Applied Mathematics and Computation 219 (2013) 4332–4339

Contents lists available at SciVerse ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

# Application of particle swarm optimization based on CHKS smoothing function for solving nonlinear bilevel programming problem

Yan Jiang [a,*], Xuyong Li [a], Chongchao Huang [b], Xianing Wu [c]

[a] State Key Laboratory of Urban and Regional Ecology, Research Center for Eco-Environmental Sciences, Chinese Academy of Sciences, Beijing 100085, PR China
[b] School of Mathematics and Statistics, Wuhan University, Wuhan 430072, PR China
[c] Sinohydro Corporation Limited, Beijing 100044, PR China

ARTICLE INFO

*Keywords:*
Nonlinear bilevel programming
Karush–Kuhn–Tucker condition
Smoothing method
Particle swarm optimization

ABSTRACT

Particle Swarm Optimization (PSO) is a new optimization technique originating from artificial life and evolutionary computation. It completes optimization through following the personal best solution of each particle and the global best value of the whole swarm. PSO can be used to solve nonlinear programming problems for global optimal solutions efficiently, so a novel approach based on particle swarm optimization is proposed to solve nonlinear bilevel programming problem (NBLP). In the proposed approach, applying Karush–Kuhn–Tucker (KKT) condition to the lower level problem, we transform the NBLP into a regular nonlinear programming with complementary constraints, which is sequentially smoothed by Chen-Harker-Kanzow-Smale (CHKS) smoothing function. The PSO approach is then applied to solve the smoothed nonlinear programming for getting the approximate optimal solution of the NBLP problem. Simulations on 5 benchmark problems and practical example about watershed water trading decision-making problem are made and the results demonstrate the effectiveness of the proposed method for solving NBLP.

## 1. Introduction

The bilevel programming problem (BLP) is a nested optimizations problem with two levels in a hierarchy, the upper and lower level decision-making. Both of them have their own objective functions and constraints. The upper level maker makes his decision firstly, followed by the lower decision make. The objective function and constraint of the upper level programming rely not only on their own decision variables but also on the optimum solution of the lower level programming. The decision maker at the lower level has to optimize its own objective function under the given parameters from decision maker at the upper level, who, in return, with complete information on the possible reactions of the lower, selects the parameters so as to optimize its own objective function. Unlike the multiple objective mathematical programming techniques, the bilevel mathematical programming emphasizes the non-cooperative character of the system. The application of this hierarchical model can be used widely in such areas as resource allocation, decentralized control, network design problem, etc. [1].

The successful application of this hierarchical model depends on how well it is solved in handling realistic complications. A significant amount of effort have been devoted to solving bilevel mathematical programming and many efficient algorithms have been proposed. To date a few algorithms exist to solve BLP, which can be classified into four types: approach of using the Karush–Kuhn–Tucker (K-K-T) condition [2–6], penalty function approach [7–10], descent approach [11,12] and evolutionary approach [13].

* Corresponding author.
*E-mail address:* lirenjy@sohu.com (Y. Jiang).

0096-3003/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.amc.2012.10.010

Recently, the evolutionary algorithms are widely used to solve different problems in optimal areas and become an alternative for solving bilevel programming for its good characteristics. In 1994, Mathieu etc. [14] firstly developed a genetic algorithm based bilevel programming algorithm. In 1998, Kemal etc. [15] proposed a dual temperature simulated annealing approach for solving bilevel programming problems. In this method, the lower level problem is stochastically relaxed with a parameter that can be used as a temperature scale in simulated annealing. Oduguwa etc. [16] proposed a bilevel genetic algorithm, which is an elitist optimization algorithm developed to encourage limited asymmetric cooperation between the two players, to solve different classes of the bilevel problems within a single framework. Wang etc. [17] proposed an evolutionary algorithm for solving nonlinear bilevel programming problem. A specific optimization problem is constructed with two objectives firstly, which then is solved by a new evolutionary algorithm. By solving the specific problem, they decrease the upper objective value, identify the quality of any feasible solution from infeasible solutions, force the infeasible solutions moving toward the feasible region and improve the feasible solutions gradually.

