http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (22): 6871-6876, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.6871.6876

Numerical Algorithm for a Class of Linear Fractional Programming Problem

[1]Hongwei Jiao, [2]Kun Li and [1]Jianping Wang
[1]School of Mathematical Science, Henan Institute of Science and Technology, 453003, Xinxiang, China
[2]Department of Basic Science, Henan Mechanical and Electrical Engineering College, 453002, Xinxiang, China

Abstract: In this study, a branch and bound algorithm is presented for globally solving a class of linear fractional programming problems. In the algorithm, a linear relaxation method is introduced to generate the linear relaxation programming problem of the investigated linear fractional programming problems. In this study, we pay more attention to the numerical experiments. Several test problems are used to verify the feasibility and computational efficiency of the proposed branch and bound algorithm.

Key words: Numerical algorithm, linear fractional programming, global optimization, linear relaxation programming, branch and bound

INTRODUCTION

In this study, we consider the same linear fractional programming problem investigated in (Feng *et al*., 2011):

$$(LFP):\ \begin{cases}\min\ \max\ \{g_1(x), g_2(x), \ldots, g_p(x)\}\\ \text{s.t.}\quad D=\{x \mid Ax \le b,\quad x \ge 0\},\end{cases}$$

Where:

$$g_j(x)=\frac{\sum_{i=1}^{n} c_{ji}x_i + d_j}{\sum_{i=1}^{n} e_{ji}x_i + f_j},\ j=1,2,\ldots,p$$

p is a given natural number:

$$\sum_{i=1}^{n} c_{ji}x_i + d_j$$

and:

$$\sum_{i=1}^{n} e_{ji}x_i + f_j$$

are all linear functions such that:

$$\sum_{i=1}^{n} c_{ji}x_i + d_j > 0$$

and:

$$\sum_{i=1}^{n} e_{ji}x_i + f_j \ge 0$$

for all x∈D.

The Linear Fractional Programming problem (LFP) has broad applications in measuring the efficiency or productivity of system, the design of electronic circuits (Schaible and Shi, 2003), production planning (Schaible, 1995), portfolio optimization (Sekitani *et al*., 1995) and so on. But to our knowledge, since the problem (LFP) is nonconvex, there exist various local optimal solutions which are not global optimal solutions. Therefore, the problem (LFP) is very difficult to be solved.

In last years, several algorithms have been presented for solving the Linear Fractional Programming problem (LFP) with assumption that the denominator and numerator of each ratio are all positive linear affine functions (Ahmad and Husain, 2006; Phuong and Tuy, 2003). But to our knowledge, less work has been still developed to globally solve the Linear Fractional Programming problem (LFP) investigated in this article.

Recently, a new linearization method is proposed for solving the problem (LFP) in (Feng *et al*., 2011), based on the new linearization technique, a branch and bound algorithm is given and the convergence of the proposed algorithm is discussed. In addition, we will pay more attention to the numerical experiments of the algorithm. The proposed algorithm is coded in C^{++} program on Intel(R) Core(TM)2 Duo CPU (1.58GHZ) microcomputer, several test examples are used to verify the feasibility and robustness of the proposed algorithm. And finally the numerical experimental results show the proposed algorithm is robust and effective.

Corresponding Author: Hongwei Jiao, School of Mathematical Science, Henan Institute of Science and Technology, 453003, Xinxiang, China

LINEAR RELAXATION PROGRAMMING

To solve the considered Linear Fractional Programming problem (LFP), first of all, for each j = 1,…, p, we solve the following two linear programming problems:

$$\begin{cases}\min\ x_i\\ \text{s.t.}\quad Ax \le b,\end{cases}$$

and:

$$\begin{cases}\max\ x_i,\\ \text{s.t.}\quad Ax \le b.\end{cases}$$

So, that we can compute the initial upper bound $\bar{x}_i^0$ and the initial lower bound $\underline{x}_i^0$ of each variable x_i, i = 1, 2,…, n and the initial rectangle $X^0 = \{x \mid \underline{x}_i^0 \le x_i \le \bar{x}_i^0,\ i = 1,\ldots,n\}$. Obviously, X^0 contain the feasible region of the problem (LFP).

