Hybrid Biogeography-based Optimization Algorithm for Solving Nonlinear Bilevel Programming

Yuejiao Wang*, Sanyang Liu & Nan Lu
School of Mathematics and Statistics, Xidian University, Xi’an, Shaanxi, China

ABSTRACT: The article deals with a class of nonlinear bilevel programming problems in which the follower’s objective is a quasi-concave function, and a new hybrid biogeography-based optimization algorithm is proposed to search for the value of the leader’s variables. First, an extreme point searching approach is constructed to obtain the optimal solution of the follower’s programming. An efficient mutation operator is then designed by using Zoutendijk feasible direction method so that it can generate high quality potential offspring. Furthermore, a new fitness function is established that can be easily used to force the individuals moving toward the feasible region and improve the feasible solutions gradually. Under some assumptions, theoretical analysis of the algorithm has been presented. Finally, to verify the effectiveness of the algorithm, numerical experiments on 8 test problems are made and performance analysis verify the proposed algorithm can converge to the global optimal solution of bilevel programming problems.

Keywords: bilevel programming; quasi-concave function; biogeography-based optimization; extreme point searching approach; mutation operator

*Corresponding author: xd07101051@126.com

1 INTRODUCTION

Consider the following nonlinear bilevel programming problem (BLPP) [1]:

\[
\begin{aligned}
&\min_{x,y} F(x,y) \\
&\text{s.t. } G(x,y) \leq 0, \text{ where } y \text{ needs to be solved} \\
&\min_{y} f(x,y) \\
&\text{s.t. } g(x,y) \leq 0,
\end{aligned}
\]

where \( x \in X \subset \mathbb{R}^n \), \( y \in Y \subset \mathbb{R}^m \) are leader’s and follower’s decision variable, respectively. Set \( X \) represents the search spaces in upper levels and Set \( Y \) represents the ones in lower levels. \( F, f : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n \) are called respectively the leader’s and follower’s objective function. \( G : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n \) and \( g : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^m \) are the leader’s and follower’s constraint function. \( F(x, y) \) and \( G(x, y) \) are non-differentiable and non-convex. And \( f(x, y) \) is continuous strictly quasi-concave in \( y \) for \( x \) fixed.

At present, the methods [2] for nonlinear bilevel programming generally can be classified as pole searching algorithm, branch and bound methods, descent iteration methods, and penalty function. Although these algorithms based on gradient have a fast rate of convergence, they require the convexity and differentiability of the objective function. When the objective function is non-differentiable or complex (multimodal), these methods will tend to be trapped in local optimum. Therefore, the solution of nonlinear BLPP is very difficult to obtain because of its inherent non-convexity, NP-hard and non-differentiability. In particular, it is more difficult to obtain the global optimal solution.

Biogeography-based optimization (BBO) algorithm is a new kind of modern intelligent evolutionary algorithm which simulates the principle of nature evolution to solve the unconstrained optimization problems, which was proposed by American scholar Dan Simon [3] in 2008.

In BBO, there are \( ps \) habitats. A habitat consists of a suitable index vector (SIV) [3, 4] like
\[ x_i = (x_{i1}, x_{i2}, \ldots, x_{in}) \quad i = 1, 2, \ldots, p \in \mathbb{R}^n, \]
which represents a feasible solution to an optimization problem.

BBO has some advantages, such as global search ability, robustness, simple operation and fast convergence speed [5]. It has a wide attention by its unique search mechanism and good performance in sensor selection, power system optimization and community detection problem in dynamic networks [3,6,7]. However, it’s difficult to verify the point according to evolution meet lower optimal constraint and it will cost a heavy amount of calculation.

Motivated by different types of modified evolutionary computation techniques [8,9,10], in this paper a new efficient mutation operator of intelligent algorithm by using traditional method is designed so that it can generate high quality potential offspring [11,12] and a new hybrid biogeography-based optimization algorithm is proposed as a new type of adaptive algorithm computation techniques [8,9,10], in this paper a method for solving nonlinear bilevel Programming.

