Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 326431, 12 pages
http://dx.doi.org/10.1155/2015/326431

Research Article

An Improved Quantum-Behaved Particle Swarm Optimization Algorithm with Elitist Breeding for Unconstrained Optimization

Zhen-Lun Yang,[1,2] Angus Wu,[3] and Hua-Qing Min[1]

[1] *School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China*
[2] *School of Information Engineering, Guangzhou Panyu Polytechnic, Guangzhou 511483, China*
[3] *Department of Electronic Engineering, City University of Hong Kong, Tat Chee Ave, Hong Kong*

Correspondence should be addressed to Hua-Qing Min; hqmin2013@yeah.net

Received 12 December 2014; Accepted 15 April 2015

Academic Editor: Thomas DeMarse

An improved quantum-behaved particle swarm optimization with elitist breeding (EB-QPSO) for unconstrained optimization is presented and empirically studied in this paper. In EB-QPSO, the novel elitist breeding strategy acts on the elitists of the swarm to escape from the likely local optima and guide the swarm to perform more efficient search. During the iterative optimization process of EB-QPSO, when criteria met, the personal best of each particle and the global best of the swarm are used to generate new diverse individuals through the transposon operators. The new generated individuals with better fitness are selected to be the new personal best particles and global best particle to guide the swarm for further solution exploration. A comprehensive simulation study is conducted on a set of twelve benchmark functions. Compared with five state-of-the-art quantum-behaved particle swarm optimization algorithms, the proposed EB-QPSO performs more competitively in all of the benchmark functions in terms of better global search capability and faster convergence rate.

1. Introduction

Particle swarm optimization (PSO), inspired by the social behavior of bird flocks [1], is an important and widely used population-based stochastic algorithm. Unlike evolutionary algorithms, PSO is computationally inexpensive and its implementation is straightforward. Each potential solution in PSO, represented by a particle, flies in a multidimensional search space with a velocity dynamically adjusted by the particle's own former information and the experience of the other particles. For its superiority, PSO has rapidly developed with applications in solving real-world optimization problems in recent years [2–5].

However, as demonstrated by van den Bergh [6], PSO is not a guaranteed global convergence algorithm according to the convergence criteria in [7]. Based on quantum mechanics and trajectory analysis of PSO [8], Sun et al. [9] proposed a variant of PSO, quantum-behaved PSO (QPSO) algorithm, which is theoretically proved to be global convergent using Markov process [10, 11]. The global convergence of QPSO guarantees to find the global optimal solution upon unlimited number of search iterations. Nevertheless, such condition is unrealistic when it comes to the real-world problems as only a finite number of iterations are allowed for the search of optimal solution on using any optimization algorithm. Thus, QPSO is also likely to be trapped in local optima or with slow convergence speed when it is used to solve complex problems. So far, many researchers developed various strategies to improve performance of QPSO in terms of convergence speed and global optimality [12–21]. However, it is rather difficult to improve the global search capability and accelerate the rate of convergence simultaneously. If any attempt focuses on avoiding being stuck at local optima, it is likely to have a slower convergence rate.

In PSO, the personal best (*pbest*) of each particle and global best (*gbest*) of the swarm found so far in the search process can be considered as the elitists of the whole swarm at any search iteration. In most of the current QPSOs, the information of elitists is either used directly or with some simple extra processing to guide the flying behavior of each particle in the search space; to the best of our knowledge, deep exploration with the elitist is not taken into account to assist the search of solutions in any work of QPSO reported in literatures. The exploration on the elitists will produce some extra information that may be beneficial for the search of the optimal solution.

In this study, a novel variant of the QPSO algorithm, called the quantum-behaved particle swarm optimization with elitist breeding (EB-QPSO), is proposed for the desirable aims to achieve better global search capability and convergence rate by employing a breeding scheme through transposon operators on elitists of the swarm. Elitist breeding is a kind of advanced elitist exploration method treating the elitist members as parents to create new diverse individuals with transposon operators. On one hand, elitist breeding helps to diversify the particle swarm during the search and thus enhance the global search capability of the algorithm. On the other hand, the new bred individuals with better fitness are selected as the new members of elitists and used to guide the swarm to perform exploration and exploitation more efficiently. Experiment results on twelve benchmark functions show that EB-QPSO outperforms the original QPSO and other four state-of-the-art QPSO variants.

The rest of this paper is organized as follows. A brief introduction of QPSO is presented in Section 2. In Section 3, an overview of related work is given. The proposed EB-QPSO algorithm is elaborated and compared with various existing QPSO algorithms over twelve benchmark functions in Sections 4 and 5, respectively. Finally, the general conclusions of the paper are given in Section 6.

2. Quantum-Behaved Particle Swarm Optimization

In the original PSO, each particle is defined by a position vector $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ which signifies a solution in the search space and associated with a velocity vector $\mathbf{v} = (v_1, v_2, \ldots, v_D)$ responsible for the exploration of the search space. Let N denote the swarm size and D the dimensionality of the search space, during the evolutionary process, the velocity and the position of each particle are updated with the following rules:

$$\begin{aligned} v_{i,d}^{t+1} &= \omega v_{i,d}^{t} + c_1 r_{i,d}^{t}\left(pbest_{i,d}^{t} - x_{i,d}^{t}\right) \\ &\quad + c_2 R_{i,d}^{t}\left(gbest_{d}^{t} - x_{i,d}^{t}\right), \\ x_{i,d}^{t+1} &= x_{i,d}^{t} + v_{i,d}^{t+1}, \end{aligned} \tag{1}$$

where i $(1 \le i \le N)$ and d $(1 \le d \le D)$, $v_{i,d}^t$ and $x_{i,d}^t$ are the dth dimension component of velocity and position of particle i in search iteration t, respectively, $pbest_{i,d}^t$ and $gbest_d^t$ are the dth dimension of the personal best of particle i and the global best of the swarm in search iteration t, respectively, ω is the inertia weight, c_1 and c_2 are two positive constant acceleration coefficients, and $r_{i,d}^t$ and $R_{i,d}^t$ are two random numbers uniformly distributed in the interval $(0, 1)$.

According to the trajectory analysis given by Clerc and Kennedy [8], the convergence of the PSO algorithm may be achieved if each particle converges to its local attractor $p_i = (p_{i,1}, p_{i,2}, \ldots, p_{i,D})$, of which the coordinates are defined as

$$p_{i,d}^t = \varphi_d^t \times pbest_{i,d}^t + \left(1 - \varphi_d^t\right) \times gbest_d^t, \tag{2}$$

where $\varphi_d^t = c_1 r_{i,d}^t / (c_1 r_{i,d}^t + c_2 R_{i,d}^t)$.

The concept of the QPSO was developed based on the analysis above. Each single particle in QPSO is treated as a spin-less one moving in quantum space and the probability of the particle's appearing at position x_i^t in the search iteration t is determined from a probability density function [22]. Employing the Monte Carlo method, each particle flies with the following rules:

$$\begin{aligned} x_{i,d}^{t+1} &= p_{i,d}^t + \alpha \left| x_{i,d}^t - mbest_d^t \right| \ln\left(\frac{1}{u_{i,d}^t}\right), \\ &\qquad\qquad \text{if rand}v \ge 0.5, \\ x_{i,d}^{t+1} &= p_{i,d}^t - \alpha \left| x_{i,d}^t - mbest_d^t \right| \ln\left(\frac{1}{u_{i,d}^t}\right), \\ &\qquad\qquad \text{if rand}v < 0.5, \end{aligned} \tag{3}$$

where α is a parameter called contraction-expansion coefficient; both $u_{i,d}^t$ and rand$v$ are random numbers uniformly distributed on $[0, 1]$; *mbest* is a global virtual point called mainstream or mean best defined as

$$mbest_d^t = \frac{1}{N} \sum_{i=1}^{N} pbest_{i,d}^t. \tag{4}$$

A time-varying decreasing method [23] usually is adapted to control the contraction-expansion coefficient defined as follows:

$$\alpha = \alpha_1 + \frac{(T - t) \times (\alpha_0 - \alpha_1)}{T}, \tag{5}$$

where α_0 and α_1 are the initial and final values of α, respectively; T is the maximum number of iterations; t is the current search iteration number.

