An Improved Teaching-Learning-Based Optimization with the Social Character of PSO for Global Optimization

Feng Zou, Debao Chen, and Jiangtao Wang

School of Physics and Electronic Information, Huaibei Normal University, Huaibei 235000, China

Correspondence should be addressed to Debao Chen; chendb_8@163.com

Received 24 June 2015; Accepted 18 November 2015

1. Introduction

Global optimization is a concerned research area in science and engineering. Many real-world optimization applications can be formulated as global optimization problems. To efficiently solve the global optimization problems, the efficient and robust optimization algorithms are needed. The traditional methods often fail to solve complex global optimization problems. A detailed overview of the research progress in deterministic global optimization can be found in [1]. To overcome the difficulties of traditional methods, some well-known metaheuristics are developed for solving global optimization problems during the last four decades. Among existing metaheuristics, particle swarm optimization (PSO) algorithm [2] plays very important role in solving global optimization problems. PSO inspired by the social behaviors of birds has been successfully utilized to optimize continuous nonlinear functions [3], but the standard PSO algorithm (SPSO) might trap into local optima when solving complex multimodal problems. To improve the global performance of PSO, some variants are developed. Linear decreasing inertia weight particle swarm optimization (LDWPSO) [4] was introduced by Shi and Eberhart to overcome the lack of velocity control in standard PSO; the inertia weight of the algorithm decreases from a large value to a small one with increasing of evolution. To improve the convergence accuracy of PSO, some modified operators are adopted to help the swarm escape from the local optima especially when the best fitness of the swarm is not changed in some continuous iteration [5–8]. To improve the convergence speed of PSO, some modifications for updating rule of particles were presented in [9], and the improved algorithm was tested on some benchmark functions. Adaptive particle swarm optimization algorithm was introduced by Zhan to improve the performance of PSO, many operators were proposed to help the swarm jump out of the local optima and the algorithms have been evaluated on 12 benchmark functions [10, 11]. Some detail surveys of development to PSO are introduced for the interested readers in [12–14].

Though swarm intelligence optimization algorithms have been successfully used to solve global optimization problems, the main limitation of the previously mentioned algorithms is that many parameters (often more than two parameters) should be determined in updating process of individuals, and the efficiency of algorithms are usually affected by these parameters. For example, there are three parameters ($\omega$, $c_1$, $c_2$).
and $c_2$) that should be determined in updating equations of PSO. Moreover, the optimal parameters of the algorithms are often difficult to be determined. To decrease the effects of parameters for the algorithms, teaching-learning-based (TLBO) algorithm [15] is proposed recently, and it has been used in some real applications [16–19]. Some variants of the TLBO algorithm have been presented for improving the performance of the original TLBO [20, 21].

Under the framework of population-based optimizations, many variations of evolutionary optimization algorithms have been designed. Each of these algorithms performs well in certain cases and none of them are dominating one another. The key reason for employing the hybridization is that the hybrid algorithm can take advantage of the strengths of each individual technique while simultaneously overcoming its main limitations. On top of this idea, many hybrid algorithms have been presented [22–27]. In the paper, to improve the performance of TLBO algorithm for solving global optimization problems, an improved TLBO algorithm with combining of the social character of PSO, named TLBOPSO, is proposed. In the improved TLBO algorithm, the teacher improves not only the performance of the mean grade of the whole class but also the performance of every student. The proposed algorithm has been evaluated on some benchmark functions, and the results are compared with some other algorithms.

The paper is organized as follows. Section 2 provides a brief description of the standard PSO algorithm. Original teaching-learning-based (TLBO) algorithm is introduced in Section 3. Section 4 provides the detail procedure of the improved teaching-learning-based optimization algorithm with combining of the social character of PSO (TLBOPSO). Some experiments are given in Section 5. Section 6 concludes the paper.

