Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 1805198, 12 pages
https://doi.org/10.1155/2019/1805198

Research Article

A Quantum Particle Swarm Optimization Algorithm with Teamwork Evolutionary Strategy

Guoqiang Liu,[1] **Weiyi Chen**,[1] **Huadong Chen**,[1] **and Jiahui Xie**[2]

[1]*School of Ordnance Engineering, Naval University of Engineering, No. 716, Jiefang Avenue, Wuhan, 430033 Hubei, China*
[2]*School of Business, Hubei University, No. 368, Youyi Avenue, Wuhan, 430062 Hubei, China*

Correspondence should be addressed to Guoqiang Liu; sdrzliuguoqiang@126.com

Received 22 January 2019; Accepted 12 March 2019; Published 14 April 2019

Guest Editor: Qijun Zheng

The quantum particle swarm optimization algorithm is a global convergence guarantee algorithm. Its searching performance is better than the original particle swarm optimization algorithm (PSO), but the control parameters are less and easy to fall into local optimum. The paper proposed teamwork evolutionary strategy for balance global search and local search. This algorithm is based on a novel learning strategy consisting of cross-sequential quadratic programming and Gaussian chaotic mutation operators. The former performs the local search on the sample and the interlaced operation on the parent individual while the descendants of the latter generated by Gaussian chaotic mutation may produce new regions in the search space. Experiments performed on multimodal test and composite functions with or without coordinate rotation demonstrated that the population information could be utilized by the TEQPSO algorithm more effectively compared with the eight QSOs and PSOs variants. This improves the algorithm performance, significantly.

1. Introduction

With the development of real-life engineering technology, the related optimization problems are more and more complicated. Experience shows that using traditional methods to solve these complex problems is inefficient. In order to cope with this limitation, a variety of artificial intelligence algorithms, such as particle swarm optimization- (PSO-) based memetic algorithm (MA) [1], simulated annealing algorithm (SABA) [2], ant colony optimization (MSCRSP–ACO) [3], bat algorithm (BA) [4], firefly algorithm [5], biogeography-based optimization (BBO) [6], the cuckoo search algorithm [7, 8], charged system search (CSS) [9], gravitational search algorithm [10], grey wolf optimizer (GWO) [11], ant colony optimization (ACO) algorithm [12], and new adaptive ant algorithm [13], have been proposed and applied to solve optimization problems. The particle swarm optimization (PSO) method is a broader class of swarm intelligence methods for solving GO problems. The method was originally proposed by Kennedy to simulate the social behaviour of flocks and was first introduced as an optimization method in 1995. Unlike other evolutionary algorithms that use evolutionary operators to manipulate individuals, PSO relies on the exchange of information between individuals. Each particle in the particle swarm algorithm flies in the searching space at a certain speed, and speed is dynamically adjusted according to the information of the particle itself.

Since 1995, many have been made to improve the performance of PSO by experts and scholars [14, 15]. However, Van den Bergh [16] proved that PSO is not a global optimization algorithm. Sun et al. [17] combined the quantum theory with the particle swarm optimization (PSO) algorithm to establish a quantum behavior particle swarm optimization algorithm (QPSO). This algorithm ensures finding the global optimal solution in the search space. Experimental results performed on various benchmark functions demonstrate that the proposed algorithm could improve the standard PSO algorithm. The global convergence of QPSO guarantees that the global optimal solution is calculated in the case of infinite searching iterations. However, in practical problems, this condition is unrealistic because any optimization algorithm allows only a limited number of iterations to search for the optimal solution. When solving complex problems, QPSO is also prone to fall into local optimum or slow convergence.

Experts and scholars have developed various strategies to improve the convergence speed and global optimal performance of QPSO. Liu et al. [18] introduced simulated annealing into the QPSO algorithm; Chen et al. [19] presented an improved GSO optimizer with quantum-behaved operator for scroungers; Li et al. [20] presented cooperative quantum-behaved particle swarm optimization (CQPSO); Li et al. [21] proposed a dynamic-context cooperative quantum-behaved particle swarm optimization algorithm; H Long et al. [22, 23] proposed a new algorithm PDCQPSO and diversity maintained into QPSO; Leandro dos Santos Coelho [24, 25] presented a novel quantum-behaved PSO (QPSO) using chaotic mutation operator, mutation operator with Gaussian probability distribution; Qin et al. [26] presented a hybrid improved QPSO algorithm (LTQPSO); Tian et al. [27] proposed a heuristic method known as quantum-behaved particle swarm optimization (QPSO) to solve the inverse advection–dispersion problem.

However, it is quite difficult to improve global search capabilities and speed up convergence at the same time. If you try to avoid falling into the local optimal state, the convergence speed may be slower, so the QPSO algorithm needs to be specifically modified to be suitable for the practical optimization problem.

In the group intelligence algorithm, how to balance the global and local search ability is a key issue that is the balance between exploration and gain. Particle swarms have some shortcomings in this respect. In terms of exploration, the fast convergence characteristics easily lead to premature convergence and, in the aspect of gain, the single search mode of particle swarm has the problem of insufficient convergence precision, and, in the multiobjective particle swarm, the frequent replacement of global optimal solution leads to more prominent problem of exploration and development. In summary, this paper proposes a new cooperative evolutionary strategy teamwork particle swarm optimization algorithm (TEQPSO) for solving nonlinear numerical problems including unimodal and multipeak test functions. The main difference between the comprehensive learning particle swarm optimizer (CLPSO) [28] and traditional QPSO is that CLQPSO uses cooperative evolution strategy to generate local attractors for each particle, as follows: (1) using two operators (cross-sequential quadratic programming operators and the cooperative learning strategy consisting of Gaussian chaotic mutation operator generates local attractors; (2) it uses a cooperative learning strategy composed of two operators to generate local attractors; (3) a probability factor p is employed to control the implementation of these two operators to realize a balance between exploration and gain of the teamwork evolutionary strategy.

2. Teamwork Evolutionary Strategy Quantum Particle Swarm Optimization

2.1. Cooperative Evolutionary Strategy Quantum Particle Swarm Optimization. Consider that the particle moved on a one-dimensional well δ. Now, its position (x) could be calculated from the following equation:

$$x = p \pm \frac{L}{2} \ln\left(\frac{1}{u}\right) \tag{1}$$

where p is the particle motion centre. In QPSO algorithm, it is also called the *attractor* of the particle. L is the characteristic length of the potential well δ where its value is directly related to the convergence speed and searching ability of the algorithm. u is a random number with a uniform distribution function in the range $(0, 1)$.

Parameter L should be appropriately determined through the QPSO algorithm. This parameter could be calculated from the following equations:

$$L_{i,j} = 2\beta \cdot \left\| mbest_j - x_{i,j} \right\| \tag{2}$$

where

$$mbest = \frac{1}{N} \sum_{i=1}^{N} pbest_i \tag{3}$$

where $pbest_i$ is the optimal position of the individual in the search history of the particle x_i. β is the Contraction-Expansion (CE) factor. This parameter should be decreased during the algorithm running.