The QPSO algorithm has simpler evolutional equation forms and fewer parameters than classical PSO, substantially facilitating the control and convergence in the search space. Without loss of generality, let f be the objective function to be minimized; the procedure for implementing the QPSO is given in Algorithm 1.

3. Related Work

Since QPSO was proposed, it has attracted much attention and different variants of QPSO have been proposed to enhance the performance from different aspects and applied to solve various real world optimization problems. In this section, a brief review of these QPSO variants will be presented. In general, most current QPSO variants can be classified into three categories, that is, the improvement based on operators from other evolutionary algorithms, hybrid search methods, and cooperative methods.

```
(1)  Procedure of QPSO
(2)    For $i = 1$ to swarm size $N$
(3)      randomize the position of each particle $x[i]$;
(4)      evaluate $x[i]$; $pbest[i] = x[i]$;
(5)    Endfor
(6)    Do
(7)      $gbest = \text{argmin}(f(pbest))$;
(8)      compute $mbest$ by (4);
(9)      For $i = 1$ to swarm size $N$
(10)       calculate $p[i]$ with (2);
(11)       update $x[i]$ with (3);
(12)       If $f(x[i]) < f(pbest[i])$
(13)         $pbest[i] = x[i]$;
(14)       Endif
(15)     Endfor
(16)   Until maximum number of iterations is reached
```

Algorithm 1: The pseudocodes for QPSO algorithm.

To improve the search efficiency, different operators derived from other evolutionary algorithms were introduced into QPSO algorithms. The mutation operator provides diversity in the search of solutions and consequently enhances the global search capability; therefore, various mutation operators based on Cauchy probability distribution [24], Gaussian probability distribution [14, 25], Lévy probability distribution [20], and chaotic sequences [26] were proposed to improve the QPSO performance in preventing premature convergence to local optima. Selection operators are beneficial in making good use of the information of elitists and the particles' position in the last iteration. Elitist selection operator and ranking strategy were introduced into QPSO to balance the convergence rate and global searching ability in [27, 28]. In [15], the *mbest* in each particle's position update procedure is determined by a random selection operator on all the *pbests* of the whole swarm. Another approach incorporating a ranking operator to select a random particle with its personal best position replacing the global best position in order to guide the particle to escape from the local optima is also proposed in [15]. Besides mutations and selections mentioned above, DE operators [29] and crossover operator [30] were also integrated into QPSO to enhance the particles' search capability.

Various search methods, including those from other optimization algorithms and the new proposed approaches, were incorporated into QPSO for better search efficiency. In [31], local search was incorporated into the main global search mechanism of QPSO to enhance the searching qualities of QPSO. The chaotic search method was integrated into QPSO to diversify the swarm in the latter period of the search process so as to alleviate the likelihood to be stuck in the local optima [32]. In [33, 34], the immune system was introduced to QPSO to effectively restrain the degeneration in the process of optimization. Simulated annealing was incorporated into QPSO to improve the ability to effectively jump out of the local optima in [35].

The third category, cooperative methods, refers to the searches performed with multiswarms or the approaches of optimizing different components of each solution vector separately. In [36], two subswarms were used to search in different layers and the cooperation of the subswarms leads to better performance in approximating solutions to the global optima. In [21, 37], a particle is split into several subparts and makes contribution to the population not only as a whole item but also in subset of the position vector.

4. The Proposed Algorithm

There are numerous variants of QPSO which have been proposed in recent literatures. In most of the QPSOs, elitists are simply stored in memory for the solution comparison or through simple processing step like mutation for the exploration search. Nevertheless, to the best of our knowledge, all the existing QPSOs seldom explore the elitist memory and get more potential information from it, which might be a significant trend to improve the performance of the algorithm.

The memory of QPSO mainly consists of the elitist individuals which are the personal best of each particle and the global best of the whole swarm. According to the iterative equation of QPSO, the memory is very important in affecting the behavior of the swarm and thus impinges the performance of the algorithm. If the personal best particles and the global best particle get trapped in local optima, the algorithm will likely converge to local optima. On the other hand, if the individuals in memory are less likely to be stuck at local optima, the algorithm can achieve better solutions.

In fact, we believe that the information from elitists can have a better use in aiding the search exploration and exploitation of the global optima. New subswarms can be generated from the elitists with proper mechanism to aim at the better search efficiency in terms of the solution quality and rate of convergence. An elitist exploration strategy, namely, elitist breeding, is proposed in this study. It makes use of the elitists generated in the evolutionary processes of algorithm to create new subswarms through the proposed breeding scheme. In the elitist breeding scheme, an elitist pool consists of personal best particles and global best particle found so far was constructed, and then the transposon operators which has the ability to enhance the diversity of solutions are selected as the breeding operators to explore the elitist memory and extract some more potential essences from the elitist individuals, thus to improve the search efficiency of QPSO. Moreover, the mechanism of updating the elitists with the new bred individuals with better fitness also provides a more efficient and precise search guidance for the swarm. The pseudocodes of the proposed EB-QPSO are given in Algorithm 2.

We can see from the pseudocodes of EB-QPSO that when the criteria is met, the personal best of each particle and the global best will be put into an elitist pool named *epool* and a new subswarm called *epool_eb* is generated through the transposon operators on the elitist pool. The size of *epool_eb* is the same as the *epool* and each individual in both of the pools is identified with its sequential order. To maintain the diversity of the swarm, each individual in *epool_eb* is only compared with the member with the same sequential order

```
(1)  **Procedure** of EB-QPSO
(2)    **For** $i = 1$ to swarm size $N$
(3)      randomize the position of each particle $x[i]$;
(4)      evaluate $x[i]$; $pbest[i] = x[i]$;
(5)    **Endfor**
(6)    **Do**
(7)      $gbest = \arg\min(f(pbest))$;
(8)      compute $mbest$ by (4);
(9)      **If** elitist breeding criterion is met
(10)       **For** $i = 1$ to swarm size $N$
(11)         $epool[i] = pbest[i]$;
(12)       **Endfor**
(13)       $epool[N+1] = gbest$;
(14)       $epool\_eb$ = transposon_op($epool$);
(15)       **For** $i = 1$ to swarm size $N$
(16)         **If** $f(epool\_eb[i]) < f(pbest[i])$;
(17)           $pbest[i] = epool\_eb[i]$;
(18)         **Endif**
(19)       **Endfor**
(20)       $gbest = \arg\min(f(pbest))$;
(21)     **Endif**
(22)     **For** $i = 1$ to swarm size $N$
(23)       calculate $p[i]$ with (2);
(24)       update $x[i]$ with (3);
(25)       **If** $f(x[i]) < f(pbest[i])$
(26)         $pbest[i] = x[i]$;
(27)       **Endif**
(28)     **Endfor**
(29)   **Until** maximum number of iterations is reached
```

Algorithm 2: The pseudocodes for the proposed EB-QPSO algorithm.

in *epool*. The *pbests* will be updated when the new generated individual is better than the corresponding one in *epool*. Here, a predefined parameter λ is used to control the frequency of elitist breeding. In every λ iteration, the breeding operation will be performed once.