In modern science and technology, many optimization problems need to be solved in real time, while these classical methods cannot render real-time solutions to these optimization problems, especially large-scale problems. As a new meta-heuristic, particle swarm optimization (PSO) [18,19] has proved to be a competitive algorithm for optimization problems compared with other algorithms such as genetic algorithm (GA) and simulating algorithm (SA). It can converge to the optimal solution rapidly [20,21], and this advantage has been attracting researchers to solve BLP problem using PSO approach. [22,23] proposed a hierarchical particle swarm optimization for solving BLP problem.

In this paper, for a class of nolinear bilevel programming (NBLP) problem, replaced the lower level problem by its Kraush-Kuhn-Tucker optimality conditions, the NBLP problem is reduced into a regular nonlinear programming with complementary constraints. It is then smoothed by CHKS smoothing function. Finally, a particle swarm optimization approach is proposed to solve the smoothed nonlinear programming for getting the approximate optimal solution of the NBLP problem. This paper is organized as follows: Section 2 introduces the formulation and basic definitions of bilevel nonlinear programming, and also introduces the smoothing method for nonlinear complementarity problem. Section 3 introduces a particle swarm optimization for solving the smoothed programming problem. Numerical examples are reported in Section 4. And the conclusion is given in Section 5.

## 2. Nonlinear bilevel programming problem and smoothing method

We consider the nonlinear bilevel programming (NBLP) formulated as follows [24]:

$$\begin{aligned}
&(UP)\ \min_{x} F(x,y),\\
&\text{t. } h(x,y) \leqslant 0,\\
&(LP)\ \min_{y} f(x,y),\\
&\text{s.t. } g(x,y) \leqslant 0.
\end{aligned} \tag{1}$$

where $x \in X \subset R^{n_1},\ y \in Y \subset R^{n_2}.F,\ f: R^{n_1} \times R^{n_2} \to R,\ h: R^{n_1} \times R^{n_2} \to R^{m_1},\ g: R^{n_1} \times R^{n_2} \to R^{m_2}$ are continuous differentiable functions. The term (UP) is called the upper-level problem and the term (LP) is called the lower-level problem and correspondingly the terms $x, y$ are the upper-level variable and the lower-level variable respectively.

The notations are defined as follow:

(a) Constraint region of the NBLP:
$S = \{(x,y) | h(x,y) \leqslant 0, g(x,y) \leqslant 0\}$
(b) Feasible set for the lower level problem for each fixed $x$:
$S(x) = \{y | g(x,y) \leqslant 0\}$
(c) Projection of onto the upper level maker's decision space:
$S(X) = \{x | \exists y, g(x,y) \leqslant 0\}$
(d) Lower level maker's rational reaction set for:
$P(x) = \{y | y \in argmin[f(x,\hat{y}) | \hat{y} \in S(x)]\}$where, $argmin[f(x,\hat{y} | \hat{y} \in S(x))] = \{y \in S(x) | f(x,\hat{y}) \leqslant f(x,y), \hat{y} \in S(x)\}$
(e) Inducible region:
$IR = \{(x,y) | (x,y) \in S, y \in P(x)\}$

Here, in order to ensure that the problem (1) is well posed, $S$ is assumed to be nonempty and compact, and that for each decision taken by the upper decision maker, the lower decision maker has some room to respond, i.e. $S(x) \neq \emptyset$. Then we can reduce the NBLP problem to the one-level programming problem:

$$\begin{aligned}
&\min_{x,y,\lambda} F(x,y),\\
&\text{s.t. } h(x,y) \leqslant 0,\\
&\nabla_y L(x,y,\lambda) = 0,\\
&\lambda^T g(x,y) = 0,\\
&g(x,y) \leqslant 0,\\
&\lambda \geqslant 0.
\end{aligned} \tag{2}$$

where $L(x,y,\lambda) = f(x,y) + \lambda^T g(x,y), \lambda \in R^{m_2}$ are Lagrange multipliers.
Its optimal solution can be defined as follow:

**Definition 1.** A point $(x,y)$ is called feasible if $(x,y) \in IR$.

**Definition 2.** A feasible point $(x^*,y^*)$ is called optimal if $(x^*,y^*) \in IR$ and $F(x^*,y^*) \leqslant F(\bar{x},\bar{y}),\ \forall(\bar{x},\bar{y}) \in IR$.
Consider the nonlinear complementarity problem (NCP for short) in the problem (2):

$$g(x,y) \leqslant 0, \quad \lambda \geqslant 0, \quad \lambda^T g(x,y) = 0. \tag{3}$$

We note that the problem (3) is a non-smooth problem, hence standard methods are not guaranteed to solve such problem [25,26]. In the last few decades, various methods have been developed for solving NCP, where the smoothing-type algorithm is one of the most effective methods for NCP. Fukushima and Pang [27] proposed a smooth method to solve the mathematical programming problem with complementarity constraints. Chen and Ma [28] propose a regularization smoothing Newton method for solving nonlinear complementarity problem with $P_0$-function based on Fischer-Burmeister function with perturbed parameter.

