Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2011, Article ID 409491, 12 pages
doi:10.1155/2011/409491

Research Article

A Branch-and-Reduce Approach for Solving Generalized Linear Multiplicative Programming

Chun-Feng Wang,[1, 2] San-Yang Liu,[1] and Geng-Zhong Zheng[3]

[1] *Department of Mathematical Sciences, Xidian University, Xi'an 710071, China*
[2] *Department of Mathematics, Henan Normal University, Xinxiang 453007, China*
[3] *School of Computer Science and Technology, Xidian University, Xi'an 710071, China*

Correspondence should be addressed to Chun-Feng Wang, wangchunfeng09@126.com

Received 15 March 2011; Accepted 12 May 2011

Academic Editor: Victoria Vampa

We consider a branch-and-reduce approach for solving generalized linear multiplicative programming. First, a new lower approximate linearization method is proposed; then, by using this linearization method, the initial nonconvex problem is reduced to a sequence of linear programming problems. Some techniques at improving the overall performance of this algorithm are presented. The proposed algorithm is proved to be convergent, and some experiments are provided to show the feasibility and efficiency of this algorithm.

1. Introduction

In this paper, the following generalized linear multiplicative programming is considered:

$$\begin{aligned} \min \quad & \prod_{i=1}^{p_0}\left(c_{0i}^T x + d_{0i}\right)^{\gamma_{0i}} \\ \text{s.t.} \quad & \prod_{i=1}^{p_j}\left(c_{ji}^T x + d_{ji}\right)^{\gamma_{ji}} \le \beta_j, \quad j = 1,\ldots,m, \\ & x \in X^0 = [l,u] \subset R^n, \end{aligned} \tag{P}$$

where $c_{ji} = (c_{ji1}, c_{ji2}, \ldots, c_{jin})^T \in R^n$, $d_{ji} \in R$, and $\beta_j \in R$, $\gamma_{ji} \in R$, $\beta_j > 0$ and for all $x \in X^0$, $c_{ji}^T x + d_{ji} > 0$, $j = 0,\ldots,m$, $i = 1,\ldots,p_j$.

Since a large number of practical applications in various fields can be put into problem (P), including VLSI chip design [1], decision tree optimization [2], multicriteria optimization problem [3], robust optimization [4], and so on, this problem has attracted considerable attention in the past years.

It is well known that the product of affine functions need not be (quasi) convex, thus the problem can have multiple locally optimal solutions, many of which fail to be globally optimal, that is, problem (P) is multiextremal [5].

In the last decade, many solution algorithms have been proposed for globally solving special forms of (P). They can be generally classified as outer-approximation method [6], decomposition method [7], finite branch and bound algorithms [8, 9], and cutting plane method [10]. However, the global optimization algorithms based on the general form (P) have been little studied. Recently, several algorithms were presented for solving problem (P) [11–15].

The aim of this paper is to provide a new branch-and-reduce algorithm for globally solving problem (P). Firstly, by using the property of logarithmic function, we derive an equivalent problem (Q) of the initial problem (P), which has the same optimal solution as the problem (P). Secondly, by utilizing the special structure of (Q), we present a new linear relaxation technique, which can be used to construct the linear relaxation programming problem for (Q). Finally, the initial nonconvex problem (P) is systematically converted into a series of linear programming problems. The solutions of these converted problems can be as close as possible to the globally optimal solution of (Q) by successive refinement process.

The main features of this algorithm: (1) the problem investigated in this paper has a more general form than those in [6–10]; (2) a new linearization method for solving the problem (Q) is proposed; (3) these generated linear relaxation programming problems are embedded within a branch and bound algorithm without increasing the number of variables and constraints; (4) some techniques are proposed to improve the convergence speed of our algorithm.

This paper is organized as follows. In Section 2, an equivalent transformation and a new linear relaxation technique are presented for generating the linear relaxation programming problem (LRP) for (Q), which can provide a lower bound for the optimal value of (Q). In Section 3, in order to improve the convergence speed of our algorithm, we present a reducing technique. In Section 4, the global optimization algorithm is described in which the linear relaxation problem and reducing technique are embedded, and the convergence of this algorithm is established. Numerical results are reported to show the feasibility of our algorithm in Section 5.

2. Linear Relaxation Problem

Without loss of generality, assume that, for $0 \le i \le T_j$, $\gamma_{ji} > 0$, $T_j+1 \le i \le p_j$, $\gamma_{ji} < 0$, $j = 0, \dots, m$, $i = 1, \dots, p_j$.