In the QPSO algorithm, each particle takes the weighted average position of the individual historical optimal position and the optimal position of the group history as its own attraction point. This calculation method could be concluded from the particle motion trajectory results [29, 30]. Although this kind of calculation is simple, it has two obvious defects: (i) apart from its own experience learning, each particle position depends on the historical optimal position of the group. This leads to rapid decline in the diversity of large groups which reduces the algorithm capability for solving complex multipeak optimization problems. (ii) The possible distribution space of each particle attraction point gradually decreases during the evolution process of the algorithm. The particles are limited to a rectangle with vertices $pbest_{i,t}$ and $gbest_{i,t}$. The $attractor_{i,t}$ gradually approaches $gbest_{i,t}$. Finally, this algorithm could not jump out of local optimum in the final stage. Now, we have

$$attractor_{i,t} = u_{i,t} pbest_{i,t} + \left(1 - u_{i,t}\right) pbest_{\mathrm{b},t} + \Delta_{i,t} \tag{4}$$

where $u_{i,t}$ is a random number with uniform distribution function over the interval $[0, 1]$. The subscript i is the number of a randomly selected particle with the best fitness value. Moreover, the ratio of the particles is selected as $m \in (0, 1]$. $\Delta_{i,t} = \{\Delta_{i,t}^1, \Delta_{i,t}^2, \ldots, \Delta_{i,t}^D\}$ is a perturbation vector defined as

$$\Delta_{i,t} = \frac{pbest_{\mathrm{a},t} - pbest_{c,t}}{2} \tag{5}$$

where subscripts b and c are two randomly selected particles in the group and $\mathrm{a} \neq \mathrm{b} \neq \mathrm{c} \neq \mathrm{i}$.

The following observation (update) equation for the particle position in the QPSO algorithm could be obtained. Consider

$$x_{i,t} = attractor_{i,t} \pm 2\beta \left\| mbest_j - x_{i,t} \right\| \tag{6}$$

2.2. TE Strategy. In the evolution procedure, some useful information about individual particles and the global optimal position may lose through the algorithm. In addition, the movement of some attractors in a worse direction leads to poor fitness in the next evolutionary process. Therefore, to improve the algorithm performance, the effective information on the individual and global optimal positions of the particles should be utilized through an appropriate method. To improve the optimization ability of the algorithm using the mentioned information, the cross-over algorithm and local search are incorporated into a cross-sequential quadratic programming (SQP) algorithm. The algorithm falls into local optimum in its final stage. This means that the individual and global optimum positions of the particles in the population are very close to each other or even the same. Considering the mentioned problem, a Gaussian chaotic mutation operator is proposed to improve the population diversity and jump out of the local optimum.

2.3. CROSS-SQP Operator

(1) Local Search Based on the SQP Algorithm. The SQP method [31] is one of the nonlinear programming methods for constrained optimization. In the SQP algorithm, the quasi-Newton update method is utilized. In this method, the Lagrange function's Hessian function in each iteration is calculated, approximately. Then, the quadratic programming subquestion is generated to calculate the searching direction of the line searching process. Thus, the following optimization problem could be written:

$$\min_{d \in R^n} \quad \frac{1}{2} d^T H_k d + \nabla f\left(x_k\right)^T d \tag{7}$$

$$\text{s.t.} \quad \nabla g\left(x_k\right)^T d + g\left(x_k\right) \leq 0 \tag{8}$$

where the subscript k is the current number of iterations; H is the *Hession* matrix that could be approximated by the quasi-Newton and the BFGS methods. The solution of the quadratic programming subproblem is employed as the linear search direction for the next iteration.

In summary, the SQP method mainly includes three stages: (i) updating the Hessian matrix of the Lagrangian function; (ii) calculating the line search and performance function; and (iii) solving the quadratic programming problem.

(2) Cross-Algorithm. The crossover operator is derived from the genetic algorithm [32]. Through cross-operation, the information exchange between individuals in the group could be performed. As a result, the excellent genes are gradually retained during the evolutionary process. Accordingly, the group evolves into a good direction. To solve complex multipeak optimization problems through the QPSO algorithm, the group diversity and the algorithm performance should be improved by incorporating the crossover operator into the algorithm. In the beginning, the measurement position $X_{i,t+1}$ corresponding to the particle x_i is generated. Then, $X_{i,t+1}$ and the individual historical optimal position $pbest_{i,t}$ are discretely intersected to generate a test position $Q_{i,t} = \{q_{i,t}^1, q_{i,t}^2, \ldots, q_{i,t}^D\}$. The crossover equation is given as

$$q_{i,t+1}^j = \begin{cases} x_{i,t+1}^j & \text{rand}_j\,(0,1) < \text{pbest or j} = \text{j}_{rand} \\ p_{ci,t}^{\ j} & other \end{cases} \tag{9}$$

where $\text{rand}_j(0,1)$ is a random number with a uniform distribution function in the interval $[0,1]$; j_{rand} is an integer number that is uniformly generated in the interval $[1, \text{D}]$; p_c is the crossover operation probability. This operation is similar to the binomial operation in the differential evolution algorithm [33].

Finally, the optimal historical position of the updated particle is calculated as

$$pbest_{i,t+1}^j = \begin{cases} q_{i,t+1}^j & f\left(q_{i,t+1}^j\right) < f\left(pbest_{i,t+1}^j\right) \\ pbest_{i,t+1}^j & other \end{cases} \tag{10}$$

where $f(*)$ is the fitness function. Without loss of generality, this paper only considers the minimization problem.

Different optimization problems require different optimal parameters at different stages of evolution. Therefore, finding an appropriate value p for all problems is difficult. In this paper, this parameter is directly encoded into each particle by an adaptive method to attain an adaptive control method. The particle i in the extended coded group could be described as

$$X_{i,t} = \left\{x_{i,t}^1, x_{i,t}^2, \ldots, x_{i,t}^D, pbest_{i,t}\right\} \tag{11}$$

For each group particle, the crossover probability could be updated according to the following adaptation rule:

$$pbest_{i,t+1} = \begin{cases} \text{rand}_1\,(0,1) & \text{rand}_2\,(0,1) < \lambda \\ pbest_{i,t} & other \end{cases} \tag{12}$$

where λ is the probability of updating parameter $pbest_i$.

The individual historical optimal position $pbest_{i,t}$ could be calculated through the CROSS-SQP operator. It could be employed as the attractor $attractor_{i,t}$ of particle x_i.

2.4. Gaussian Chaotic Mutation Operator. Necessary conditions for global convergence of the QPSO algorithm could be obtained through a convergence analysis approach; that is, each particle x_i, $j(t)$ converges on $k_i = (k_{i1}, k_{i2}, \ldots k_{in})^{\text{T}}$. Its expression is

$$k_{i,j}\,(t) = \varphi \bullet pbest_{i,j}\,(t) + \left(1 - \varphi\right) \bullet gbest_j\,(t) \tag{13}$$

where $\varphi \in (0,1)$, $pbest_{i,j}(t)$ is the historical optimal position of the particle and $gbest_j(t)$ is the optimal particle position.

In the TEQPSO algorithm, the dimension of the particle is considered as one. The chaotic mapping relationship is established by calculating the distance between positions $x_i(t)$ and $k_i(t)$ for particle i. In addition, the chaotic search range for each generated particle is dynamically adjusted,

iteratively. The searching range of particles is calculated as follows:

$$x_{i,\min} = \begin{cases} k_i(t) - \dfrac{\left|k_i(t) - x_i(t)\right|}{u} & k_i(t) \neq x_i(t) \\ k_i(t) - \left|k_i(t) \cdot z\right| & k_i(t) = x_i(t) \end{cases} \quad (14)$$

$$x_{i,\max} = \begin{cases} k_i(t) + \dfrac{\left|k_i(t) - x_i(t)\right|}{u} & k_i(t) \neq x_i(t) \\ k_i(t) + \left|k_i(t) \cdot (1-u)\right| & k_i(t) = x_i(t) \end{cases} \quad (15)$$