The problem (3) with nonlinear complementarity condition is non-convex and non-differential, and even not satisfies the regularity assumptions. For this reason, we apply smoothing method to solve this problem.

**Definition 3.** The Chen-Mangasarian smoothing function is $\phi : R^2 \to R$ defined by $\phi(a,b) = a + b - \sqrt{(a-b)^2}$. By introducing a smoothing parameter $\epsilon \in R$ into $\phi$, we obtain the Chen-Harker-Kanzow-Smale (CHKS) smoothing function [29–31]: $\phi_\epsilon(a,b) = a + b - \sqrt{(a-b)^2 + 4\epsilon^2}$.

The Chen-Mangasarian smoothing function has the property $\phi(a,b) = 0$ if and only if $a \geqslant 0,\ b \geqslant 0,\ ab = 0$, but it is non-differentiable at $a = b = 0$. But, the CHKS smoothing function has the property $\phi(a,b,\epsilon) = 0$ if and only if $a \geqslant 0,\ b \geqslant 0,\ ab = \frac{\epsilon}{2}$ for $\epsilon \geqslant 0$, and the function is smooth with respect to $a,b$ for $\epsilon \geqslant 0$. Hence, by applying the CHKS smoothing function $\phi(a,b,\epsilon) = a + b - \sqrt{(a-b)^2 + 4\epsilon^2}$, the problem (3) can be approximated by:

$$\begin{cases} \nabla_y L(x,y,\lambda) = 0, \\ \lambda_j - g_j(x,y) - \sqrt{(\lambda_i + g_j(x,y))^2 + 4\epsilon^2} = 0, \quad j = 1,2,\ldots,m_2. \end{cases} \tag{4}$$

Hence, the problem (2) can be transformed as follows:

$$\begin{aligned} &\min_{x,y,\lambda} F(x,y), \\ &\text{s.t. } h(x,y) \leqslant 0, \\ &\nabla_y L(x,y,\lambda) = 0, \\ &\lambda_j - g_j(x,y) - \sqrt{(\lambda_i + g_j(x,y))^2 + 4\epsilon^2} = 0, \quad j = 1,2,\ldots,m_2. \end{aligned} \tag{5}$$

The smoothing factor $\epsilon$ is treated as a variable rather than a parameter in the problem (4) and (5), which avoids the difficulty of not satisfying the regularity assumptions induced by complementarity condition in the problem (2). For a given $\epsilon \geqslant 0$ and $x$ fixed, if one wants to get the optimal solution $y$ of the lower level problem, what he needs to do is to solve the problem (4). If $(x,y)$ satisfies all constraints of the lower level problem, then $f(x,y)$ gets its optimal value at $y$. Thus what we need to do is to solve the following single programming problem to get the optimal solution $x$ and $(x,y)$ is then the approximate optimal solution of $F(x,y)$:

$$\begin{aligned} &\min_{x,y,\lambda} F(x,y), \\ &\text{s.t. } h(x,y) \leqslant 0. \end{aligned} \tag{6}$$

For the above problem, we use PSO to solve it.

## 3. Particle swarm optimization for NLBP

### 3.1. Overview of particle swarm optimization

PSO is a population-based heuristic algorithm that simulates the social behavior as birds flocking to a promising position to achieve precise objectives in a multidimensional space. In PSO, the population is referred as a swarm and individuals are called particles. Like other evolutionary algorithms, PSO performs searches using a population of individuals that are updated from iteration to iteration. To find the optimal or approximately optimal solution, each particle changes its searching direction according to two factors, its own best previous experience and the best experience of the entire swarm in the past.

