithms in [14, 25] on the following problems:

Example 4.

$$\begin{array}{ll} \min & \sum_{i=1}^{p} \dfrac{\sum_{j=1}^{n} c_{ij}x_j + c_{0i}}{\sum_{j=1}^{n} d_{ij}x_j + d_{0i}} \\ \text{s.t.} & Ax \le b, x \ge 0, \end{array}$$

where $c_{0i} = 0.5, d_{0i} = 5$, the parameters c_{ij}, d_{ij} are randomly generated in $[0, 0.5]$ and $[0, 5]$, respectively, and all elements of $A \in R^{m\times n}$, $b \in R^m$ are the random numbers in the interval $[0.1, 20]$ and $[0, 1]$, respectively. In Tables 3-4, we summarize the average numerical results of the BB Algorithm and the algorithms in [14, 25] for five instances of Example 4.

Table 3: Computational results for Example 4 with $p \le 5$

(p,m,n)	BB Algorithm			Algorithm in [14]			Algorithm in [25]	
	Opt.val	CPU	Iter	Opt.val	CPU	Iter	Opt.val	CPU
(2,200,2000)	0.1998	0.55	4.2	0.1998	1.00	8.8	0.1998	0.75
(2,400,4000)	0.1999	2.83	3.6	0.1999	4.75	6.2	0.1998	4.95
(2,600,6000)	0.1999	7.07	3.4	0.1999	9.82	4.6	0.1999	11.71
(2,800,8000)	0.2000	10.23	1.6	0.2000	13.16	2.0	0.1999	19.46
(2,1000,10000)	0.1999	19.54	2.4	0.1999	28.29	3.6	0.1999	27.30
(3,10,100)	0.2984	0.17	71.8	0.2984	0.89	705.0	0.2984	1183.15
(3,30,300)	0.2984	0.23	43.0	0.2984	1.86	643.2	0.2984	1781.90
(3,50,500)	0.2989	0.39	54.4	0.2989	2.98	537.6	0.2989	2956.26
(3,250,2500)	0.2996	2.45	14.8	0.2996	9.74	65.4	...	...
(3,500,5000)	0.2998	8.43	6.8	0.2998	25.06	24.6	...	...
(4,100,1000)	0.3992	2.61	124.4	0.3992	31.64	1690.0	...	...
(4,300,3000)	0.3997	11.56	36.2	0.3997	97.35	326.0	...	...
(4,500,5000)	0.3998	18.34	17.6	0.3998	100.77	113.8	...	...
(4,700,7000)	0.3999	28.38	11.8	0.3999	106.83	57.0	...	...
(5,200,2000)	0.4992	21.77	266.0	0.4992	574.36	6928.6	...	...
(5,400,4000)	0.4996	50.61	94.0	0.4996	798.59	1466.4	...	...
(5,600,6000)	0.4998	58.21	42.6	0.4998	803.59	635.2	...	...
(5,800,8000)	0.4998	109.38	38.6	0.4998	984.22	378.6	...	...

The numerical results for Example 4 with the change of p from 2 to 5 are reported in Table 3. As can be seen from Table 3, for finding the global solutions for all tested instances, the CPU time required by the proposed algorithm is the least among three methods (BB Algorithm and the algorithms in [14, 25]). The average CPU time cost by BB Algorithm increases relatively slowly as n increases, compared with the methods in [14,25]. In addition, the algorithm in [25] fails to find the global solutions within 3600s in the case that $p > 3$ or the value of (p, m, n) exceeds $(3, 250, 2500)$. This is because the approach in [25] has to solve the linear programming problems related to each grid node, it may not be very effective when the number of grid node increases.

In order to further test the performance of BB Algorithm, we increased the size of the instances to be tested. Table 4 lists the computational results for Example 4 with $p > 5$. From Table 4, it can be seen that the average CPU time spent by the BB Algorithm is much less than that of [14] for the large scale instances of Example 4. The average CPU time of the algorithm in [14] grows fast as (m, n) increases, compared with BB Algorithm. Moreover, the BB Algorithm performs more effectively than the method in [14] when $p \geq 7$. The main reason for this phenomenon is that the number of linear problems need to be solved at each iteration in BB Algorithm is one less than that of the algorithm in [14], and the structure of the linear problem in our method is relatively simple. Based on the above observation, it can be concluded that BB Algorithm can efficiently solve the tested instances.