For all $x \in X = [\underline{x}, \bar{x}] \subseteq X^0$, define:

$$L_j = \min_{x\in X} \sum_{i=1}^{n} e_{ji}x_i + f_j,$$

$$U_j = \max_{x\in X} \sum_{i=1}^{n} e_{ji}x_i + f_j,$$

$$g_j^L(x) = \sum_{i=1,c_{ji}>0}^{n} \frac{c_{ji}x_i}{U_j} + \sum_{i=1,c_{ji}<0}^{n} \frac{c_{ji}x_i}{L_j} + \sum_{i=1,d_j>0}^{n} \frac{d_j}{U_j} + \sum_{i=1,d_j<0}^{n} \frac{d_j}{L_j},$$

$$g_j^U(x) = \sum_{i=1,c_{ji}>0}^{n} \frac{c_{ji}x_i}{L_j} + \sum_{i=1,c_{ji}<0}^{n} \frac{c_{ji}x_i}{U_j} + \sum_{i=1,d_j<0}^{n} \frac{d_j}{U_j} + \sum_{i=1,d_j>0}^{n} \frac{d_j}{L_j}.$$

Theorem 1: Feng *et al.* (2011). For all x∈X for each j = 1,…, p, we can get the following conclusions:

- $g_j^L(x) \le g_j(x) \le g_j^U(x)$
- For each j = 1,…, p, the maximal errors of bounding $g_j(x)$ using $g_j^L(x)$ and $g_j^U(x)$ satisfying:

$$\lim_{w\to 0} \Delta_{max}^L = \lim_{w\to 0} \nabla_{max}^U = 0$$

Where:

$$w = \|\bar{x} - \underline{x}\|,$$

$$\Delta_{max}^L = \max_{x\in X}[g_j(x) - g_j^L(x)],$$

$$\nabla_{max}^U = \max_{x\in X}[g_j^U(x) - g_j(x)]$$

Proof: Feng *et al.* (2011).

Obviously, by theorem 1 we get that, for each j = 1, 2,…, p, the linear function $g_j^L(x)$ will enough approximate the function $g_j(x)$ as w→0.

Therefore, for $\forall X^k \subseteq X^0$, we can construct the linear relaxation programming problem of the problem (LFP) in X^k as follows:

$$(\text{LRP}):\ \begin{cases}\min\ \max\limits_{1\le j\le p}\ y\\ \text{s. t.}\ \ g_j^L(x) \le y,\\ \qquad Ax \le b,\\ \qquad x \in X^k.\end{cases}$$

By the above construction method of the relaxation linear programming, for $\forall X^k \subseteq X^0$, the problem LRP ($X^k$) provides a valid lower bound for the optimal value of the problem LFP (X^k).

BRANCH AND BOUND AGLORITHM AND ITS CONVERGENCE

The important step for the construction of a branch and bound procedure for globally solving the problem (LFP) is calculation of lower bounds for the initial problem and its sub-problems. A lower bound of the global optimal value of the problem (LFP) and its partitioned sub-problems can be calculated by solving a series of linear relaxation programming problems. Based on the above proposed linear relaxation programming problem, the problem (LRP) provides a valid lower bound for the global optimal value of the problem (LFP) over the rectangle X^k.

The important condition for guarantee that the proposed algorithm converges to the global optimal solution of the problem (LFP) is the selection of a suitable partitioning operation. Here, we still choose standard bisection method proposed in (Feng *et al.*, 2011).

In the following, for convenience in expression, we assume that $LB(X^k)$ be the global optimal value of the problem $LPR(X^k)$ and $x^k = x(X^k)$ be the global optimal solution of the problem $LPR(X^k)$. Step of the proposed algorithm is given as follows:

Step 0: Let the initial convergence tolerance ε the initial the number of iteration k: = 0; the initial set of active node $\Omega_0 = X^0$; the initial upper bound $UB_0 = +\infty$; the initial set of feasible points F: = ∅

Solve the problem $LRP(X^0)$ to obtain its optimal value LB_0: = (X^0) and optimal solution x^0: = $x(X^0)$, respectively. If x^0 is feasible to the problem (LFP), then we may update F and UB_0, if necessary.