Each particle is a potential solution to the optimization problem. A particle represents a point in a $D$-dimension space, and its status is characterized through its position and velocity. The position for the particle $i$ at iteration $k$ can be represented by a $D$-dimensional vector $X_i^k = \{x_{i1}^k, x_{i2}^k, \ldots, x_{iD}^k\}$. The velocity of this particle can be represented by another $D$-dimensional vector $V_i^k = \{v_{i1}^k, v_{i2}^k, \ldots, v_{iD}^k\}$. The fitness of each particle can be evaluated using the objective function of optimization problem. The best previous position of the $i$th particle memorized until iteration $k$ represented as $PB_i^k = \{p_{i1}^k, p_{i2}^k, \ldots, p_{iD}^k\}$. The best position in the entire swarm is denoted as $GB^k = \{g_1^k, g_2^k, \ldots, g_D^k\}$.

To search for the optimal solution, each particle changes its velocity and position according to the following two formulas:

$$v_{id}^k = w \cdot v_{id}^{k-1} + c_1 \cdot r_1 \cdot (p_{id}^{k-1} - x_{id}^{k-1}) + c_2 \cdot r_2 \cdot (g_d^{k-1} - x_{id}^{k-1}). \tag{7}$$
$$x_{id}^k = x_{id}^{k-1} + v_{id}^k, \quad d = 1, 2, \ldots, D. \tag{8}$$

where $w$ is called the inertia weight that controls the impact of previous velocity of particle on its current one. $r_1, r_2$ are random numbers between zero and one, $c_1, c_2$ are positive constant parameters called acceleration factors which control the maximum step size.

In PSO, Eq. (7) is used to calculate the new velocity according to its previous velocity and to the distance of its current position from both its own best historical position and the best position of the entire population. Generally, the value of each component in $V$ can be clamped to the range $[-V_{\max}, V_{\max}]$ to control excessive roaming of particles outside the search space. Then the particle flies toward a new position according equation Eq. (8). This process is repeated until a user-defined stopping criterion is reached.

### 3.2. Fitness function

In the problem (6), let $\varepsilon = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_{m_1})^T$, where $\varepsilon_i$ are small positive numbers and tend to zero with the increasing of the generations, $\overline{S} = \{(x, y) \in S | h(x, y) \leqslant \varepsilon\}$, $b(x, y) = max(max(h(x, y) - \varepsilon), 0)$. The fitness function $\overline{F}(x, y)$ is defined as following [32]:

$$\overline{F}(x,y) = \begin{cases} F(x,y) & b(x,y) = 0, \\ F_{max} + b(x,y) & b(x,y) \neq 0. \end{cases} \tag{9}$$

where the parameter $F_{max}$ is the objective function value of the worst feasible solution in the population. If there is no feasible solution in the population, $F_{max}$ returns a value zero. In the process of evolution, infeasible solutions gradually close to feasible region under the pressure of $F_{max} + b(x, y)$, and after they enter into feasible region, they close to optimal solution under the pressure of $F(x, y)$.

### 3.3. Proposed algorithm for NBLP

**Step1:** In this step, the population size $M$ and the maximal iterated generation $T$ are given. Randomly generate a set of initial population in $X$. Set $t = 1$.
**Step2:** For a given $\epsilon \geqslant 0$ and each $x \in X$ fixed, we solve the problem (4) for $y$.
**Step3:** For each $y \in Y$, we solve the problem (6) using PSO for optimal solution $x$ and optimal value $F$. Set $x^* = x,\ y^* = y,\ F^* = F,\ t = t + 1$.
**Step4:** The algorithm terminates when the generation $t$ is greater than the maximal iterated generation $T$. Otherwise, set $t = t + 1$ and goto **Step 2**, then we can get the another solution $(x', y')$ and value $F(x', y')$. If $F' < F^*$, then set $x^* = x',\ y^* = y',\ F^* = F'$.
**Step5:** After $T$ iterations, we will get the approximate optimal solution $(x^*, y^*)$ and approximate optimal value $F(x^*, y^*)$ of NBLP.