By using the property of logarithmic function, the equivalent problem (Q) of (P) can be derived, which has the same optimal solution as (P),

$$\begin{aligned}
\min \quad & \phi_0(x) = \sum_{i=1}^{T_0} \gamma_{0i} \ln\left(c_{0i}^T x + d_{0i}\right) + \sum_{i=T_0+1}^{p_0} \gamma_{0i} \ln\left(c_{0i}^T x + d_{0i}\right) \\
\text{s.t.} \quad & \phi_j(x) = \sum_{i=1}^{T_j} \gamma_{ji} \ln\left(c_{ji}^T x + d_{ji}\right) + \sum_{i=T_j+1}^{p_j} \gamma_{ji} \ln\left(c_{ji}^T x + d_{ji}\right) \le \ln \beta_j, \\
& x \in X^0 = [l, u] \subset R^n, \quad j = 1, \dots, m.
\end{aligned} \tag{Q}$$

Thus, for solving problem (P), we may solve its equivalent problem (Q) instead. Toward this end, we present a branch-and-reduce algorithm. In this algorithm, the principal aim is to construct linear relaxation programming problem (LRP) for (Q), which can provide a lower bound for the optimal value of (Q).

Suppose that $X = [\underline{x}, \overline{x}]$ represents either the initial rectangle of problem (Q), or modified rectangle as defined for some partitioned subproblem in a branch and bound scheme. The problem (LRP) can be realized through underestimating every function $\phi_j(x)$ with a linear relaxation function $\phi_j^l(x)$ $(j = 0, \ldots, m)$. All the details of this linearization method for generating relaxations will be given below.

Consider the function $\phi_j(x)$ $(j = 0, \ldots, m)$. Let $\phi_{j1}(x) = \sum_{i=1}^{T_j} \gamma_{ji} \ln(c_{ji}^T x + d_{ji})$, and $\phi_{j2}(x) = \sum_{i=T_j+1}^{p_j} \gamma_{ji} \ln(c_{ji}^T x + d_{ji})$, then, $\phi_{j1}(x)$ and $\phi_{j2}(x)$ are concave function and convex function, respectively.

First, we consider the function $\phi_{j1}(x)$. For convenience in expression, we introduce the following notations:

$$\begin{gathered}
X_{ji} = c_{ji}^T x + d_{ji} = \sum_{t=1}^{n} c_{jit} x_t + d_{ji}, \\
\underline{X}_{ji} = \sum_{t=1}^{n} \min\{c_{jit}\underline{x}_t, c_{jit}\overline{x}_t\} + d_{ji}, \\
\overline{X}_{ji} = \sum_{t=1}^{n} \max\{c_{jit}\underline{x}_t, c_{jit}\overline{x}_t\} + d_{ji}, \\
K_{ji} = \frac{\ln(\overline{X}_{ji}) - \ln(\underline{X}_{ji})}{\overline{X}_{ji} - \underline{X}_{ji}}, \\
f_{ji}(x) = \ln(c_{ji}^T x + d_{ji}) = \ln(X_{ji}), \\
h_{ji}(x) = \ln(\underline{X}_{ji}) + K_{ji}(X_{ji} - \underline{X}_{ji}) = \ln(\underline{X}_{ji}) + K_{ji}\left(\sum_{t=1}^{n} c_{jit} x_t + d_{ji} - \underline{X}_{ji}\right).
\end{gathered} \tag{2.1}$$

By Theorem 1 in [11], we can derive the lower bound function $\phi_{j1}^l(x)$ of $\phi_{j1}(x)$ as follows:

$$\phi_{j1}^l(x) = \sum_{i=1}^{T_j} \gamma_{ji} h_{ji}(x) \le \sum_{i=1}^{T_j} \gamma_{ji} f_{ji}(x) = \phi_{j1}(x). \tag{2.2}$$

