An Evolutionary Algorithm for Solving Nonlinear Bilevel Programming Based on a New Constraint-Handling Scheme

Yuping Wang, Yong-Chang Jiao, and Hong Li

***Abstract*—In this paper, a special nonlinear bilevel programming problem (nonlinear BLPP) is transformed into an equivalent single objective nonlinear programming problem. To solve the equivalent problem effectively, we first construct a specific optimization problem with two objectives. By solving the specific problem, we can decrease the leader's objective value, identify the quality of any feasible solution from infeasible solutions and the quality of two feasible solutions for the equivalent single objective optimization problem, force the infeasible solutions moving toward the feasible region, and improve the feasible solutions gradually. We then propose a new constraint-handling scheme and a specific-design crossover operator. The new constraint-handling scheme can make the individuals satisfy all linear constraints exactly and the nonlinear constraints approximately. The crossover operator can generate high quality potential offspring. Based on the constraint-handling scheme and the crossover operator, we propose a new evolutionary algorithm and prove its global convergence. A distinguishing feature of the algorithm is that it can be used to handle nonlinear BLPPs with nondifferentiable leader's objective functions. Finally, simulations on 31 benchmark problems, 12 of which have nondifferentiable leader's objective functions, are made and the results demonstrate the effectiveness of the proposed algorithm.**

***Index Terms*—Constraint handling, evolutionary algorithm (EA), global optimization, nonlinear bilevel programming.**

I. Introduction

We consider the following nonlinear bilevel programming problem (BLPP):

$$\begin{cases} \min_{x\in X}\ F(x,y) \\ \text{s.t. } G(x,y)\le 0 \\ \min_{y\in Y}\ f(x,y) \\ \text{s.t. } g(x,y)\le 0 \end{cases} \tag{1}$$

where $F, f : R^n \times R^m \to R^1, G : R^n \times R^m \to R^p, g : R^n \times R^m \to R^q, F(x,y)$, and $G(x,y)$ are nondifferentiable and nonconvex, and $f(x,y)$ and $g(x,y)$ are differentiable and convex in y for x fixed. Sets X and Y represent the search spaces in upper and lower levels, respectively, which are given by

$$X = \{(x_1,\ldots,x_n)^\top \in R^n \,|\, \mu_i^l \le x_i \le \mu_i^u,\quad i=1,\ldots,n\}$$

and

$$Y = \{(y_1,\ldots,y_m)^\top \in R^m \,|\, \xi_j^l \le y_j \le \xi_j^u,\quad j=1,\ldots,m\}$$

where $\mu_i^l, \mu_i^u, \xi_j^l$, and ξ_j^u $(i = 1,2,\ldots,n, j = 1,2,\ldots,m)$ are all real constants. $F(x,y)$ is called the upper-level objective function or leader's objective function, and $f(x,y)$ is called the lower-level objective function or follower's objective function.

The bilevel programming problem (BLPP) can be viewed as a static version of the noncooperative, two-person game introduced by Von Stackelberg [9] in the context of unbalanced economic markets. In this model, the control of the decision variables is partitioned among the two players: the leader and the follower. Each player seeks to optimize his (or her) objective function. The leader goes first by choosing a vector $x \in R^n$ in an attempt to optimize his (or her) objective function $F(x,y)$. In doing so, he (or she) must anticipate all possible responses of the follower. The follower observes a leader's decision and reacts by selecting a vector $y \in R^m$ that optimizes his (or her) objective function $f(x,y)$. Because the set of feasible choices available to either player is interdependent, the leader's decision affects both the follower's objective and decision space and vice-versa.

BLPP has a wide variety of applications, such as network design [10], transport system planning [11], and management and economics [12]. Examples include the two-level design/planning problem, general resource allocation problems for hierarchical decentralized systems, min-max problems, as well as satisfaction optimization problems. However, even for the linear BLPP, the simplest one among the BLPP family, where all of the functions are linear, [1] and [2] have shown that it is strongly Nondeterministic Polynomial-time hard (NP-hard). Therefore, as a strong NP-hard problem, the nonlinear BLPP is a more complex and challenging problem.

Over the decades, the vast majority of research on BLPP is concentrated on the linear version of the problem. For nonlinear BLPP, most of the existing algorithms can yield a local minimum only [3]. For some more specific nonlinear BLPPs, only a few procedures (e.g., [3]–[6]) have been proposed to find a global minimum. For example, when all of the functions are convex and twice differentiable, and the lower-level objective $f(x,y)$ is strictly convex with respect to y, Amouzegar [4] proposed an algorithm for finding a global minimum of the

Manuscript received September 1, 2003; revised February 13, 2004. This work was supported in part by the National Natural Science Foundations of China under Grant 60374063 and Grant 60171045, in part by grants of the Scientific Research Foundation (SRF) for the Returned Overseas Chinese Scholars (ROCS), State Education Ministry (SEM), the Natural Science Foundation of Shaanxi Province, China (2001SL06), and the Excellent Young Teachers Program, Ministry of Education, China. This paper was recommended by Guest Editor Y. Jin.

Y. Wang and H. Li are with the Faculty of Science, Xidian University, Xi'an 710071, China (e-mail: ywang@xidian.edu.cn).

Y.-C. Jiao is with the Institute of Antennas and Electromagnetic Scattering, Xidian University, Xi'an 710071, China (e-mail: ychjiao@xidian.edu.cn).

Digital Object Identifier 10.1109/TSMCC.2004.841908

nonlinear BLPP, based on the branch-and-bound method. Few works, to our knowledge, are concentrated on the nonlinear BLPP with nondifferentiable nonconvex leader's objective and constraint functions.

Evolutionary algorithms (EAs) have been successfully used for some complex optimization problems. We will propose a new EA for the complex nonlinear BLPP.

In this paper, we first transform the nonlinear BLPP into an equivalent nonlinear optimization problem with a single nondifferentiable and nonconvex objective function [3]. In order to solve the equivalent problem effectively, we construct a specific optimization problem with two objectives. By solving the specific problem, we can decrease the objective function, identify the quality of any feasible solution from infeasible solutions and the quality of two feasible solutions for the equivalent problem, force the infeasible solutions moving toward the feasible region, and improve the feasible solutions gradually. An EA with specifically designed crossover and mutation operators is then proposed. Moreover, a new constraint-handling method with linear and nonlinear schemes is presented and cooperated with the EA. The linear constraint-handling scheme can make the individual satisfy all linear constraints, and the nonlinear scheme can make the individual satisfy the nonlinear constraints approximately, while maintaining the satisfaction of the linear constraints. Once a potential offspring is generated, the constraint-handling scheme is used to improve it further. As a result, the potential solutions become better and better. Finally, the new algorithm is tested on 31 benchmark problems. Our results show that the proposed algorithm is effective and can find better solutions than the compared algorithms.

The paper is organized as follows. In Section II, the nonlinear BLPP is transformed into an equivalent single objective optimization problem. In Section III, a new EA for nonlinear BLPP is proposed. In Section IV, the global convergence of the proposed algorithm is proved. Section V presents the simulation results and Section VI summarizes our conclusions.

II. Equivalent Problem With Single Objective

First, we briefly introduce some related definitions and notations. In BLPP (1), for x fixed, the problem

$$\begin{cases} \min_{y\in Y}\ f(x,y) \\ \text{s.t. } g(x,y)\le 0 \end{cases} \tag{2}$$

is called the lower-level problem or the follower's problem. For fixed y, the problem

$$\begin{cases} \min_{x\in X}\ F(x,y) \\ \text{s.t. } G(x,y)\le 0 \end{cases} \tag{3}$$

is called the upper-level problem or the leader's problem.

Definition 1: [3], [4]

a) Denote the constraint region of the BLPP by

$$S=\{(x,y)\,|\,x\in X, y\in Y, G(x,y)\le 0, g(x,y)\le 0\}.$$

b) Denote the feasible region of the follower's problem for each fixed $x\in X$ by

$$S(x)=\{y\in Y\,|\,g(x,y)\le 0\}.$$

c) Projection of S onto the leader's decision space is defined as

$$S(X)=\{x\in X\,|\,\exists y\in Y,(x,y)\in S\}.$$

d) The follower's rational reaction set for $x\in S(X)$ is defined as

$$P(x)=\{y\in Y\,|\,y\in \arg\min\{f(x,\hat{y})\,|\,\hat{y}\in S(x)\}\}.$$

e) Denote the inducible region by

$$\mathrm{IR}=\{(x,y)\,|\,(x,y)\in S, y\in P(x)\}.$$

In order to ensure that problem (1) is well posed, we also assume that S is nonempty and compact, that for all decisions taken by the leader, each follower has some room to respond (i.e., $S(x)\neq\emptyset$). The rational reaction set $P(x)$ represents these responses while the inducible region IR stands for the set over which the leader may optimize. Thus, in terms of the aforementioned notations, problem (1) can be written as

$$\min\{F(x,y):(x,y)\in \mathrm{IR}\}.$$

Since the lower-level problem is a differentiable convex programming for each fixed x, it is equivalent to Karush–Kuhn–Tucker stationary-point problem [3], [7], [8]

$$\begin{cases} \nabla_y f(x,y)+(\nabla_y g(x,y))^{\top}U=0 \\ U^{\top}g(x,y)=0 \\ g(x,y)\le 0 \\ U\ge 0 \end{cases} \tag{4}$$

where $(\nabla_y g(x,y))^{\top}$ is an $m\times q$ matrix whose ith column is $\nabla_y g_i(x,y)$ (i.e., the transpose of the Jacobian matrix of g with respect to y at (x,y), and U is a q dimensional column vector denoting the Lagrangian multipliers). Replacing the follower's problem by (4), we can rewrite the nonlinear BLPP (1) into the following equivalent single-level mathematical programming problem ([3], [7], [8]):

$$\begin{cases} \min_{x,y,U}\ F(x,y) & \\ \text{s.t. } E_i(x,y,U)\le 0, & i=1,\dots,u \\ h_k(x,y,U)=0, & k=1,\dots,m+1 \\ A_r x+B_r y\le b_r, & r=1,\dots,p+q-u \\ U\ge 0. & \end{cases} \tag{5}$$

where $E_i(x,y,U)$'s represent the nonlinear components of $G(x,y)$ and $g(x,y)$, $h_k(x,y,U)$'s the left-hand side functions of the first two equations in (4), and $A_r x+B_r y-b_r$'s are all of the linear components of $G(x,y)$ and $g(x,y)$.