## 4. Computational tests

### 4.1. Numerical examples

In this section, 5 benchmark problems were used for simulations to test the feasibility and efficiency of the proposed algorithm. For notational simplicity, denote $x = (x_1, \ldots, x_n)^T$ and $y = (y_1, \ldots, y_m)^T$ in these test problems. The details of these problems are as follows:

(1) Reference [5]: Let $n = 2$ and $m = 2$

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

(2) Reference [9]: Let $n = 2$ and $m = 2$

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

(3) Reference [16]: Let $n = 1$ and $m = 2$

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

(4) Reference [16]: Let $n = 1$ and $m = 1$

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

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

$$\begin{aligned}&\min_{0\leqslant x\leqslant 50} F(x,y) = |\sin(2x_1 + 2x_2 - 3y_1 - 3y_2 - 60)|,\\ &\text{s.t. } x_1 + x_2 + y_1 - 2y_2 \leqslant 40,\\ &\min_{-10\leqslant y\leqslant 20} f(x,y) = 2x_1^2 + y_1^2 - 5y_2,\\ &\text{s.t. } 2y_1 - x_1 + 10 \leqslant 0,\\ &2y_2 - x_2 + 10 \leqslant 0.\end{aligned}$$

Applying KKT condition, firstly we transform these problems into single level programming problems. And then adopting the CHSK smoothing function: $\phi_\epsilon(a,b) = a + b - \sqrt{(a-b)^2 + 4\epsilon^2}$, we approximate these problems into smoothing problems. Finally, we apply PSO to solve the smoothing programming problems.

In the proposed algorithm, the population size is set $M = 40$, and the algorithm stops after 100 generations for these problems. The parameter $\omega$ used is recommended from reference [33] with a linearly decreasing, which changes from 0.9 to 0.4. The acceleration constants $c_1,\ c_2$ are both 2.0. $X_{\max}$ and $V_{\max}$ are set to be equal. We execute the algorithm in 50 independent runs on each of the above 5 benchmark problems on a microcomputer and record the following data:

(1) The best solution $(x^*, y^*)$ obtained in 50 independent runs and the upper objective value $F(x^*, y^*)$ as well as the lower objective value $f(x^*, y^*)$ at the best solution $(x^*, y^*)$.
(2) The worst solution $(\bar{x}, \bar{y})$ obtained in 50 independent runs and the upper objective value $F(\bar{x}, \bar{y})$ as well as the lower objective value $f(\bar{x}, \bar{y})$ at the best solution $(\bar{x}, \bar{y})$.

**Table 1**
Compare of the best and worst results with those of references.

| Prob. | $\epsilon$ | Results obtained by the proposed algorithm | | | | Results in Ref. | |
|---|---|---|---|---|---|---|---|
| | | $F(x^*, y^*)$ | $f(x^*, y^*)$ | $F(\bar{x}, \bar{y})$ | $f(\bar{x}, \bar{y})$ | $F(x,y)$ | $f(x,y)$ |
| | 0.1 | −13.984 | 3.37 | −13.0199 | −0.95 | | |
| 1 | 0.01 | −14 | 4 | −13.987 | 3.43 | −12.68 | −1.016 |
| | 0.001 | −14 | 4 | −14 | 4 | | |
| | 0.1 | 5 | 0 | 5.001 | 0 | | |
| 2 | 0.01 | 5 | 0 | 5 | 0 | 5 | 0 |
| | 0.001 | 5 | 0 | 5 | 0 | | |
| | 0.1 | −1.028 | 7.016 | −0.9996 | 6.932 | | |
| 3 | 0.01 | −1.055 | 7.100 | −1.042 | 7.061 | 3.57 | 2.4 |
| | 0.001 | −1.216 | 7.638 | −1.072 | 7.155 | | |
| | 0.1 | 100.421 | 0.001 | 100.744 | 0.003 | | |
| 4 | 0.01 | 100 | 0 | 100.040 | 0.0001 | 100.58 | 0.001 |
| | 0.001 | 100 | 0 | 100 | 0 | | |
| | 0.1 | 0 | 100 | 0 | 100 | | |
| 5 | 0.01 | 0 | 100 | 0 | 100 | 0 | 100 |
| | 0.001 | 0 | 100 | 0 | 100 | | |

We compare the obtained results with those presented in the corresponding references. All of the results are presented in Table 1. Table 2 summarizes the best and worst solutions found by the proposed algorithm in 50 independent runs. For convenience, Ref. stands for the algorithm in the corresponding references. $F(x,y)$ and $f(x,y)$ are upper and lower objective values at the solution $(x,y)$, respectively.

From these tables, we can see that except for the worst solution of the Problem 2 and 4 for $\epsilon = 0.1$, all of the solutions found by the proposed method are better than or equal to those by the compared algorithms in the references. For the Problem 1 and 3, the solutions found by the proposed method are much better than those by the compared algorithms. We can also see that the solutions found by the algorithms in [1,16] for Problems 1 and 3 are not the global optimal solutions. In a word, the results show that the proposed method ban better solve most test problems than the compared algorithms in the references.