Second, we consider function $\phi_{j2}(x)$ $(j = 0, \ldots, m)$. Since $\phi_{j2}(x)$ is a convex function, by the property of the convex function, we have

$$\phi_{j2}(x) \ge \phi_{j2}(x_{\text{mid}}) + \nabla\phi_{j2}(x_{\text{mid}})^T (x - x_{\text{mid}}) = \phi_{j2}^l(x), \tag{2.3}$$

where $x_{\text{mid}} = (1/2)(\underline{x} + \overline{x})$,

$$\nabla \phi_{j2}(x) = \begin{pmatrix} \dfrac{\gamma_{j,T_j+1} c_{j,T_j+1,1}}{c_{j,T_j+1}^T x + d_{j,T_j+1}} + \dfrac{\gamma_{j,T_j+2} c_{j,T_j+2,1}}{c_{j,T_j+2}^T x + d_{j,T_j+2}} + \cdots + \dfrac{\gamma_{j,p_j} c_{j,p_j,1}}{c_{j,p_j}^T x + d_{j,p_j}} \\ \vdots \\ \dfrac{\gamma_{j,T_j+1} c_{j,T_j+1,n}}{c_{j,T_j+1}^T x + d_{j,T_j+1}} + \dfrac{\gamma_{j,T_j+2} c_{j,T_j+2,n}}{c_{j,T_j+2}^T x + d_{j,T_j+2}} + \cdots + \dfrac{\gamma_{j,p_j} c_{j,p_j,n}}{c_{jp_j}^T x + d_{jp_j}} \end{pmatrix}. \tag{2.4}$$

Finally, from (2.2) and (2.3), for all $x \in X$, we have

$$\phi_j^l(x) = \phi_{j1}^l(x) + \phi_{j2}^l(x) \le \phi_j(x). \tag{2.5}$$

Theorem 2.1. *For all $x \in X$, consider the functions $\phi_j(x)$ and $\phi_j^l(x)$, $j = 0, \ldots, m$. Then, the difference between $\phi_j^l(x)$ and $\phi_j(x)$ satisfies*

$$\phi_j(x) - \phi_j^l(x) \longrightarrow 0, \quad \text{as } \|\overline{x} - \underline{x}\| \longrightarrow 0, \tag{2.6}$$

where $\|\overline{x} - \underline{x}\| = \max\{\overline{x}_i - \underline{x}_i \mid i = 1, \ldots, n\}$.

Proof. Let $\Delta^1 = \phi_{j1}(x) - \phi_{j1}^l(x)$, $\Delta^2 = \phi_{j2}(x) - \phi_{j2}^l(x)$. Since $\phi_j(x) - \phi_j^l(x) = \phi_{j1}(x) - \phi_{j1}^l(x) + \phi_{j2}(x) - \phi_{j2}^l(x) = \Delta^1 + \Delta^2$, we only need to prove $\Delta^1 \to 0$, $\Delta^2 \to 0$ as $\|\overline{x} - \underline{x}\| \to 0$.

First, consider Δ^1. By the definition of Δ^1, we have

$$\Delta^1 = \phi_{j1}(x) - \phi_{j1}^l(x) = \sum_{i=1}^{T_j} \gamma_{ji} \left(f_{ji}(x) - h_{ji}(x) \right). \tag{2.7}$$

Furthermore, by Theorem 1 in [11], we know that $f_{ji}(x) - h_{ji}(x) \to 0$ as $\|\overline{x} - \underline{x}\| \to 0$. Thus, we have $\Delta^1 \to 0$ as $\|\overline{x} - \underline{x}\| \to 0$.

Second, consider Δ^2. From the definition of Δ^2, it follows that

$$\begin{aligned} \Delta^2 &= \phi_{j2}(x) - \phi_{j2}^l(x) \\ &= \phi_{j2}(x) - \phi_{j2}(x_{\text{mid}}) - \nabla \phi_{j2}(x_{\text{mid}})^T (x - x_{\text{mid}}) \\ &= \nabla \phi_{j2}(\xi)^T (x - x_{\text{mid}}) - \nabla \phi_{j2}(x_{\text{mid}})^T (x - x_{\text{mid}}) \\ &\le \left\| \nabla^2 \phi_{j2}(\eta) \right\| \|\xi - x_{\text{mid}}\| \|x - x_{\text{mid}}\|, \end{aligned} \tag{2.8}$$

where ξ, η are constant vectors, which satisfy $\phi_{j2}(x) - \phi_{j2}(x_{\text{mid}}) = \nabla\phi_{j2}(\xi)^T(x - x_{\text{mid}})$ and $\nabla\phi_{j2}(\xi) - \nabla\phi_{j2}(x_{\text{mid}}) = \nabla^2\phi_{j2}(\eta)^T(\xi - x_{\text{mid}})$, respectively. Since $\nabla^2\phi_{j2}(x)$ is continuous, and X is a compact set, there exists some $M > 0$ such that $\|\nabla^2\phi_{j2}(x)\| \le M$. From (2.8), it implies that $\Delta^2 \le M\|\overline{x} - \underline{x}\|^2$. Furthermore, we have $\Delta^2 \to 0$ as $\|\overline{x} - \underline{x}\| \to 0$.