6 Conclusion

In this article, we consider how to globally solve a class of sum of linear ratios problems (P) which have significant applications in many fields. The original problem (P) is equivalently transformed into problem (EP) by introducing variables, and then problem (EP) is represented as a two-layer problem (EP1). A new convex relaxation technique is presented based on problem (EP1), which can provide a reliable lower bound for the optimal value of problem (P). Through combining the convex relaxation technique and standard bisection rule, a branch and bound method is designed for globally solving problem (P), and we also analyse the convergence and computational complexity of the proposed algorithm. Numerical results show that the developed algorithm performs better than the known methods in the literature.

Notice that the complexity of BB Algorithm grows exponentially in terms of p, which indicates that BB Algorithm may not be effective for the instances with large p. More work should be considered for designing practical deleting strategy, adaptive branching rule and some local search techniques to improve the computational efficiency in the future. Besides, to enhance the tightness of relaxation problem by exploiting some valid cut techniques may bring further improvement to our branch and bound algorithm. Another interesting direction for future research is to investigate whether similar global approaches can be developed for generic fractional programming problems with nonconvex feasible set, or the fractional programming problems in uncertain variable environment. More study is needed to address these issues.

Table 4: Computational results for Example 4 with $p > 5$

(p, m, n)	BB Algorithm			Algorithm in [14]			Algorithm in [25]	
	Opt.val	CPU	Iter	Opt.val	CPU	Iter	Opt.val	CPU
(6,100,1000)	0.5992	20.67	902.4	0.5992	683.78	25456.4	...	...
(6,300,3000)	0.5997	53.64	172.2	0.5997	1021.42	3288.6	...	...
(6,500,5000)	0.5998	85.87	95.8	0.5998	1327.02	1346.0	...	...
(6,700,7000)	0.5998	162.75	91.0	0.5998	2234.58	1272.8	...	...
(7,80,800)	0.6990	197.06	10901.4	0.6990	3100.38	102754.0	...	...
(7,200,2000)	0.6992	213.17	2621.2	0.6992	3542.67	38442.0	...	...
(7,300,3000)	0.6993	560.42	1928.2	...	...	...	...	...
(7,400,4000)	0.6995	732.00	1362.2	...	...	...	...	...
(8,100,200)	0.7993	51.72	6909.8	0.7993	2387.38	87095.4	...	...
(8,100,1000)	0.7986	617.45	21543.4	...	...	...	...	...
(8,200,2000)	0.7994	1063.12	12112.4	...	...	...	...	...
(9,100,100)	0.8995	136.28	13306.4	0.8995	2606.24	97426.2	...	...
(9,100,300)	0.8992	427.38	26626.2	...	...	...	...	...
(9,100,500)	0.8988	1463.48	52671.6	...	...	...	...	...
(10,50,50)	0.9993	345.80	29934.0	...	...	...	...	...
(10,100,100)	0.9995	385.70	29943.4	...	...	...	...	...
(10,100,500)	0.9990	1719.67	63211.2	...	...	...	...	...

References

[1] Y. Almogy and O. Levin, Parametric analysis of a multi-stage stochastic shipping problem, *Oper. Res.* 69 (1970) 359–370.

[2] E.B. Bajalinov, *Linear-Fractional Programming: Theory, Methods, Applications, and Software*, Springer US, 2003.

[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.

[4] H.P. Benson, Branch and bound outer approximation algorithm for sum-of-ratios fractional programs, *J. Optimiz. Theory. App.* 146 (2010) 1–18.

[5] J.G. Carlsson and 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.

[6] D.Z. Chen, O.Daescu, Y. Dai, N. Katoh, X.D. Wu and J.H. Xu, Optimizing the sum of linear fractional functions and applications, in: *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, tSociety for Industrial and Applied Mathematics, 2000, pp. 707–716.

[7] J.C. Choi and D.L. Bricker, Effectiveness of a geometric programming algorithm for optimization of machining economics models, *Comput. Oper. Res.* 23 (1996) 957–961.

[8] D. Depetrini and M. Locatelli, Approximation algorithm for linear fractional multiplicative problems, *Math. Program.* 128 (2011) 437–443.

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

[10] R.W. Freund and F. Jarre, Solving the sum-of-ratios problem by an interior-point method, *J. Glob. Optim.* 19 (2001) 83–102.

[11] X.L. Huang, Y.L. Gao, B. Zhang and X. Liu, An effective computational algorithm for the global solution of a class of linear fractional programming, *Math. Probl. Eng.* (2020), https://doi.org/10.1155/2020/3580419.

[12] IBM ILOG CPLEX. IBM ILOG CPLEX 12.3 User's Manual for CPLEX. 89 (2011).

[13] H.W. Jiao, L. Cai, Z.S. Hou and C.Y. Bai, An effective algorithm for globally solving sum of linear ratios problems, *Journal of Control Science and Engineering.* 2017 (2017), https://doi.org/10.1155/2017/8138975.