## *4.2. Practical example*

In this section, a watershed water trading decision-making model based on bilevel programming was used to test the feasibility of the proposed algorithm [34]. The upper decision-maker is the watershed management agency as the planning, controlling and coordinating center of watershed and each user is the lower decision-maker. The nonlinear bilevel programming formulated as follows:

$$\begin{aligned}
&\max_{w,t,r_1,g_1,r_2,g_2} F = 0.4w + t(q_1+q_2) + f_1 + f_2,\\
&\text{s.t. } r_1 + r_2 + w = 90,\\
&q_1 + q_2 + w \leqslant 90,\\
&g_1 + g_2 = 20,\\
&r_1 \geqslant 38, \quad r_2 \geqslant 42, \quad g_1 \geqslant 7, \quad g_2 \geqslant 8, \quad w \geqslant 6, \quad 0.3 \leqslant t \leqslant 2.0,\\
&\max_{q_1,l_1} f_1 = 0.7q_1 - q_1 t - 0.3(45-q_1)^2 + (r_1 - q_1)[0.9 - 0.01(r_1 + r_2 - q_1 - q_2)],\\
&-0.2(0.2q_1 - l_1)^2 + (g_1 - l_1)[0.8 - 0.01(g_1 + g_2 - l_1 - l_2)],\\
&\max_{q_2,l_2} f_2 = 0.8q_2 - q_2 t - 0.2(47-q_2)^2 + (r_2 - q_2)[0.9 - 0.01(r_1 + r_2 - q_1 - q_2)],\\
&-0.1(0.3q_2 - l_2)^2 + (g_2 - l_2)[0.8 - 0.01(g_1 + g_2 - l_1 - l_2)],\\
&\text{s.t. } l_1 + l_2 \leqslant 20,\\
&q_1, \quad l_1 \geqslant 0,\\
&q_2, \quad l_2 \geqslant 0.
\end{aligned}$$

where $q_1$ and $q_2$ are actual water intake volume of water consumer A and water consumer B respectively. $l_1$ and $l_2$ are waste water discharge volume of two users respectively. $r_1$ and $r_2$ are water rights of two users respectively. $g_1$ and $g_2$ are emission rights of two users respectively. $w$ is ecological water requirement of watershed. $t$ is water resource fees.

Applying KKT condition and the CHSK smoothing function, we transform the watershed water trading decision-making problem into a single level smoothing programming problem. Then, we apply PSO to solve the programming problem. The parameters are selected as Section 4.1. We execute the algorithm in 10 independent runs and record the best solutions and the upper objective value as well as the lower objective values at the best solutions. Table 3 compares the obtained results

**Table 2**
Compare of the best and worst solutions with those of references.

| Prob. | $\epsilon$ | Solutions obtained by the proposed algorithm | | Solutions in Ref. |
|---|---|---|---|---|
| | | $(x^*, y^*)$ | $(\bar{x}, \bar{y})$ | $(x, y)$ |
| | 0.1 | (0,2,2,0.126) | (0,2,2,0.99) | |
| 1 | 0.01 | (0,2,2,0) | (0,2,2,0.114) | (0,2,1.875,0.9063) |
| | 0.001 | (0,2,2,0) | (0,2,2,0) | |
| | 0.1 | (25,30,5,10) | (24.999,30,4.999,10) | |
| 2 | 0.01 | (25,30,5,10) | (25,30,5,10) | (25,30,5,10) |
| | 0.001 | (25,30,5,10) | (25,30,5,10) | |
| | 0.1 | (1.794,0.965,0) | (1.780,0.976,0) | |
| 3 | 0.01 | (1.807,0.954,0) | (1.801,0.959,0) | (NA) |
| | 0.001 | (1.881,0.885,0) | (1.816,0.974,0) | |
| | 0.1 | (10.021,9.978) | (10.037,9.954) | |
| 4 | 0.01 | (10,10) | (10.002,9.993) | (10.03,9.969) |
| | 0.001 | (10,10) | (10,10) | |
| | 0.1 | (0,0,-10,-10) | (0,0,-10,-10) | |
| 5 | 0.01 | (0,0,-10,-10) | (0,0,-10,-10) | (0,30,-10,10) |
| | 0.001 | (0,0,-10,-10) | (0,0,-10,-10) | |