If $UB_0 - LB_0 \le \varepsilon$, then algorithm stops with x^0 be the global optimization solution of the problem (LFP). Otherwise, continue to step 1:

Step 1: Let:

$$H(x) = \max\{g_1(x), g_2(x),\ldots, g_p(x)\}$$

and $UB_k = \min_{x\in F} h(x)$, if F: = ∅, then denote the best known feasible solution by $\tilde{x} = \arg\min_{x\in F} h(x)$.

Step 2: Partition the selected rectangle X^k into two new sub-hyper-rectangles according to the above proposed partitioning operation. Denote the set of new partitioned sub-hyper-rectangles as $\overline{X}^k$. For each $X\in\overline{X}^k$, solve the problem PLR(X) to obtain its optimal value LB(X) and optimal solution x(X). If LB(X)>UB_k, then let $\overline{X}^k := \overline{X}^k \setminus X$.

Step 3: If LB(X)>UB_k and x(X) is feasible to the problem (LFP), then we may update UB_k, F and b, if necessary and let $\Omega_k = (\Omega_k \setminus X)\cup\overline{X}^k$, update the lower bound $LB_k = \inf_{X\in\Omega_k} LB(X)$

Step 4: Set $\Omega_{k+1} = \Omega_k \setminus \{X : UB_k - LB(X) \le \varepsilon, X\in\Omega_k\}$. If Ω_{k+1} = ∅, then algorithm stops, UB_k is the global optimal value for the (LFP) and $\tilde{x}$ is a global optimization solution for problem (LFP); Otherwise, let k: = k+1, select X^k such that $X^k = \arg\min_{X\in\Omega_k} LB(X)$, x^k: = x(X^k), return to step 1

Theorem 2: If the proposed algorithm terminates finitely at iteration k, then a global ε-optimal solution x^k of the problem (LFP) is obtained. Otherwise it must produce an infinite sequence $\{x^k\}$ of iterations of which any limitation point x* must be a global optimal solution of the problem (LFP) and $\lim_{k\to\infty} UB_k = \lim_{k\to\infty} LB_k = v$ where v is the optimal solution for the problem (LFP).

Proof: If the proposed algorithm stops finitely at iteration k, then, when the algorithm stops, it can follow that $UB_k = LB_k$. Hence, on the basis of the updating step of the upper bound of the algorithm, we can follow that x^k is the optimal solution of the problem (LFP).

If the proposed algorithm does not stop finitely at some stage k, then it will generate an infinite sequence $\{X^k\}$. Since, the used branching operation is exhaustive, it is obvious that the whole sequence $\{X^k\}$ must converge to a singleton. On the basis of the updating step of the upper bound and the lower bound for the proposed algorithm, we get easily that $\{UB_k\}$ and $\{LB_k\}$ are non-increasing sequence and non-decreasing sequence, respectively. Hence, we can get that $\{UB_k\text{-}LB_k\}$ is a non-increasing sequence, by the conclusions of the Theorem 1, it follows that the sequence $\{UB_k\text{-}LB_k\}$ converges to zero. Meanwhile, by the structure of the branch and bound algorithm, we have $LB_k \le v \le UB_k$ for all k. Hence, we have $\lim_{k\to\infty} UB_k = \lim_{k\to\infty} LB_k = v$.

On the basis of the updating method of the upper bound UB_k, it follows that all x^k obtained at iteration k are all feasible to the problem (LFP) and $UB_k = \max\{g_1(x^k), g_2(x^k),\ldots, g_p(x^k)\}$.

Thus, the limitation x* of the sequence $\{x^k\}$ is feasible to the problem (LFP) with the objective function value v = $\max\{g_1(x^*), g_2(x^*),\ldots, g_p(x^*)\}$. Hence, the conclusion of the theorem is concluded.

NUMERICAL EXPERIMENTS

To validate the feasibility and robustness of the presented global optimization algorithm, some test problems are implemented on Intel(R) Core(TM)2 Duo CPU (1.58GHZ) microcomputer, the algorithm is coded in C++, the simplex method is used to solve linear relaxation programming problems and the convergence tolerance is set to $\varepsilon = 5\times10^{-8}$. These test examples and their computational results are given as follows.