Taken together above, it implies that $\phi_j(x) - \phi_j^l(x) = \Delta^1 + \Delta^2 \to 0$ as $\|\overline{x} - \underline{x}\| \to 0$, and the proof is complete. □

From Theorem 2.1, it follows that the function $\phi_j^l(x)$ can approximate enough the function $\phi_j(x)$ as $\|\overline{x} - \underline{x}\| \to 0$.

Based on the above discussion, the linear relaxation programming problem (LRP) of (Q) over X can be obtained as follows:

$$\begin{aligned} \min \quad & \phi_0^l(x) \\ \text{s.t.} \quad & \phi_j^l(x) \le \ln\beta_j, \quad j = 1,\ldots,m, \\ & x \in X = [\underline{x}, \overline{x}] \subset R^n. \end{aligned} \tag{LRP}$$

Obviously, the feasible region for the problem (Q) is contained in the new feasible region for the problem (LRP), thus, the minimum $V(\text{LRP})$ of (LRP) provides a lower bound for the optimal value $V(\text{Q})$ of problem (Q) over the rectangle X, that is $V(\text{LRP}) \le V(\text{Q})$.

3. Reducing Technique

In this section, we pay our attention on how to form the new reducing technique for eliminate the region in which the global minimum of (Q) does not exist.

Assume that UB is the current known upper bound of the optimal value ϕ_0^* of the problem (Q). Let

$$\begin{gathered} \alpha_t = \sum_{i=1}^{T_0}\gamma_{0i}K_{0i}c_{0it} + \nabla\phi_{j2}(x_{\text{mid}})_t, \quad t = 1,\ldots,n, \\ T = \sum_{i=1}^{T_0}\gamma_{0i}\left[\ln(\underline{X}_{0i}) + K_{0i}d_{0i} - K_{0i}\underline{X}_{0i}\right] + \phi_{02}(x_{\text{mid}}) - \nabla\phi_{02}(x_{\text{mid}})^T x_{\text{mid}}, \\ \rho_k = \text{UB} - \sum_{t=1, t\neq k}^{n} \min\{\alpha_t\underline{x}_t, \alpha_t\overline{x}_t\} - T, \quad k = 1,\ldots,n. \end{gathered} \tag{3.1}$$

The reducing technique is derived as in the following theorem.

Theorem 3.1. *For any subrectangle $X = (X_t)_{n\times 1} \subseteq X^0$ with $X_t = [\underline{x}_t, \overline{x}_t]$. If there exists some index $k \in \{1,2,\ldots,n\}$ such that $\alpha_k > 0$ and $\rho_k < \alpha_k\overline{x}_k$, then there is no globally optimal solution of* (Q) *over X^1; if $\alpha_k < 0$ and $\rho_k < \alpha_k\underline{x}_k$, for some k, then there is no globally optimal solution of* (Q) *over X^2,* *where*

$$X^1 = \left(X_t^1\right)_{n\times 1} \subseteq X, \quad \textit{with } X_t^1 = \begin{cases} X_t, & t \neq k, \\ \left(\dfrac{\rho_k}{\alpha_k}, \overline{x}_k\right] \bigcap X_t, & t = k, \end{cases}$$
$$X^2 = \left(X_t^2\right)_{n\times 1} \subseteq X, \quad \textit{with } X_t^2 = \begin{cases} X_t, & t \neq k, \\ \left[\underline{x}_k, \dfrac{\rho_k}{\alpha_k}\right) \bigcap X_t, & t = k. \end{cases} \tag{3.2}$$

Proof. First, we show that for all $x \in X^1$, $\phi_0(x) > \text{UB}$. Consider the kth component x_k of x. Since $x_k \in (\rho_k/\alpha_k, \overline{x}_k]$, it follows that

$$\frac{\rho_k}{\alpha_k} < x_k \leq \overline{x}_k. \tag{3.3}$$

From $\alpha_k > 0$, we have $\rho_k < \alpha_k x_k$. For all $x \in X^1$, by the above inequality and the definition of ρ_k, it implies that

$$\text{UB} - \sum_{t=1, t\neq k}^{n} \min\left\{\alpha_t \underline{x}_t, \alpha_t \overline{x}_t\right\} - T < \alpha_k x_k, \tag{3.4}$$

that is

$$\begin{aligned} \text{UB} &< \sum_{t=1, t\neq k}^{n} \min\left\{\alpha_t \underline{x}_t, \alpha_t \overline{x}_t\right\} + \alpha_k x_k + T \\ &\leq \sum_{t=1}^{n} \alpha_t x_t + T = \phi_0^l(x). \end{aligned} \tag{3.5}$$