2. Particle Swarm Optimizers

In the standard PSO (SPSO) algorithm, a swarm of particles are represented as potential solutions; each particle searches for an optimal solution to the objective function in the search space. The ith particle is associated with two vectors, that is, the velocity vector $V_i = [v^1_i, v^2_i, ..., v^D_i]$ and the position vector $X_i = [x^1_i, x^2_i, ..., x^D_i]$, where $D$ is the dimensions of the variable. Each particle dynamically updates its position in a limited searching domain according to the best position of current iteration and the best position which it has achieved so far. The velocity and position of the $i$th particle are updated as follows:

$$v^d_i = \omega v^d_i + c_1 r_1 (P_{best}^d_i - x^d_i) + c_2 r_2 (G_{best}^d - x^d_i),$$  \hspace{1cm} (1)

$$x^d_i = x^d_i + v^d_i,$$  \hspace{1cm} (2)

where $c_1$ and $c_2$ are the acceleration coefficients, the value of them often equals 2, $r_1$ and $r_2$ are random numbers between 0 and 1. $\omega$ is the inertia weight which influences the convergent character of the swarm. Large inertial weight benefits global searching performance, while small one facilitates local searching. $x^d_i$ is the position of the $i$th particle on the $d$th dimension, $P_{best}^d_i$ is the best position which the $i$th has achieved so far, and $G_{best}^d$ is the best position of the swarm in current iteration. In general, the velocity of all the particles is limited by the maximum velocity ($V_{max}$), and positions of them are limited by the maximum position ($P_{max}$) and the minimum position ($P_{min}$). To improve the performance of PSO, the method LDWPSO in which the inertia weight decreases linearly from a relatively large value to a small one is proposed [4], and the changed weight with evolution iteration is shown as follows:

$$w = w_{max} - (w_{max} - w_{min}) \frac{t}{max \text{gen}},$$  \hspace{1cm} (3)

where $w_{max} = 0.9$, $w_{min} = 0.4$ are the maximum and minimum values of inertia weight, respectively, $g$ is the current generation, $max \text{gen}$ is the maximum evolutionary iteration. The adaptive weights make swarm have good global searching ability at the beginning of iterations and good local searching ability near the end of runs. Algorithm 1 presents the detail steps of LDWPSO algorithm.

3. Teaching-Learning-Based Optimization (TLBO) Algorithm

TLBO is one of the recently proposed population-based algorithms and it simulates the teaching-learning process of the class [15]. The main idea of the algorithm is based on the influence of a teacher on the output of learners in a class and the interaction between the learners. The TLBO algorithm does not require any specific parameters. TLBO requires only common controlling parameters like population size and number of generations for its working. The use of teaching in the algorithm is to improve the average grades of the class. The algorithm contains two phases: teacher phase and learner phase. The detail description of the algorithm can be found in [15]. In this paper, only the two main phases of TLBO are introduced as follows.

3.1. Teacher Phase. In the teacher phase of TLBO algorithm, the task of the teacher is to increase the mean grades of the class. Suppose that an objective function is $f(X)$ with $n$-dimensional variables, the $i$th student can be represented as $X_i = [x_{i1}, x_{i2}, ..., x_{in}]$. At any iteration $g$, assume that the population size of the class is $m$, the mean result of students in current iteration is $X_{gmean} = (1/m)[\sum_{i=1}^{m} x_{i1}, \sum_{i=1}^{m} x_{i2}, \ldots, \sum_{i=1}^{m} x_{in}]$. The student with the best fitness is chosen as the teacher of current iteration; it is represented as $X_{teacher}$. All the students will update their position as follows:

$$X_{i,new} = X_{i,old} + r_1 (X_{teacher} - T_F X_{gmean}),$$  \hspace{1cm} (4)

where $X_{i,new}$ and $X_{i,old}$ are the new and the old position of the $i$th student and $r_1$ is the random number in the range $[0, 1]$. If the new solution is better than the old one, the old position of individual will be replaced by the new position. The value of $T_F$ is randomly decided by the algorithm according to

$$T_F = \text{round} [1 + \text{rand}(0, 1) \{2 − 1\}] .$$  \hspace{1cm} (5)
Initialize:
Initialize maxgen, w_{max}, w_{min}, V_{max}, P_{max}, P_{min}. Population size (Ps), Dimension size (D) and the initial swarm