Example 1:

$$\begin{aligned}
&\min\max\ \{g_1(x), g_2(x)\}\\
&\text{s.t.}\quad x_1 + x_2 - x_3 \le 1.2,\\
&\quad -x_1 + x_2 - x_3 \le -0.9,\\
&\quad 12x_1 + 5x_2 + 12x_3 \le 41,\\
&\quad 12x_1 + 12x_2 + 7x_3 \le 51,\\
&\quad -6x_1 + x_2 + x_3 \le -1,\\
&\quad 1.0 \le x_1 \le 1.2,\\
&\quad 0.55 \le x_2 \le 0.65,\\
&\quad 1.35 \le x_3 \le 1.45.
\end{aligned}$$

Where:

$$g_1(x) = \frac{2.2x_1 + 2.3x_2 - x_3 + 1.8}{1.2x_1 - x_2 + 1.3x_3}$$

$$g_2(x) = \frac{3.2x_1 - x_2 + 1.4x_3}{8.3x_1 + 4.2x_2 - x_3}$$

Select the feasible error $\varepsilon_f = 0.001$, numerical results are given as follows: the number of iteration is 207, the maximal number of active nodes necessary is 181, the execution time in seconds t = 0.361621 sec, obtain the optimal value V* = 1.504141260, the global optimization solution is $x^*_1 = 1.0$, $x^*_2 = 0.55$, $x^*_3 = 1.45$.

Example 2:

$$\begin{aligned}
&\min\max\ \{g_1(x), g_2(x), g_3(x), g_4(x)\}\\
&\text{s.t.}\quad x_1 + x_2 - x_3 \le 1.1,\\
&\quad -x_1 + x_2 - x_3 \le -0.5,\\
&\quad 12x_1 + 5x_2 + 12x_3 \le 43,\\
&\quad 12x_1 + 12x_2 + 7x_3 \le 56,\\
&\quad -6x_1 + x_2 + x_3 \le -2.5,\\
&\quad 1.0 \le x_1 \le 2.0,\\
&\quad 0.5 \le x_2 \le 2.0,\\
&\quad 0.5 \le x_3 \le 2.0.
\end{aligned}$$

Where:

$$g_1(x)=\frac{3x_1+4x_2-x_3+0.6}{2x_1-x_2+x_3+0.6}$$
$$g_2(x)=\frac{4x_1-x_2+3x_3+0.5}{10x_1+5x_2-x_3+0.5}$$
$$g_3(x)=\frac{5x_1-x_2+5x_3+0.5}{12x_1+6x_2-x_3}$$
$$g_4(x)=\frac{5x_1-x_2+6x_3+0.5}{12x_1+7x_2-x_3+0.9}$$

Select the feasible error $\varepsilon_f = 0.005$, numerical results are given as follows: the number of iteration is 497, the maximal number of active nodes necessary 360, the execution time in seconds t = 1.03535 sec, obtain the optimal value V* = 0.981545886, the global optimal solution is $x^*_1 = 1.390625000$, $x^*_2 = 0.5$, $x^*_3 = 1.982309883$.

Example 3:

$$\begin{aligned}&\min\max\ \{g_1(x),g_2(x),g_3(x),g_4(x),g_5(x)\}\\ &\text{s.t.}\quad 2x_1+x_2-x_3\le 5,\\ &\quad -2x_1+x_2-2x_3\le -0.1,\\ &\quad 11x_1+6x_2+12x_3\le 46,\\ &\quad 11x_1+13x_2+6x_3\le 53,\\ &\quad -7x_1+x_2+x_3\le -1,\\ &\quad 1.0\le x_1\le 2.0,\\ &\quad 0.35\le x_2\le 0.9,\ 1.0\le x_2\le 1.55.\end{aligned}$$

Where:

$$g_1(x)=\frac{3x_1+4x_2-x_3+0.8}{3x_1-x_2+x_3+0.8}$$
$$g_2(x)=\frac{4x_1-x_2+3x_3+0.6}{9x_1+9x_2-x_3+0.6}$$
$$g_3(x)=\frac{5x_1-x_2+5x_3+0.6}{12x_1+6x_2-x_3+0.6}$$
$$g_4(x)=\frac{5x_1-x_2+6x_3+0.5}{13x_1+7x_2-x_3+0.9}$$
$$g_5(x)=\frac{7x_1-x_2+7x_3+0.6}{12x_1+6x_2-x_3+0.9}$$

Select the feasible error $\varepsilon_f = 0.001$, numerical results are given as follows: the number of iteration is 970, the maximal number of active nodes necessary 773, the execution time in seconds t = 2.18618 sec, obtain the optimal value V* = 0.897576556, the global optimal solution is $x^*_1 = 1.998046875$, $x^*_2 = 0.35$, $x^*_3 = 1.066406250$.