This article is organized as follows. In the next section, assumptions and lemmas are as follows:

2.1 Research situation of BLPPs

The vast majority of the existing researches on BLPPs is concentrated on the linear BLPPs and some special nonlinear BLPPs in which all the functions involved are convex and twice differentiable, especially the follower’s objective is strictly convex.

Few works are concentrated on BLPPs, involving non-convex follower’s functions. For example, the leader’s and follower’s objective are both concave functions [13, 14].

Two level objective functions are all continuous and strictly quasi concave. Literature [15] proves the global optimal solution can be obtained at boundaries of the feasible region under the linear constraints; When there are no upper constraints and the constraint region is polyhedron, literature [16] testifies the optimal solution to a quasi-concave programming can be obtained at one of extreme points of the feasible region, but this conclusion does not apply when there are upper constraints.

2.2 The extreme point searching approach to solve the lower problem

As for the bilevel programming model like (1), the assumptions and lemmas are as follows:

Definition 1: Let \( f(x) \) be continuous strictly quasi convex function at every point of a set \( \Omega \). If \( f(x) \not= f(y) \) for any \( x, y \in \Omega \), then \( f(\lambda x + (1-\lambda)y) < \max(f(x), f(y)), \quad \forall \lambda \in (0,1) \). If \( -f \) is (strictly) quasi convex function at every point of a set \( \Omega \), then \( f \) is (strictly) quasi concave function at every point of a set \( \Omega \).

Assumption 2: The feasible region \( \Sigma \) and follower’s feasible region \( R \) of (1) are non-empty bounded closed sets.

Lemma 3[17]: Suppose \( f(x) \) is a continuous and strictly quasi convex function at \( R = \{x | Ax = b, x \geq 0\} \), the optimal solution of \( \max f(x) \) can be obtained at one of extreme points of \( R \) if it exists.

To sum up, the follower’s optimal solution of (1) can be obtained at the boundary of follower’s feasible region \( R \) if it exists, and it is one of extreme points at least. By Lemma 3, we can obtain the follower’s optimal solution by inspecting function value of the follower’s extreme points.

Now we give the extreme point searching approach.

For the minimization model, the follower’s problem is as follows

\[
\begin{align*}
\min_{y \in \mathcal{Y}} f(x, y) \\
\text{s.t.} \quad A(x)y - b(x) \geq 0, \quad y \geq 0
\end{align*}
\]

(2)

where \( f(x, y) \) is continuous strictly quasi concave in \( y \) for \( x \) fixed. We assume that the optimal solution of the follower’s problem is unique.

Feasible region of the follower’s problem is defined as \( R = \{y | Ay \geq b, y \geq 0\} \), where \( A = (a_1, a_2, \ldots, a_m)^{\top} \), \( a_i = (a_{i1}, a_{i2}, \ldots, a_{in})^{\top}, \quad i = 1, 2, \ldots, m \), \( y = (y_1, y_2, \ldots, y_n)^{\top} \), \( b = (b_1, b_2, \ldots, b_n)^{\top} \), the follower’s problem can be transformed into an equivalent problem:

\[
\begin{align*}
\min_{y \in \mathcal{Y}} f(x, y) \\
a_i y \geq b_i, i = 1 \cdots m \\
y_j, \quad j = 1 \cdots n
\end{align*}
\]

(3)

Suppose that \( R \) is bounded. Then \( R = \{y | a_i y \geq b_i, i = 1 \cdots m; y \geq 0, j = 1 \cdots n\} \subseteq \{y | e^{\top} y = 1\} \)

The extreme point searching algorithm [18,19] for solving (3) is constructed as follows:

**Algorithm 1: The extreme point searching algorithm**

Consider \( \Lambda = \{y | e^{\top} y \geq \alpha, e^{\top} y = 1, y \geq 0\} \) where \( c = (c_1, c_2, \ldots, c_s), \alpha \in E^{\top}, e = (1, 1, \ldots, 1)^{\top} \in E^n \)
If \((c_i - a)(c_i - a) < 0, c_i \neq c_j, 1 \leq k < l \leq n\),
then \(d = \left(0, \ldots, 0, \frac{(c_i - a)}{(c_i - c_j)}, 0, \ldots, 0, -\frac{(c_i - a)}{(c_i - c_j)}, 0, \ldots, 0\right)^\top\)
is an extreme point of \(\Lambda\).
If \((c_i - a) \geq 0, 1 \leq k \leq n\), then
\(d = (0, \ldots, 0, 1, 0, \ldots, 0)^\top\) is an extreme point of \(\Lambda\).