Optimize:
for gen = 1 : maxgen
    Calculate the inertia weight \( w \) according to (3)
    for \( i = 1 : Ps \)
        Calculate fitness of all particles;
        Calculate the best position \( G_{best}^d \) of current generation;
        Calculate the best position \( P_{best}^d \) of \( i \)th particle which it has achieved so far;
        for \( d = 1 : D \)
            Update the velocity of \( i \)th particle according to (1)
            If \( v_i^d > V_{max} \) then \( v_i^d = V_{max} \)
            If \( v_i^d < -V_{max} \) then \( v_i^d = -V_{max} \)
            Update the position of \( i \)th particle according to (2)
            If \( X_i^d > P_{max} \) then \( X_i^d = P_{max} \)
            If \( X_i^d < P_{min} \) then \( X_i^d = P_{min} \)
        end
    end
end

Algorithm 1: LDWPSO algorithm.

3.2. Learner Phase. In learner phase of TLBO algorithm, learners can increase their knowledge from others. A learner interacts randomly with other learners for enhancing his or her knowledge. The learning of learner phase can be expressed as follows. For the \( i \)th individual \( X_i \) in the \( t \)th generation, randomly select the \( k \)th individual \( X_k \), which is different from \( X_j \), and the updated formula of \( X_i \) is defined in (6) and (7).

If \( X_i \) is better than \( X_k \) according to their fitness, then

\[
X_{i,\text{new}} = X_{i,\text{old}} + r_j (X_i - X_k), \tag{6}
\]

else

\[
X_{i,\text{new}} = X_{i,\text{old}} + r_j (X_k - X_i), \tag{7}
\]

where \( r_j \) is the random number in the range [0, 1].

If the new position \( X_{i,\text{new}} \) is better than the old one \( X_{i,\text{old}} \), the old position \( X_{i,\text{old}} \) is replaced by the new \( X_{i,\text{new}} \); otherwise, the position of the \( i \)th individual is not changed. The detail algorithm is shown in Algorithm 2.

4. Teaching-Learning-Based Optimization Algorithm with PSO (TLBO-PSO)

4.1. The Main Idea of TLBO-PSO. As previously reviewed, the main idea of TLBO is to imitate the teaching-learning process in a class. The teacher tries to disseminate knowledge among the learners to increase the knowledge level of the whole class, and the learners also study knowledge from the others to improve its grade. Algorithm displays that all individuals update their positions based on the distance between one or two times of the mean solution and the teacher in teacher phase. An individual also renewed its position based on the distance between it and a randomly selected individual from the class. Equation (4) indicates that the teacher only improves the grades of the students by using the mean grade of the class; the distance between the teacher and the students is not considered in teacher phase. In PSO, the difference between the current best individual and the individual can help the individual improve its performance. Based on this idea, the method in PSO is introduced into TLBO to improve the learning efficiency of the TLBO algorithm. The main change for TLBO is represented in updating equation. Equation (4) in TLBO is modified as

\[
X_{i,\text{new}} = X_{i,\text{old}} + r_1 (X_{\text{teacher}} - T_F X_{g,\text{mean}}) + r_2 (X_{\text{teacher}} - X_{i,\text{old}}),
\]

where \( r_1, r_2 \) are the random number in the range [0, 1]. With this modification, the performance of TLBO algorithm might be improved.