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

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

[16] A.A. Khan, R.S. Adve and W. Yu, Optimizing downlink resource allocation in multiuser mimo networks via fractional programming and the Hungarian algorithm, *IEEE. Trans. Wirel. Commun.* 19 (2020) 5162–5175.

[17] H. Konno and N. Abe, Minimization of the sum of three linear fractional functions, *J. Glob. Optim.* 15 (1999) 419–432.

[18] H. Konno and M. Inori, Bond portfolio optimization by bilinear fractional programming, *J. Oper. Res. Soc. Jpn.* 32 (1989) 143–158.

[19] H. Konno and H. Watanabe, Bond portfolio optimization problems and their applications to index tracking: a partial optimization approach, *Journal of the Operations Research Society of Japan-Keiei Kagaku.* 39 (1996) 295–306.

[20] H. Konno, Y. Yajima and T. Matsui, Parametric simplex algorithms for solving a special class of nonconvex minimization problem, *J. Glob. Optim.* 1 (1991) 65–81.

[21] X. Liu, Y.L. Gao, B. Zhang and F.P. Tian, A new global optimization algorithm for a class of linear fractional programming, *Mathematics* 7 (2019): 867.

[22] Y.E. Nesterov and A.S. Nemirovskii, An interior-point method for generalized linear-fractional programming, *Math. Program.* 69 (1995) 177–204.

[23] B.A. Ozkok, An iterative algorithm to solve a linear fractional programming problem, *Comput. Ind. Eng.* 140 (2020) 106234.1–106234.7.

[24] N.T.H. Phuong and H. Tuy, A unified monotonic approach to generalized linear fractional programming, *J. Glob. Optim.* 26 (2003) 229–259.

[25] P.P. Shen, B.D. Huang and L.F. Wang, Range division and linearization algorithm for a class of linear ratios optimization problems, *J. Comput. Appl. Math.* 350 (2019) 324–342.

[26] N.V. Sahinidis, BARON: a general purpose global optimization software package, *J. Glob. Optim.* 8 (1996) 201–205.

[27] S. Schaible, *Fractional Programming*, Springer, Berlin, 1995.

[28] P.P. Shen, W.M. Li and Y.C. Liang, Branch-reduction-bound algorithm for linear sum-of-ratios fractional programs, *Pacific J. Optim.* 11 (2015) 79–99.

[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.P. Shen and C.F. Wang, Global optimization for sum of linear ratios problem with coefficients, *Appl. Math. Comput.* 176 (2006), 219–229.

[31] P.P. Shen and C.F. Wang, Global optimization for sum of generalized fractional functions, *J. Comput. Appl. Math.* 214 (2008) 1–12.

[32] C.C. Skiscim and S. W. Palocsay, Minimum spanning trees with sums of ratios, *J. Glob. Optim.* 19 (2001) 103–120.

[33] C.F. Wang and P.P. Shen, A global optimization algorithm for linear fractional programming, *Appl. Math. Comput.* 204 (2008) 281–287.

[34] Y. Xia, L.F. Wang and S. Wang, Minimizing the sum of linear fractional functions over the cone of positive semidefinite matrices: Approximation and applications, *Oper. Res. Lett.* 46 (2018) 76–80.

[35] Y. Xia, L.F. Wang and X.H. Wang, Globally minimizing the sum of a convex-concave fraction and a convex function based on wave-curve bounds, *J. Glob. Optim.* 77 (2020) 301–318.

[36] Y.H. Zhang and C.F. Wang, A new global algorithm for sum of linear ratios problem, *Acta Mathematicae Applicatae Sinica.* 35 (2012) 42–48.

[37] Y.H. Zhang and C.F. Wang, A new branch and reduce approach for solving generalized linear fractional programming, *Eng. Lett.* 25 (2017) 262–267.

[38] Y.F. Zhao and T. Zhao, A reduced space branch and bound algorithm for a class of sum of ratios problems, *Open. Math.* 16 (2018) 539–552.

Manuscript received 25 June 2021
revised 1 September 2021
accepted for publication 30 September 2021

Bingdi Huang
College of Mathematics and Information Science
Henan Normal University, Xinxiang 453007, China
E-mail address: huangbingdihtu@163.com

Peiping Shen
School of Mathematics and Statistics
North China University of Water Resources and Electric Power
Zhengzhou 450046, China;
College of Mathematics and Information Science
Henan Normal University
Xinxiang 453007, China
E-mail address: shenpeipingncwu@126.com