The Gaussian chaotic variation in [34] could be employed to improve the single-objective PSO algorithm into the multiobjective PSO algorithm. Accordingly, we have

$$x_i^k(t) = \begin{cases} x_i^k(t) + \alpha g_i^k \left|l_g - x_i^k(t)\right| \left(x_{i,\max} - x_i'^k(t)\right) & \alpha \geq 0 \\ x_i^k(t) + \alpha g_i^k \left|l_g - x_i^k(t)\right| \left(x_i'^k(t) - x_{i,\min}\right) & \alpha \leq 0 \end{cases} \quad (16)$$

where α is the random number in the interval $[-1, 1]$, $|l_g - x_i^k|$ is the distance from the global optimal position of the particle i and $x_i'^k(t)$ is the introduced logistic chaotic map value that is calculated such that the omnidirectional ergodicity property for the variation is realized, or

$$x_i'^k(t) = x_{i,\min} + r_i^k \left(x_{i,\max} - x_{i,\min}\right) \quad (17)$$

$$r_i^k = 4 r_i^k \left(1 - r_i^k\right) \quad (18)$$

The initial value of r_i^k is considered as $r_{i,0}^k \in \text{rand}(0, 1)$ and $r_{i,0}^k \neq 0.25, 0.5, 0.75$, $x_{i,\max}$ and $x_{i,\min}$ are the upper and lower bounds of the particle search space, and $s_i^k(t)$ is a Gaussian process with the following distribution function:

$$s_i^k(t) = \exp\left(-\frac{\left(p_g(t) - x_i^k(t)\right)^2}{2\sigma^2(k)}\right) \quad (19)$$

$$\sigma^2(k) = \sigma^2 e^{-k/c} \quad (20)$$

where k is the current number of iterations, T is the maximum number of iterations, and σ_0 is the initial variance.

The detailed operation flow of the Gaussian chaotic mutation operator is shown in Table 2. Before applying Gaussian chaotic mutation operators, the number of Gaussian variations should be determined first. If the number of variations is too large, the calculating time of the algorithm increases greatly while if the number of mutations is too small, the probability that the algorithm jumps out of the local optimum decreases.

2.5. "Exploration" and "Gain" Analysis in Cooperative Evolution Strategy. "Exploration" in the algorithm refers to adding cross-sequential quadratic programming operators to enhance the local search capability of the optimization algorithm. The "gain" refers to accessing the areas that were visited in the historical search to enhance the global search or the refining capability of the optimization algorithm [32]. The crossover operator and SQP strategy have been employed in the cross-sequential quadratic programming operator to strengthen the local search capability or "exploration" in the teamwork evolutionary strategy. The descendants of the Gaussian chaotic mutation operator generated by the Gaussian chaotic mutation may appear in the new searching space. Therefore, the Gaussian chaotic operator is nearer to the "gain" in the teamwork evolutionary strategy.

The TEQPSO algorithm could be summarized through the following steps.

(1) Generate particle groups $P_i = (x_1, x_2, \ldots, x_n)$ in the decision space, randomly. Now, initialize the global and individual optimal values. Calculate the fitness value $f(x_i)$ for each particle, the maximum number of iterations T, and the upper and lower bounds ($x_{i,\max}$ and $x_{i,\min}$) for the particle x_i.

(2) Calculate the current optimal position $gbest_i$ for particle x_i and the optimal position $gbest_i$ for all particles. Let $stop_i$ be zero for each particle x_i, where $stop_i$ indicates that individual optimal position for the particle x_i remains unchanged.

(3) For the acquisition method of the attractor $attractor_i$ of the particle x_i: (i) if $stop_i \leq T$, then $attractor_i$ is obtained according to the CROSS-SQP operator; (ii) if $stop_i \neq T \& pbest_i = gbest$, then $attractor_i$ is obtained according to the Gaussian chaotic operator. The Gaussian chaotic operator operation is carried out by the steps given in Table 2; (iii) if $stop_i \neq T \ \& \ pbest_i \neq gbest$, according to the CROSS-SQP operator $attractor_i$, this operator is applied using the corresponding steps given in Table 1. Otherwise, $attractor_i$ is obtained according to the Gaussian chaotic mutation operator where its steps are illustrated in Table 2.

(4) Update the particle swarm and fitness values. If $f(x_{i+1}) > f(x_i), pbest_i = x_i, stop_i = 0$; if $f(x_{i+1}) < f(x_i)$, $pbest_i = pbest_{i+1}, stop_i = stop_i + 1$.

(5) Determine whether the algorithm satisfies the termination condition; then output $pbest_i$. Otherwise, return to step (3).

3. Test Functions and Settings

3.1. Test Function. To evaluate the performance of the algorithm, the optimization function defined in IEEE-CEC 2014 is selected, as shown in Table 3. The MATLAB code for these functions can be downloaded from http://www.ntu.edu.sg/home/epnsugan/. Table 3 gives some main information about the relevant test functions. It could be seen from Table 3 that the twelve test functions could be divided into four categories, in which the function types of functions F_1 and F_2 are identical to single-mode functions, the functions of $F_3 \sim F_6$ are consistently multimodal functions, F_7 and F_8 are rotating multimode, and $F_9 \sim F_{12}$ are multiobjective test functions. For the state function, to rotate the multimodal function, first construct an orthogonal matrix N and multiply the variables x by the multimodal function to obtain a new variable y. This variable could be utilized to calculate the fitness value of the function [38].

Table 1: CROSS-SQP operator algorithm.

CROSS-SQP operator
(1) Update the Hessian matrix of the Lagrangian function and calculate the attraction points of the particles according to Equations (1)-(6);
(2) Solve the SQP problem using the BFGS method according to Equations (7) and (8);
(3) Generate the measurement position of the particles;
(4) Evaluate the fitness value of the test location;
(5) Update the crossover probability
(6) Select the best individual $pbest_{i,t}$ in the calculated sample as $attractor_{i,t}$ of the particle x_i

Table 2: Gaussian chaotic mutation operator flow.

Gaussian chaotic mutation algorithm
(1) Calculate the attractor fixed point $attractor_i$, according to Equation (6).
(2) Set the chaotic map search interval $x_{i\max}$ and $x_{i\min}$ according to Equations (14) and (15).
(3) The particle x_i is subjected to the chaotic mapping operation corresponding to Equations (17) and (18).
(4) Apply the Gaussian mutation operation on particle x_i according to Equations (19) and (20).
(5) Is it better to evaluate the fitness function of the current value of the particle of the chaotic map sequence? $pbest_{i,j}(t)$ and $gbest_j(t)$ update $pbest_{i,j}(t)$ and $gbest_j(t)$.
(6) Select the best individual $pbest_{i,t+1}$ in the calculated sample as the attractor $attractor_{i,t}$ of x_i.

Table 3: CEC 2014 related test functions.