In original TLBO algorithm, all genes of each individual should be compared with those of all other individuals for removing the duplicate individual. This operator is a heavy computation cost process, and the function evaluations required are not clearly known. In our opinion, duplication testing is not needed in every generation especially in the beginning of evolution. Assume that individuals \( i \) and \( k \) have the same genes in \( t \) generation, and the new position of these two individuals might be different when \( T_F \) is a random number (1 or 2). At the beginning of evolution, all individuals generate better positions easily. Random operator for \( T_F \) may benefit for maintaining diversity of class, but, in the anaphase of evolution, the positions of individuals might be close to each other. When the mean solution equals the best solution, the individual does not change (\( T_F = 1 \)) or the large change of genes might destroy the individual in large degree (\( T_F = 2 \)) so that the better individual is difficult to be generated. To decrease the computational effort of comparing all the individuals, the removing of the duplicate individual process in the original TLBO is deleted in the improved
Initialize $N$ (number of learners) and $D$ (number of dimensions)
Initialize learners $X$ and evaluate all learners $X$
Donate the best learner as $X_{\text{teacher}}$ and the mean of all learners $X$ as $X_{\text{gmean}}$

while (stopping condition not met)
    for each learner $X_i$ of the class % Teaching phase
        $T_F = \text{round}(1 + \text{rand}(0, 1))$
        $X_{i, \text{new}} = X_{i, \text{old}} + r(X_{\text{teacher}} - T_F X_{\text{gmean}})$
        Accept $X_{i, \text{new}}$ if $f(X_{i, \text{new}})$ is better than $f(X_{i, \text{old}})$
    endfor
    for each learner $X_i$ of the class % Learning phase
        Randomly select one learner $X_k$, such that $i \neq k$
        if $f(X_i)$ better $f(X_k)$
            $X_{i, \text{new}} = X_{i, \text{old}} + r(X_i - X_k)$
        else
            $X_{i, \text{new}} = X_{i, \text{old}} + r(X_k - X_i)$
        endif
        Accept $X_{i, \text{new}}$ if $f(X_{i, \text{new}})$ is better than $f(X_{i, \text{old}})$
    endfor
    Update $X_{\text{teacher}}$ and $X_{\text{gmean}}$
endwhile

Algorithm 2: TLBO().

Algorithm 3: Subroutine for mutation.

TLBO algorithm and a mutation operator according to the best fitness of successive generations is introduced in the paper. If the best fitness of continuous $n$ generations is not changed or changed slightly, an individual will be randomly selected according to mutation possibility $p_c$ to be mutated. To maintain the global performance of the algorithm, the best individual is not mutated. The subroutine for the mutation is described as shown in Algorithm 3.

In Algorithm 3, $n$ is the setting generation, $p_c$ is the mutation possibility, and $\epsilon$ is a small number which is given by the designer. $\alpha$ is a mutation parameter between 0.01 and 0.1.

4.2. The Steps of TLBO-PSO

Step 1. Set the maximum $X_{\text{max}}$ and minimum $X_{\text{min}}$ of position, the maximal evolution generation $\text{genmax}$, mutation possibility $p_m$ and mutation parameter $\alpha$, the population size popsize, and the dimension size of the task. Initialize the initial population pop as follows:

$$pop = X_{\text{min}} + r \cdot (X_{\text{max}} - X_{\text{min}}), \tag{9}$$

where $r$ is the random number in the range [0, 1].

Step 2. Evaluate the individual, select the best individual $X_{\text{teacher}}$ as the teacher, and calculate the mean solution $X_{\text{gmean}}$ of the population.

Step 3. For each individual, update its position according to (8). If $X_{i, \text{new}}$ is better than $X_{i, \text{old}}$, then $X_{i, \text{old}} = X_{i, \text{new}}$.