Example 4:

$$\begin{aligned}&\min\max\ \{g_1(x),g_2(x),g_3(x),g_4(x),g_5(x)\}\\ &\text{s.t.}\quad 2x_1+2x_2-x_3\le 3,\\ &\quad -2x_1+x_2-3x_3\le -1,\\ &\quad 11x_1+7x_2+12x_3\le 47,\\ &\quad 13x_1+13x_2+6x_3\le 56,\\ &\quad -6x_1+2x_2+3x_3\le -1,\\ &\quad 1.0\le x_1\le 2.0,\ 0.35\le x_2\le 0.9,\\ &\quad 1.0\le x_3\le 1.55.\end{aligned}$$

Where:

$$g_1(x)=\frac{6x_1+4x_2-x_3+0.7}{6x_1-x_2+2x_3+0.7}$$
$$g_2(x)=\frac{9x_1-x_2+4x_3+0.7}{9x_1+3x_2-x_3+0.7}$$
$$g_3(x)=\frac{4x_1-x_2+6x_3+0.8}{11x_1+7x_2-x_3+0.8}$$
$$g_4(x)=\frac{7x_1-x_2+7x_3+0.8}{11x_1+9x_2-x_3+0.8}$$
$$g_5(x)=\frac{7x_1-x_2+7x_3+0.7}{12x_1+7x_2-x_3+0.7}$$

Select the feasible error $\varepsilon_f = 0.001$, numerical results are given as follows: the number of iteration is 1183, the maximal number of active nodes necessary 1056, the execution time in seconds t = 2.63849 sec, obtain the optimal value V* = 1.124582064, the global optimal solution is $x^*_1 = 1.311523438$, $x^*_2 = 0.865625000$, $x^*_3 = 1.034375000$.

Example 5:

$$\begin{aligned}&\min\max\ \{g_1(x),g_2(x),g_3(x),g_4(x),g_5(x)\}\\ &\text{s.t.}\quad 2x_1+2x_2-x_3\le 2,\\ &\quad -2x_1+x_2-3x_3\le -0.5,\\ &\quad 11x_1+7x_2+12x_3\le 45,\\ &\quad 13x_1+13x_2+6x_3\le 54,\\ &\quad -6x_1+2x_2+3x_3\le -1,\\ &\quad 1.0\le x_1\le 2.0,\ 0.35\le x_2\le 0.9,\\ &\quad 1.0\le x_3\le 1.55.\end{aligned}$$

Where:

$$g_1(x)=\frac{6x_1+3x_2-x_3+0.7}{6x_1-0.5x_2+2x_3+0.7}$$

$$g_2(x)=\frac{8x_1-x_2+4x_3+0.7}{8x_1+3x_2-x_3+0.7}$$

$$g_3(x)=\frac{5x_1-x_2+6x_3+0.8}{12x_1+7x_2-x_3+0.8}$$

$$g_4(x)=\frac{8x_1-x_2+7x_3+0.8}{10x_1+9x_2-x_3+0.8}$$

$$g_5(x)=\frac{7x_1-x_2+7x_3+0.6}{11x_1+7x_2-x_3+0.6}$$

Select the feasible error $\varepsilon_f = 0.001$, numerical results are given as follows: the number of iteration is 273, the maximal number of active nodes necessary 213, the execution time in seconds t = 0.63138 sec, obtain the optimal value V* = 1.281956612, the global optimal solution is $x^*_1 = 1.092708333$, $x^*_2 = 0.865625000$, $x^*_3 = 1.275000000$.