**Table 3**
Compare of the results with those in reference.

| Parameter | $\epsilon = 0.1$ | $\epsilon = 0.01$ | $\epsilon = 0.001$ | Results in Ref. |
|---|---|---|---|---|
| $q_1$ | 41.5521 | 41.0789 | 42.0000 | 41.2016 |
| $q_2$ | 41.5388 | 42.3652 | 41.9680 | 42.5388 |
| $l_1$ | 6.2456 | 6.5345 | 6.9984 | 6.4772 |
| $l_2$ | 9.7766 | 9.5568 | 9.1751 | 9.1611 |
| $r_1$ | 39.5373 | 40.5868 | 39.9679 | 39.4861 |
| $r_2$ | 43.8375 | 42.8572 | 44.0000 | 44.2542 |
| $g_1$ | 8.7039 | 9.1045 | 9.0000 | 9.0015 |
| $g_2$ | 11.2961 | 10.8955 | 11.0000 | 10.9985 |
| $w$ | 6.6252 | 6.5559 | 6.0321 | 6.2596 |
| $t$ | 0.3000 | 0.3000 | 0.3000 | 0.3226 |
| $F$ | 57.14132 | 57.77584 | 58.97837 | 58.4482 |
| $f_1$ | 12.26299 | 12.76638 | 13.40286 | 10.964 |
| $f_2$ | 17.30098 | 17.35385 | 17.97226 | 17.964 |

with the presented in the reference. From the table we can see that for $\epsilon = 0.001$, the solutions found by the proposed method are better than those given in the references. Therefore, the proposed method can also be applied to the practical problem.

## 5. Conclusions

Bilevel programming problem are intrinsically non-convex and it is difficult to solve for the global optimum solution. In this paper, we present a method based on particle swarm optimization to solve nonlinear bilevel programming problem. In this algorithm, KKT condition is used to the lower level problem to transform the NBLP into a single nonlinear programming problem with complementary constraints. And CHKS smoothing function is adopted to avoid the difficulty of dealing with the non-differentiable because of the complementary condition. A particle swarm optimization approach is then applied to solve the smoothed nonlinear programming. In order to enhance the capability of handling constraints, the proposed algorithm introduces the rule of tournament selection and designs the special fitness function, which makes iterative point close to feasible region gradually under the pressure of tournament selection, and close to global optimum solution finally. The numerical computation and practical test results indicate the proposed algorithm is effective.

## Acknowledgements

This work was supported by the State Key Development Program for Basic Research of China (No. 2009CB421104) and the National Natural Science Foundation of China (Nos. 50809004 & 41071323).

## References

[1] J.F. Bard, Coordination of a multidivisional organization through two levels of management, International Journal of Management Science 11 (1983) 457–468.
[2] J.F. Bard, J.E. Falk, An explicit solution to the multi-level programming problem, Computers & Operations Research 9 (1982) 77–100.
[3] J.F. Bard, An algorithm for solving the general bilevel programming problem, Mathematics of Operations Research 8 (1983) 260–272.
[4] T.A. Edmunds, J.F. Bard, Algorithm for nonlinear bilevel mathematical programs, IEEE Transactions on Systems, Men, and Cybernetics 21 (1991) 83–89.