Step 4. For each individual, randomly select another individual, update its position according to (6) and (7), and choose
Table 1: Nine tested functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Formula</th>
<th>Range</th>
<th>( f_{\text{min}} )</th>
<th>Acceptance</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 ) (Sphere)</td>
<td>( f_1 \left( x \right) = \sum_{i=1}^{D} x_i^2 )</td>
<td>([-100,100])</td>
<td>0</td>
<td>1E-6</td>
</tr>
<tr>
<td>( f_2 ) (Quadric)</td>
<td>( f_2 \left( x \right) = \sum_{i=1}^{D} \left( \sum_{j=1}^{D} x_j \right)^2 )</td>
<td>([-100,100])</td>
<td>0</td>
<td>1E-6</td>
</tr>
<tr>
<td>( f_3 ) (Sum Square)</td>
<td>( f_3 \left( x \right) = \sum_{i=1}^{D} i x_i^2 )</td>
<td>([-100,100])</td>
<td>0</td>
<td>1E-6</td>
</tr>
<tr>
<td>( f_4 ) (Zakharov)</td>
<td>( f_4 \left( x \right) = \sum_{i=1}^{D} x_i^2 + \left( \sum_{i=1}^{D} 0.5 i x_i \right)^2 + \left( \sum_{i=1}^{D} 0.5 i x_i \right)^4 )</td>
<td>([-10,10])</td>
<td>0</td>
<td>1E-6</td>
</tr>
<tr>
<td>( f_5 ) (Rosenbrock)</td>
<td>( f_5 \left( x \right) = \sum_{i=1}^{D-1} \left[ 100 \left( x_{i+1}^2 - x_i \right)^2 + (x_i - 1)^2 \right] )</td>
<td>([-2.048,2.048])</td>
<td>0</td>
<td>1E-6</td>
</tr>
<tr>
<td>( f_6 ) (Ackley)</td>
<td>( f_6 \left( x \right) = 20 - 20 \exp \left( -\frac{1}{5} \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2} \right) - \exp \left( \frac{1}{D} \sum_{i=1}^{D} \cos \left( 2\pi x_i \right) \right) + e )</td>
<td>([-32.768,32.768])</td>
<td>0</td>
<td>1E-6</td>
</tr>
<tr>
<td>( f_7 ) (Rastrigin)</td>
<td>( f_7 \left( x \right) = \sum_{i=1}^{D} \left( x_i^2 - 10 \cos \left( 2\pi x_i \right) \right) + 10 )</td>
<td>([-5.12,5.12])</td>
<td>0</td>
<td>1E-6</td>
</tr>
<tr>
<td>( f_8 ) (Weierstrass)</td>
<td>( f_8 \left( x \right) = \sum_{i=1}^{D} \left( \sum_{k=0}^{k_{\text{max}}} \left( a^k \cos \left( 2\pi b^k \left( x_i + 0.5 \right) \right) \right) \right) - D \sum_{k=0}^{\infty} \left( a^k \cos \left( 2\pi b^k \times 0.5 \right) \right) )</td>
<td>([-0.5,0.5])</td>
<td>0</td>
<td>1E-1</td>
</tr>
<tr>
<td>( f_9 ) (Griewank)</td>
<td>( f_9 \left( x \right) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos \left( \frac{x_i}{\sqrt{i}} \right) + 1 )</td>
<td>([-600,600])</td>
<td>0</td>
<td>1E-1</td>
</tr>
</tbody>
</table>

the better solution from \( X_{i,\text{old}} \) and \( X_{i,\text{new}} \) as the new position of the individual.

Step 5. Execute mutation operator for the population according to Algorithm 3.

Step 6. If the ended condition of TLBO-PSO is not satisfied, the algorithm will go back to Step 2, or it is terminated.

5. Simulation Experiments

To test the performance of the improved TLBO-PSO algorithm, nine benchmark functions are simulated in this section. These nine benchmark functions are listed in Table 1. In particular, in order to compare the performance of the proposed TLBO-PSO with variants of TLBO and PSO, CLPSO [22], TLBO [15], and ETLBO [21] are selected and simulated.