Functions	Formulations	Initialization	Max.range
Sphere	$f_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^{\mathrm{D}}$	$[-100, 50]^{\mathrm{D}}$
Rosenbrock	$f_2(x) = \sum_{i=1}^{D-1} \left(100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right)$	$[-2048, 2048]^{\mathrm{D}}$	$[-2048, 2048]^{\mathrm{D}}$
Ackley	$f_3(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\left(2\pi x_i\right)\right) + 20 + e$	$[-32768, 32768]^{\mathrm{D}}$	$[-32768, 16]^{\mathrm{D}}$
Griewank	$f_4(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^{\mathrm{D}}$	$[-600, 200]^{\mathrm{D}}$
Step	$f_5(\mathrm{x}) = \sum_{i=1}^{D} \left(\left\lfloor x_i + 0.5\right\rfloor\right)^2$	$[-100, 100]^{\mathrm{D}}$	$[-100, 100]^{\mathrm{D}}$
Rastrigin	$f_6(x) = \sum_{i=1}^{D} \left(x_i^2 - 10\cos\left(2\pi x_i\right) + 10\right)$	$[-5.12, 5.12]^{\mathrm{D}}$	$[-5.12, 2]^{\mathrm{D}}$
Rotated Griewank	$f_7(x) = \frac{1}{4000}\sum_{i=1}^{n} y_i^2 - \prod_{i=1}^{n}\cos\left(\frac{y_i}{\sqrt{i}}\right) + 1,\ y = N * x$	$[-600, 600]^{\mathrm{D}}$	$[-600, 200]^{\mathrm{D}}$
Rotated Rastrigin	$f_8(x) = \sum_{i=1}^{D} \left(y_i^2 - 10\cos\left(2\pi y_i\right) + 10\right),\ y = N * x$	$[-0.5, 0.5]^{\mathrm{D}}$	$[-0.5, 0.2]^{\mathrm{D}}$

Among them, $N = \begin{bmatrix} N_{11} & N_{12} & \cdots & N_{1D} \\ N_{21} & N_{22} & \cdots & N_{2D} \\ \cdots & \cdots & & \cdots \\ N_{D1} & N_{D2} & \cdots & N_{DD} \end{bmatrix}$, $x = [x_1, x_2, \ldots, x_D]^T$, and $y = [y_1, y_2, \ldots, y_D]^T$; then $y_i = n_{i1}x_1 + n_{i2}x_2 + \ldots + n_{iD}x_D$.

3.2. Algorithm Parameter Analysis. The TEQPSO algorithm mainly uses three new parameters: probability factor p, the number of mutations c, and the maximum number of iterations T. These three parameters could be analysed accurately by selecting each function from three types of functions ($f_1, f_3, f_4, f_6, f_7, f_8$). The TEQPSO algorithm could work effectively if appropriate values are chosen for p, c, and T. To find the optimal values for these parameters, two of them are considered constant while the other one changes within the setting range. Now, the parameters giving the best fitness values for $f_1, f_3, f_4, f_6, f_7, f_8$ functions are selected as the optimal parameters. This ensures the accuracy of the algorithm and reduces the statistical error. The experimental results given below are obtained from 50 independent running statistics of the algorithm. For each test function, each algorithm performs a subfunction evaluation (FEs) 2.5×10^5 times.

(1) Probability Factor p. In the TEQPSO algorithm, the probability factor p is introduced to decide whether to use the crossover operator or the Gaussian chaotic mutation operator. This makes an effective balance between the local

Table 4: Parameter settings of each algorithm.

References	Algorithm	Parameter settings
[34]	QPSO	$\alpha = 1 \sim 0.5$
[35]	CSPSO	$w = 0.4,\ c_1 = c_2 = 2,\ p_v = 0.8$
[28]	CLPSO	$w = 0.9 \sim 0.4,\ c = 1.495,\ V_{\max} = 0.2 \times range$
[36]	WQPSO	$\alpha = 1.5 \sim 0.5, \beta = 1.0 \sim 0.5$
[20]	CQPSO	$\alpha = 1 \sim 0.5, p_0 = 0.9, \tau = 0.2$
[37]	COQPSO	$\alpha = 1 \sim 0.5,\ G = 5,\ k = 2,\ p = 0.7$
This paper	TEQPSO	$\alpha = 1 \sim 0.5,\ T = 9,\ c = 8,\ p = 0.15$

and global search effects. To investigate the effect of different probability factors on the algorithm performance and find the best probability factor, the number of mutations c=8 and the stagnation factor $T = 9$ are selected. Figure 1(a) shows the average result of 50 independent running times of the TEQPSO algorithm. As what could be seen from Figure 1(a), good results are obtained for most of the test functions for p=0.15. Thus, p=0.15 could be a suitable value for the probability factor of the TEQPSO algorithm.

(2) Number of Mutations c. During the chaotic operator execution, the number of mutations c determines the number of chaotic variations in the particles. If large value is chosen for this parameter, the algorithm will waste too much computation time. Considering small values for this parameter reduces the capability of the algorithm to jump out of the local optimum. Therefore, an appropriate value for number of mutations should be chosen. Figure 1(b) gives the statistical results of the algorithm running for different values of c. The probability factor p=0.15 and the stagnation factor T=9 are considered. It could be seen from Figure 1(b) that superior results for most test functions could be obtained by choosing $c = 8$.

(3) Stagnation Factor T. In the TEQPSO algorithm, if the particle does not change its individual optimal position in Tth iteration, a cooperative learning strategy is adopted to construct the attractor so that the particle jumps out of the current position. Therefore, the stop factor T could be utilized to control the constructing attractor frequencies using the teamwork evolutionary strategy. To derive the appropriate stopping factor, the probability factor p=0.15 and the number of mutations c=8 are selected. The running results of the algorithm for different values of the stagnation factors are presented in Figure 1(c). It is obvious that the algorithm gives better results for most of the test functions for T=9.

3.3. Related Algorithm Parameter Settings. We make two groups of experiments, with each to test the standard QPSO [34], the CSPSO [35], the CLPSO [28], the WQPSO [36], the CQPSO [20], the COQPSO [37], and TEQPSO in this paper. The parameter settings of the TEQPSO algorithm and other algorithms are shown in Table 4. When solving the 10-dimensional (10D) problem, the population size is set to 10, and the maximum fitness evaluation (FE) is set to 400000. When solving the 30-dimensional (30D) problem, the population size is set to 40 and the maximum FE is set to 400000. All experiments were performed 30 times, and the mean and standard deviation from the calculation results on each algorithm were given. In solving multiobjective optimization problems, because the PSO has higher computational efficiency, usually the fitness calculation takes the most time. Therefore, this article does not compare the algorithm-related calculation times of these algorithms.

4. Experimental Results and Analysis

4.1. Results of 10-Dimensional (10D) Problems. Table 5 shows the mean and variance of seven algorithms running 30 times for eight test functions. As can be seen from Table 5, the TEQPSO algorithm obtains the highest f_1 precision solution to the unimodal function; the mean and standard deviation values of TEQPSO are 0.00e+00 and 0.00e+00, followed by WQPSO, COQPSO, CQPSO, QPSO, CLPSO, and CSPSO. The CLPSO algorithm has the best performance in the unimodal function f_2. The mean and standard deviation values of CLPSO are 6.18e+00 and 2.89e+00, followed by CSPSO, COQPSO, QPSO, WQPSO, TEQPSO, and CQPSO. Then, the TEQPSO algorithm achieves the best results of the multimodal function $f_3 \sim f_6$, with the mean and variance values of 0.00e+00 and 0.00e+00. For the other six algorithms of functions f_3, f_4, and f_6, they fall into local optimum; only TEQPSO obtains the best solution. The results show that the TEQPSO algorithm can maintain better population diversity, enhance the ability of the algorithm to jump out of local optimum, and have higher search accuracy on complex multimodal functions. The functions f_7 and f_8 are rotational mode function, and the other five algorithms are difficult to solve. Only CLPSO and TEQPSO obtain better calculation results. CLPSO obtains the highest precision solution in the rotation function f_7. The mean and standard deviation values of CLPSO are 6.41e-05 and 3.58e-05, followed by CSPSO, CQPSO, WQPSO, COQPSO, QPSO, and TEQPSO. And TEQPSO obtains the most accurate solution in the rotation function f_8. The mean and standard deviation of TEQPSO are 1.88e+00 and 1.05e+00, followed by WQPSO, CQPSO, COQPSO, CLPSO, QPSO, and CSPSO. In summary, the quality of TEQPSO solutions to functions f_1, f_3, f_4, f_5, f_6, and f_8 is significantly better than the other six algorithms.