Transposon operators were firstly proposed by Man et al. [38] and mainly used in multiobjective evolutionary algorithms (MOEAs). A transposon is made of consecutive genes located in the randomly assigned position in each chromosome while the transposon operators are lateral movement operations that happen in one chromosome or between different ones. Generally speaking, there exist two types of transposon operators, that is, cut-and-paste and copy-and-paste, which are demonstrated in Figures 1 and 2. The transposon operations conducted within an individual chromosome or on a different chromosome are chosen randomly. Moreover, the size of each transposon can be greater than one and is decided by a parameter called jumping percentage while the number of transposons is also a predefined parameter. Another parameter, the jumping rate, is assigned to determine the probability of the activation of transposon operations.

As demonstrated in Figure 3, each particle which can be regarded as a chromosome consists of the same number of genes as the size of its position vector and each gene holds a real number of the corresponding decision variable.

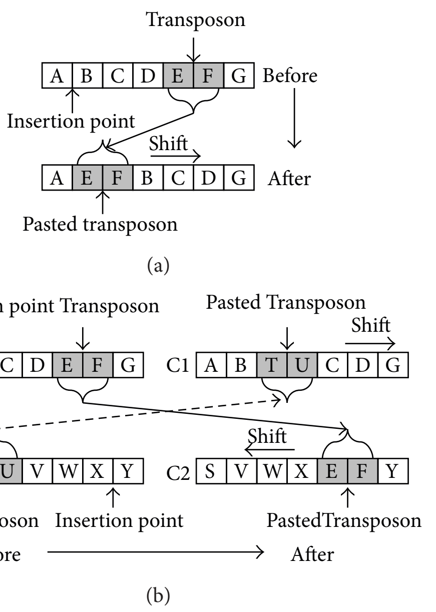


Figure 1: Cut-and-paste transposon operator. (a) Cut-and-paste in same chromosome. (b) Cut-and-paste in different chromosomes.

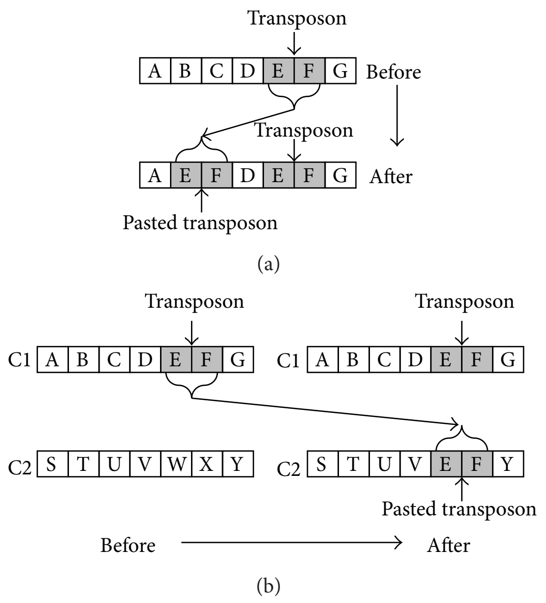
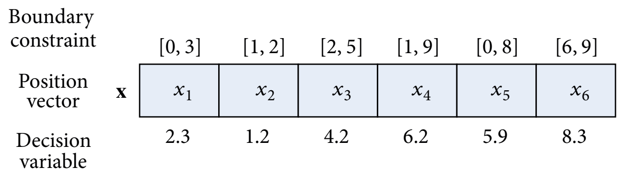


Figure 2: Copy-and-paste transposon operator. (a) Copy-and-paste in same chromosome. (b) Copy-and-paste in different chromosomes.



Boundary constraint		[0, 3]	[1, 2]	[2, 5]	[1, 9]	[0, 8]	[6, 9]
Position vector	**x**	x_1	x_2	x_3	x_4	x_5	x_6
Decision variable		2.3	1.2	4.2	6.2	5.9	8.3

Figure 3: An example of the particle is represented as a chromosome.

```
(1)  Function $epool\_eb$ = transposon_op($epool$)
(2)    generate the $epool\_ratio$ with the $epool$ based on (6)
(3)    **For** $i = 1$ to $N + 1$
(4)      **For** $j = 1$ to ***number of transposon***
(5)        **If** rand(0, 1) < ***jumping rate***
(6)          $C_1 = i$;
(7)          $C_2 = ceil(\text{rand}(0, 1) \times (N + 1))$
(8)          **If** $C_1 == C_2$
(9)            **If** rand(0, 1) > 0.5
(10)             perform cut and paste operation in $epool\_ratio$ [$C_1$];
(11)           **Else**
(12)             perform copy and paste operation in $epool\_ratio$ [$C_1$];
(13)           **Endif**
(14)         **Else**
(15)           **If** rand(0, 1) > 0.5
(16)             perform cut and paste operation in $epool\_ratio$ [$C_1$] and $epool\_ratio$ [$C_2$];
(17)           **Else**
(18)             perform copy and paste operation in $epool\_ratio$ [$C_1$] and $epool\_ratio$ [$C_2$];
(19)           **Endif**
(20)         **Endif**
(21)       **Endif**
(22)     **Endfor**
(23)   **Endfor**
(24)   restore the individuals from $epool\_ratio$ based on (7) and save to $epool\_eb$
```

Algorithm 3: The pseudocodes for the transposon operations in the proposed algorithm.

It is clear that the number of the chromosomes is the same as the number of particles in the swarm. With such representation, the transposon operators can be integrated into PSOs. However, since different decision variables might have different boundary constraints, the transposon operators might generate invalid individual. As illustrated in Figure 3, the boundary violation occurs if x_1 is copied to the position of x_2. To overcome the boundary violation problem caused by transposon operators, according to the description in [39], the position vector of each particle is normalized to a chromosome consists of the ratio value of each variable occupying in its own boundary range before performing the transposon operations. The conversion equation is defined as follows:

$$\text{conv}(x) = \frac{x - \text{lb}(x)}{\text{ub}(x) - \text{lb}(x)}, \tag{6}$$

where lb(x) and ub(x) represent the lower and upper bounds of x, respectively. After the transposon operations, the value of each gene should be restored back to its corresponding positional value in the search space according to the equation as follows:

$$\text{resto}(x) = \text{conv}(x) \times (\text{ub}(x) - \text{lb}(x)) + \text{lb}(x). \tag{7}$$

In transposon operations, a temporary pool called *epool_ratio* is used to store all the normalized chromosomes converted from the individuals in *epool* according to (6). When the transposon operations have been done, each individual will be restored from *epool_ratio* and saved to *epool_eb*. The pseudocodes of the function *transposon_op* are given in Algorithm 3.

5. Experiment Result and Discussion

To validate the performance of the proposed EB-QPSO, it was tested on a set of twelve widely adopted benchmark testing functions and compared with the original QPSO [9] and four state-of-the-art QPSO variants including WQPSO [28], CAQPSO [14], QPSO-RM [15] and QPSO-RO [15], which have been studied thoroughly in the corresponding literatures, on solution accuracy, convergence speed, and algorithm reliability. These QPSO variants are very competitive in comparing with some other forms of PSO algorithms including PSO with inertia weight (PSO-In) [40], PSO with constriction factor (PSOCo) [41], the Standard PSO [42], Gaussian PSO [43], Gaussian Bare Bones PSO [44], Exponential PSO (PSO-E) [45], Lévy PSO [46], comprehensive learning PSO (CLPSO) [47], dynamic multiple swarm PSO (DMS-PSO) [48], and fully informed particle swarm (FIPS) [49].

5.1. Test Instances and Algorithmic Configuration.

The test set consisted of twelve unconstrained test instances widely used in early researches reported in literatures [50–53] as listed in Table 1 with seven unimodal functions (f_1–f_7) and five multimodal functions (f_8–f_{12}). These benchmark instances pose different difficulties to optimization algorithms. The more detailed description of each function can be found in [51]. Following [51], an "acceptance" value is defined to gauge the acceptability of a solution found by the QPSOs.