5.1. Parameters Setting. To reduce statistical errors, each function in the paper is independently simulated 30 runs, and their mean results are used in the comparison. The value of function is defined as the fitness function. All the experiments are carried out on the same machine with a Celeron 2.26 GHz CPU, 512-MB memory, and Windows XP operating system with Matlab 7.0. All functions are simulated in 10 and 30 dimensions. The nine functions are summarized in Table 1. “Range” is the lower and upper bounds of the variables. “\( f_{\text{min}} \)” is the theory global minimum solution. “Acceptance” is the acceptable solutions of different functions. For CLPSO, TLBO, and ETLBO algorithms, the training parameters are the same as those used in the corresponding references except that the maximal FEs are 50000 and the size of population is 30. The size of elitism is 2 in ETLBO algorithm. When the best fitness is changed to be smaller than \( \epsilon \), the large value of mutation will possibly be chosen. In our experiments, the mutation possibility \( p_c \) is 0.6, \( \epsilon \) is 0.01, and \( \alpha \) is 0.05.

5.2. Comparisons on the Solution Accuracy. The performance of different algorithms for 10- and 30-dimensional functions in terms of the best solution, the mean (Mean), and standard deviation (STD) of the solutions obtained in the 30 independent runs is listed in Tables 2 and 3. Boldface in the tables indicates the best solution among those obtained by all four contenders.

The results for 10-dimensional functions in Table 2 display that the improved TLBO (TLBO-PSO) outperforms all the other algorithms in terms of the mean best solutions and standard deviations (STD) for functions \( f_1, f_2, f_3, \) and \( f_4 \). For function \( f_5 \), ETLBO has the best performance. For function \( f_6 \), the mean best solution of TLBO-PSO is the best, and the STD of TLBO is the smallest. For functions \( f_7 \) and \( f_9 \), CLPSO has the best performance. For function \( f_8 \), three TLBOs have the same solutions in terms of the mean best solutions and standard deviations (STD).

The results for 30-dimensional functions in Table 3 indicate that TLBO-PSO also has the best performance in terms of the mean best solutions and standard deviations (STD) for functions \( f_1, f_2, f_3, \) and \( f_4 \) with 30 dimensions. TLBO has the best performance for function \( f_5 \). For function \( f_6 \), TLBO and ETLBO have the same performance in terms of the mean best solutions and standard deviations (STD). For function \( f_7 \), the mean best solution of TLBO is the smallest, and the standard
deviation of TLBO-PSO is the smallest. Three TLBOs have the same solutions for functions $f_8$ and $f_9$.

5.3. Comparisons on the Convergence Speed and Successful Ratios. The mean number of function evaluations (FEs) is often used to measure the convergence speed of algorithms. In this paper, the mean value of FEs is used to measure the speed of all algorithms. The average FEs (when the algorithm is globally convergent) of all algorithms for nine functions with 10 and 30 dimensions are shown in Tables 4 and 5. Boldface in the tables indicates the best result among those obtained by all algorithms. When the algorithm is not globally convergent in all 30 runs, the mFEs is represented as “NaN.” The successful ratios of different algorithms for the nine functions are also shown in the tables.

Tables 4 and 5 display that mFEs of TLBO-PSO are the smallest for large part of functions except that for function $f_5$. The merit in terms of mean FEs for 10-dimensional function $f_1$ is globally convergent) of all algorithms for nine functions with 10 and 30 dimensions are shown in Tables 4 and 5. Boldface in the tables indicates the best result among those obtained by all algorithms. When the algorithm is not globally convergent in all 30 runs, the mFEs is represented as “NaN.” The successful ratios of different algorithms for the nine functions are also shown in the tables.