Assume that the entire extreme points of \(\Lambda\) are denoted by \(d^1, d^2, \ldots, d^m\), then \(\Lambda = \{D\alpha | \alpha \geq 0, e^\top \alpha = 1, \alpha \in E\}\)
where
\(D = (d^1, d^2, \ldots, d^m)^\top\).

First we calculate the extreme points of \(R^\ell = \Lambda^\ell = \{y | a_i y \geq b_i, e^\top y = 1, y \geq 0\}\) which contains a structural constraint referring to the above-mentioned rules, then calculate the new extreme point set to obtain all extreme points of the feasible region by adding constraints, finally we obtain the optimal solution by comparing function values of extreme points[18,19].

3 A NEW MUTATION OPERATOR BASED ON ZOUTENDIJK FEASIBLE DIRECTION METHOD

A hybrid BBO algorithm is proposed by combining BBO with extreme point searching approach which utilizes the extreme points of the feasible region to seek the follower’s optimal solution. Each initial point we obtain from the hybrid BBO algorithm satisfies the follower’s programming and the process costs a small amount of calculation.

We design a new efficient mutation operator of BBO algorithm by using Zoutendijk feasible direction method so that it can generate high quality potential offspring. Then the algorithm can be used to handle nonlinear BLPPs with non-differentiable leader’s objective functions and quasi concave follower’s objective functions.

3.1 A new mutation operator based on Zoutendijk feasible direction method

**Algorithm 2: A mutation operator designed by Zoutendijk feasible direction method**

**Step1:** For each habitat, update the probability \(P(s_i)\) that the habitat \(i\) contains exactly \(s_i\) species.

Then calculate \(m(s_i)\) (referring to **Algorithm 4**). For \(k\) elite habitats, let mutation rate \(m(s_i) = 0\). Then mutate each non-elite habitat based on \(m(s_i)\).

Step2: Suppose \((x_{i1}^{(k)}, y_{i1}^{(k)})\) is an offspring of migration from the current population \(POP(k)\), and then do mutation operation towards a favorable direction by \(m(s_i)\) to get a new individual \((p_{i1}^{(k)}, q_{i1}^{(k)})\).

Denote
\[
(p_{i1}^{(k)}, q_{i1}^{(k)})^\top = (x_{i1}^{(k)}, y_{i1}^{(k)})^\top + \lambda d(x_{i1}^{(k)}, y_{i1}^{(k)})^\top,
\]
where \(d(x_{i1}^{(k)}, y_{i1}^{(k)})^\top\) is feasible descent direction at \((x_{i1}^{(k)}, y_{i1}^{(k)})^\top\), \(\lambda\) is search step length.

**Step3:** For the minimization fitness functions model

\[
\begin{align*}
\min Cost(x_{i1}^{(k)}, y_{i1}^{(k)}) \\
A(x_{i1}^{(k)}, y_{i1}^{(k)}) &\geq b \\
E(x_{i1}^{(k)}, y_{i1}^{(k)}) &= e
\end{align*}
\]

we assume \(A = \left(\begin{array}{c} A_1 \\ A_2 \end{array}\right), b = \left(\begin{array}{c} b_1 \\ b_2 \end{array}\right), \) where \(A\) is coefficient matrix of all the active constraints, \(b\) is the corresponding constant vector.