Example 6:

$$\begin{aligned}\min\max\ & \{g_1(x),g_2(x),g_3(x),g_4(x),g_5(x)\}\\ \text{s.t.}\quad & 2x_1+2x_2-x_3\le 2,\\ & -2x_1+x_2-3x_3\le -0.5,\\ & 11x_1+7x_2+12x_3\le 46,\\ & 13x_1+13x_2+6x_3\le 58,\\ & -6x_1+2x_2+3x_3\le -1,\\ & 1.0\le x_1\le 2.0,\ 0.35\le x_2\le 0.9,\\ & 1.0\le x_3\le 1.55.\end{aligned}$$

Where:

$$g_1(x)=\frac{7x_1+3x_2-x_3+0.4}{7x_1-0.5x_2+2x_3+0.4}$$

$$g_2(x)=\frac{6x_1-x_2+4x_3+0.6}{6x_1+3x_2-x_3+0.6}$$

$$g_3(x)=\frac{6x_1-x_2+7x_3+0.6}{11x_1+7x_2-2x_3+0.6}$$

$$g_4(x)=\frac{6x_1-x_2+7x_3+0.6}{7x_1+9x_2-x_3+0.6}$$

$$g_5(x)=\frac{5x_1-x_2+7x_3+0.7}{12x_1+7x_2-x_3+0.6}$$

Select the feasible error $\varepsilon_f = 0.001$, numerical results are given as follows: the number of iteration is 150, the maximal number of active nodes necessary is 143, the execution time in seconds t = 0.381588 sec, obtain the optimal value V* = 1.349330947, the global optimal solution is $x^*_1 = 1.092708333$, $x^*_2 = 0.865625000$, $x^*_3 = 1.275000000$.

Example 7:

$$\begin{aligned}\min\max\ & \{g_1(x),g_2(x),g_3(x),g_4(x),g_5(x)\}\\ \text{s.t.}\quad & 2x_1+2x_2-x_3\le 2,\\ & -2x_1+x_2-3x_3\le -0.4,\\ & 11x_1+7x_2+12x_3\le 50,\\ & 13x_1+13x_2+6x_3\le 59,\\ & -6x_1+2x_2+3x_3\le -1,\\ & 1.0\le x_1\le 2.1,\ 0.35\le x_2\le 0.8,\\ & 1.1\le x_3\le 1.55.\end{aligned}$$

Where:

$$g_1(x)=\frac{4x_1+4x_2-x_3+0.2}{4x_1-0.5x_2+2x_3+0.2}$$

$$g_2(x)=\frac{5x_1-x_2+4x_3+0.4}{5x_1+3x_2-x_3+0.4}$$

$$g_3(x)=\frac{4x_1-x_2+4x_3+0.5}{8x_1+7x_2-x_3+0.5}$$

$$g_4(x)=\frac{5x_1-x_2+7x_3+0.5}{6x_1+9x_2-x_3+0.6}$$

$$g_5(x)=\frac{4x_1-x_2+6x_3+0.9}{13x_1+8x_2-x_3+0.9}$$

Select the feasible error $\varepsilon_f = 0.001$, numerical results are given as follows: the number of iteration is 226, the maximal number of active nodes necessary is 146, the execution time in seconds t = 0.63138 sec, obtain the optimal value V* = 1.523500353, the global optimal solution is $x^*_1 = 1.048958333$, $x^*_2 = 0.743750000$, $x^*_3 = 1.268750000$.

Example 8:

$$\begin{aligned}\min\max\ & \{g_1(x),g_2(x),g_3(x),g_4(x),g_5(x)\}\\ \text{s.t.}\quad & 2x_1+2x_2-x_3\le 1,\\ & -2x_1+x_2-3x_3\le -0.3,\\ & 11x_1+7x_2+12x_3\le 45,\\ & 13x_1+13x_2+6x_3\le 50,\\ & -6x_1+2x_2+3x_3\le -1,\\ & 1.0\le x_1\le 1.2,\ 0.35\le x_2\le 0.7,\\ & 1.2\le x_3\le 1.55.\end{aligned}$$

Where:

$$g_1(x)=\frac{3x_1+3x_2-x_3+0.8}{3x_1-0.5x_2+2x_3+0.8}$$

$$g_2(x)=\frac{5x_1-x_2+5x_3+0.3}{5x_1+5x_2-x_3+0.3}$$

$$g_3(x)=\frac{5x_1-x_2+5x_3+0.5}{9x_1+6x_2-x_3+0.5}$$

$$g_4(x)=\frac{6x_1-x_2+6x_3+0.5}{6x_1+12x_2-x_3+0.6}$$

$$g_5(x)=\frac{11x_1-x_2+6x_3+0.2}{12x_1+8x_2-x_3+0.2}$$

Select the feasible error $\varepsilon_f = 0.001$, numerical results are given as follows: the number of iteration is 48, the maximal number of active nodes necessary is 49, the execution time in seconds t = 0.168447 sec, obtain the optimal value V* = 1.646398127, the global optimal solution is x^*_1 = 1.116666667, x^*_2 = 0.656250000, x^*_3 = 1.462500000.