For the simulation experiment, the contraction expansion (CE) coefficients of the proposed EB-QPSO algorithm and the compared five QPSOs decreased linearly from α_0 to α_1 during the search process according to (5). α_0 and α_1 were set at 0.6 and 0.5 for EB-QPSO, QPSO-RM and QPSO-RO

Table 1: The test instances used in the experiment.

Name	Function	D	Search space	Global $f_{\min}$	Acceptance
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	30	$[-100, 100]^D$	0	0.01
Schwefel's P2.22	$f_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	$[-10, 10]^D$	0	0.01
Quadric	$f_3(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100, 100]^D$	0	100
Rosenbrock	$f_4(x) = \sum_{i=1}^{D-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + \left(x_i - 1 \right)^2 \right]$	30	$[-100, 100]^D$	0	100
Step	$f_5(x) = \sum_{i=1}^{D} \left(\lvert x_i + 0.5 \rvert \right)^2$	30	$[-100, 100]^D$	0	0
Quadric Noise	$f_6(x) = \sum_{i=1}^{D} i x_i^4 + \text{rand}[0, 1]$	30	$[-1.28, 1.28]^D$	0	0.01
Schwefel	$f_7(x) = \sum_{i=1}^{D} -x_i \sin\left(\sqrt{x_i} \right) + 12569.5$	30	$[-500, 500]^D$	0	2569.5
Rastrigin	$f_8(x) = \sum_{i=1}^{D} \left[x_i^2 - 10 \cos\left(2\pi x_i \right) + 10 \right]$	30	$[-5.12, 5.12]^D$	0	50
Noncontinuous Rastrigin	$f_9(x) = \sum_{i=1}^{D} \left[y_i^2 - 10 \cos\left(2\pi y_i \right) + 10 \right]$ where $y_i = \begin{cases} x_i, & \lvert x_i \rvert < 0.5 \\ \dfrac{\text{round}(2x_i)}{2}, & \lvert x_i \rvert \geq 0.5 \end{cases}$	30	$[-5.12, 5.12]^D$	0	50
Ackley	$f_{10}(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos 2\pi x_i \right) + 20 + e$	30	$[-32, 32]^D$	0	0.01
Griewank	$f_{11}(x) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	$[-600, 600]^D$	0	0.01
Generalized penalized	$f_{12}(x) = \frac{\pi}{D} \left\{ 10 \sin^2\left(\pi y_i \right) + \sum_{i=1}^{D-1} \left(y_i - 1 \right)^2 \left[1 + 10 \sin\left(\pi y_{i+1} \right) \right] \right\} + \left(y_D - 1 \right)^2 + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ where $y_i = 1 + \frac{1}{4}(x_i + 1), u(x, a, k, m) = \begin{cases} k\left(x_i - a \right)^m, & x_i > a \\ 0, & -a \leq x_i \leq a \\ k\left(-x_i - a \right)^m, & x_i < a \end{cases}$	30	$[-50, 50]^D$	0	0.01

while 1.0 and 0.5 were used for QPSO, WQPSO, and CAQPSO accordingly. The other parameters of the proposed EB-QPSO algorithm for the experiment were chosen at values as given in Table 2. For the compared QPSO algorithms, parameters settings other than CE coefficients were used with values according to their corresponding references.

For each test problem, 50 simulation trials were conducted for each of the compared algorithms with random initial populations. For a fair comparison among all the QPSO algorithms, they were tested with the same population size of 20 at each trial run. Furthermore, the maximum number of objective function evaluations (FEs) was set at 4.0×10^4 for each test instance for all the algorithms. Since all the algorithms in the simulation experiment are stochastic, not solely compared on the success rate, it is essential to have the statistical analysis conducted so as to provide confidential

Table 2: The parameters setting for the proposed EB-QPSO.

Parameter	Value
CE coefficient (α)	$\alpha_0 = 0.6, \alpha_1 = 0.5$
Jumping percentage	6
Jumping rate	0.1
Number of transposons	6

Table 3: Search result comparisons among six QPSOs.

Problems	EB-QPSO $\tilde{x}$ (IQR)	QPSO $\tilde{x}$ (IQR)	WQPSO $\tilde{x}$ (IQR)	CAQPSO $\tilde{x}$ (IQR)	QPSO-RM $\tilde{x}$ (IQR)	QPSO-RO $\tilde{x}$ (IQR)	
f_1	**0.000E + 00** (**0.000E + 00**)	1.974E − 12 (5.003E − 12)	4.373E − 26 (1.039E − 25)	5.271E + 00 (6.794E + 00)	6.074E − 31 (2.290E − 29)	2.649E − 05 (4.408E − 04)	+
f_2	**0.000E + 00** (**0.000E + 00**)	5.497E − 09 (1.344E − 08)	1.459E − 18 (2.283E − 18)	1.225E − 01 (5.969E − 02)	1.319E − 18 (7.520E − 18)	5.124E − 06 (8.130E − 05)	+
f_3	**3.919E − 04** (**2.019E − 03**)	8.796E + 02 (3.557E + 02)	1.140E + 01 (1.122E + 01)	2.620E + 04 (6.306E + 03)	1.493E + 01 (1.476E + 01)	9.875E + 02 (7.522E + 02)	+
f_4	**6.290E − 02** (**1.828E − 01**)	8.830E + 01 (1.324E + 02)	2.476E + 01 (6.817E + 01)	2.643E + 05 (5.027E + 05)	2.879E + 01 (1.004E + 02)	8.907E + 02 (1.238E + 03)	+
f_5	**0.000E + 00** (**0.000E + 00**)	2.178E − 12 (1.149E − 11)	1.209E − 03 (2.756E − 04)	5.168E + 00 (6.310E + 00)	1.972E − 31 (5.276E − 30)	5.042E − 06 (3.926E − 05)	+
f_6	**3.138E − 03** (**2.605E − 03**)	7.314E − 03 (3.798E − 03)	9.824E − 03 (6.296E − 03)	1.219E − 01 (7.925E − 02)	1.096E − 02 (5.525E − 03)	1.246E − 02 (9.904E − 03)	+
f_7	**3.818E − 04** (**1.819E − 12**)	3.432E + 03 (8.739E + 02)	3.273E + 03 (8.595E + 02)	7.901E + 03 (4.248E + 02)	2.496E + 03 (5.330E + 02)	3.020E + 03 (5.892E + 02)	+
f_8	**0.000E + 00** (**0.000E + 00**)	2.933E + 01 (1.344E + 01)	2.600E + 01 (6.959E + 00)	2.335E + 02 (3.252E + 01)	6.847E + 01 (7.292E + 01)	2.289E + 01 (7.699E + 00)	+
f_9	**0.000E + 00** (**0.000E + 00**)	4.525E + 01 (1.901E + 01)	3.018E + 01 (1.617E + 01)	2.288E + 02 (3.662E + 01)	1.410E + 02 (4.711E + 01)	3.250E + 01 (1.113E + 01)	+
f_{10}	**7.994E − 15** (**7.105E − 15**)	2.774E − 07 (5.001E − 07)	1.004E − 13 (4.263E − 14)	2.023E + 01 (1.035E + 01)	1.510E − 14 (1.243E − 14)	1.712E + 00 (1.252E + 00)	+
f_{11}	**0.000E + 00** (**0.000E + 00**)	7.396E − 03 (1.907E − 02)	7.475E − 03 (1.671E − 02)	1.045E + 00 (5.193E − 02)	7.396E − 03 (1.232E − 02)	1.015E − 01 (1.938E − 01)	+
f_{12}	**1.571E − 32** (**0.000E + 00**)	4.510E − 12 (4.574E − 11)	4.557E − 05 (1.012E − 05)	2.077E + 01 (6.941E + 01)	1.057E − 29 (9.400E − 27)	1.039E − 01 (4.960E − 01)	+

comparisons. The median ($\tilde{x}$) and interquartile range (IQR) of each test instance were recorded as the measures of location (or central tendency) and statistical dispersion. To provide the confidential comparisons, the statistical analysis as in [54, 55] was used. The general structure of statistical analysis is given in Figure 4.