Thus, for all $x \in X^1$, we have $\phi_0(x) \geq \phi_0^l(x) > \text{UB} \geq \phi_0^*$, that is, for all $x \in X^1$, $\phi_0(x)$ is always greater than the optimal value ϕ_0^* of the problem (Q). Therefore, there cannot exist globally optimal solution of (Q) over X^1.

For all $x \in X^2$, if there exists some k such that $\alpha_k < 0$ and $\rho_k < \alpha_k \underline{x}_k$, from arguments similar to the above, it can be derived that there is no globally optimal solution of (Q) over X^2. □

4. Algorithm and Its Convergence

In this section, based on the former results, we present a branch-and-reduce algorithm to solve the problem (Q). There are three fundamental processes in the algorithm procedure: a reducing process, a branching process, and an updating upper and lower bounds process.

Firstly, based on Section 3, when some conditions are satisfied, the reducing process can cut away a large part of the currently investigated feasible region in which the global optimal solution does not exist.

The second fundamental process iteratively subdivides the rectangle X into two subrectangles. During each iteration of the algorithm, the branching process creates a more refined partition that cannot yet be excluded from further consideration in searching for a global optimal solution for problem (Q). In this paper we choose a simple and standard bisection rule. This branching rule is sufficient to ensure convergence since it drives the intervals shrinking to a singleton for all the variables along any infinite branch of the branch and bound tree. Consider any node subproblem identified by rectangle $X = \{x \in R^n \mid \underline{x}_i \leq x_i \leq \overline{x}_i, = 1, \dots, n\} \subseteq X^0$. This branching rule is as follows.

(i) Let $p = \arg\max\{\overline{x}_i - \underline{x}_i \mid i = 1, \dots, n\}$.

(ii) Let $\gamma = (\underline{x}_p + \overline{x}_p)/2$.

(iii) Let

$$\begin{aligned} \overline{\mathrm{X}} &= \left\{x \in R^n \mid \underline{x}_i \leq x_i \leq \overline{x}_i,\ i \neq p,\ \underline{x}_p \leq x_p \leq \gamma\right\}, \\ \overline{\overline{\mathrm{X}}} &= \left\{x \in R^n \mid \underline{x}_i \leq x_i \leq \overline{x}_i,\ i \neq p,\ \gamma \leq x_p \leq \overline{x}_p\right\}. \end{aligned} \tag{4.1}$$

By this branching rule, the rectangle X is partitioned into two subrectangles $\overline{\mathrm{X}}$ and $\overline{\overline{\mathrm{X}}}$.

The third process is to update the upper and lower bounds of the optimal value of (Q). This process needs to solve a sequence of linear programming problems and to compute the objective function value of (Q) at the midpoint of the subrectangle X for the problem (Q). In addition, some bound tightening strategies are applied to the proposed algorithm.

The basic steps of the proposed algorithm are summarized as follows. In this algorithm, let $\mathrm{LB}(\mathrm{X}^k)$ be the optimal value of (LRP) over the subrectangle $X = X^k$, and $x^k = x(X^k)$ be an element of corresponding arg min. Since $\phi_j^l(x)$ $(j = 0, \dots, m)$ is a linear function, for convenience in expression, assume that it is expressed as follows $\phi_j^l(x) = \sum_{t=1}^n a_{jt}x_t + b_j$, where $a_{jt}, b_j \in R$. Thus, we have $\min_{x \in X}\phi_j^l(x) = \sum_{t=1}^n \min\{a_{jt}\underline{x}_t, a_{jt}\overline{x}_t\} + b_j$.

4.1. Algorithm Statement

Step 1 (initialization). Let the set all active node $Q_0 = \{\mathrm{X}^0\}$, the upper bound $\mathrm{UB} = +\infty$, the set of feasible points $F = \emptyset$, some accuracy tolerance $\epsilon > 0$ and the iteration counter $k = 0$.