III. New EA

To solve problem (5) efficiently, we construct a specific optimization problem with two objectives in this section. By solving

the specific problem, we can decrease the leader's objective function, identify the quality of any feasible solution from infeasible solutions and the quality of two feasible solutions for the equivalent single objective optimization problem, force the infeasible solutions moving toward the feasible region, and improve the feasible solutions gradually. A new crossover operator is then carefully designed based on the two-objective optimization problem. Moreover, a new constraint-handling method is presented to cooperate the genetic operators. Finally, an effective genetic algorithm is proposed.

A. Optimization Problem With Two Objectives

For convenience, set $\Theta = (x^\top, y^\top, U^\top)^\top, \Theta^* = (x^{*\top}, y^{*\top}, U^{*\top})^\top$, and $\Theta_i = (xi^\top, yi^\top, Ui^\top)^\top (i = 0,1,2)$. Denote $E(\Theta) = E(x, y, U)$, and

$$\begin{cases} E_{u+k}(\Theta) = |h_k(\Theta)| \\ E_{u+m+1+r}(\Theta) = A_r x + B_r y - b_r \end{cases} \tag{6}$$

where $k = 1, \ldots, m+1$ and $r = 1, \ldots, p+q-u$. Construct the following optimization problem with two objectives:

$$\min \{\tilde{F}(\Theta), E(\Theta)\} \tag{7}$$

where

$$\begin{aligned} E(\Theta) &= \max\{E_1(\Theta), \ldots, E_J(\Theta)\} \\ J &= p + q + m + 1 \\ \tilde{F}(\Theta) &= \begin{cases} F(x,y), & \text{if } E(\Theta) = 0 \\ F(\hat{x},\hat{y}) + E(\Theta), & \text{otherwise} \end{cases} \\ F(\hat{x},\hat{y}) &= \max\{F(x,y) \,|\, E(\Theta) = 0\} \end{aligned}$$

or an upperbound of $F(x,y)$ on set $\{(x,y) \,|\, E(\Theta) = 0\}$.

The definition of $\tilde{F}(\Theta)$ in problem 7 guarantees that any feasible solution of problem (5) is better than all of the infeasible solutions of the problem. For two feasible solutions of problem (5), one with smaller function value $F(x,y)$ is better than the other. Furthermore, solving 7 can force the population to gradually go forward to the feasible region of problem (5) [by optimizing the second objective function $E(\Theta)$], and can also improve the objective function $F(x,y)$ of problem (5) [by optimizing the first objective function $\tilde{F}(\Theta)$].

Definition 2: For problem 7, if two points Θ and Θ_1 satisfy the following conditions:

- $(\tilde{F}(\Theta), E(\Theta)) \leq (\tilde{F}(\Theta_1), E(\Theta_1))$.
- At least one of $\tilde{F}(\Theta) < \tilde{F}(\Theta_1)$ and $E(\Theta) < E(\Theta_1)$ holds.

Then Θ is said to dominate Θ_1.

Definition 3: Let Ω be a set of points. A point $\Theta \in \Omega$ is said to be nondominated regarding the set Ω if and only if there exists no point $\Theta_1 \in \Omega$ such that Θ_1 dominates Θ.

Definition 4: Let Ω denote the set of all points generated by an algorithm up to now, then the set of all nondominated points regarding Ω is called the current nondominated set.

Note that if Θ dominates Θ_1, then Θ is at least as good as Θ_1 with respect to both objectives in (7) (the first condition in Definition 2), and Θ is strictly better than Θ_1 with respect to at least one objective (the second condition in Definition 2).

Usually, a bilevel programming may not be equivalent to a corresponding two-objective programming (with the upper-level and the lower-level objectives as two objectives) (i.e., an optimal solution of the bilevel programming is not necessarily the Pareto optimal solution of the two-objective programming, and vice versa). However, two-objective programming (7) is equivalent to the bilevel programming (1) and single-level programming (5), respectively, as shown in the following theorem.

Theorem 1: Suppose that there exists an optimal solution for bilevel programming (1), then Θ^* is a Pareto optimal solution for two-objective programming (7) if and only if it is an optimal solution for single-level programming (5) and, thus, Θ^* is an optimal solution for bilevel programming (1).

Proof: Since there exists an optimal solution for problem (1), and problem (1) is equivalent to problem (5), [3], then $(x^{*\top}, y^{*\top})^\top$ is an optimal solution for problem (1) if and only if there exists U^* such that Θ^* is an optimal solution for problem (5), [3]. Therefore, we only need to prove that Θ^* is a Pareto optimal solution for problem (7) if and only if it is an optimal solution for problem (5).

Note that $E(\Theta) = 0$ if and only if Θ is a feasible solution of problem (5) [i.e., Θ satisfies constraints of problem (5)]. If Θ^* is an optimal solution for problem (5), then all constraints of problem (5) should be satisfied at Θ^* and $F(x^*, y^*) = \min\{F(x,y) \,|\, E(\Theta) = 0\}$. For any feasible solution Θ of problem (5), we have

$$\tilde{F}(\Theta^*) = F(x^*, y^*) \leq F(x,y) = \tilde{F}(\Theta).$$

For any infeasible solution Θ of problem (5) (i.e., Θ does not satisfy at least one constraint of problem (5)), it follows by definition of $\tilde{F}(\Theta)$ of problem (7) that

$$\tilde{F}(\Theta^*) \leq \tilde{F}(\Theta).$$

Thus, $\tilde{F}(\Theta^*)$ attains the minimum value at Θ^*. Note that $E(\Theta) \geq 0$ for any Θ, and $E(\Theta) = 0$ if and only if Θ is a feasible solution of problem (5). Thus $E(\Theta)$ also attains the minimum value at Θ^*. Therefore, Θ^* is a Pareto optimal solution of problem (7).

Now we prove the necessity. Suppose that Θ^* is a Pareto optimal solution of problem (7). Let Θ denote an optimal solution of problem (1). Since problem (1) is equivalent to problem (5), then there exists $\bar{U}$ such that $\bar{\Theta} = (\bar{x}^\top, \bar{y}^\top, \bar{U}^\top)^\top$ is an optimal solution of problem (5). Of cause, $\bar{\Theta}$ is a feasible solution of problem (5), and $E(\bar{\Theta}) = 0$. If $E(\Theta^*) > 0$, then Θ^* is an infeasible solution of problem (5). By definition of $\tilde{F}(\Theta)$, we have $\tilde{F}(\Theta^*) \geq \tilde{F}(\bar{\Theta})$. Note that $E(\Theta^*) > 0 = E(\bar{\Theta})$. This contradicts that Θ^* is a Pareto optimal solution of problem (7). Thus, $E(\Theta^*) = 0 = E(\bar{\Theta})$ (i.e., Θ^* is a feasible solution of problem (5)), and we have $\tilde{F}(\Theta^*) \geq \tilde{F}(\bar{\Theta})$. When $\tilde{F}(\Theta^*) > \tilde{F}(\bar{\Theta})$, Θ^* is not a Pareto optimal solution of problem (7). Thus

$$\tilde{F}(\Theta^*) = \tilde{F}(\bar{\Theta}).$$

Therefore, Θ^* is also an optimal solution of problem (5). This completes the proof. ■

The definition of problem (7) guarantees that any feasible solution of problem (5) dominates all infeasible solutions of the problem. For two feasible solutions, one with the smaller function value $F(x,y)$ dominates the other. Thus, two-objective optimization problem (7) can effectively improve the potential solutions.

B. Crossover Operator

A specific crossover operator is designed for problem (7). The operator forces $\tilde{F}(\Theta)$ to decrease and forces Θ to enter into the feasible region of problem (5), which can be done by the following. For a parent Θ, look for a point Θ_1 in the current nondominated set of problem (7) dominating Θ, then

$$D = \Theta_1 - \Theta \tag{8}$$

is generally a descent direction for both $\tilde{F}(\Theta)$ and $E(\Theta)$ at point Θ. Extensive simulation results in Section V support this assertion. A line search is executed in this direction to determine a solution

$$\Theta_2 = \Theta + \beta(\Theta_1 - \Theta) \tag{9}$$

that dominates Θ_1, where $\beta \in (0, \epsilon)$ and ϵ is a positive number determined by upper and lower bounds of variables x and y. Formula (9) defines a crossover operator for problem (7), which forces $\tilde{F}(\Theta)$ to decrease and forces Θ to enter into the feasible region of problem (5).