**Step4:** If \(A(x_{i1}^{(k)}, y_{i1}^{(k)}) = b_1, A'(x_{i1}^{(k)}, y_{i1}^{(k)}) > b_2\) then solve the following optimization problem to obtain \(d(x_{i1}^{(k)}, y_{i1}^{(k)})\) by performing Penalty function method (Algorithm 3):

\[
\begin{align*}
\min V^\top \left(Cost(x_{i1}^{(k)}, y_{i1}^{(k)})\right) d(x_{i1}^{(k)}, y_{i1}^{(k)})^\top \\
\text{s.t. } & \quad A d(x_{i1}^{(k)}, y_{i1}^{(k)})^\top \geq 0 \\
& \quad E d = 0
\end{align*}
\]

where \(E \in R^n\) is identity matrix. The solutions can be divided into the following two states:

If \(V^\top \left(Cost(x_{i1}^{(k)}, y_{i1}^{(k)})\right) d(x_{i1}^{(k)}, y_{i1}^{(k)})^\top < 0\), then \(d(x_{i1}^{(k)}, y_{i1}^{(k)})^\top\) is feasible descent direction at \((x_{i1}^{(k)}, y_{i1}^{(k)})^\top\). For simplicity, successive verify if \(Cost(p_{i1}^{(k)}, q_{i1}^{(k)}) < Cost(x_{i1}^{(k)}, y_{i1}^{(k)})\) at a certain fixed value \(\lambda_i > 0\). If yes, \((p_{i1}^{(k)}, q_{i1}^{(k)})\) becomes a new individual after mutation, if not, let \(\lambda = \frac{\lambda_i}{2}\), continue to verify to obtain “<” until a limited num-
ber of times, if not still, let \( \left( \rho_i^{(t)}, q_i^{(t)} \right) = \left( x_i^{(t)}, y_i^{(t)} \right) \).

If \( \nabla^T \left( \text{Cost} \left( x_i^{(t)}, y_i^{(t)} \right) \right) d \left( x_i^{(t)}, y_i^{(t)} \right)^2 = 0 \),
then \( \left( x_i^{(t)}, y_i^{(t)} \right)^2 \) is \( K-T \) point of (4).

Step5: If \( A \left( x_i^{(t)}, y_i^{(t)} \right) > b \), individual \( \left( x_i^{(t)}, y_i^{(t)} \right) \) of mutation is in the interior of the feasible region. Hence, it is not necessary to solve (5), denote \( d \left( x_i^{(t)}, y_i^{(t)} \right)^2 = -\nabla \left( \text{Cost} \left( x_i^{(t)}, y_i^{(t)} \right) \right) \)
directly.

The individual \( \left( x_i^{(t)}, y_i^{(t)} \right) \) will evolve along the negative gradient direction \( -\nabla \left( \text{Cost} \left( x_i^{(t)}, y_i^{(t)} \right) \right) \)
when it is in the interior of the feasible region. When it is on the boundary of some constraints, we will project the gradient \( \nabla \left( \text{Cost} \left( x_i^{(t)}, y_i^{(t)} \right) \right) \) into space which
is decided by coefficient matrix \( A \) of all the active constraints. Not only can individual evolve along projection which is a feasible descent direction, but also will it be contained within the feasible region.

The purpose of designing like this is to force the individual of mutation moving toward the feasible region by using feasible descent directions, and which will improve the individual gradually and guarantee the effectiveness and simplicity of operation.

3.2 Penalty function method

Nelder-Mead simplex method modified by Nelder and Mead in 1965 [20] is a kind of common direct search method for multidimensional unconstrained problems. Although Nelder-Mead method has the slower convergence speed, it is demonstrated effective and feasible as time goes on. The basic thought is as follows:

Nelder-Mead simplex method is used to solve a series of unconstrained optimizations in the penalty function. We propose a penalty function method based on Nelder-Mead to solve (5).

Algorithm 3: A penalty function method to solve (5).

Step1: The parameters are chosen as follows: Initial point \( d^{(i)} \in \mathbb{R}^m \), \( m \) is the dimension of variable \( d \left( x_i^{(t)}, y_i^{(t)} \right)^2 \), initial penalty factor \( M_0 = 100 \), convergence precision \( \varepsilon = 1e^{-6} \), penalty factor coefficient \( c = 10 \) and the largest number of iterations \( k = 10000 \). Let \( k = 1 \).