### Table 2: The mean best solutions and standard deviations by various methods for 10D functions.

<table>
<thead>
<tr>
<th>Methods</th>
<th>CLPSO</th>
<th>TLBO</th>
<th>ETLBO</th>
<th>TLBO-PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>STD</td>
<td>Mean</td>
<td>STD</td>
</tr>
<tr>
<td>$f_1$</td>
<td>4.68E-19</td>
<td>4.90E-19</td>
<td>2.12E-185</td>
<td>0.00E+00</td>
</tr>
<tr>
<td>$f_2$</td>
<td>6.73E-01</td>
<td>4.40E-01</td>
<td>5.79E-80</td>
<td>1.71E-79</td>
</tr>
<tr>
<td>$f_3$</td>
<td>1.14E-20</td>
<td>8.02E-21</td>
<td>1.02E-185</td>
<td>0.00E+00</td>
</tr>
<tr>
<td>$f_4$</td>
<td>3.32E-03</td>
<td>3.25E-03</td>
<td>4.44E-87</td>
<td>1.36E-86</td>
</tr>
<tr>
<td>$f_5$</td>
<td>2.25E+00</td>
<td>1.00E+00</td>
<td>6.54E-01</td>
<td>6.56E-01</td>
</tr>
<tr>
<td>$f_6$</td>
<td>3.75E-10</td>
<td>2.83E-10</td>
<td>3.55E-15</td>
<td>0.00E+00</td>
</tr>
<tr>
<td>$f_7$</td>
<td>1.63E-09</td>
<td>2.42E-09</td>
<td>2.17E+00</td>
<td>1.86E+00</td>
</tr>
<tr>
<td>$f_8$</td>
<td>1.16E-11</td>
<td>1.55E-11</td>
<td>0.00E+00</td>
<td>0.00E+00</td>
</tr>
<tr>
<td>$f_9$</td>
<td>1.06E-03</td>
<td>2.29E-03</td>
<td>9.96E-03</td>
<td>1.02E-02</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>CLPSO</th>
<th>TLBO</th>
<th>ETLBO</th>
<th>TLBO-PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>STD</td>
<td>Mean</td>
<td>STD</td>
</tr>
<tr>
<td>$f_1$</td>
<td>1.50E-03</td>
<td>5.04E-04</td>
<td>1.63E-152</td>
<td>2.38E-152</td>
</tr>
<tr>
<td>$f_2$</td>
<td>9.37E+03</td>
<td>2.23E+03</td>
<td>1.78E-33</td>
<td>3.34E-33</td>
</tr>
<tr>
<td>$f_3$</td>
<td>1.39E-04</td>
<td>4.19E-05</td>
<td>1.45E-152</td>
<td>2.79E-152</td>
</tr>
<tr>
<td>$f_4$</td>
<td>1.45E+02</td>
<td>4.10E+01</td>
<td>5.72E-18</td>
<td>7.60E-18</td>
</tr>
<tr>
<td>$f_5$</td>
<td>3.60E+01</td>
<td>1.03E+01</td>
<td>2.42E+01</td>
<td>6.48E-01</td>
</tr>
<tr>
<td>$f_6$</td>
<td>3.19E-02</td>
<td>1.27E-02</td>
<td>3.55E-15</td>
<td>0.00E+00</td>
</tr>
<tr>
<td>$f_7$</td>
<td>9.88E+00</td>
<td>2.48E+00</td>
<td>1.31E+01</td>
<td>5.78E+00</td>
</tr>
<tr>
<td>$f_8$</td>
<td>6.53E-02</td>
<td>1.08E-02</td>
<td>0.00E+00</td>
<td>0.00E+00</td>
</tr>
<tr>
<td>$f_9$</td>
<td>8.76E-03</td>
<td>3.43E-03</td>
<td>0.00E+00</td>
<td>0.00E+00</td>
</tr>
</tbody>
</table>

### Table 3: The mean best solutions and standard deviations by various methods for 30D functions.

### Table 4: The mean FEs needed to reach an acceptable solution and reliability “ratio” being the percentage of trial runs reaching acceptable solutions for 10-dimensional functions.
Figure 1: Continued.
Figure 1: Convergence performance of the four different methods for 30-dimensional functions.

Table 5: The mean FEs needed to reach an acceptable solution and reliability "ratio" being the percentage of trial runs reaching acceptable solutions for 30-dimensional functions.