C. New Constraint-Handling Method

Constraint handling is an important issue for constrained problems. The well-known penalty methods/Lagrange multiplier methods cooperated with EA are mainly used to solve single-objective problems, in which the constraints are handled by integrating them into an objective function, but they are not suitable to a two-objective optimization problem (7).

In this subsection, a new constraint-handling method is proposed. Like crossover or mutation, the constraint-handling scheme is an independent part, which can be applied to the two-objective optimization problem (7).

For any infeasible point $\Theta_0 = (x0^\top, y0^\top, U0^\top)^\top$ of the problem, this method generates a point satisfying all of the linear constraints and makes all of the nonlinear constraints satisfy approximately. With this strategy, all of the constraints are handled efficiently. The method consists of two steps. The first step is to handle linear constraints, and the second step is to handle nonlinear constraints while maintaining the satisfaction of linear constraints.

1) Handling Linear Constraints: Denote $l_i(x,y) = A_i x + B_i y$. Then, $-\nabla l_i(x,y) = -[A_i, B_i]^\top$ is a constant column vector in R^{n+m}. Note that $-[A_i, B_i]^\top$ is the steepest descent direction of the function $l_i(x,y)$ at any point $(x^\top, y^\top)^\top$. For arbitrary point $(x0^\top, y0^\top)^\top$ not satisfying linear constraints, denote $b_i^0 = l_i(x0, y0)$ (i.e., $b_i^0 = A_i x0 + B_i y0, i = 1, \ldots, p+q-u$). We try to look for a descent direction d for all functions $l_i(x,y)$ for $i = 1, \ldots, p+q-u$ and to move point $(x0^\top, y0^\top)^\top$ to a point $(x^\top, y^\top)^\top = (x0^\top, y0^\top)^\top + d$ satisfying all linear constraints. In order to do so, we have to consider two cases. For convenience, denote $C_i = [A_i, B_i]$ for $i = 1, \ldots, p+q-u$.

Case 1: $C_1, C_2, \ldots, C_{p+q-u}$ are linearly independent row vectors. Denote $I = p+q-u$ and define

$$d = \sum_{j=1}^{I} \lambda_j(-\nabla l_j(x,y)) = -\sum_{j=1}^{I} \lambda_j C_j^\top \tag{10}$$

such that

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x0 \\ y0 \end{pmatrix} + d = \begin{pmatrix} x0 \\ y0 \end{pmatrix} + \begin{pmatrix} d_x \\ d_y \end{pmatrix} \tag{11}$$

satisfying all linear constraints

$$A_i(x0 + d_x) + B_i(y0 + d_y) \le b_i, \qquad i = 1, \ldots, I \tag{12}$$

where $d = (d_x^\top, d_y^\top)^\top, d_x \in R^n$ and $d_y \in R^m$ are column vectors. It follows from $b_i^0 = A_i x0 + B_i y0$ that (12) is equivalent to $A_i d_x + B_i d_y \le b_i - b_i^0, i = 1, \ldots, I$, or

$$-\sum_{j=1}^{I} \lambda_j C_i C_j^\top \le b_i - b_i^0, \quad i = 1, \ldots, I. \tag{13}$$

Denote $C = (C_1^\top, \ldots, C_I^\top)^\top, \Lambda = (\lambda_1, \ldots, \lambda_I)^\top, \delta = (b_1 - b_1^0, \ldots, b_I - b_I^0)^\top$, and

$$T = -\begin{bmatrix} C_1C_1^\top & C_1C_2^\top & \cdots & C_1C_I^\top \\ C_2C_1^\top & C_2C_2^\top & \cdots & C_2C_I^\top \\ \cdots & \cdots & \cdots & \cdots \\ C_IC_1^\top & C_IC_2^\top & \cdots & C_IC_I^\top \end{bmatrix} = -CC^\top$$

then the inequalities (13) can be written as

$$T\Lambda \le \delta. \tag{14}$$

For arbitrarily chosen $\delta^1 \in R^I$ satisfying $\delta^1 \le \delta$ (i.e., each element of δ^1 is less than or equal to the corresponding element of δ), then we can get a Λ by solving the following linear equations:

$$T\Lambda = \delta^1. \tag{15}$$

Because the rows of C are linearly independent, T is positive definite. Equation (15) has a unique solution Λ, which must satisfy inequality (14). Thus, $(x^\top, y^\top)^\top$ defined by (11) must satisfy all linear constraints. Being a constant positive define matrix, T has a factorization $T = L\tilde{D}L^\top$, where L is a lower triangular matrix with 1's on the diagonal, and $\tilde{D}$ is a diagonal matrix with positive diagonal elements. We only need to make the factorization of T once. With the factorization, solving (15) more than one time is an easy job.

Case 2: $C_1, C_2, \ldots, C_{p+q-u}$ are linearly dependent. Suppose that $I = \operatorname{rank}(C_1, \ldots, C_{p+q-u})$ and $C_1, \ldots, C_I$ are linearly independent without loss of generality. Define d and $(x^\top, y^\top)^\top$ by (10) and (11), respectively.

Similar to case 1, for arbitrary $\delta^1 = (\delta_1^1, \ldots, \delta_I^1)^\top \in R^I$ satisfying $\delta^1 \leq \delta$, we can get a unique solution Λ by (15). Then, $(x^\top, y^\top)^\top$ defined by (11) satisfies the first I inequalities in (12). However, it may not satisfy the last $(p+q-u-I)$ linear constraints. In fact, for each $j > I$, since there exists constants $\theta_i^j (i = 1, 2, \ldots, I)$ such that $C_j = \sum_{i=1}^{I} \theta_i^j C_i$, and $C_j \begin{pmatrix} x0 \\ y0 \end{pmatrix} = b_j^0, C_i d = \delta_i^1$ for $i = 1, \ldots, I$, then

$$\begin{aligned} C_j \begin{pmatrix} x \\ y \end{pmatrix} &= C_j \begin{pmatrix} x0 \\ y0 \end{pmatrix} + C_j d \\ &= b_j^0 + C_j d \\ &= b_j^0 + \sum_{i=1}^{I} \theta_i^j C_i d \\ &= b_j^0 + \sum_{i=1}^{I} \theta_i^j \delta_i^1. \end{aligned}$$

The jth linear constraint in (12) holds if and only if

$$b_j^0 + \sum_{i=1}^{I} \theta_i^j \delta_i^1 \leq b_j \tag{16}$$

holds.

To ensure the first I linear constraints hold, δ^1 should satisfy $\delta^1 \leq \delta$. Furthermore, in order to make the jth linear constraint hold for $j > I, \delta^1$ should also satisfy (16).

If there exists some $j \in \{I+1, \ldots, p+q-u\}$ such that inequality (16) does not hold, we have to decrease the left-hand side of inequality (16). In addition to $\delta^1 \leq \delta$, if δ^1 is modified in the following way:

$$\delta_i^1 = \begin{cases} \delta_i, & \text{if } \theta_i^j \leq 0 \\ -M_i^j, & \text{if } \theta_i^j > 0 \end{cases}, \qquad i = 1, 2, \ldots, I$$

where $-M_i^j \leq \delta_i$ ($-M_i^j < 0$ should gradually decrease), then each term $\theta_i^j \delta_i^1$ in the left-hand side of inequality (16) will decrease, which will make inequality (16) hold. If there are several unsatisfied linear constraints, δ^1 can be modified in a similar way. Thus, we can choose δ^1 in the set

$$\{\delta^1 \leq \delta \,|\, \delta^1 \text{ satisfies (16)}, I+1 \leq j \leq p+q-u\}. \tag{17}$$

Based on aforementioned argument, a scheme for handling linear constraints can be constructed as follows.

- For any $(x0^\top, y0^\top)^\top$ not satisfying all linear constraints, choose δ^1 according to formula (17) and get Λ by solving (15).
- A point $(x^\top, y^\top)^\top$, satisfying all linear constraints can be generated by

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x0 \\ y0 \end{pmatrix} - \sum_{j=1}^{I} \lambda_j C_j^\top. \tag{18}$$

2) Handling Nonlinear Constraints: Handling nonlinear constraints consists of two steps. The first step is handling equality constraints, and the second one is handling all constraints. For convenience, we use the notations in (5) and 6).

First Step: For each fixed point $(x1^\top, y1^\top)^\top$ generated by the scheme for handling linear constraints in (18), if $(x1^\top, y1^\top, U0^\top)^\top$ does not satisfy the equality constraints, we handle equality constraints by solving the linear equations

$$h_k(x1, y1, U) = 0, \qquad k = 1, \ldots, m+1 \tag{19}$$

to get its solution $U1$. Then, Θ_1 satisfies all linear and equality constraints.

Second Step: For any point Θ_1 satisfying all linear and equality constraints, but not satisfying nonlinear inequality constraints, choose a point Θ_2 among all nondominated points generated up to now such that

$$(E_1(\Theta_2), \ldots, E_J(\Theta_2)) \leq (E_1(\Theta_1), \ldots, E_J(\Theta_1)) \tag{20}$$

and there exists at least one $i \in \{1, 2, \ldots, J\}$ such that

$$E_i(\Theta_2) < E_i(\Theta_1). \tag{21}$$

Point Θ_2 satisfying conditions (20) and (21) is better than point Θ_1 with respect to $(E_1(\Theta), \cdots, E_J(\Theta))$. Furthermore, there is no point generated up to now to be better than point Θ_2. Thus

$$D^1 = \Theta_2 - \Theta_1 \tag{22}$$

is generally a descent direction for all constraint functions $E_1(\Theta), \ldots, E_J(\Theta)$ at point Θ_1 for problem (5). For saving the computation amount, an inexact line search is executed in this direction with the aim of identifying a solution

$$\Theta = \Theta_1 + \alpha(\Theta_2 - \Theta_1), \qquad \alpha \geq 0 \tag{23}$$

that satisfies nonlinear inequality constraints approximately.