Solve the problem (LRP) for $X = \mathrm{X}^0$. Let $\mathrm{LB}_0 = \mathrm{LB}(\mathrm{X}^0)$ and $x^0 = x(X^0)$. If x^0 is a feasible point of (Q), then let

$$\mathrm{UB} = \phi_0\left(x^0\right), \qquad F = F \bigcup \left\{x^0\right\}. \tag{4.2}$$

If $\mathrm{UB} < \mathrm{LB}_0 + \epsilon$, then stop: x^0 is an ϵ-optimal solution of (Q). Otherwise, proceed.

Step 2 (updating the upper bound). Select the midpoint x_{mid}^k of X^k; if x_{mid}^k is feasible to (Q), then $F = F \cup \{x_{\text{mid}}^k\}$. Let the upper bound $\mathrm{UB} = \min\{\phi_0(x_{\text{mid}}^k), \mathrm{UB}\}$ and the best known feasible point $x^* = \arg\min_{x \in F}\phi_0(x)$.

Step 3 (branching and reducing). Using the branching rule to partition X^k into two new subrectangles, and denote the set of new partition rectangles as $\overline{X}^k$. For each $X \in \overline{X}^k$, utilize the reducing technique of Theorem 3.1 to reduce box X, and compute the lower bound $\phi_j^l(x)$ of $\phi_j(x)$ over the rectangle X. If for $j = 1, \dots, m$, there exists some j such that $\min_{x \in X} \phi_j^l(x) > \ln \beta_j$, or for $j = 0$, $\min_{x \in X} \phi_0^l(x) > \text{UB}$, then the corresponding subrectangle X will be removed from $\overline{X}^k$, that is, $\overline{X}^k = \overline{X}^k \setminus X$, and skip to the next element of $\overline{X}^k$.

Step 4 (bounding). If $\overline{X}^k \neq \emptyset$, solve (LRP) to obtain $\text{LB}(X)$ and $x(X)$ for each $X \in \overline{X}^k$. If $\text{LB}(X) > \text{UB}$, set $\overline{X}^k = \overline{X}^k \setminus X$; otherwise, update the best available solution UB, F and x^* if possible, as in the Step 2. The partition set remaining is now $Q_k = (Q_k \setminus X^k) \bigcup \overline{X}^k$, and a new lower bound is $\text{LB}_k = \inf_{X \in Q_k} \text{LB}(X)$.

Step 5 (convergence checking). Set

$$Q_{k+1} = Q_k \setminus \{X \mid \text{UB} - \text{LB}(X) \leq \epsilon, X \in Q_k\}. \tag{4.3}$$

If $Q_{k+1} = \emptyset$, then stop: UB is the ϵ-optimal value of (Q), and x^* is an ϵ-optimal solution. Otherwise, select an active node X^{k+1} such that $X^{k+1} = \arg\min_{X \in Q_{k+1}} \text{LB}(X)$, $x^{k+1} = x(X^{k+1})$. Set $k = k + 1$, and return to Step 2.

4.2. Convergence Analysis

In this subsection, we give the global convergence properties of the above algorithm.

Theorem 4.1 (convergence). *The above algorithm either terminates finitely with a globally ϵ-optimal solution, or generates an infinite sequence $\{x^k\}$ which any accumulation point is a globally optimal solution of* (Q).

Proof. When the algorithm is finite, by the algorithm, it terminates at some step $k \geq 0$. Upon termination, it follows that

$$\text{UB} - \text{LB}_k \leq \epsilon. \tag{4.4}$$

From Step 1 and Step 5 in the algorithm, a feasible solution x^* for the problem (Q) can be found, and the following relation holds

$$\phi_0(x^*) - \text{LB}_k \leq \epsilon. \tag{4.5}$$

Let v denote the optimal value of problem (Q). By Section 2, we have

$$\text{LB}_k \leq v. \tag{4.6}$$

Since x^* is a feasible solution of problem (Q), $\phi_0(x^*) \geq v$. Taken together above, it implies that

$$v \leq \phi_0(x^*) \leq \text{LB}_k + \epsilon \leq v + \epsilon, \tag{4.7}$$

and so x^* is a global ϵ-optimal solution to the problem (Q) in the sense that

$$v \leq \phi_0(x^*) \leq v + \epsilon. \tag{4.8}$$

When the algorithm is infinite, by [5], a sufficient condition for a global optimization to be convergent to the global minimum, requires that the bounding operation must be consistent and the selection operation is bound improving.