Example 9:

$$\begin{aligned}
&\min\max\ \{g_1(x),g_2(x),g_3(x),g_4(x),g_5(x)\}\\
&\text{s.t.}\quad 2x_1+2x_2-x_3\le 1.3,\\
&\quad -2x_1+2x_2-3x_3\le -0.35,\\
&\quad 12x_1+7x_2+12x_3\le 46,\\
&\quad 12x_1+13x_2+6x_3\le 52,\\
&\quad -6x_1+2x_2+3x_3\le -1,\\
&\quad 1.0\le x_1\le 1.25,\\
&\quad 0.35\le x_2\le 0.75, 1.2\le x_3\le 1.55.
\end{aligned}$$

Where:

$$g_1(x)=\frac{2x_1+2x_2-x_3+0.6}{2x_1-0.5x_2+2x_3+0.6}$$

$$g_2(x)=\frac{6x_1-1.1x_2+5x_3+0.2}{6x_1+5x_2-1.1x_3+0.2}$$

$$g_3(x)=\frac{5.2x_1-x_2+5.2x_3+0.6}{9.1x_1+6.1x_2-x_3+0.5}$$

$$g_4(x)=\frac{6.2x_1-x_2+6.3x_3+0.55}{6.1x_1+12.1x_2-x_3+0.65}$$

$$g_5(x)=\frac{11.1x_1-x_2+6.3x_3+0.26}{12.1x_1+8.2x_2-x_3+0.25}$$

Select the feasible error $\varepsilon_f = 0.001$, numerical results are given as follows: the number of iteration is 39, the maximal number of active nodes necessary is 38, the execution time in seconds t = 0.111753 sec, obtain the optimal value V* = 1.668691804, the global optimal solution is x^*_1 = 1.097916667, x^*_2 = 0.600000000, x^*_3 = 1.462500000.

Numerical results show that our algorithm can globally solve the problem (LFP) on a microcomputer. By the numerical results of the test examples 1-9, we can follow that the proposed algorithm is competitive and can be used to globally solve the problem (LFP).

CONCLUDING REMARKS

In this study, we present a numerical algorithm and test examples for a class of linear fractional programming problems which is investigated in recent literature (Feng *et al*., 2011). Based on the known linearization technique proposed in recent literature (Feng *et al*., 2011), a branch and bound algorithm is proposed and implemented which is convergent to the global optimal solution of the linear fractional programming problem (LFP). Several test examples are used to verify the feasibility and computational efficiency of the proposed algorithm.

ACKNOWLEDGMENT

This study is supported by the Foundation for University Key Teacher by the Ministry of Education of Henan Province (2010GGJS-140).

REFERENCES

Ahmad, I. and Z. Husain, 2006. Duality in nondifferentiable minimax fractional programming with generalized convexity. Applied Math. Comput., 176: 545-551.

Feng, Q.G., H.P. Mao and H.W. Jiao, 2011. A feasible method for a class of mathematical problems in manufacturing system. Key Eng. Mater., 460-461: 806-809.

Phuong, N.T.H. and H. Tuy, 2003. A unified monotonic approach to generalized linear fractional programming. J. Global Optim., 26: 229-259.

Schaible, S. and J. Shi, 2003. Recent developments in fractional programming: Single-ratio and max-min case. Proceedings of the Third International Conference on Nonlinear Analysis and Convex Analysis, August 25-29, 2003, Tokyo, Japan, pp: 493-506.

Schaible, S., 1995. Fractional Programming. In: Handbook of Global Optimization, Horst, R. and P.M. Pardalos (Eds.). Kluwer Academic Publishers, Dordrecht, Boston, London, pp: 495-608.

Sekitani, K., J. Shi and Y. Yamamoto, 1995. General fractional programming: Min-max convex quadratic case. Proceedings of the 3rd Conference of the Association of Asian-Pacific Operational Research Societies (APORS) within IFORS, July 26-29, 1994, Fukuoka, Japan, pp: 505-514.