Kolmogorov-Smirnov test is firstly conducted to check whether the value of the results follow the normal (Gaussian) distribution or not. If the results do not follow the normal distribution, the nonparametric Kruskal-Wallis test is performed in order to compare the median result of each algorithm; if not, the Levene test is used to check the homogeneity of the variances. A Welch test is performed to verify the confidence of comparisons if the samples have different variance, otherwise ANOVA test is done to accomplish this task. The confidence level of 95% is used in the statistical analysis. The results are shown in the last column of Tables 3 and 4 with the symbol "+" indicating that the performance difference between the proposed EB-QPSO and best algorithm among the other QPSO algorithms is statistically significant, while the symbol "−" representing the difference is insignificant.

5.2. Comparisons on the Solution Accuracy. The performance results on the solution accuracy of each of the algorithms in the simulation experiment are shown in Table 3 in terms of the median and interquartile range of the solutions obtained in the 50 independent runs by each algorithm. The best result among those obtained by all six contenders for each problem is highlighted with boldface.

From the results in Table 3, we can see that the proposed EB-QPSO algorithm obtains the best values in all of the 12 test instances. It is worth mentioning that EB-QPSO achieves the solutions which have values reached or approximated to the

Table 4: The FEs number needed to reach an acceptable solution for six QPSOs.

Problems	EB-QPSO $\tilde{x}$ (IQR)	QPSO $\tilde{x}$ (IQR)	WQPSO $\tilde{x}$ (IQR)	CAQPSO $\tilde{x}$ (IQR)	QPSO-RM $\tilde{x}$ (IQR)	QPSO-RO $\tilde{x}$ (IQR)	
f_1	**2.220E + 03** (**1.800E + 02**)	1.720E + 04 (6.100E + 02)	1.647E + 04 (5.000E + 02)	/ (0.000E + 00)	8.910E + 03 (6.400E + 02)	6.580E + 03 (2.860E + 03)	+
f_2	**2.230E + 03** (**1.400E + 02**)	1.694E + 04 (6.950E + 02)	1.593E + 04 (3.600E + 02)	/ (0.000E + 00)	9.140E + 03 (6.750E + 02)	4.700E + 03 (9.450E + 02)	+
f_3	**9.230E + 03** (**2.815E + 03**)	/ (0.000E + 00)	3.323E + 04 (2.275E + 03)	/ (0.000E + 00)	3.206E + 04 (3.995E + 03)	/ (0.000E + 00)	+
f_4	**4.730E + 03** (**1.470E + 03**)	1.743E + 04 (1.957E + 04)	1.788E + 04 (3.900E + 03)	/ (0.000E + 00)	9.190E + 03 (1.214E + 04)	/ (0.000E + 00)	+
f_5	**1.095E + 04** (**1.075E + 03**)	/ (0.000E + 00)	/ (0.000E + 00)	/ (0.000E + 00)	/ (3.667E + 04)	/ (0.000E + 00)	+
f_6	**1.099E + 04** (**7.035E + 03**)	2.974E + 04 (8.495E + 03)	2.563E + 04 (3.415E + 04)	/ (0.000E + 00)	/ (3.366E + 04)	/ (0.000E + 00)	+
f_7	**5.200E + 03** (**9.500E + 02**)	/ (0.000E + 00)	/ (0.000E + 00)	/ (0.000E + 00)	5.510E + 03 (1.420E + 04)	/ (0.000E + 00)	+
f_8	**4.700E + 03** (**8.000E + 02**)	2.920E + 04 (3.910E + 03)	2.749E + 04 (2.985E + 03)	/ (0.000E + 00)	/ (3.404E + 04)	1.008E + 04 (6.340E + 03)	+
f_9	**3.400E + 03** (**6.000E + 02**)	3.073E + 04 (3.500E + 04)	3.384E + 04 (6.245E + 03)	/ (0.000E + 00)	/ (0.000E + 00)	2.092E + 04 (1.468E + 04)	+
f_{10}	**2.490E + 03** (**2.050E + 02**)	1.809E + 04 (6.800E + 02)	1.726E + 04 (7.250E + 02)	/ (0.000E + 00)	1.021E + 04 (9.850E + 02)	/ (0.000E + 00)	+
f_{11}	**4.220E + 03** (**2.130E + 03**)	1.865E + 04 (1.997E + 04)	1.742E + 04 (1.969E + 04)	/ (0.000E + 00)	1.043E + 04 (1.160E + 04)	/ (0.000E + 00)	+
f_{12}	**2.010E + 03** (**4.350E + 02**)	1.754E + 04 (1.400E + 03)	1.735E + 04 (1.785E + 03)	/ (0.000E + 00)	9.500E + 03 (2.230E + 03)	/ (8.655E + 03)	+

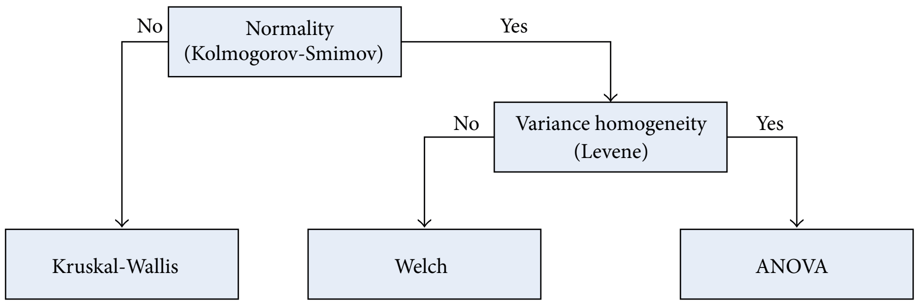
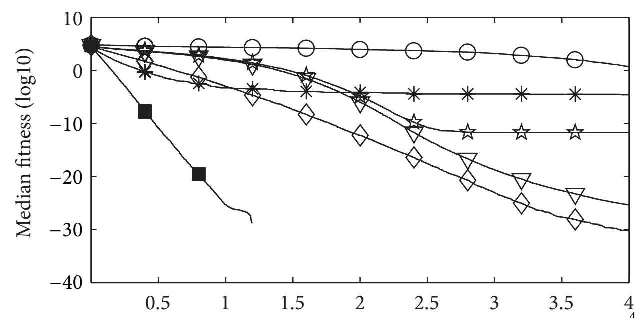


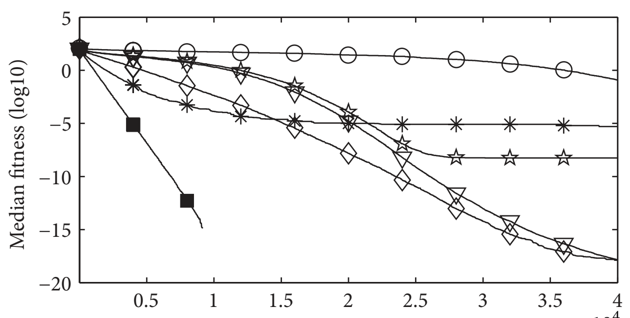
Figure 4: The general structure of the statistical analysis performed in this work.

global optima on problems f_3, f_4, f_7, f_8, and f_9 whereas the other compared algorithms are unable to do so. It illustrates that the proposed EB-QPSO having the ability to avoid being stuck at the local optima with the benefits from the elitist breeding through transposon operators. The difference in performance of the proposed algorithm comparing to the other five QPSO is statistically significant as indicated in the statistical analysis result shown in the last column of the table.