Based on these two cases, a new constraint-handling method can be presented as follows.

Algorithm 1:

Step 1) For any infeasible point Θ_0 of problem (5), if it does not satisfy linear constraints, we use linear constraint-handling scheme (18) to get a point $(x1^\top, y1^\top)^\top$; otherwise, let $x1 = x0, y1 = y0$, and go to Step 2).

Step 2) If $(x1^\top, y1^\top, U0^\top)^\top$ does not satisfy equality constraints, we view $x1$ and $y1$ as constants, and use the equality constraint-handling scheme (19) to get a point Θ_1. Otherwise, let $U1 = U0$, go to Step 3).

Step 3) If Θ_1 does not satisfy nonlinear inequality constraints, we use formula (23) to get a point Θ.

Note that $(x1^\top, y1^\top)^\top$ generated in the first step satisfies all of the linear constraints of problem (5), and Θ_1 determined by the second step not only satisfies all of the linear constraints, but also satisfies the equality constraints of problem (5). Moreover, Θ generated in the third step satisfies approximately nonlinear constraints while maintaining the satisfaction of linear constraints.

Therefore, the benefit of the constraint handling is to enhance the search ability of algorithm such that it can force the infeasible solution to become the feasible solution approximately. From this point of view, it can improve the quality of the individuals and speed the movement of the individuals toward optimal

solution. Thus, it can effectively handle all of the linear and nonlinear constraints and speed the convergence.

D. Proposed EA

Denote $\Psi = \{(x,y)\,|\,x \in X, y \in Y\}$. The proposed algorithm for nonlinear BLPP is presented as follows:

Algorithm 2:

Step 1) (*Initialization*) Randomly generate $N_{\rm POP}$ initial points $\{\bar{\Theta}_i\,|\,i = 1,\ldots,N_{\rm POP}\}$, where $\{(\bar{xi},\bar{yi})\,|\,i = 1,\ldots,N_{\rm POP}\} \subset \Psi$, and $\bar{U}i$'s are all random vectors in $[0,1]^q$. Apply Algorithm 1 to each of these points to get the initial population ${\rm POP} = \{\Theta_i\,|\,i = 1,\ldots,N_{\rm POP}\}$. Let ${\rm OP} = {\rm POP}$, and choose offspring set $O = \phi$.

Step 2) Define two-objective optimization problem (7). Compute its current nondominated set of set OP, and denote it by NDOM.

Step 3) (*Crossover*) Randomly select $\lceil p_c \times N_{\rm POP}\rceil$ parents from the current population POP, where $\lceil\sigma\rceil$ stands for the largest integer not greater than σ, p_c is the probability of crossover. Do crossover for each selected parent by using (8) and (9) to get $\lceil p_c \times N_{\rm POP}\rceil$ offspring.

Step 4) (*Mutation*) Randomly select $\lceil p_m \times N_{\rm POP}\rceil$ parents from the current population POP, where p_m is the probability of mutation. Suppose that Θ is a selected parent, then Θ is to become $\bar{\Theta} = \Theta + \Delta\Theta$ by mutation, where

$$\Delta\Theta \sim N(0,\sigma) = (N(0,\sigma_1),\ldots,N(0,\sigma_{n+m+q}))^{\top}$$

with the independent components, and $N(0,\sigma_i)$ represents Gaussian distribution with mean 0 and deviation σ^2 (i.e., the ith component of $\Delta\Theta$ obeys Gaussian distribution with mean 0 and deviation σ^2). Then, the resulting point $\bar{\Theta}$ is an offspring of $\Theta = (x^{\top}, y^{\top}, U^{\top})^{\top}$.

Step 5) (*Constraint handling*) The set of all offspring generated in steps 3 and 4 is denoted by $O1$. Use Algorithm 1 to each offspring in $O1$ to get an improved offspring. The set of all of the improved offspring is denoted by $O2$.

Step 6) (*Selection*) Let ${\rm OP} = {\rm NDOM} \cup O1 \cup O2$. Select the best point from the set ${\rm POP} \cup {\rm OP}$ and then randomly select $N_{\rm POP} - 1$ points from it. These points form the next population POP, and go to Step 2).

IV. Global Convergence

Suppose that $F(x,y)$ and $G(x,y)$ are continuous, and $f(x,y)$ and $g(x,y)$ are differentiable and convex in y for x fixed. As mentioned in Section II under these assumptions, the problem (1) is equivalent to problem (5). We only need to prove the global convergence of the proposed algorithm for problem (5). For convenience, we denote the feasible set of problem (5) by Ω.

A. Preliminaries

We briefly introduce some concepts and conclusions in probability ([17]).

Definition 5: Suppose that $\{\xi_k\}$ is a random sequence on probability space. If there is a random variable ξ such that $\lim_{k\to\infty} P\{\|\xi_k - \xi\| < \varepsilon\} = 1$ for $\forall \varepsilon > 0$, then $\{\xi_k\}$ is called to converge to ξ in probability, where $P\{A\}$ represents the probability of random event A.

Definition 6: Suppose that $\{\xi_k\}$ is a random sequence on probability space. If there is a random variable ξ such that $P\{\lim_{k\to\infty}\xi_k = \xi\} = 1$, or $P\{\bigcap_{t=1}^{\infty}\bigcup_{k\geq t}\{\|\xi_k - \xi\| \geq \varepsilon\}\} = 0$ for $\forall\varepsilon > 0$, then $\{\xi_k\}$ is called to converge to ξ with probability one.

B. Global Convergence

Assumptions:

A1. The feasible set Ω of problem (5) is a bounded closed set in R^{n+m+q}, and for $\forall\Theta \in \Omega$, the Lebesgue measure of the intersection of Ω and any neighborhood of Θ is positive.

A2. The objective function $F(\Theta)$ is continuous on $[\Theta_l, \Theta_u] \supseteq \Omega$.

For $\forall\varepsilon > 0$, if we denote

$$Q_1 = \{\Theta \in \Omega\,|\,|F(\Theta) - F^*| < \varepsilon\}; \quad Q2 = \Omega\backslash Q1 \tag{24}$$

where $F^* = F(\Theta^*), \Theta^* \in \Theta_{\rm opt}$, then all ${\rm POP}(k)$'s generated by Algorithm 2 can be classified into two states.

1) If there exists at least one point in ${\rm POP}(k)$ belonging to $Q1$, then ${\rm POP}(k)$ is called to be in state S_1, where ${\rm POP}(k)$ represents the population at generation k.
2) If all points in ${\rm POP}(k)$ belong to Q_2, then ${\rm POP}(k)$ is called to be in state S_2.

Theorem 2: Let $p_{ij} = P\{{\rm POP}(k)$ in state $S_i, {\rm POP}(k+1)$ in state $S_j\}, i,j = 1,2$, then under Assumptions A1 and A2, we have the following conclusions:

1) For any ${\rm POP}(k)$ in state S_1, it must satisfy $p_{11} = 1$.
2) For any ${\rm POP}(k)$ in state S_2, there exists a constant $c \in (0,1)$ such that $p_{22} < c$.

Proof: It follows from the selection scheme of Algorithm 2 that ${\rm POP}(k)$ in state S_1 cannot be evolved to a population in state S_2. Thus, the first conclusion is true.

It follows from the Assumption A1 that $\Theta_{\rm opt} = \{\Theta\,|\,\arg\min_{\Theta\in\Omega} F(\Theta)\} \neq \emptyset$. For any global optimal solution $\Theta^* \in \Theta_{\rm opt}$, since $F(\Theta)$ is continuous on Ω, then there exists $\gamma > 0$ such that

$$|F(\Theta) - F(\Theta^*)| < \varepsilon/2 \tag{25}$$

when

$$\|\Theta - \Theta^*\| \leq \gamma, \qquad \Theta \in \Omega. \tag{26}$$

If we denote

$$Nb(\Theta^*,\gamma) = \{\Theta \in \Omega : \|\Theta - \Theta^*\| \leq \gamma\} \tag{27}$$

then

$$Nb(\Theta^*,\gamma) \subseteq Q_1. \tag{28}$$

For ${\rm POP}(k)$ in state S_2, suppose that $\Theta \in {\rm POP}(k)$ is any individual that undergoes the mutation and it generates an offspring $\bar{\Theta} = \Theta + \Delta\Theta$ by mutation in step