[5] M.A. Amouzegar, A global optimization method for nonlinear bilevel programming problems, IEEE Transactions on Systems, Men, and Cybernetics 29 (1999) 771–777.
[6] J.B.E. Etoa, Solving quadratic convex bilevel programming problems using a smoothing method, Applied Mathematics and Computation 217 (2011) 6680–6690.
[7] Y. Ishizuka, E. Aiyoshi, A new computational method for Stackelberg and min-max problems by use of a penalty method, IEEE Transactions on Automatic Control 26 (1981) 460–466.
[8] E. Aiyoshi, K. Shimuzu, A solution method for the static constrained Stackelberg problem via penalty method, IEEE Transactions on Automatic Control 29 (1984) 1112–1114.
[9] Y. Ishizuka, E. Aiyoshi, Double penalty method for bilevel optimization problems, Annals of Operations Research 34 (1992) 73–88.
[10] Y.B. Lv, T.S. Hu, G.M. Wang, Z.P. Wan, A penalty function method based on KuhnCTucker condition for solving linear bilevel programming, Applied Mathematics and Computation 188 (2007) 808–813.
[11] G. Savard, J. Gauvin, The steepest descent direction for the nonlinear bilevel programming problem, Operations Research Letters 15 (1994) 265–272.
[12] J.E. Falk, J.M. Liu, On bilevel programming, Part I: general nonlinear cases Mathematical Programming 70 (1995) 47–72.
[13] G.M. Wang, X.J. Wang, Z.P. Wan, Y.B. Lv, A globally convergent algorithm for a class of bilevel nonlinear programming problem, Applied Mathematics and Computation 188 (2007) 166–172.
[14] R. Mathieu, L. Pittard, G. Anandalingam, Genetic algorithm based approach to bi-level linear programming, Operations Research 28 (1994) 1–21.
[15] H.S. Kemal, R.C. Amy, A dual temperature simulated annealing approach for solving bilevel programming problems, Computers and Chemical Engineering 23 (1998) 11–25.
[16] V. Oduguwa, R. Roy, Bi-level optimization using genetic algorithm, IEEE International Conference on Artificial Intelligence Systems (2002) 23–128.
[17] Y.P. Wang, Y.C. Jiao, H. Li, An evolutionary algorithm for solving nonlinear bilevel programming based on a new constraint-handling scheme, IEEE Transactions on Systems Men and Cybernetics 35 (2005) 221–232.
[18] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings Sixth Symposium on Micro Machine and Human Science, IEEE Service Center, Piscataway, NJ, 1995, pp. 39–43.
[19] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, 1995, pp. 1942–1948.
[20] Y. Jiang, T.S. Hu, C.C. Huang, X.N. Wu, An improved particle swarm optimization algorithm, Applied Mathematics and Computation 193 (2007) 231–239.
[21] Y. Jiang, C.M. Liu, C.C. Huang, X.N. Wu, Improved particle swarm algorithm for hydrological parameter optimization, Applied Mathematics and Computation 217 (2010) 3207–3215.
[22] X.Y. Li, P. Tian, X.P. Min, A hierarchical particle swarm optimization for solving bilevel programming problems, LNAI (2006) 169–1178.
[23] R.R. Kuo, C.C. Huang, Application of particle swarm optimization algorithm for solving bi-level linear programming problem, Computers and Mathmatics with Applications 58 (2009) 678–685.
[24] Z.H. GÜMÜS, C.A. Floudas, A shuffled Global optimization of nonlinear bilevel programming problems, Journal of Global Optimization Journal of Global Optimization 20 (2001) 1–21.
[25] Y. Chen, M. Florian, The nonlinear bilevel programming problem: formulations regularity and optimality conditions, Optimization 32 (1995) 193–209.
[26] H. Scheel, S. Scholtes, Mathematical programs with complementarity constraints: stationarity optimality and sensitivity, Mathematics of Operations Research 25 (1995) 1–22.
[27] M. Fukushima, J.S. Pang, Convergence of a smoothing continuation method for mathematical problem with complementarity constraints, Lecture Notes in Economics and Mathematical Systems 477 (1999) 105–116.
[28] X.H. Chen, C.F. Ma, Regularization smoothing Newton method for solving nonlinear complementarity problem, Nonlinear Analysis: Real World Applications 10 (2009) 1702–1711.
[29] B. Chen, P.T. Harker, A non-interior-point continuation method for linear complementarity problem, SIAM Journal on Matrix Analysis and Applications 14 (1993) 1168–1190.
[30] C. Kanzow, Some noninterior continuation methods for linear complementarity problems, SIAM Journal on Matrix Analysis and Applications 17 (1996) 851–868.
[31] S. Smale, Algorithms for solving equations, in: Proceeding of International Congress of Mathematicians, American Mathematics Society, Providence, Rhode Island, 1987, pp. 72–195.
[32] D. Kalyanmoy, An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering 186 (2) (2000) 311–338.
[33] Y. Shi, R. Eberhart, A modified particle swarm optimizer, IEEE International Conference on Evolutionary Computation Anchorage Alaska (1998) 9–73.
[34] C.Y. Wu, Y.Z. Zhao, Watershed water trading decision-making model based on bilevel programming, Operations Research and Management Science (in Chinese) 20 (3) (2011) 30–37.