In order to compare the convergence of several algorithms, Figures 2–9 show the convergence of the seven algorithms under eight test functions. It can be seen from

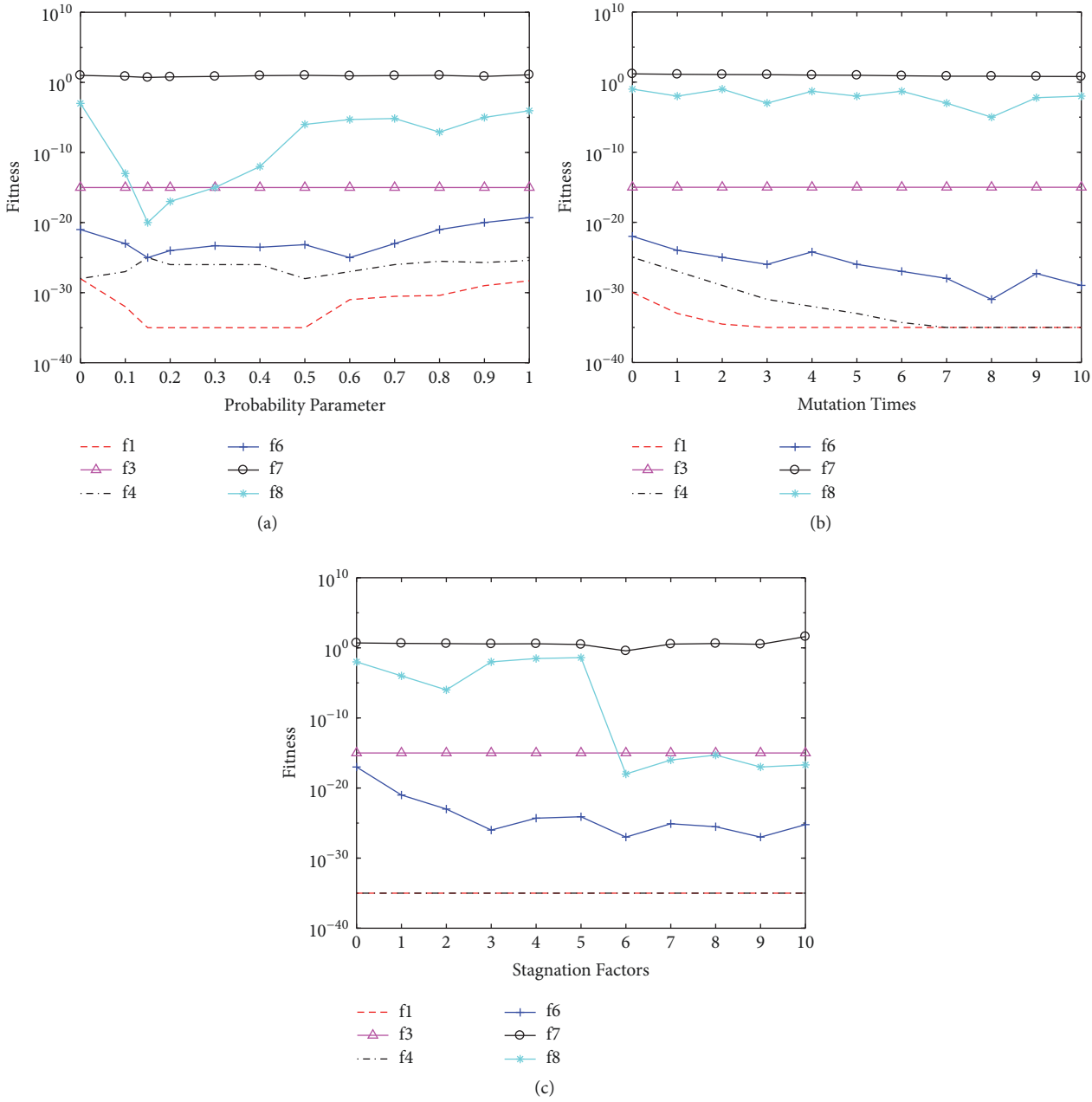

FIGURE 1: The change in the fitness value of the test function for different parameters: (a) probability parameter; (b) mutation times; (c) stagnation factors.

the figure that, compared with the TEQPSO, the other four quantum particle swarm algorithms have obvious faster convergence speed, but they are also very easy to fall into local optimum. The other two particle swarm optimization algorithms converge slowly and perform better in unimodal functions. Therefore, after adopting the teamwork evolutionary strategy, compared with the other four quantum particle swarm optimization algorithms, the convergence speed of the TEQPSO is slowed down, but the ability of the algorithm to jump out of the local optimal region is also significantly enhanced.

4.2. Results of the 30-Dimensional (30D) Problem. Table 6 shows the mean and variance of seven algorithms running 30 times on the eight test functions. The best results of seven algorithms are shown in bold.

It can be seen from Table 6 that TEQPSO algorithm obtains the most accurate solution in the unimodal function f_1, which jumps out of the local optimum; the mean and standard deviation values of TEQPSO are 0.00e+00 and 0.00e+00, followed by WQPSO, COQPSO, CQPSO, QPSO, CLPSO, and CSPSO. The CSPSO algorithm has the best performance in the unimodal function f_2; the mean and

Table 5: Comparison results of seven algorithms in 10-dimensional test function.

Func	QPSO Mean (Std)	CSPSO Mean (Std)	CLPSO Mean (Std)	WQPSO Mean (Std)	CQPSO Mean (Std)	COQPSO Mean (Std)	TEQPSO Mean (Std)
f_1	3.11e-41 (2.84e-29)	1.52e-23 (7.29e-22)	6.55e-31 (3.89e-31)	9.11e-56 (5.36e-57)	6.04e-48 (6.22e-49)	3.17e-52 (3.00e-52)	**0.00e+00 (0.00e+00)**
f_2	5.18e+01 (4.70e-01)	3.22e+01 (1.79e+01)	**6.18e+00 (2.89e+00)**	7.01e+01 (4.28e-01)	3.56e+01 (2.33e+01)	1.08e+02 (6.44e+01)	7.25e+01 (4.34e-01)
f_3	4.00e-03 (1.72e-03)	3.74e-13 (2.00e-12)	6.82e-15 (3.17e-15)	5.70e-19 (3.78e-19)	8.03e-02 (5.34e-02)	6.22e-23 (3.25e-24)	**0.00e+00 (0.00e+00)**
f_4	2.02e-02 (2.15e-02)	8.69e+01 (9.55e+00)	6.04e+00 (6.77e+00)	7.33e-2 (6.84e-01)	4.17e+00 (2.86e+00)	1.84e-01 (2.00e-01)	**0.00e+00 (0.00e+00)**
f_5	**0.00e+00 (0.00e+00)**	2.08e-03 (2.33-03)	3.41e-03 (3.27e-03)	**0.00e+00 (0.00e+00)**	**0.00e+00 (0.00e+00)**	**0.00e+00 (0.00e+00)**	**0.00e+00 (0.00e+00)**
f_6	4.80e+00 (1.32e+00)	5.87e+05 (1.77e+05)	5.01e+04 (1.63e+04)	4.05e+00 (9.40e+00)	5.02e+01 (1.59e+01)	4.35e+01 (1.01e+01)	**3.85e+00 (1.25e-03)**
f_7	7.08e+00 (1.08e-01)	5.92e-02 (7.37e-01)	**6.41e-05 (3.58e-05)**	1.07e+00 (3.45e-01)	5.74e-01 (8.43e+00)	3.18e+00 (8.50e-00)	7.02e+01 (3.39e+00)
f_8	8.24e+01 (4.03e+01)	9.58e+01 (2.68e+01)	5.88e+01 (2.95e+01)	2.01e+00 (1.55e+00)	9.97e+00 (2.47e+01)	1.31e+01 (3.58e+00)	**1.88e+00 (1.05e+00)**

Table 6: Comparison results of seven algorithms in 30-dimensional test function.