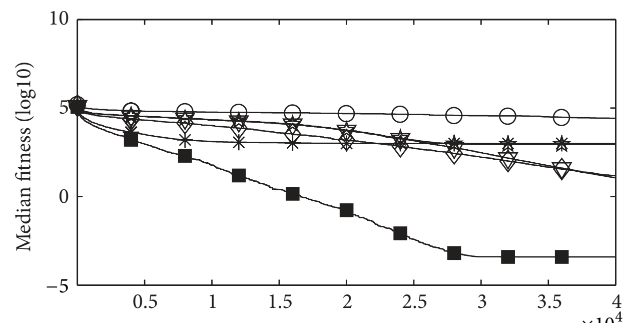
5.3. Comparisons on the Convergence Speed and Reliability. Another salient yardstick for evaluating the algorithm performance is the speed in approximating the global optimum. As shown in Table 4, EB-QPSO entirely offers a much higher convergence speed which is measured by the median and interquartile range of FEs number needed to reach an acceptable solution. Note that here "/" represents the corresponding algorithm cannot reach an acceptable solution in at least one-half of all the trials. To compare the convergence characteristics, Figure 5 graphically presents the convergence processes in terms of median results obtained in the 50 runs of all six contenders in solving the 12 test instances. It is worth mentioning that the global optima of all the test instances are at "0" and the logarithm of "0" has no mathematical meaning and cannot be displayed graphically. For any algorithm, if the optima is found, it will stop searching for solution further even though the maximum number of function



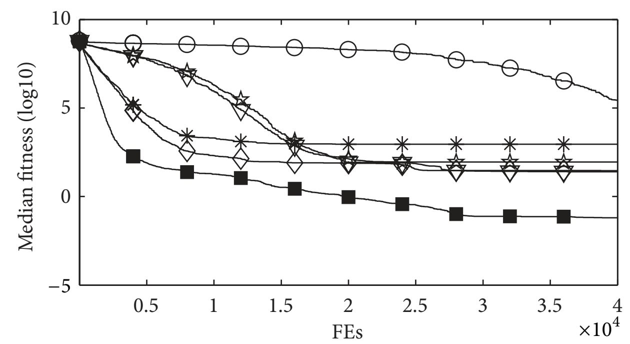
(f_1)



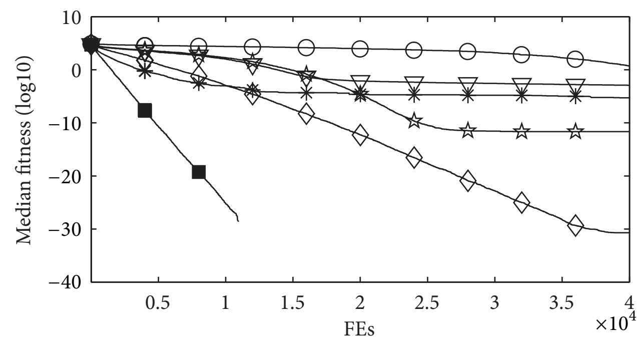
(f_2)



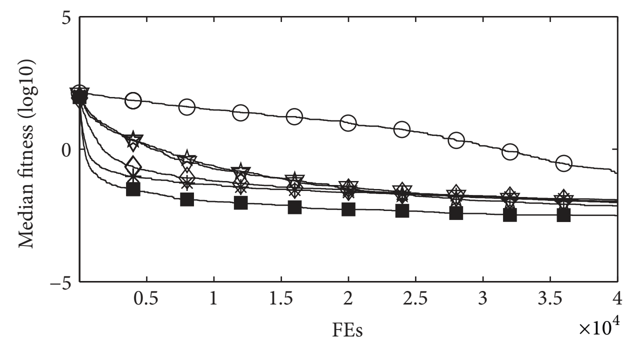
(f_3)



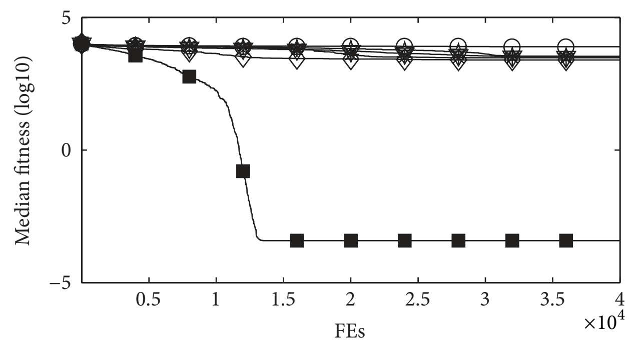
(f_4)



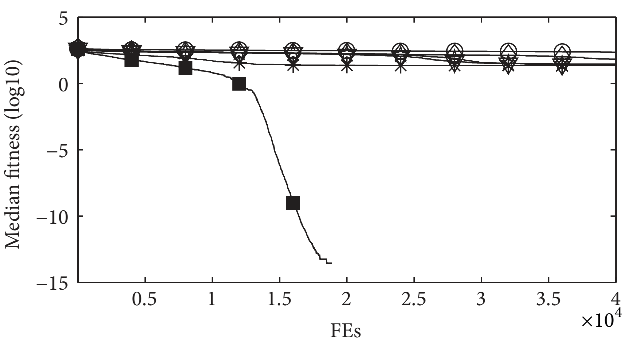
(f_5)



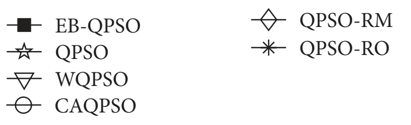
(f_6)



(f_7)



(f_8)



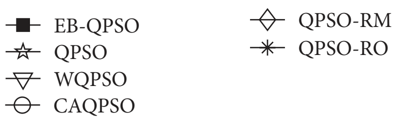


Figure 5: Continued.