A bounding operation is called consistent if at every step any unfathomed partition can be further refined, and if any infinitely decreasing sequence of successively refined partition elements satisfies

$$\lim_{k \to \infty} (\text{UB} - \text{LB}_k) = 0, \tag{4.9}$$

where LB_k is a computed lower bound in stage k and UB is the best upper bound at iteration k not necessarily occurring inside the same subrectangle with LB_k. Now, we show that (4.9) holds.

Since the employed subdivision process is rectangle bisection, the process is exhaustive. Consequently, from Theorem 2.1 and the relationship $V(\text{LRP}) \leq V(\text{Q})$, the formulation (4.9) holds, this implies that the employed bounding operation is consistent.

A selection operation is called bound improving if at least one partition element where the actual lower bound is attained is selected for further partition after a finite number of refinements. Clearly, the employed selection operation is bound improving because the partition element where the actual lower bound is attained is selected for further partition in the immediately following iteration.

From the above discussion, and Theorem IV.3 in [5], the branch-and-reduce algorithm presented in this paper is convergent to the global minimum of (Q). □

5. Numerical Experiments

In this section, some numerical experiments are reported to verify the performance of the proposed algorithm. The algorithm is coded in Matlab 7.1. The simplex method is applied to solve the linear relaxation programming problems. The test problems are implemented on a Pentium IV (3.06 GHZ) microcomputer, and the convergence tolerance is set at $\epsilon = 1.0e - 4$ in our experiments.

Example 5.1 (see [12, 15]).

$$\begin{aligned}
\min \quad & (x_1+x_2+1)^{2.5}(2x_1+x_2+1)^{1.1}(x_1+2x_2+1)^{1.9} \\
\text{s.t.} \quad & (x_1+2x_2+1)^{1.1}(2x_1+2x_2+2)^{1.3} \le 50, \\
& 1 \le x_1 \le 3, \quad 1 \le x_2 \le 3.
\end{aligned} \tag{5.1}$$

Example 5.2 (see [15]).

$$\begin{aligned}
\min \quad & (2x_1+x_2-x_3+1)^{-0.2}(2x_1-x_2+x_3+1)(x_1+2x_2+1)^{0.5} \\
\text{s.t.} \quad & (3x_1-x_2+1)^{0.3}(2x_1-x_2+x_3+2)^{-0.1} \le 10, \\
& (1.2x_1+x_2+1)^{-1}(2x_1+2x_2+1)^{0.5} \le 12, \\
& (x_1+x_2+2)^{0.2}(1.5x_1+x_2+1)^{-2} \le 15, \\
& 1 \le x_1 \le 2, \quad 1 \le x_2 \le 2, \quad 1 \le x_3 \le 2.
\end{aligned} \tag{5.2}$$

Example 5.3 (see [12, 15]).

$$\begin{aligned}
\min \quad & (x_1+x_2+x_3)(2x_1+x_2+x_3)(x_1+2x_2+2x_3) \\
\text{s.t.} \quad & (x_1+2x_2+x_3)^{1.1}(2x_1+2x_2+x_3)^{1.3} \le 100, \\
& 1 \le x_1 \le 3, \quad 1 \le x_2 \le 3, \quad 1 \le x_3 \le 3.
\end{aligned} \tag{5.3}$$

Example 5.4 (see [13, 16]).

$$\begin{aligned}
\min \quad & (-x_1+2x_2+2)(4x_1-3x_2+4)(3x_1-4x_2+5)^{-1}(-2x_1+x_2+3)^{-1} \\
\text{s.t.} \quad & x_1+x_2 \le 1.5, \\
& x_1-x_2 \le 0, \\
& 0 \le x_1 \le 1, \quad 0 \le x_2 \le 1.
\end{aligned} \tag{5.4}$$

Example 5.5 (see [11, 15]).

$$\begin{aligned}
\min \quad & (2x_1+x_2+1)^{1.5}(2x_1+x_2+1)^{2.1}(0.5x_1+2x_2+1)^{0.5} \\
\text{s.t.} \quad & (x_1+2x_2+1)^{1.2}(2x_1+2x_2+2)^{0.1} \le 18, \\
& (1.5x_1+2x_2+1)(2x_1+2x_2+1)^{0.5} \le 25, \\
& 1 \le x_1 \le 3, \; 1 \le x_2 \le 3.
\end{aligned} \tag{5.5}$$

Table 1: Computational results of test problems (2.2)–(4.9).