Func	QPSO Mean (Std)	CSPSO Mean (Std)	CLPSO Mean (Std)	WQPSO Mean (Std)	CQPSO Mean (Std)	COQPSO Mean (Std)	TEQPSO Mean (Std)
f_1	6.50e-14 (1.38e-14)	5.11e-9 (7.07e-9)	4.01e-10 (4.28e-10)	4.30e-33 (6.72e-33)	7.82e-17 (1.04e-17)	1.81e-30 (3.44e-30)	**0.00e+00 (0.00e+00)**
f_2	1.57e+02 (8.2e+01)	**6.78e-01 (1.39e-01)**	2.78e+00 (1.46e+00)	6.4e+01 (4.86e+01)	3.56e+01 (2.33e+01)	4.11e+02 (3.58e+02)	1.02e+01 (1.6e+00)
f_3	6.01e-02 (9.79e-01)	4.94e-12 (7.61-12)	7.05e-13 (9.20e-12)	2.07e-13 (6.37e-13)	3.08e-02 (5.64e-02)	4.59e-10 (1.59e-10)	**7.40e-17 (4.94e-17)**
f_4	2.59e-01 (1.74e-01)	7.69e+00 (4.55e+00)	2.04e+00 (1.57e+00)	5.18e-01 (4.64e-01)	2.97e-01 (2.66e-01)	1.04e-01 (9.40e+00)	**0.00e+00 (0.00e+00)**
f_5	2.18e-12 (2.15e-12)	4.78e-02 (7.61-02)	6.55e-03 (3.84-03)	1.29e-05 (7.15e-04)	2.18e-06 (1.15e-06)	2.18e-13 (2.15e-13)	**0.00e+00 (0.00e+00)**
f_6	2.88e+01 (1.34e+01)	3.37e+02 (1.75e+02)	1.28e+02 (1.11e+02)	2.60e+01 (6.95e+00)	2.35e+02 (3.25e+01)	6.87e+01 (7.29e+01)	**0.00e+00 (0.00e+00)**
f_7	3.54e-01 (4.20e-01)	6.02e-01 (5.55e-01)	3.78e-01 (2.02e-01)	**8.27e-03 (1.55e-03)**	2.04e-02 (9.47e-01)	1.28e-01 (1.89e-01)	4.00e-03 (2.31e-03)
f_8	1.07e+01 (5.33e+00)	4.11e+02 (1.06e+02)	3.56e+02 (1.83e+02)	5.24e+01 (1.55e+01)	8.87e+01 (2.47e+01)	1.88e+01 (5.95e+00)	**3.77e+00 (1.82e+00)**

standard deviation values of CSPSO are 6.78e-01 and 1.39e-01, followed by CLPSO, TEQPSO, CQPSO, WQPSO, QPSO, and COQPSO.

Then, the algorithm of TEQPSO obtains the best results on the multimodal functions $f_3 \sim f_6$; the mean and standard deviation values of f_3 are 7.40e-17 and 4.94e-17 and the mean and standard deviation values of $f_4 \sim f_6$ are 0.00e+00 and 0.00e+00. For the functions f_4, f_5, and f_6, the other six algorithms fall into local optimum and only TEQPSO obtains the best solution. The results show that TEQPSO algorithm can maintain better population diversity, enhance the ability of the algorithm to jump out of local optimum, and have higher search accuracy on complex multimodal functions. The functions f_7 and f_8 are rotational modal functions. The other five algorithms are difficult to solve the functions f_7 and f_8. Only WQPSO and TEQPSO can obtain better calculation results. WQPSO obtains the highest precision solution in the rotation function f_7; the mean and standard deviation values of WQPSO are 8.27e-03 and 1.55e-03, followed by WQPSO, TEQPSO, CQPSO, CSPSO,

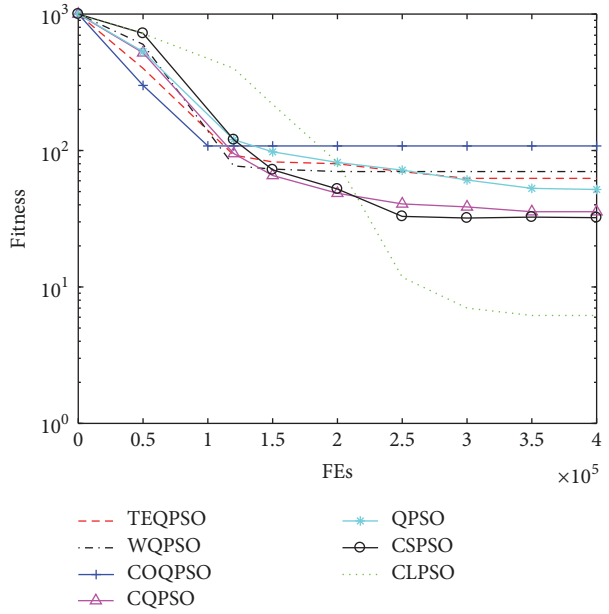

Figure 2: Comparison of convergence process with seven algorithms in sphere function.

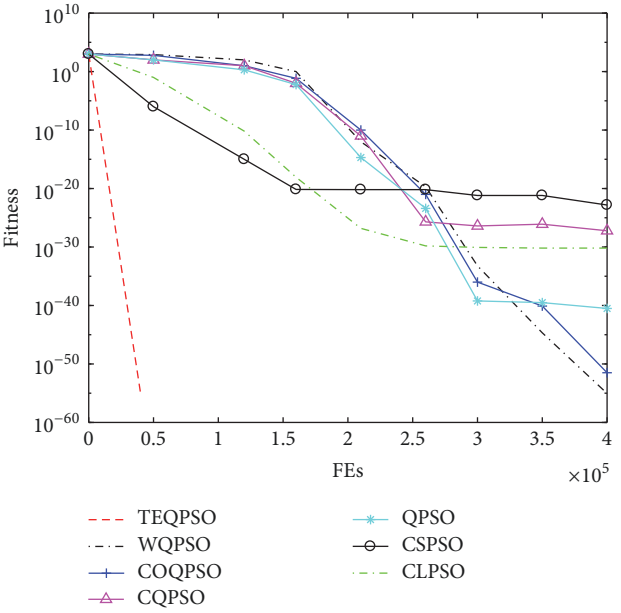

Figure 3: Comparison of convergence process with seven algorithms in Rosenbrock function.

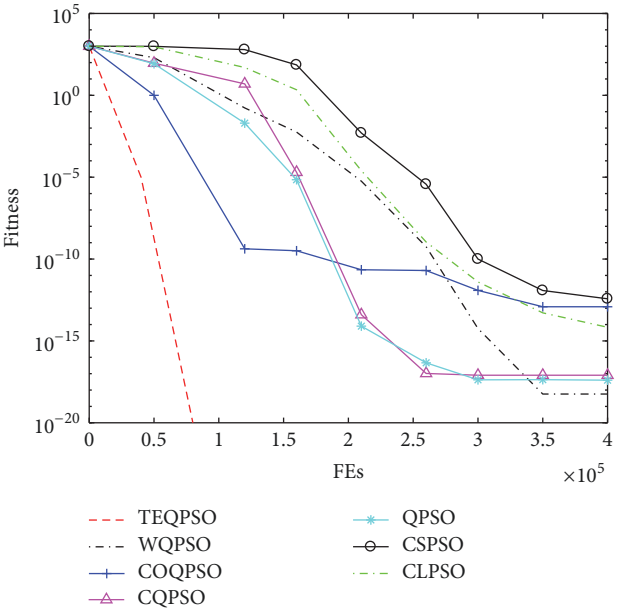

Figure 4: Comparison of convergence process with seven algorithms in Ackley function.