4 of Algorithm 2, where $\Delta\Theta = (\Delta\theta_1, \ldots, \Delta\theta_{n+m+q})^\top \sim (N(0,\sigma_1), \ldots, N(0,\sigma_{n+m+q}))^\top$ (i.e., $\Delta\theta_i$ obeys Gaussian distribution with mean 0 and deviation $\sigma_i^2 (i = 1 \sim n+m+q)$, and $\Delta\theta_1, \ldots, \Delta\theta_{n+m+q}$ are independent). Since $\Delta\theta_i \sim N(0,\sigma_i)$ and

$$\begin{aligned} P\{\bar{\Theta} \in Nb(\Theta^*,\gamma)\} &= P\{\Theta + \Delta\Theta \in Nb(\Theta^*,\gamma)\} \\ &\le \prod_{i=1}^{n+m+q} P\{|\theta_i + \Delta\theta_i - \theta_i^*| \le \gamma\} \end{aligned}$$

then

$$P\{\bar{\Theta} \in Nb(\Theta^*,\gamma)\} \le \prod_{i=1}^{n+m+q} \int_{\theta_i^*-\theta_i-\gamma}^{\theta_i^*-\theta_i+\gamma} \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{t^2}{2\sigma_i^2}}\, dt \tag{29}$$

where θ_i and θ_i^* represent the ith components of Θ and Θ^*, respectively. If we denote

$$P_1(\Theta) = P\{(\Theta + \Delta\Theta) \in Nb(\Theta^*,\gamma)\}, \qquad \Theta \in \Omega \tag{30}$$

it follows from the measure of $Nb(\Theta^*,\gamma)$ being positive that $P_1(\Theta) > 0$ and from formula (29) that $P_1(\Theta) < 1$. Thus

$$0 < P_1(\Theta) < 1, \qquad \Theta \in \Omega. \tag{31}$$

Since each $\Delta\theta_i$ is a continuously random variable with Gaussian distribution, it easily follows from formulas (29) and (30) that $P_1(\Theta)$ is continuous on Ω. Also note that Ω is a closed bounded set; thus, there exists $\tilde{\Theta} \in \Omega$ such that $P_1(\tilde{\Theta}) = \min\{P_1(\Theta) \,|\, \Theta \in \Omega\}$ and

$$0 < P_1(\tilde{\Theta}) < 1. \tag{32}$$

It directly follows from formulae (28), (32), and the definition of p_{21} that

$$p_{21} \ge P_1(\Theta) \ge P_1(\tilde{\Theta}). \tag{33}$$

Denote

$$c = 1 - P_1(\tilde{\Theta}) \tag{34}$$

then by formula (32), we have

$$0 < c < 1. \tag{35}$$

It follows from (33), (34), and $p_{21} + p_{22} = 1$ that

$$p_{22} = 1 - p_{21} \le 1 - P_1(\tilde{\Theta}) = c.$$

This completes the proof. ∎

Theorem 3: Suppose that $\{POP(k)\}$ is a population sequence generated by Algorithm 2, and there exists at least one point $\Theta \in \Omega$ in initial population. Denote $\Theta^*(k) = \arg\min\{F(\Theta) : \Theta \in \mathrm{POP}(k)\}$, then, under the Assumptions (A1) and (A2), we have

$$P\{\lim_{k\to\infty} F(\Theta^*(k)) = F(\Theta^*)\} = 1 \tag{36}$$

that is, the population sequence converge to global optimal solution with probability one.

Proof: For any $\varepsilon > 0$, denote $p_k = P\{|F(\Theta^*(k)) - F(\Theta^*)| \ge \varepsilon\}$, then

$$p_k = \begin{cases} 0, & \text{if } \Theta^*(t) \in Q_1 \text{ for some } 1 \le t \le k \\ \overline{p_k}, & \text{if } \Theta^*(j) \notin Q_1,\ j = 1, \ldots, k. \end{cases} \tag{37}$$

It follows from Theorem 2 that

$$\bar{p_k} = P\{\Theta^*(t) \notin Q_1, t = 1, 2, \ldots, k\} = p_{22}^k \le c^k. \tag{38}$$

Thus

$$\sum_{k=1}^{\infty} p_k \le \sum_{k=1}^{\infty} c^k = \frac{c}{1-c} < \infty. \tag{39}$$

By the Borel–Cantelli Lemma [17], we have

$$P\left\{\bigcap_{t=1}^{\infty} \bigcup_{k\ge t} \{|F(\Theta^*(k)) - F(\Theta^*)| \ge \varepsilon\}\right\} = 0.$$

Thus, the conclusion can be obtained by Definition 6. ∎

V. Simulation Results

A. Test Functions

In this section, 31 benchmark problems were used for simulations. The first 19 problems are selected from [3], [4], [13]–[16], [18], [19]. In order to illustrate the efficiency of the proposed algorithm for solving complex nonlinear BLPPs with nondifferentiable, even nonconvex, leader's objective functions, we construct 12 new benchmark problems, problems 20 to 31. The leader's objective functions of problems 20 and 29 are nondifferentiable and convex, while the leader's objective functions of problems 21 to 28, 30, and 31 are nondifferentiable and nonconvex. For notational simplicity, denote $x = (x_1, \ldots, x_n)^\top$ and $y = (y_1, \ldots, y_m)^\top$ in the test problems. The details of these problems are as follows:

1) Reference [13]: Let $n = 2$ and $m = 2$

$$\begin{aligned} \min_x\ & F(x,y) = (x_1 - 30)^2 + (x_2 - 20)^2 - 20y_1 + 20y_2 \\ & \text{s.t. } x_1 + 2x_2 \ge 30, \quad x_1 + x_2 \le 25, \quad x_2 \le 15 \\ \min_{0\le y\le 10}\ & f(x,y) = (x_1 - y_1)^2 + (x_2 - y_2)^2. \end{aligned}$$

2) Reference [14]: Let $n = 2$ and $m = 2$

$$\begin{aligned} \min_{0\le x\le 50}\ & F(x,y) = 2x_1 + 2x_2 - 3y_1 - 3y_2 - 60 \\ & \text{s.t. } x_1 + x_2 + y_1 - 2y_2 \le 40 \\ \min_{-10\le y\le 20}\ & f(x,y) = (y_1 - x_1 + 20)^2 + (y_2 - x_2 + 20)^2 \\ & \text{s.t. } x_1 - 2y_1 \ge 10, \quad x_2 - 2y_2 \ge 10. \end{aligned}$$

3) Reference [4]: Let $n = 2$ and $m = 2$

$$\begin{aligned}
&\min_x F(x,y) = -x_1^2 - 3x_2^2 - 4y_1 + y_2^2 \\
&\quad \text{s.t. } x_1^2 + 2x_2 \le 4, \quad x \ge 0 \\
&\min_y f(x,y) = 2x_1^2 + y_1^2 - 5y_2 \\
&\text{s.t. } x_1^2 - 2x_1 + x_2^2 - 2y_1 + y_2 \ge -3 \\
&x_2 + 3y_1 - 4y_2 \ge 4, \quad y \ge 0.
\end{aligned}$$

4) Reference [3]: Let $n = 2$ and $m = 3$

$$\begin{aligned}
&\min_{x \ge 0} F(x,y) = -8x_1 - 4x_2 + 4y_1 - 40y_2 - 4y_3 \\
&\min_y f(x,y) = x_1 + 2x_2 + y_1 + y_2 + 2y_3 \\
&\text{s.t. } y_2 + y_3 - y_1 \le 1, \quad 2x_1 - y_1 + 2y_2 - 0.5y_3 \le 1 \\
&2x_2 + 2y_1 - y_2 - 0.5y_3 \le 1, \quad y \ge 0.
\end{aligned}$$

5–9) Reference [15]: Let $n = 2$ and $m = 2$

$$\begin{aligned}
&\min_x F(x,y) = rx^\top x - 3y_1 - 4y_2 + 0.5y^\top y \\
&\min_y f(x,y) = 0.5y^\top Hy - b(x)^\top y \\
&\text{s.t.} - 0.333y_1 + y_2 - 2 \le 0 \\
&\qquad y_1 - 0.333y_2 - 2 \le 0, \quad y \ge 0
\end{aligned}$$

where $H_1 = \begin{bmatrix} 1 & -2 \\ -2 & 5 \end{bmatrix}, H_2 = \begin{bmatrix} 1 & 3 \\ 3 & 10 \end{bmatrix}$

$$\begin{aligned}
&5: r = 0.1, \quad H = H_1, \quad b(x) = x \\
&6: r = 1, \quad H = H_1, \quad b(x) = x \\
&7: r = 0, \quad H = H_2, \quad b(x) = x \\
&8\text{–}9: r = 0.1, \quad H = H_2, b(x) = x \text{ for } 8 \\
&b(x) = \begin{bmatrix} -1 & 2 \\ 3 & -3 \end{bmatrix} x \text{ for } 9.
\end{aligned}$$

10) Reference [16]: Let $a_{ij} = y_i + y_j, n = 4$, and $m = 4$

$$\begin{aligned}
&\min_x F(x,y) = a_{13}(a_{13} - 200) + a_{24}(a_{24} - 160) \\
&\text{s.t. } x_1 + x_2 + x_3 + x_4 \le 40, \quad 0 \le x_1 \le 10 \\
&\qquad 0 \le x_2 \le 5, \quad 0 \le x_3 \le 15, \quad 0 \le x_4 \le 20 \\
&\min_y f_1(x,y) = (y_1 - 4)^2 + (y_2 - 13)^2 \\
&\quad \text{s.t. } 0.4y_1 + 0.7y_2 - x_1 \le 0, \quad 0 \le y_1 \le 20 \\
&\qquad 0.6y_1 + 0.3y_2 - x_2 \le 0, \quad 0 \le y_2 \le 20 \\
&\min_y f_2(x,y) = (y_3 - 35)^2 + (y_4 - 2)^2 \\
&\quad \text{s.t. } 0.4y_3 + 0.7y_4 - x_3 \le 0, \quad 0 \le y_3 \le 40 \\
&\qquad 0.6y_3 + 0.3y_4 - x_4 \le 0, \quad 0 \le y_4 \le 40.
\end{aligned}$$