Step2: For initial point \( d^{(i)} \), we use Nelder-Mead simplex method to solve the follower’s unconstrained optimization:

\[
\min F \left( x, y, M_k \right) = C \times f \left( x, y \right) + M_k \left( \sum \left( \max \left\{ g_i \left( x, y \right), 0 \right\} \right)^2 + \sum h_j \left( x, y \right) \right)
\]

where \( g_i \left( x, y \right) \) is the inequality constraint of (4), \( h_j \left( x, y \right) \) is the equality constraint of (4), we assume its optimal solution \( d^{(i)} = d \left( M_k \right) \).

Step3: The algorithm stops after \( M_k \left( \sum \left( \max \left\{ g_i \left( x, y \right), 0 \right\} \right)^2 + \sum h_j \left( x, y \right) \right) < \varepsilon \) or 10000 generations, then exports \( d^{(i)} \); If not, let \( M_{k+1} = c M_k, k = k + 1 \), go to step 2 for the next iteration.

3.3 Hybrid BBO algorithm design

Before we propose the hybrid BBO algorithm, we give some definitions about the habitats initializing, fitness functions and the mapping functions from fitness functions to the species count.

1) Habitats initializing

Set \( X \) represents the search spaces in upper and lower levels of variable \( x \), respectively, which are given by

\[
X = \left\{ x = \left( x_1, x_2, \ldots, x_n \right) | l_i \leq x_i \leq u_i, i = 1, 2, \ldots, n \right\}
\]

We initialize the habitats \( X \left( 0 \right) \) \( i = 1, 2, \ldots, ps \) with \( X \left( 0 \right) \cdot U \left( l_j, u_j \right), j = 1, 2, \ldots, n \), where \( U \left( l_j, u_j \right) \) is uniform distribution from \( l_j \) to \( u_j \). Finally it generates the initial population \( POP = \left\{ pop_{ps} \right\}_{ps = 1} \), where \( pop_{ps} \) is element in \( \left[ l_j, u_j \right] \).

2) Fitness function

The decision problem discussed in the article is a minimization problem. Fitness functions satisfy the following conditions: the smaller fitness function value, the more good its solution. Fitness function called Cost is defined as:

\[
\text{Cost} \left( x, y \right) = F \left( x, y \right) + M \left( \sum \left( \max \left\{ g_i \left( x, y \right), 0 \right\} \right)^2 + \sum h_j \left( x, y \right) \right)
\]

where \( F \left( x, y \right) \) represents the leader’s objective function of (1), \( g_i \left( x, y \right), k = 1, 2, \ldots, m \) is the lead-
er’s inequality constraint, \( h_j(x, y), j = 1, 2, \cdots, p \) is the leader’s equality constraint.

The definition of the fitness functions guarantees that any feasible solution dominates all infeasible solutions of the problem. For two feasible solutions, one with the smaller function value \( f(x, y) \) dominates the other. Thus, it forces the algorithm to move toward the potential solutions and improves the feasible solutions gradually.

3) Map fitness functions to species count

Firstly all the habitats are sorted in ascending order, and then we establish a mapping function from fitness functions to species count \([21]\). Suppose \( P = ps \).

Denote the species count of the habitats by

\[
P_i = \min_{x \in S} f(x, y), \quad i = 1, 2, \cdots, ps
\]

(9)

where \( s_i \) has been sorted in ascending order.

3.4 A new hybrid BBO algorithm for nonlinear BLPP

Algorithm 4: Biogeography-based Optimization Based on Zoutendijk Feasible Direction method, \( \text{BBOZFD} \)

01 Initialization: Preset habitat amount \( ps \), dimensions of \( x \) and \( y \) are \( n_x \) and \( n_y \), respectively, maximum species count \( S_{max} \), maximum mutation rate \( m_{max} \) and maximum generations. Generate

\[
x = (x_1, x_2, \cdots, x_n), \quad i = 1, 2, \cdots, ps
\]

in accordance with the section 3.3.1.