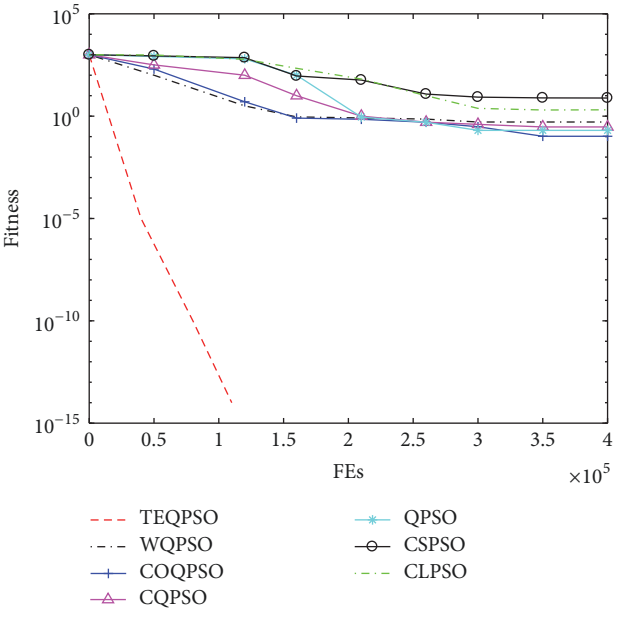

Figure 5: Comparison of convergence process with seven algorithms in Griewank function.

QPSO, CLPSO, and COQPSO. TEQPSO obtains the highest precision solution in the rotation function f_8; the mean and standard deviation values of TEQPSO are 3.77e+00 and 1.82e+00, followed by QPSO, COQPSO, WQPSO, CQPSO, CSPSO, and CLPSO. In summary, the precision of the TEQPSO solutions in functions f_1, f_3, f_4, f_5, f_6, and f_8 is better than the other six algorithms significantly. Since the convergence characteristics of seven algorithms under 30-dimensional test function are similar to the 10-dimensional convergence characteristics, the convergence characteristics of seven algorithms under 30D are not given in this paper.

4.3. Calculation Results Discussion. By analyzing the experimental results of the TEQPSO algorithm in the case of 10D and 30D, it could be concluded that this algorithm does not work well for unimodal functions. The TEQPSO algorithm gives superior results for multimodal functions compared with other QPSO algorithms. Jumping out of local

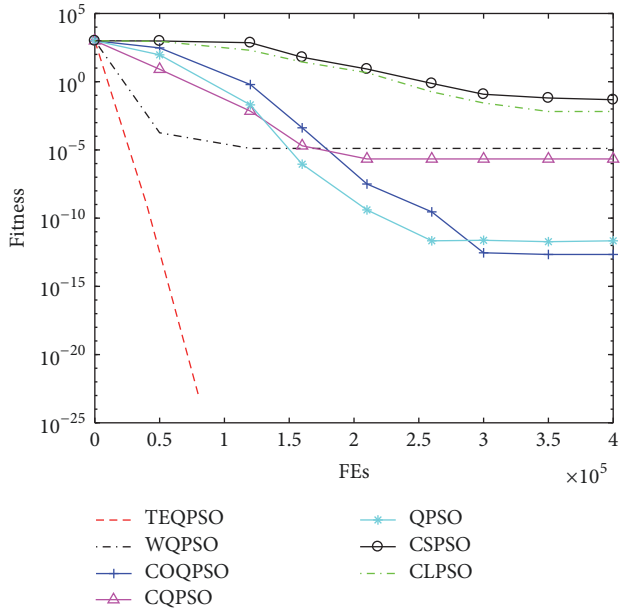

Figure 6: Comparison of convergence process with seven algorithms in step function.

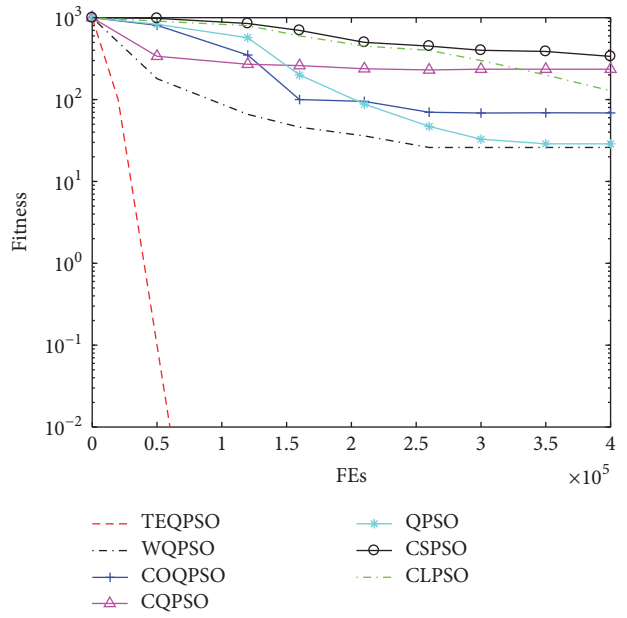

Figure 7: Comparison of convergence process with seven algorithms in Rastrigin function.

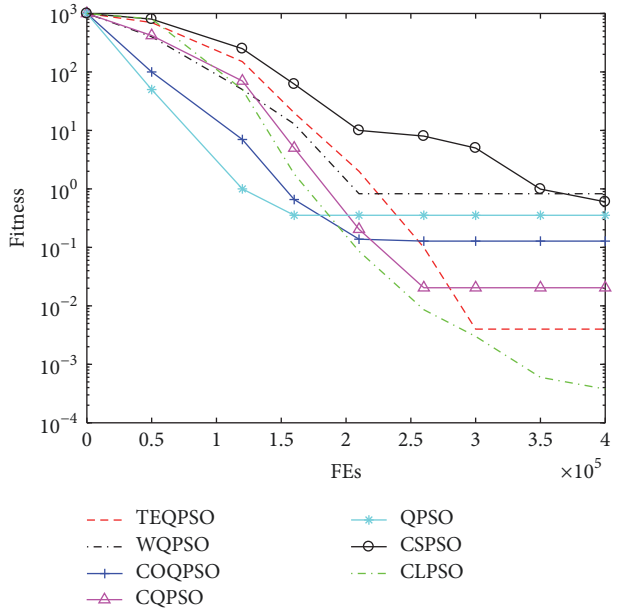

Figure 8: Comparison of convergence process with seven algorithms in rotated Griewank function.