11) Reference [18]: Let $m = 2$ and $n = 1$

$$\begin{aligned}
&\max_{0 \le x \le 1} F(x,y) = 100x + 1000y_1 \\
&\max_{y_1, y_2} f(x,y) = y_1 + y_2 \\
&\text{s.t. } x + y_1 - y_2 \le 1, \quad y_1 + y_2 \le 1.
\end{aligned}$$

12) Reference [18]: Let $m = 1$ and $n = 1$

$$\begin{aligned}
&\min_x F(x,y) = (x-1)^2 + (y-1)^2 \\
&\min_y f(x,y) = 0.5y^2 + 500y - 50xy.
\end{aligned}$$

13) Reference [18]: Let $m = 1$ and $n = 1$

$$\begin{aligned}
&\min_x F(x,y) = x^2 + (y-10)^2 \\
&\text{s.t. } -x + y \le 0, \quad 0 \le x \le 15 \\
&\min_y f(x,y) = (x + 2y - 30)^2 \\
&\quad \text{s.t. } x + y \le 20, \quad 0 \le y \le 20.
\end{aligned}$$

14) Reference [18]: Let $m = 2$ and $n = 1$

$$\begin{aligned}
&\min_{x \ge 0} F(x,y) = (x-1)^2 + 2y_1 - 2x \\
&\min_{y_1, y_2 \ge 0} f(x,y) = (2y_1 - 4)^2 + (2y_2 - 1)^2 + xy_1 \\
&\text{s.t. } 4x + 5y_1 + 4y_2 \le 12, \quad 4y_2 - 4x - 5y_1 \le -4 \\
&\qquad 4x - 4y_1 + 5y_2 \le 4, \quad 4y_1 - 4x + 5y_2 \le 4.
\end{aligned}$$

15) Reference [19]: Let $n = 3, f_i(y_i) = y_{i1} \sin y_{i2} + y_{i2} \sin y_{i1}$, and $h_i = y_{i1} y_{i2} \sin x_i, y_i = (y_{i1}, y_{i2})^\top$

$$\begin{aligned}
&\max_x F(x, y_1, y_2) = h_1 + h_2 + h_3 \\
&\text{s.t. } x_1 + x_2 + x_3 \le 10, \quad x \ge 0 \\
&\max_{y_1} f_1(y_1), \quad \text{s.t. } y_{11} + y_{12} \le x_1, \quad y_1 \ge 0 \\
&\max_{y_2} f_2(y_2), \quad \text{s.t. } y_{21} + y_{22} \le x_2, \quad y_2 \ge 0 \\
&\max_{y_3} f_3(y_3), \quad \text{s.t.} \quad y_{31} + y_{32} \le x_3, \quad y_3 \ge 0.
\end{aligned}$$

16) Reference [19]: The problem is same as 15 except for

$$F(x, y_1, y_2) = h_1 + 2h_2 + 3h_3.$$

17) Reference [19]: Let $m = 2$ and $n = 2$

$$\begin{aligned}
&\max_x F(x,y) = \frac{(x_1 + y_1)(x_2 + y_2)}{1 + x_1 y_1 + x_2 y_2} \\
&\text{s.t. } x_1^2 + x_2^2 \le 100, \quad x_1, \quad x_2 \ge 0 \\
&\max_{y_1, y_2} f(x,y) = -F(x,y) \\
&\quad \text{s.t. } 0 \le y_1 \le x_1, \quad 0 \le y_2 \le x_2.
\end{aligned}$$

18) Reference [19]: Let $n = 2, h_i = (y_{i1} + y_{i2})^2, k_j = y_{1j}/y_{2j}$

$$\begin{aligned}
&\min_x F(x, y_1, y_2, y_3) = \frac{3h_1 + 5h_2 + 10h_3}{2x_1^2 + x_2^2 + 3x_1 x_2} \\
&\text{s.t. } x_1 + 2x_2 \le 10 \quad x_1 > 0, \quad x_2 > 0 \\
&\min_{y_1} f_1(y_1) = y_{11}^2 + y_{12}^2 \\
&\text{s.t. } y_{11} + y_{21} + y_{31} \ge x_1, \quad y_{11} \ge 1 \\
&\qquad y_{12} + y_{22} + y_{32} \ge x_2, \quad y_{12} \ge 2 \\
&\min_{y_2 \ge 0} f_2(y_2) = y_{21} + y_{22} + k_1 + k_2 \\
&\min_{y_3} f_3(y_3) = \frac{(y_{31} - y_{21})^2}{y_{31}} + \frac{(y_{32} - y_{22})^2}{y_{32}} \\
&\text{s.t. } 2y_{31} + 3y_{32} = 5, \quad y_{31} > 0, \quad y_{32} > 0
\end{aligned}$$

where y_i is the same as that in 15.

19) Reference [20]: Let $m = 6$ and $n = 2$

$$\min_{x\geq 0} F(x,y) = -8x_1 - 4x_2 + 4y_1 - 40y_2 - 4y_3$$
$$\min_{y\geq 0} f(x,y) = \frac{1 + x_1 + x_2 + 2y_1 - y_2 + y_3}{6 + 2x_1 + y_1 + y_2 - 3y_3}$$
$$\text{s.t. } -y_1 + y_2 + y_3 + y_4 = 1$$
$$2x_1 - y_1 + 2y_2 - 0.5y_3 + y_5 = 1$$
$$2x_2 + 2y_1 - y_2 - 0.5y_3 + y_6 = 1.$$

20): Let $m = 2, n = 2$, and $h_i = (y_i - x_i + 20)^2$

$$\min_{0\leq x\leq 50} F(x,y) = |2x_1 + 2x_2 - 3y_1 - 3y_2 - 60|$$
$$\text{s.t. } x_1 + x_2 + y_1 - 2y_2 \leq 40$$
$$\min_{-10\leq y\leq 20} f(x,y) = h_1 + h_2$$
$$\text{s.t. } 2y_1 - x_1 + 10 \leq 0, \quad 2y_2 - x_2 + 10 \leq 0.$$

21): The problem is the same as 20 except for

$$F(x,y) = |\sin(2x_1 + 2x_2 - 3y_1 - 3y_2 - 60)|.$$

22): The problem is the same as 20 except for

$$F(x,y) = |\tan(2x_1 + 2x_2 - 3y_1 - 3y_2 - 60)|.$$

23): Let $m = n = 2$, and $h = (x_1 - 30)^2 + (x_2 - 20)^2$

$$\min_{x} F(x,y) = |h - 20y_1 + 20y_2 - 225|$$
$$\text{s.t. } 30 - x_1 - 2x_2 \leq 0, \quad x_1 + x_2 - 25 \leq 0, \quad x_2 \leq 15$$
$$\min_{0\leq y\leq 10} f(x,y) = (x_1 - y_1)^2 + (x_2 - y_2)^2.$$

24): The problem is the same as 23 except for

$$F(x,y) = |\sin(h - 20y_1 + 20y_2 - 225)|.$$

25): The problem is the same as 23 except for

$$F(x,y) = |\tan(h - 20y_1 + 20y_2 - 225)|.$$

26): The problem is the same as 14 except for

$$F(x,y) = |(x-1)^2 + 2y_1 - 2x + 1.2097|.$$

27): The problem is the same as 26 except for

$$F(x,y) = |\sin((x-1)^2 + 2y_1 - 2x + 1.2097)|.$$

28): The problem is the same as 26 except for

$$F(x,y) = |\tan((x-1)^2 + 2y_1 - 2x + 1.2097)|.$$

29): The problem is the same as 19 except for

$$F(x,y) = |-8x_1 - 4x_2 + 4y_1 - 40y_2 - 4y_3 + 29.2|.$$

30): The problem is the same as 29 except for

$$F(x,y) = |\sin(-8x_1 - 4x_2 + 4y_1 - 40y_2 - 4y_3 + 29.2)|.$$

31): The problem is the same as 29 except for

$$F(x,y) = |\tan(-8x_1 - 4x_2 + 4y_1 - 40y_2 - 4y_3 + 29.2)|.$$

TABLE I
COMPARISON OF THE BEST RESULTS FOUND BY NEA AND THE RELATED ALGORITHMS FOR PROBLEMS 1–19

	$F(x^*,y^*)$		$f(x^*,y^*)$	
No.	NEA	Ref	NEA	Ref
1([13])	225	225	100	100
2([14])	0	5	100	0
3([4])	-12.68	-12.68	-1.016	-1.016
4([3])	-29.2	-29.2	3.2	3.2
5([15])	-8.92	-8.92	-6.14	-6.05
6([15])	-7.58	-7.56	-0.574	-0.580
7([15])	-11.999	-12	-163.42	-112.71
8([15])	-3.6	-3.6	-2	-2
9([15])	-3.92	-3.15	-2	-16.29
10([16])	-6600	-6600	$f_1 = 23.6358$ $f_2 = 30.5833$	$f_1 = 11.924$ $f_2 = 64.684$
10([19])	-6600	-6599.99	$f_1 = 23.6358$ $f_2 = 30.5833$	$f_1 = 23.47$ $f_2 = 30.83$
11([18])	1000	1000	1	1
12([18])	81.3279	82,44	-0.3359	0.271
13([18])	100.0001	100.58	3.5e-11	0.001
14([18])	-1.2098	3.57	7.6168	2.4
15([19])	9.5644	9.566	$f_1 = 1.66614$ $f_2 = 7.099$ $f_3 = 0$	$f_1 = 1.609$ $f_2 = 7.099$ $f_3 = 0$
16([19])	27.8148	27.8222	$f_1 = 0$ $f_2 = 1.5906$ $f_3 = 7.1266$	$f_1 = 0$ $f_2 = 1.595$ $f_3 = 7.120$
17([19])	1.9802	1.9760	-1.9802	-1.9454
18([19])	1.5101	1.5100	$f_1 = 11.6551$ $f_2 = 6.1552$ $f_3 = 0.5219$	$f_1 = 12.323$ $f_2 = 6.225$ $f_3 = 0.835$
19([20])	-29.2	-29.2	0.3148	0.3148