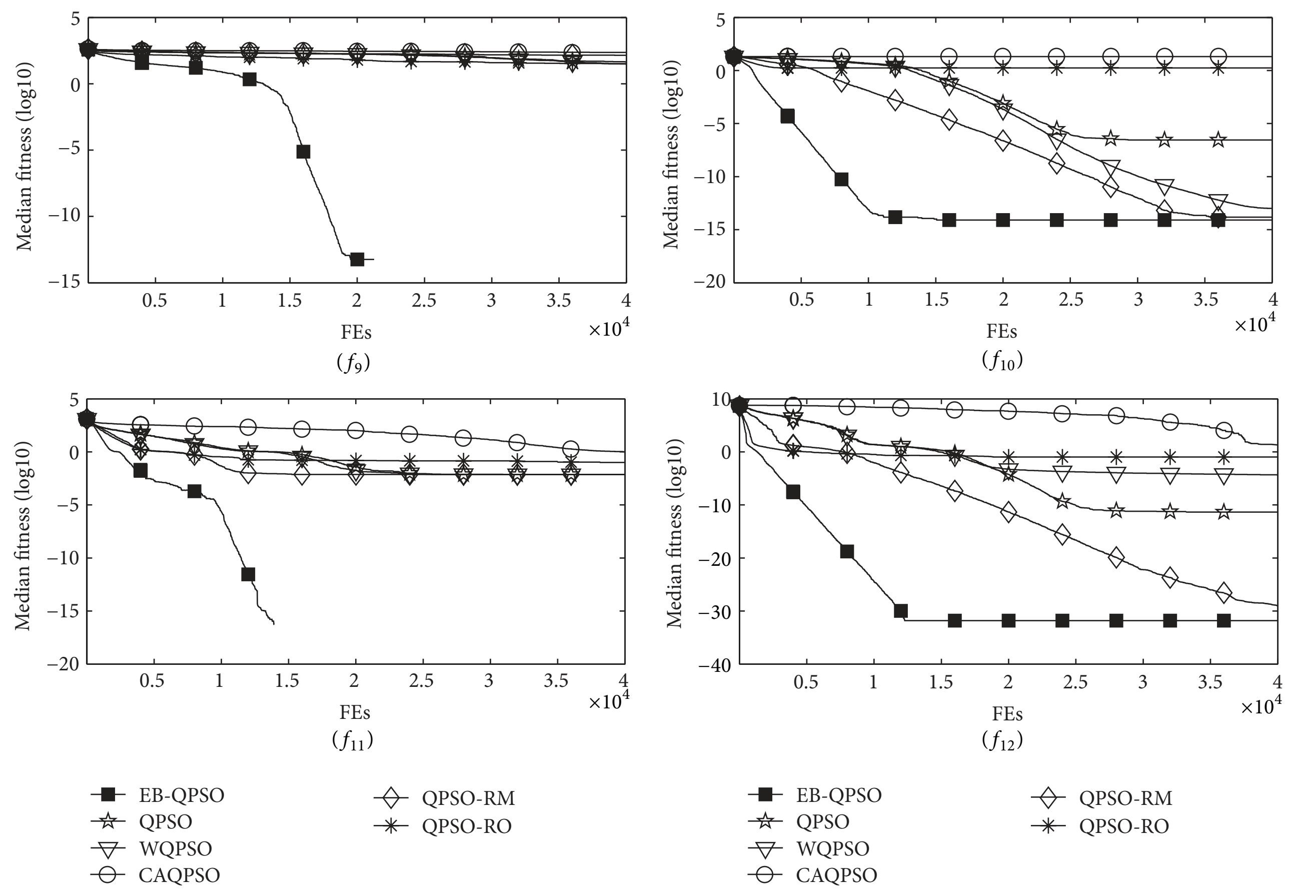


Figure 5: Convergence performance of the six QPSOs on the 12 test instances.

Table 5: Reliability comparisons among six QPSOs.

Problems	EB-QPSO	QPSO	WQPSO	CAQPSO	QPSO-RM	QPSO-RO
f_1	100.0%	100.0%	100.0%	0.0%	100.0%	100.0%
f_2	100.0%	100.0%	100.0%	0.0%	100.0%	98.0%
f_3	100.0%	0.0%	100.0%	0.0%	100.0%	0.0%
f_4	100.0%	56.0%	76.0%	0.0%	68.0%	2.0%
f_5	100.0%	0.0%	0.0%	0.0%	28.0%	0.0%
f_6	100.0%	82.0%	52.0%	0.0%	42.0%	22.0%
f_7	100.0%	4.0%	10.0%	0.0%	52.0%	18.0%
f_8	100.0%	98.0%	98.0%	0.0%	40.0%	100.0%
f_9	100.0%	62.0%	88.0%	0.0%	0.0%	92.0%
f_{10}	100.0%	100.0%	100.0%	0.0%	100.0%	12.0%
f_{11}	100.0%	62.0%	58.0%	0.0%	68.0%	4.0%
f_{12}	100.0%	90.0%	96.0%	0.0%	88.0%	44.0%

evaluation limit is not reached; thus, upon such condition, the corresponding convergence curve for a particular test case will be stopped at that point, It is the case for problems f_1, f_2, f_5, f_8, f_9, and f_{11} as the search convergences to the global optima. It shows that the proposed EB-QPSO has the best convergent efficiency amongst the compared algorithms in all 12 test instances.

Reliability here refers to the success rate; that is, the percentage of trial runs reaching acceptable solutions. Table 5 reveals that EB-QPSO is able to reach acceptable solutions in all the trials over all the test instances whereas the compared algorithms cannot.

6. Conclusion

QPSO is a promising optimization technique which has shown its superiority in solving wide range of optimization problems. However, it is still a difficult problem to improve the global search capability and accelerate convergence speed of QPSO simultaneously. In this paper, we presented an improved quantum-behaved particle swarm optimization algorithm with elitist breeding (EB-QPSO) for unconstrained optimization. The novel elitist breeding scheme acts on the elitists found during the evolutionary process to jump out of the likely local optima and guide the swarm to perform exploration and exploitation more efficiently and thus improves the performance of QPSO in terms of better global search capability and faster convergence speed. The performance of EB-QPSO has been compared against the existing QPSO algorithms, the original, WQPSO, CAQPSO, QPSO-RM, and QPSO-RO on a test suite consisting of twelve benchmark functions. All simulation results have demonstrated that EB-QPSO has superiority over other QPSOs in solution accuracy, convergence speed, and reliability significantly. Besides, EB-QPSO can locate the global optima of most of the test functions while the other algorithms cannot. Our further work will concentrate on applying the EB-QPSO algorithm to the real-world optimization problems and on integrating the approach of elitist breed into other swarm intelligence algorithms. In this paper, the proposed EB-QPSO was studied empirically; the theoretical analysis of the global convergence of EB-QPSO will be developed in the future.

Conflict of Interests

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

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, Perth, Australia, November-December 1995.

[2] S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho, "OPSO: orthogonal particle swarm optimization and its application to task assignment problems," *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, vol. 38, no. 2, pp. 288–298, 2008.

[3] B. Liu, L. Wang, and Y.-H. Jin, "An effective PSO-based memetic algorithm for flow shop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 1, pp. 18–27, 2007.

[4] P. Melin, F. Olivas, O. Castillo, F. Valdez, J. Soria, and M. Valdez, "Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic," *Expert Systems with Applications*, vol. 40, no. 8, pp. 3196–3206, 2013.

[5] A. Selakov, D. Cvijetinović, L. Milović, S. Mellon, and D. Bekut, "Hybrid PSO-SVM method for short-term load forecasting during periods with significant temperature variations in city of Burbank," *Applied Soft Computing Journal*, vol. 16, pp. 80–88, 2014.

[6] F. van den Bergh, *An Analysis of Particle Swarm Optimizers*, Department of Computer Science, University of Pretoria, 2002.

[7] F. J. Solis and R. J. Wets, "Minimization by random search techniques," *Mathematics of Operations Research*, vol. 6, no. 1, pp. 19–30, 1981.

[8] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[9] J. Sun, B. Feng, and W. Xu, "Particle swarm optimization with particles having quantum behavior," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '04)*, pp. 325–331, June 2004.

[10] S. Jun, *Particle Swarm Optimization with Particles Having Quantum Behavior*, Department of Control Theory and Engineering, Jiangnan University, 2009.

[11] J. Sun, C.-H. Lai, and X.-J. Wu, *Particle Swarm Optimisation: Classical and Quantum Perspectives*, CRC Press, 2012.

[12] H. Long, J. Sun, X. Wang, C.-H. Lai, and W. Xu, "Using selection to improve quantum-behaved particle swarm optimisation," *International Journal of Innovative Computing and Applications*, vol. 2, no. 2, pp. 100–114, 2009.

[13] J. Sun, W. Xu, and W. Fang, "A diversity-guided quantum-behaved particle swarm optimization algorithm," in *Proceedings of the 6th International Conference on Simulated Evolution and Learning*, pp. 497–504, Hefei, China, October 2006.

[14] J. Sun, W. Fang, V. Palade, X. Wu, and W. Xu, "Quantum-behaved particle swarm optimization with Gaussian distributed local attractor point," *Applied Mathematics and Computation*, vol. 218, no. 7, pp. 3763–3775, 2011.

[15] J. Sun, X. Wu, V. Palade, W. Fang, C.-H. Lai, and W. Xu, "Convergence analysis and improvements of quantum-behaved particle swarm optimization," *Information Sciences*, vol. 193, pp. 81–103, 2012.