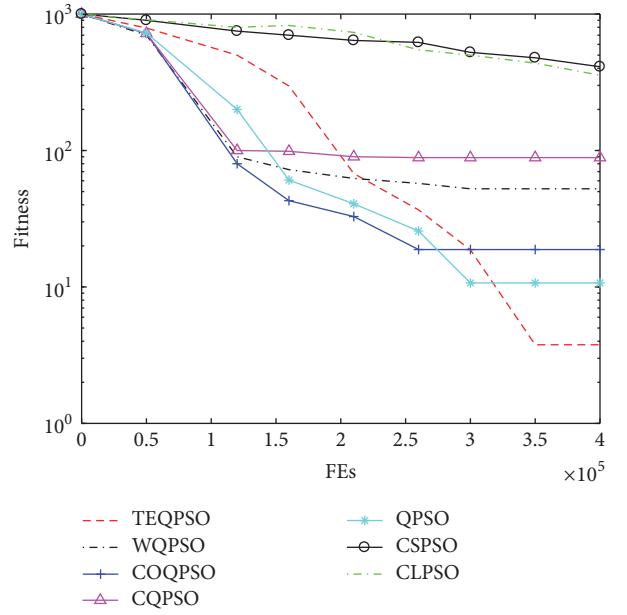

Figure 9: Comparison of convergence process with seven algorithms in rotated Rastrigin function.

optimum and achieving better search accuracy for rotating multimodal functions, which is very helpful for solving complex problems, are other advantages of this algorithm. According to the "No Free Lunch" theorem, the TEQPSO algorithm leads to higher search accuracy for multimodal and rotating multimodal functions. However, its convergence speed is significantly slower than other QPSO algorithms. Therefore, the TEQPSO algorithm gives obvious advantages in solving multiobjective and complex problems. In addition, the calculation effect is better.

5. Conclusion

This paper proposes a QPSO algorithm for the teamwork evolutionary strategy. The algorithm adopts a novel learning strategy, namely, teamwork evolutionary strategy, which consists of cross-sequential quadratic programming and Gaussian chaotic mutation operators. The new strategy

allows the particle to have more samples to learn and larger potential space to fly. Through analysis and experiments, it is derived that the teamwork evolutionary strategy increases the TEQPSO algorithm ability to utilize the information in the group. In comparison with the eight QSOs and PSOs variants, it could be concluded that the TEQPSO algorithm significantly improves the algorithm performance for multi-peak cost functions while the TEQPSO algorithm is effective in solving single-peak cost functions.

Data Availability

The relevant raw data of this article should be all downloaded from http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2014. All the test functions in this paper can also be found in the reference J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization."

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] B. Liu, L. Wang, and Y. 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.

[2] H. Zhang, G. Bai, and C. Liu, "A broadcast path choice algorithm based on simulated annealing for Wireless Sensor Network," in *Proceedings of the IEEE Int Conf on Automation and Logistics*, pp. 310–314, IEEE, Zhengzhou, China, 2012.

[3] Z. Zhang, N. Zhang, and Z. Feng, "Multi-satellite control resource scheduling based on ant colony optimization," *Expert Systems with Applications*, vol. 41, no. 6, pp. 2816–2823, 2014.

[4] A. H. Gandomi, X. S. Yang, A. H. Alavi, and S. Talatahari, "Bat algorithm for constrained optimization tasks," *Neural Computing and Applications*, vol. 22, no. 6, pp. 1239–1255, 2013.

[5] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Mixed variable structural optimization using firefly algorithm," *Computers & Structures*, vol. 89, no. 23-24, pp. 2325–2336, 2011.

[6] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[7] X. Li, J. N. Wang, and M. Yin, "Enhancing the performance of cuckoo search algorithm using orthogonal learning method," *Neural Computing and Applications*, vol. 24, no. 6, pp. 1233–1247, 2014.

[8] X. T. Li and M. H. Yin, "A particle swarm inspired cuckoo search algorithm for real parameter optimization," *Soft Computing*, vol. 20, no. 4, pp. 1389–1413, 2016.

[9] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: charged system search," *Acta Mechanica*, vol. 213, no. 3-4, pp. 267–289, 2010.

[10] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[11] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[12] Q. Li, C. Zhang, P. Chen et al., "Improved ant colony optimization algorithm based on particle swarm optimization," *Control and Decision*, vol. 28, no. 6, pp. 873–878, 2013.

[13] C. Zhang, L. Tu, and J. Wang, "Application of Self adaptive ant colony optimization in TSP," *Journal of Central South University: Science and Technology*, vol. 46, no. 8, pp. 2944–2949, 2015.

[14] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

[15] L. Messerschmidt and A. P. Engelbrecht, "Learning to play games using a PSO-based competitive learning approach," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 280–288, 2004.

[16] F. Van den Bergh, *An Analysis of Particle Swarm Optimizers [Ph.D. thesis]*, University of Pretoria, November 2001.

[17] J. Sun, C.-H. Lai, W.-B. Xu, Y. Ding, and Z. Chai, "A modified quantum-behaved particle swarm optimization," in *Proceedings of the 7th International Conference on Computational Science (ICCS '07)*, pp. 294–301, Beijing, China, May 2007.

[18] J. Liu, J. Sun, and W. B. Xu, "Improving quantum-behaved particle swarm optimization by simulated annealing," *Computational Intelligence and Bioinformatics*, pp. 130–136, 2006.

[19] D. Chen, J. Wang, F. Zou, W. Hou, and C. Zhao, "An improved group search optimizer with operation of quantum-behaved swarm and its application," *Applied Soft Computing*, vol. 12, no. 2, pp. 712–725, 2012.

[20] Y. Li, R. Xiang, L. Jiao, and R. Liu, "An improved cooperative quantum-behaved particle swarm optimization," *Soft Computing*, vol. 16, no. 6, pp. 1061–1069, 2012.

[21] Y. Li, L. Jiao, R. Shang, and R. Stolkin, "Dynamic-context cooperative quantum-behaved particle swarm optimization based on multilevel thresholding applied to medical image segmentation," *Information Sciences*, vol. 294, pp. 408–422, 2015.

[22] H. Long, S. Wu, and H. Fu, "Parallel diversity-controlled quantum-behaved particle swarm optimization algorithm," in *Proceedings of the 2014 Tenth International Conference on Computational Intelligence and Security (CIS)*, pp. 74–79, Kunming, China, November 2014.

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

[24] Q. Qian, M. Tokgo, C. Kim, C. Han, J. Ri, and K. Song, "A hybrid improved quantum-behaved particle swarm optimization algorithm using adaptive coefficients and natural selection method," in *Proceedings of the 7th International Conference on Advanced Computational Intelligence, ICACI 2015*, pp. 312–317, Xiamen, China, March 2015.

[25] 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.

[26] H. Long and S. Wu, "Quantum-behaved particle swarm optimization with diversity-maintained," in *Ecosystem Assessment and Fuzzy Systems Management*, Berlin, Germany, 2014.

[27] 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.

[28] 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.

[29] 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.

[30] R. B. Wilson, *A Simplicial Algorithm for Concave Programming*, Graduate School of Business Administration, Harvard University, Boston, Mass, USA, 1963.

[31] J. H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, Mass, USA, 1992.

[32] W. Gong, Z. Cai, and Y. Wang, "Repairing the crossover rate in adaptive differential evolution," *Applied Soft Computing*, vol. 15, pp. 149–168, 2014.

[33] M. Han, J. Fan, and J. Wang, "A dynamic feedforward neural network based on gaussian particle swarm optimization and its application for predictive control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 22, no. 9, pp. 1457–1468, 2011.

[34] J. Sun, B. Feng, and W. Xu, "Particle swarm optimization with particles having quantum behavior," in *Proceedings of the 2004 Congress on Evolutionary Computation*, vol. 1, pp. 325–331, 2004.

[35] A. Meng, Z. Li, H. Yin, S. Chen, and Z. Guo, "Accelerating particle swarm optimization using crisscross search," *Information Sciences*, vol. 329, pp. 52–72, 2016.

[36] 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.

[37] S. Lu and C. Sun, "Quantum-behaved particle swarm optimization with cooperative-competitive coevolutionary," in *Proceedings of the 2008 International Symposium on Knowledge Acquisition and Modeling (KAM)*, pp. 593–597, Wuhan, China, December 2008.

[38] R. Salomon, "Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions," *BioSystems*, vol. 39, no. 3, pp. 263–278, 1996.