Example	Methods	Optimal solution	Optimal value	Iter	Time
1	[12]	(1.0, 1.0)	997.661265160	49	0
	[15]	(1.0, 1.0)	997.6613	5	0.0984
	ours	(1.0, 1.0)	997.6613	1	0.0160
2	[15]	(1.0, 2.0, 1.0)	3.7127	10	0.2717
	ours	(1.0, 2.0, 1.0)	3.7127	1	0.0150
3	[12]	(1.0, 1.0, 1.0)	60.0	64	0
	[15]	(1.0, 1.0, 1.0)	60.0	1	0.0126
	ours	(1.0, 1.0, 1.0)	60.0	1	0.0148
4	[13]	(0.0, 0.0)	0.533333333	3	0
	[16]	(0.0, 0.0)	0.533333	16	0.05
	ours	(0.0, 0.0)	0.5333	2	0.0221
5	[11]	(1.0, 1.0)	275.074284	1	0
	[15]	(1.0, 1.0)	275.0743	1	0.0105
	ours	(1.0, 1.0)	275.0743	1	0.0102

The results of problems (2.2)–(4.9) are summarized in Table 1, where the following notations have been used in row headers: Iter: number of algorithm iterations; Time: execution time in seconds.

The results in Table 1 show that our algorithm is both feasible and efficient.

Acknowledgments

The authors are grateful to the responsible editor and the anonymous referees for their valuable comments and suggestions, which have greatly improved the earlier version of this paper. This paper is supported by the National Natural Science Foundation of China (60974082) and the Fundamental Research Funds for the Central Universities (K50510700004).

References

[1] M. C. Dorneich and N. V. Sahinidis, "Global optimization algorithms for chip design and compaction," *Engineering Optimization*, vol. 25, pp. 131–154, 1995.

[2] K. P. Bennett, "Global tree optimization: a non-greedy decision tree algorithm," *Computing Sciences and Statistics*, vol. 26, pp. 156–160, 1994.

[3] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objective*, Cambridge University, Cambridge, Mass, USA, 1993.

[4] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios, "Robust optimization of large-scale systems," *Operations Research*, vol. 43, no. 2, pp. 264–281, 1995.

[5] R. Horst and H. Tuy, *Global Optimization: Deterministic Approache*, Springer, Berlin, Germany, 2nd edition, 1993.

[6] Y. Gao, C. Xu, and Y. Yang, "An outcome-space finite algorithm for solving linear multiplicative programming," *Applied Mathematics and Computation*, vol. 179, no. 2, pp. 494–505, 2006.

[7] H. P. Benson, "Decomposition branch-and-bound based algorithm for linear programs with additional multiplicative constraints," *Journal of Optimization Theory and Applications*, vol. 126, no. 1, pp. 41–61, 2005.

[8] H. S. Ryoo and N. V. Sahinidis, "Global optimization of multiplicative programs," *Journal of Global Optimization*, vol. 26, no. 4, pp. 387–418, 2003.

[9] T. Kuno, “A finite branch-and-bound algorithm for linear multiplicative programming,” *Computational Optimization and Applications*, vol. 20, no. 2, pp. 119–135, 2001.
[10] H. P. Benson and G. M. Boger, “Outcome-space cutting-plane algorithm for linear multiplicative programming,” *Journal of Optimization Theory and Applications*, vol. 104, no. 2, pp. 301–322, 2000.
[11] P. Shen and H. Jiao, “Linearization method for a class of multiplicative programming with exponent,” *Applied Mathematics and Computation*, vol. 183, no. 1, pp. 328–336, 2006.
[12] H. Jiao, “A branch and bound algorithm for globally solving a class of nonconvex programming problems,” *Nonlinear Analysis. Theory, Methods & Applications*, vol. 70, no. 2, pp. 1113–1123, 2009.
[13] P. Shen, X. Bai, and W. Li, “A new accelerating method for globally solving a class of nonconvex programming problems,” *Nonlinear Analysis. Theory, Methods & Applications*, vol. 71, no. 7-8, pp. 2866–2876, 2009.
[14] X. G. Zhou and K. Wu, “A method of acceleration for a class of multiplicative programming problems with exponent,” *Journal of Computational and Applied Mathematics*, vol. 223, no. 2, pp. 975–982, 2009.
[15] C. F. Wang and S. Y. Liu, “A new linearization method for generalized linear multiplicative programming,” *Computers & Operations Research*, vol. 38, no. 7, pp. 1008–1013, 2011.
[16] N. V. Thoai, “A global optimization approach for solving the convex multiplicative programming problem,” *Journal of Global Optimization*, vol. 1, no. 4, pp. 341–357, 1991.