B. Results

We execute the proposed algorithm in 50 independent runs on each of the above 31 benchmark problems on a microcomputer Intel Pentium IV/1.2 GHz and record the following data:

- best solution (x^*, y^*) and the worst solution $(\bar{x}, \bar{y})$ found in 50 independent runs;
- leader's objective values $F(x^*, y^*)$ at the best solution $(x^*), (y^*)$ and $F(\bar{x}, \bar{y})$ at the worst solution $(\bar{x}, \bar{y})$;
- follower's objective values $f(x^*, y^*)$ at the best solution $(x^*), (y^*)$ and $f(\bar{x}, \bar{y})$ at the worst solution $(\bar{x}, \bar{y})$;
- mean values of CPU time (denoted by CPU in short) and Mean Numbers of Individuals (MNI) used by the proposed algorithm for each of these benchmark problems.

The parameters are chosen as follows: the population size $N_{\text{POP}} = 30$, the probability of crossover $p_c = 0.8$, and the probability of mutation $p_m = 0.2$. The algorithm stops after 100 generations.

We compare the obtained results with those presented in the corresponding references. For problems 1 to 19, all of the results are presented in Tables I, II, V, and VI. Tables III and VII summarize the optimal results found by the proposed algorithm in

TABLE II
WORST RESULTS FOUND BY NEA FOR PROBLEMS 1–19

	NEA	
No.	$F(\bar{x},\bar{y})$	$f(\bar{x},\bar{y})$
1	225	100
2	0	100
3	-12.65	-1.021
4	-29.2	3.2
5	-8.92	-6.14
6	-7.575	-0.5792
7	-11.999	-163.42
8	-3.6	-2
9	-3.918	-1.956
10	-6600	$f_1 = 23.6358, f_2 = 30.5833$
10	-6600	$f_1 = 23.6358,\ f_2 = 30.5833$
11	1000	1
12	81.3279	-0.3359
13	100.014	4.93e-7
14	-1.2091	7.6145
15	9.5640	$f_1 = 1.607,\ f_2 = 7.098, f_3 = 0$
16	27.8143	$f_1 = 0, f_2 = 1.5817, f_3 = 7.1404$
17	1.98	-1.98
18	1.5101	$f_1 = 11.6551,\ f_2 = 6.1552,\ f_3 = 0.5219$
19	-29.2	0.3148

TABLE III
BEST AND THE WORST RESULTS FOUND BY NEA FOR PROBLEMS 20–31

	Best		Worst	
No.	$F(x^*,y^*)$	$f(x^*,y^*)$	$F(\bar{x},\bar{y})$	$f(\bar{x},\bar{y})$
20	0	100	5	0
21	0	100	0.3894	7.85
22	0	100	0	100
23	0	100	0	100
24	6.86e-15	91.45	6.46e-14	75.84
25	1.47e-14	8.18	4.41e-14	13.16
26	2.22e-16	7.62	1.21e-6	7.17
27	1.22e-16	2.50	1.22e-16	2.50
28	1.22e-16	2.50	1.22e-16	2.50
29	8.99e-13	0.3148	7.08e-11	0.3148
30	4.90e-16	0.2125	4.90e-16	0.2125
31	7.35e-16	0.2367	6.25e-15	0.4557

50 independent runs for problems 20–31, respectively. For convenience, NEA stands for the proposed algorithm. In Tables I and V, Ref stands for the algorithm in the corresponding references indicated in the parentheses behind the problem numbers. In Table V, NA means that the result is not available for the algorithm.

From Table I, we can see that for problems 1–19 except for problems 7 and 18, the solutions found by NEA are better than or equal to those by the compared algorithms in the references. For problems 2, 9, 13, and 14, the solutions found by NEA are much better than those by the compared algorithms. The results also show that the solutions found by the algorithms in [14], [15], and [18] for problems 2, 9, 13, and 14 are not the global optimal solutions. Although the solutions found by NEA for problems

TABLE IV
MEAN VALUES OF CPU TIME (SECONDS) AND MEAN NUMBERS OF INDIVIDUALS USED BY NEA IN 50 INDEPENDENT RUNS FOR PROBLEMS 1–31, WHERE MNI REPRESENTS MEAN NUMBER OF INDIVIDUALS USED

No.	CPU	MNI	No.	CPU	MNI
1	13.314	85499	2	37.308	256227
3	14.42	92526	4	45.39	291817
5	11.854	71273	6	11.754	77292
7	13.184	92492	8	11.82	77302
9	11.908	71302	10	82.126	533810
11	5.690	4276	12	4.218	28468
13	5.888	4339	14	25.332	163701
15	812.328	5266950	16	712.47	4647726
17	177.672	1074742	18	116.672	747335
19	107.55	697511	20	33.354	213522
21	41.87	263344	22	43.050	270461
23	15.798	106760	24	13.546	92526
25	13.724	93512	26	27.090	170818
27	23.642	149467	28	22.114	142350
29	132.702	854103	30	122.226	783636
31	121.272	782925			

TABLE V
COMPARISON OF THE BEST SOLUTIONS FOUND BY NEA AND THE RELATED ALGORITHMS IN 50 INDEPENDENT RUNS FOR PROBLEMS 1–19

	(x^*,y^*)	
No.	NEA	Ref
1([13])	(20,5,10,5)	(20,5,10,5)
2([14])	(0,30,-10,10)	(25,30,5,10)
3([4])	(4.4e-7,2,1.875,0.9063)	(0,2,1.875,0.9063)
4([3])	(1.25e-13,0.9,0,0.6,0.4)	(0,0.9,0,0.6,0.4)
5([15])	(1.03,3.097,2.59,1.79)	(0.97,3.14,2.6,1.8)
6([15])	(0.27,0.49,2.34,1.036)	(0.28,0.48,2.34,1.03)
7([15])	(12.47,67.511,2.999,2.999)	(20.26,42.8,3,3)
8([15])	(2,-2.84e-8, 2, 0)	(2, 0.06, 2, 0)
9([15])	(-0.381,0.8095,2,0)	(2.42,-3.65,0,1.58)
10([16])	(7.034,3.122,11.938,17.906, 0.25,9.906,29.844,0)	(7.91,4.37,11.09,16.63, 2.29,10,27.21,0)
10([19])	(7.034,3.122,11.938,17.906, 0.25,9.906,29.844,0)	(7.05,4.13,11.93,17.89, 0.26,9.92,29.82,0)
11([18])	(1.4e-12,1,7.07e-13)	NA
12([18])	(10.0164,0.8197)	(10.04,0.1429)
13([18])	(10.000,10.000)	(10.03,9.969)
14([18])	(1.8888,0.8889,0)	NA
15([19])	(1.95,8.05,0,0.975, 0.975,1.314,6.736,0,0)	(1.946,8.054,0,0.973, 0.973,1.315,6.793,0,0)
16([19])	(0,1.933,8.067,0,0, 0.9665,0.9665,1.317,6.75)	(0,1.936,8.064,0,0, 0.968,0.968,1.317,6.747)
17([19])	(7.0709,7.0713,7.0709,7.0713)	(7.0854,7.0291,7.0854,0)
18([19])	(5.5988,2.2005,2.7668,2, 1.6634,1.4142,1.1686,0.8876)	(5.768,2.116,2.885,2, 1.699,1.414,1.183,0.878))
19([20])	(0,0.9,0,0.6,0.4,0,0,0)	(0,0.9,0,0.6,0.4,0,0,0)

TABLE VI
WORST SOLUTIONS FOUND BY NEA IN 50 INDEPENDENT RUNS FOR PROBLEMS 1–19

No.	$(\bar{x}, \bar{y})$	No.	$(\bar{x}, \bar{y})$
1	(20,5,10,5)	2	(0,30,-10,10)
3	(3.1e-6,2, 1.865,0.900)	4	(1.25e-13,0.9, 0,0.6,0.4)
5	(1.04,3.097, 2.59,1.79)	6	(0.27,0.495, 2.34,1.036)
7	(12.47,67.5, 2.999,2.999)	8	(2,-2.84e-8, 2,0)
9	(-0.381,0.8205, 2,0)	10	(7.034,3.122,11.938, 17.906,0.25, 9.906,29.844,0)
11	(1.11e-7,1,5.569e-8)	12	(10.0164,0.8197)
13	(10.0007,9.9993)	14	(1.8885,0.8892,0)
15	(1.945,8.054,5.1e-5, 0.973,0.973,1.314, 6.739,0,0)	16	(5.7e-6,1.926,8.073, 0,0,0.9632, 0.9632,1.318,6.755)
17	(7.0726,7.0696, 6.9349,6.9245)	18	(5.5988,2.2005,2.7668, 2,1.6634,1.4142, 1.1686,0.8876)
19	(0,0.9,0,0.6, 0.4,0,0,0)		