[16] N. Tian, J. Sun, W. Xu, and C.-H. Lai, "An improved quantum-behaved particle swarm optimization with perturbation operator and its application in estimating groundwater contaminant source," *Inverse Problems in Science and Engineering*, vol. 19, no. 2, pp. 181–202, 2011.

[17] L. D. S. Coelho, "Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1676–1683, 2010.

[18] H. X. Long and S. L. Wu, "Quantum-behaved particle swarm optimization with diversity-maintained," in *Ecosystem Assessment and Fuzzy Systems Management*, vol. 254, pp. 207–219, Springer, Berlin, Germany, 2014.

[19] R. Nie, X. Xu, and J. Yue, "A novel quantum-inspired particle swarm algorithm and its application," in *Proceedings of the 6th International Conference on Natural Computation (ICNC '10)*, pp. 2556–2560, IEEE, Shandong, China, August 2010.

[20] Y. Peng, Y. Xiang, and Y. Zhong, "Quantum-behaved particle swarm optimization algorithm with Lévy mutated global best position," in *Proceedings of the 4th International Conference on Intelligent Control and Information Processing (ICICIP '13)*, pp. 529–534, Beijing, China, June 2013.

[21] H. Gao, W. Xu, J. Sun, and Y. Tang, "Multilevel thresholding for image segmentation through an improved quantum-behaved particle swarm algorithm," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 4, pp. 934–946, 2010.

[22] J. Liu, W. Xu, and J. Sun, "Quantum-behaved Particle Swarm Optimization with mutation operator," in *Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI '05)*, pp. 237–240, November 2005.

[23] J. Sun, W. Fang, X. Wu, V. Palade, and W. Xu, "Quantum-behaved particle swarm optimization: analysis of individual particle behavior and parameter selection," *Evolutionary Computation*, vol. 20, no. 3, pp. 349–393, 2012.

[24] J. Liu, J. Sun, and W. Xu, "Quantum-behaved particle swarm optimization with adaptive mutation operator," in *Proceedings of the 2nd International Conference on Natural Computation*, pp. 959–967, Xi'an, China, September 2006.

[25] L. S. Coelho, "Novel Gaussian quantum-behaved particle swarm optimiser applied to electromagnetic design," *IET Science, Measurement & Technology*, vol. 1, no. 5, pp. 290–294, 2007.

[26] L. D. S. Coelho, "A quantum particle swarm optimizer with chaotic mutation operator," *Chaos, Solitons & Fractals*, vol. 37, no. 5, pp. 1409–1418, 2008.

[27] M. Xi, J. Sun, and W. Xu, "Quantum-behaved particle swarm optimization with elitist mean best position," in *Complex Systems and Applications—Modeling, Control and Simulations*, pp. 1643–1647, 2007.

[28] M. Xi, J. Sun, and W. Xu, "An improved quantum-behaved particle swarm optimization algorithm with weighted mean best position," *Applied Mathematics and Computation*, vol. 205, no. 2, pp. 751–759, 2008.

[29] W. Fang, J. Sun, and W.-B. Xu, "Improved quantum-behaved particle swarm optimization algorithm based on differential evolution operator and its application," *Journal of System Simulation*, vol. 20, no. 24, pp. 6740–6744, 2008.

[30] M. L. Xi and J. Sun, "A modified binary quantum-behaved particle swarm optimization algorithm with bit crossover operator," *Advanced Materials Research*, vol. 591–593, pp. 1376–1380, 2012.

[31] J. Wang and Y. Zhou, "Quantum-behaved particle swarm optimization with generalized local search operator for global optimization," in *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence: Third International Conference on Intelligent Computing, ICIC 2007, Qingdao, China, August 21–24, 2007. Proceedings*, vol. 4682 of *Lecture Notes in Computer Science*, pp. 851–860, Springer, Berlin, Germany, 2007.

[32] K. Yang and H. Nomura, "Quantum-behaved particle swarm optimization with chaotic search," *IEICE Transactions on Information and Systems*, vol. 91.D, no. 7, pp. 1963–1970, 2008.

[33] J. Liu, J. Sun, W. Xu, and X. Kong, "Quantum-behaved particle swarm optimization based on immune memory and vaccination," in *Proceedings of the 2006 IEEE International Conference on Granular Computing*, pp. 453–456, Atlanta, Ga, USA, May 2006.

[34] J. Liu, J. Sun, and W. Xu, "Quantum-behaved particle swarm optimization with immune operator," in *Proceedings of the 16th International Symposium on Methodologies for Intelligent Systems*, pp. 77–83, Bari, Italy, September 2006.

[35] J. Liu, J. Sun, and W. Xu, "Improving quantum-behaved particle swarm optimization by simulated annealing," in *Proceedings of the 2006 International Conference on Intelligent Computing*, pp. 130–136, Kunming, China, August 2006.

[36] B. Qu, Z. Jiao, and B. Xu, "Research on quantumbehaved particle swarms cooperative optimization," *Computer Engineering and Applications*, vol. 44, no. 7, pp. 72–74, 2008.

[37] S. Lu and C. Sun, "Coevolutionary quantum-behaved particle swarm optimization with hybrid cooperative search," in *Proceedings of the Pacific-Asia Workshop on Computational Intelligence and Industrial Application (PACIIA '08)*, pp. 109–113, December 2008.

[38] K. F. Man, T. M. Chan, K. S. Tang, and S. Kwong, "Jumping-genes in evolutionary computing," in *Proceedings of the 30th Annual Conference of IEEE Industrial Electronics Society*, vol. 2, pp. 1268–1272, November 2004.

[39] K. Li, S. Kwong, R. Wang, K.-S. Tang, and K.-F. Man, "Learning paradigm based on jumping genes: a general framework for enhancing exploration in evolutionary multiobjective optimization," *Information Sciences*, vol. 226, pp. 1–22, 2013.

[40] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, May 1998.

[41] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '99)*, pp. 1951–1957, IEEE, Washington, DC , USA, July 1999.

[42] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '07)*, pp. 120–127, IEEE, Honolulu, Hawaii, USA, April 2007.

[43] B. R. Secrest and G. B. Lamont, "Visualizing particle swarm optimization—gaussian particle swarm optimization," in *Proceedings of the Swarm Intelligence Symposium (SIS '03)*, pp. 198–204, IEEE, April 2003.

[44] J. Kennedy, "Bare bones particle swarms," in *Proceedings of the 2003 IEEE Conference on Swarm Intelligence Symposium*, pp. 80–87, April 2003.

[45] R. A. Krohling and L. D. S. Coelho, "PSO-E: Particle swarm with exponential distribution," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 1428–1433, July 2006.

[46] T. J. Richer and T. M. Blackwell, "The Lévy particle swarm," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 808–815, July 2006.

[47] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[48] J. J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Proceedings of the IEEE Conference on Swarm Intelligence Symposium (SIS '05)*, pp. 124–129, June 2005.

[49] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.

[50] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[51] Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 39, no. 6, pp. 1362–1381, 2009.

[52] B. Akay and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.

[53] J. Xie, Y. Zhou, and H. Chen, "A novel bat algorithm based on differential operator and Lévy flights trajectory," *Computational Intelligence and Neuroscience*, vol. 2013, Article ID 453812, 13 pages, 2013.

[54] A. J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J. J. Durillo, and A. Beham, "AbYSS: adapting scatter search to multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 4, pp. 439–457, 2008.

[55] Y. Qi, F. Liu, M. Liu, M. Gong, and L. Jiao, "Multi-objective immune algorithm with Baldwinian learning," *Applied Soft Computing Journal*, vol. 12, no. 8, pp. 2654–2674, 2012.