02 For each \( x_i \), perform Algorithm 1 to obtain the follower’s corresponding optimal solution

\[
y = (y_1, y_2, \cdots, y_n), \quad j = 1, 2, \cdots, ps
\]

which constitutes the initial population

\[
\text{POP}(0) = \{ pop_j(0) \} \quad \text{ps}, \quad n = n_x + n_y
\]

(0)\( \text{POP}(0) \) is fed back to upper level to find out and retain the best solution \( \text{argmin} \{ F(x, y) \} \). Let \( k = 0 \).

03 For \( (x_i, y_j) = (x_1, x_2, \cdots, x_n, y_1, y_2, \cdots, y_n) \), calculate fitness functions \( \text{Cost}(x_i, y_j) \) according to (8) in ascending order, \( i = 1, 2, \cdots, ps \).

04 Map \( \text{Cost}(x, y) \) to species count \( S_i \) according to (9), calculate immigration rate \( \lambda(s_i) \), emigration rate \( \mu(s_i) \) and the probability

\[
P(s_i) \quad \text{that the habitat } i \text{ contains exactly } s_i \text{ species } i = 1, 2, \cdots, ps
\]

05 Migration:

For \( i = 1, 2, \cdots, ps \)

Select \( (x_i, y_j) \) with probability \( P_{rand} \)

For \( j = 1 \cdots n \)

If \( r_j(i, j) < \lambda \), \( \text{Index} = 1 \)

\[
\lambda = \frac{(\lambda_{max} - \lambda_{min})}{(\lambda_{max} - \lambda_{min})}, \quad r_j = \text{rand}(ps, n)
\]

While \( r_j(i, j) > \text{Index} \) \& \& \( \text{Index} < ps \)

\[
r_j = r_j * (\mu_i + \mu_j + \cdots + \mu_{ps})
\]

\[
m = (\mu_1 + \mu_2 + \mu_3 + \mu_4 + \cdots + \mu_{ps})
\]

\[
\text{Index} = \text{Index} + 1
\]

End

\[
(x, y)_j = (x, y)_{\text{Index}, j}
\]

End

End

End

06 Mutation:

For \( i = ps/2 \) to \( ps \)

Calculate \( m = m_{max} \left( \frac{1 - P(s_i)}{P_{max}} \right) \)

Perform Algorithm 2 to get a new individual

\[
(p_i^{(s)}, q_i^{(s)})
\]

End

End

07 Go to step 02 for the next iteration. Let \( k = k + 1 \).

This loop can be terminated after a predefined number of generations, or after an acceptable problem solution.

4 THEORETICAL ANALYSIS OF ALGORITHM 2

As for the following general minimization fitness functions model equivalent to (4),

\[
\begin{align*}
\min & \text{ Cost}(x, y) \\
A(x, y) & \geq b \\
E(x, y) & = e
\end{align*}
\]

We assume \( X \in S \) ( \( S \) represents the feasible region), \( d \) is feasible direction at \( X \), then \( \exists \delta > 0 \), \( \forall \lambda \in (0, \delta) \) such that \( X + \lambda d \in S \). Thus,

\[
\begin{align*}
A(X) & \geq b \\
E(X) & = e
\end{align*}
\]
For two systems like \(C\mathcal{O}S\) solution. Assume \(\mathcal{O}S\) is the corresponding constant vector, then we have
\[
\begin{align*}
A\overline{x} = b_i \quad &\quad A_i(\overline{x} + \lambda d) \geq b_i \\
A\overline{x} \geq b_2 \quad &\quad A_i(\overline{x} + \lambda d) \geq b_2, \text{ and hence de-} \\
E\overline{x} = e \quad &\quad E(\overline{x} + \lambda d) = e \\
\end{align*}
\]
\begin{align*}
A_i d \geq 0 \\
Ed = 0.
\end{align*}