TABLE VII
BEST AND THE WORST SOLUTIONS FOUND BY NEA IN 50 INDEPENDENT RUNS FOR PROBLEMS 20–31

	Best	Worst
No.	(x^*, y^*)	$(\bar{x}, \bar{y})$
20	(0,30,-10,10)	(25,30,5,10)
21	(0,30,-10,10)	(22.1,30.2,4.9,10.1)
22	(0,30,-10,10)	(0,30,-10,10)
23	(20,5,10,5)	(20,5,10,5)
24	(19.562976441763, 5.272238949356, 10,5.272238949356)	(18.708384798780, 5.956612550211, 5.956612550211,10,)
25	(6.204879134651, 12.85944033285, 6.204879134651,10)	(8.449938905967, 13627299028810, 8.449938905967,10)
26	(1.888846884437, 0.888922492450,0)	(1.888840237598, 0.888927809922,0)
27	(0.664849225213, 1.574632531088, 0.072173355300)	(0.664849225213, 1.574632531088, 0.072173355300)
28	(0.664849225213, 1.574632531088, 0.072173355300)	(0.664849225213, 1.574632531088, 0.072173355300)
29	(0,0.9,0,0.6, 0.4,0,0,0)	(0,0.9,0,0.6, 0.4,0,0,0)
30	(0.176021285929, 0.566577633841,0, 0.323978714071,0, 0.676021285928, 0,0.190823446389)	(0.176021285929, 0.566577633841,0, 0.323978714071,0, 0.676021285928, 0,0.190823446389)
31	(0.353445693914, 0.415176570929,0, 0.146554306086,0, 0.853445693914,0, 0.316201164228)	(0.768250479304, 0.243070313001, 0.350120110869,0, 0.372761695478, 0.977358415391,0,0)

7 and 18 are a little bit worse than those by the compared algorithms in [15] and [18], they are almost as good as those by the compared algorithms.

For problem 15, it seems that the solution found by NEA is worse than that found by the algorithm in [19]. However, the solution $y_2^* = (1.315, 6.793)^\top$ (Table V) found by the algorithm in [19] for the second follower's problem is not a feasible solution and $F(x^*, y^*) = 9.566$ given in [19] was wrong. It should be 9.6356. In fact, the solution found by NEA is the global optimal.

For problem 16, similarly, it seems that the solution found by NEA is worse than that found by the algorithm in [19]. However, The solution $y_3^* = (1.317, 6.747)^\top$ (Table V) found by algorithm in [19] for the third follower's problem is not an optimal solution. In fact, its true optimal solution for x^* given in [19] should be $y_3^* = (1.3165, 6.7475)^\top$, which corresponds to a worse solution $F(x^*, y^*) = 27.8141$ than that found by NEA.

Table II shows that all of the worst solutions found by NEA in 50 independent runs for problems 1–19 are the same as or very close to the global optimal solutions. This means that the proposed algorithm is stable.

Table III shows that the proposed algorithm performs well for problems 20 and 31. In 50 independent runs, we found the accurate global optimal solutions for problem 20 in 44 runs and for problem 21 in 48 runs, and found the high precision approximately global optimal solutions for problems 22–31 in all runs. These results also indicate that the proposed algorithm can be used to solve nonlinear BLPP with the nondifferentiable nonconvex leader's problem, which is beyond the scope of the existing algorithms.

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

VI. CONCLUSION

In this paper, we first transform the nonlinear bilevel programming problem into an equivalent single objective optimization problem, which is much easier to be solved by EAs than the original one. To solve the equivalent problem efficiently, we construct a specific two-objective optimization problem and propose a new constraint-handling method. By solving the two-objective optimization problem, we can decrease the leader's objective function values, identify the quality of solutions, and make the solutions go better and better. The new constraint-handling method can make all of the linear constraints satisfied and all of the nonlinear constraints satisfied approximately. Based on these strategies, we then develop a new EA for nonlinear bilevel programming problems. The simulation is executed on 31 benchmark problems and the results demonstrate that the proposed algorithm is effective and can find the better solutions

than the compared algorithms. Moreover, the proposed algorithm has the ability of solving complex nonlinear BLPP with nondifferentiable nonconvex leader's problem, which makes the algorithm applicable to some complex nonlinear BLPPs in engineering applications.

ACKNOWLEDGMENT

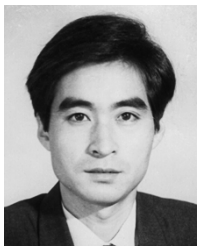
The authors would like to thank anonymous referees for their valuable comments and suggestions.

REFERENCES

[1] R. G. Jeroslow, "The polynomial hierarchy and simple model for competitive analysis," *Math. Programming*, vol. 32, pp. 146–164, 1985.
[2] J. F. Bard, "Some properties of the bilevel programming problem," *J. Optim. Theory Appl.*, vol. 68, no. 2, pp. 371–378, 1991.
[3] ——, *Practical Bilevel Optimization*. Norwell, MA: Kluwer, 1998.
[4] M. A. Amouzegar, "A global optimization method for nonlinear bilevel programming problems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 6, pp. 771–777, Dec. 1999.
[5] L. Vicente, G. Savard, and J. Judice, "Descent approach for quadratic bilevel programming," *J. Optim. Theory Appl.*, vol. 81, pp. 379–399, 1994.
[6] F. Al-Khayyal, R. Horst, and P. Pardalos, "Global optimization of concave function subject to quadratic constraints: An application in nonlinear bilevel programming," *Ann. Oper. Res.*, vol. 34, pp. 125–147, 1992.
[7] M. S. Bazarar, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming*, 2nd ed. New York: Wiley, 1993.
[8] M. Simaan and J. B. Cruz, Jr., "On the Stackelberg strategy in nonzero-sum games," *J. Optim. Theory Appl.*, vol. 11, no. 5, pp. 535–555, 1973.
[9] H. Von Stackelberg, *The Theory of the Market Economy*. Oxford, U.K.: Oxford Univ. Press, 1952.
[10] P. Marcotte, "Network optimization with continuous parameters," *Transp. Sci.*, vol. 17, pp. 181–197, 1983.
[11] S. Suh and T. Kim, "Solving nonlinear bilevel programming models of equilibrium network design problem: A comparative review," *Ann. Oper. Res.*, vol. 34, pp. 203–218, 1992.
[12] T. Miller, T. Friesz, and R. Robin, "Heuristic algorithms for delivered price spatially competitive network facility location problems," *Ann. Oper. Res.*, vol. 34, pp. 177–202, 1992.
[13] K. Shimizu and E. Aiyoshi, "A new computational method for Syackelberg and min-max problems by use of a penalty method," *IEEE Trans. Autom. Control*, vol. AC-26, no. 2, pp. 460–466, Apr. 1981.
[14] E. Aiyoshi and K. Shimuzu, "A solution method for the static constrained Stackelberg problem via penalty method," *IEEE Trans. Autom. Control*, vol. AC-29, no. 12, pp. 1112–1114, Dec. 1984.
[15] J. V. Outrata, "On the numerical solution of a class of Stackelberg problems," *Zeitschrift Fur Operation Research*, vol. 34, pp. 255–278, 1990.
[16] J. F. Bard, "Convex two-level optimization," *Math. Programming*, vol. 40, pp. 15–27, 1988.
[17] A. O. Allen, *Probability, Statistics, and Queuing Theory with Computer Science Applications*, 2nd ed. New York: Academic, 1990.
[18] V. Oduguwa and R. Roy, "Bi-level optimization using genetic algorithm," in *Proc. IEEE Int. Conf. Artificial Intelligence Systems*, 2002, pp. 123–128.
[19] B.-D. Liu, "Stackelberg–Nash equilibrium for multilevel programming with multiple followers using genetic algorithms," *Comput. Math. Appl.*, vol. 36, no. 7, pp. 79–89, 1998.
[20] H. I. Calvete and C. Gale, "Theory and methodology: The bilevel linear/linear fractional programming problem," *Eur. J. Oper. Res.*, vol. 114, pp. 188–197, 1999.

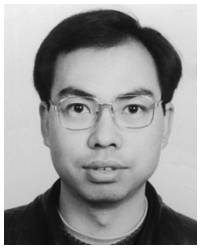
Yuping Wang received the B.Sc. degree in mathematics from Northwest University, Xi'an, China, in 1983, and the Ph.D. degree in computational mathematics from Xi'an Jiaotong University, Xi'an, China, in 1993.

Currently, he is a Professor with the Department of Mathematics Science, Faculty of Science, Xidian University, Xi'an, China. His research interests include evolutionary computation, optimization theory, algorithms, and applications.



Yong-Chang Jiao received the Ph.D. degree in electrical engineering from Xidian University, Xi'an, China, in 1990.

Currently, he is a Full Professor with the Institute of Antennas and Electromagnetic Scattering, Xidian University, where he has been since 1990. He has published many papers in technical journals and conference proceedings. His current research interests include optimization algorithms and applications, evolutionary computation, as well as analysis and synthesis of antennas.



Hong Li received the B.Sc. degree in mathematics from Shaanxi Normal University, Xi'an, China, in 1995, and the M.S. degree in applied mathematics from Xidian University, Xi'an, China, in 2003.

Currently, he is a Lecturer with the Department of Mathematics Science, Faculty of Science, Xidian University. His research interests include evolutionary computation, optimization theory, algorithms, and applications.