<table>
<thead>
<tr>
<th>Methods</th>
<th>CLPSO mFEs</th>
<th>CLPSO Ratios</th>
<th>TLBO mFEs</th>
<th>TLBO Ratios</th>
<th>ETLBO mFEs</th>
<th>ETLBO Ratios</th>
<th>TLBO-PSO mFEs</th>
<th>TLBO-PSO Ratios</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>0.0%</td>
<td>3502.8</td>
<td>100.0%</td>
<td>4039.5</td>
<td>100.0%</td>
<td>2327.9</td>
<td>100.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>0.0%</td>
<td>13845.3</td>
<td>100.0%</td>
<td>13699.8</td>
<td>100.0%</td>
<td>6802.7</td>
<td>100.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>0.0%</td>
<td>3214.1</td>
<td>100.0%</td>
<td>3716.9</td>
<td>100.0%</td>
<td>2093</td>
<td>100.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>0.0%</td>
<td>26934.9</td>
<td>100.0%</td>
<td>27876.2</td>
<td>100.0%</td>
<td>14067.3</td>
<td>100.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>( f_5 )</td>
<td>0.0%</td>
<td>NaN</td>
<td>0.0%</td>
<td>NaN</td>
<td>0.0%</td>
<td>NaN</td>
<td>0.0%</td>
<td>NaN</td>
</tr>
<tr>
<td>( f_6 )</td>
<td>0.0%</td>
<td>5039.1</td>
<td>100.0%</td>
<td>5744.6</td>
<td>100.0%</td>
<td>3305.4</td>
<td>100.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>( f_7 )</td>
<td>0.0%</td>
<td>NaN</td>
<td>0.0%</td>
<td>NaN</td>
<td>0.0%</td>
<td>4629</td>
<td>10.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>( f_8 )</td>
<td>0.0%</td>
<td>7188.8</td>
<td>100.0%</td>
<td>8125.9</td>
<td>100.0%</td>
<td>4315.3</td>
<td>100.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>( f_9 )</td>
<td>100.0%</td>
<td>2046.8</td>
<td>100.0%</td>
<td>2361</td>
<td>100.0%</td>
<td>1492.5</td>
<td>100.0%</td>
<td>100.0%</td>
</tr>
</tbody>
</table>
CLPSO. For function $f_7$, CLPSO can converge to optimia with 100% successful ratios for 10-dimensional function. They all cannot converge to 30-dimensional function except that the successful ratio of TLBO-PSO is 10%. The convergence speed of average best fitness of four algorithms is shown in Figure 1. The figures indicate that the TLBO-PSO has the best performance for large part of functions. According to the theorem of “no free lunch” [28], one algorithm cannot offer better performance than all the others on every aspect or on every kind of problem. This is also observed in our experimental results. For example, the merit of TLBO-PSO is worse than those of TLBO and ETLBO for 10-dimension function but it is better than large part of algorithms for other functions.

6. Conclusions

An improved TLBO-PSO algorithm which is considering the difference between the solutions of the best individual and the individual that want to be renewed is designed in the paper. The mutation operator is introduced to improve the global convergence performance the algorithm. The performance of TLBO-PSO is improved for larger part of functions especially for 30-dimension functions in terms of convergence accuracy and mFEs. The local convergence of TLBO-PSO is major caused by the lost diversity in the later stage of the evolution.

Further works include researches into adaptive selection of parameters to make the algorithm more efficient. Moreover, it needs to seek a better method to improve the TLBO algorithm for functions with optimal parameters displaced from all zeroes. Furthermore, the algorithm may be applied to constrained, dynamic optimization domain. It is expected that TLBO-PSO will be used in real-world optimization problems.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is partially supported by National Natural Science Foundations of China under Grants 61304082 and 61572224 and the National Science Fund for Distinguished Young Scholars under Grant 61425009, and it is partially supported by the Major Project of Natural Science Research in Anhui Province under Grant KJ2015ZD36.

References