Then, we solve the following general minimization problem equivalent to (5):
\[
\begin{align*}
\min & \quad \nabla^T(C\mathcal{O}(x, y)) d \\
\text{s.t.} & \quad A d \geq 0 \\
& \quad Ed = 0
\end{align*}
\]

(11) exists the optimal solution \(d\) when
\[
\nabla^T(C\mathcal{O}(x, y)) d < 0 , \text{ which implies } \\
\quad A d \geq 0 \\

\quad Ed = 0.
\]

5 NUMERICAL EXPERIMENTS

To verify the feasibility and effectiveness of the proposed BBOZFD algorithm, eight different types of numerical examples are made, and performance comparison verified BBOZFD algorithm can converge to the global optimal solution of bilevel programming problems.

The follower’s objective functions of examples 1 to 8[25-29] are continuous and strictly quasi concave, while the follower’s constraints are all linear.

The parameters are chosen as follows: habitat amount \(ps = 50\), maximum mutation rate \(m = 0.005\), convergence precision \(e = 1.0e - 006\) and an elitism parameter \(k = 2\). The algorithm generates 10000 loops.

The algorithm was tested on each of the above 8 functions for numerical experiments. All experiments were performed at a PC with CPU of 3.20 GHz and RAM of 2.00 GB, and all codes were finished in MATLAB. We executed the proposed algorithm in 50 independent runs and recorded the following data:

- best solution \((x^*, y^*)\) and worst solution \((x, y)\) found in 50 independent runs;
- leader’s objective value \(F(x, y^*)\) at the best solution \((x, y^*)\) and \(F(x^*, y)\) at the worst solution \((x^*, y)\);
- follower’s objective value \(f(x, y^*)\) at the best solution \((x, y^*)\) and \(f(x^*, y)\) at the worst solution \((x^*, y)\);
- mean values of CPU time

We compare the obtained results with those presented in the corresponding references (REF). For examples 1 to 8, all of the results are presented in Table 1 and Table 2.

From Table 1, we can see that for examples 1, 2, 3, 4, 5 and 8, the solutions found by BBOZFD are better than or equal to those by the compared algorithms in the references.

For example 7, the solutions found by BBOZFD are much better than those by the compared algorithms.
The results also show that the solution $(x^*, y^*) = (0.0, 0.9, 0.6, 0.4)'$ found by the algorithms in [14] for example 7 is not global optimal solution. The optimal value of the leader’s and follower’s objective function should be $F(x^*, y^*) = -30.0152 , f(x^*, y^*) = 3.198$.

Although the solutions found by BBOZFD for example 6 are a little bit worse than those by the compared algorithms, the optimal solution isn’t in the follower’s feasible region. The defects may be caused by the algorithm itself.

Table 1 lists mean values of CPU time (denoted by CPU in short). It can be seen that the proposed algorithm can find global or near global optimal solutions for all test functions in a short time by using a relatively small number of individuals. Therefore, the proposed algorithm is efficient and effective.

For test examples 1, 4 and 7, it can be seen from Table 2 that the worst solutions found by BBOZFD in 50 independent runs are all high-quality approximate global solutions and even better than the best solutions obtained by the compared algorithms in Table 1.

From the above discussions, one can see that our algorithm is effective, is stable, and performs better than the compared algorithms.

6 CONCLUSIONS

In this article, a hybrid BBO algorithm is proposed to solve nonlinear BLPPs under the assumption that the follower’s optimal solution is unique. First of all, the extreme point searching approach is combined with BBO to make each initial point we obtain from the hybrid algorithm satisfy the follower’s programming and the process costs a small amount of calculation. Moreover, Zoutendijk feasible direction method is introduced to the mutation operator of bio-inspired algorithm to generate high quality potential offspring. Furthermore, global convergence is proved and numerical experiments demonstrate that the proposed algorithm has a better performance than the compared algorithms. A distinguishing feature of the article is that the proposed algorithm can be used to handle nonlinear BLPPs with non-differentiable leader’s objective functions and quasi concave follower’s objective functions.

ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China (Grants No.61373174).

REFERENCES


