View metadata, citation and similar papers at core.ac.uk brought to you by CORE
provided by Carleton University's Institutional Repository

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

A Simple Brain Storm Optimization Algorithm with a Periodic Quantum Learning Strategy

ZHENSHOU SONG[1], JIAQI PENG[1], CHUNQUAN LI[1,2], and PETER X. LIU[1,2,] Senior, IEEE

[1]School of Information Engineering, Nanchang University, Nanchang 330029, CHINA
[2]Department of Systems and Computer Engineering, Carleton University, Ottawa, ON K1S 5B6, CANADA

Corresponding author: Chunquan Li (e-mail: lichunquan@ ncu.edu.cn).

This work was supported by the China Scholarship Council under the State Scholarship Fund (CSC No. 201606825041), by the National Natural Science Foundation of China under Grants 61503177 and 81660299, by the Science and Technology Department of Jiangxi Province of China under Grants 20161ACB21007, 20171BBE50071, and 20171BAB202033, and by the Education Department of Jiangxi province of China under Grant GJJ14228. Zhenshou Song and Jiaqi Peng contribute equally to this article.

ABSTRACT Brain storm optimization (BSO) is a young and promising population-based swarm intelligence algorithm inspired by the human process of brainstorming. The BSO algorithm has been successfully applied to both science and engineering issues. However, thus far, most BSO algorithms are prone to fall into local optima when solving complicated optimization problems. In addition, these algorithms adopt complicated clustering strategies such as K-means clustering, resulting in large computational burdens. The paper proposes a simple BSO algorithm with a periodic quantum learning strategy (SBSO-PQLS), which includes three new strategies developed to improve the defects described above. First, we develop a simple individual clustering (SIC) strategy that sorts individuals according to their fitness values and then allocates all individuals into different clusters. This reduces computational burdens and resists premature convergence. Second, we present a simple individual updating (SIU) strategy by simplifying the individual combinations and improving the step size function to enrich the diversity of newly generated individuals and reduces redundancy in the pattern for generating individuals. Third, a quantum-behaved individual updating with periodic learning (QBIU-PL) strategy is developed by introducing a quantum-behaved mechanism into SBSO-PQLS. QBIU-PL provides new momentum, enabling individuals to escape local optima. With the support of these three strategies, SBSO-PQLS effectively improves its global search capability and computational burdens. SBSO-PQLS is compared with seven other BSO variants, Particle Swarm Optimization (PSO), and Differential Evolution (DE) on CEC2013 benchmark functions. The results show that SBSO-PQLS achieves a better global search performance than do the other nine algorithms.

INDEX TERMS Global optimization, Brain storm optimization (BSO), Periodic quantum learning strategy.

I. INTRODUCTION

Brain storm optimization (BSO), invented by Shi in 2011, is a young and competitive population-based swarm intelligence optimization approach [1], [2]. The fundamental principle of the BSO algorithm is to emulate the human process of brainstorming, in which several people gather together to discuss issues. This group brainstorming effort can result in a broad range of ideas for handling complicated issues [3]. In the BSO algorithm, each individual in the solution space can be regarded as a single idea in the brainstorming process. All the ideas are divided into several clusters using the K-means algorithm. The best idea in each cluster acts as a clustering center, and each idea can be updated by integrating a Gaussian factor with ideas from other clusters. In brief, the BSO algorithm is composed of the following main processes: individual initialization, individual clustering, cluster center disruption, individual updating, and individual selection. The BSO algorithm has been successfully applied in various science and engineering fields. For instance, Xue et al. developed a multi-objective optimization algorithm based on BSO algorithm [4]. María and Miguel improved both the energy consumption and execution time by using a multi-objective BSO algorithm [5]. Jordehi adopted the BSO algorithm to determine the optimal location and setting of flexible AC transmission system [6]. Sun et al. developed a closed-loop brain storm optimization to deal with the optimal formation reconfiguration of multiple satellites with impulse control [7]. Qiu and Duan developed a modified BSO to

2169-3536 © 2017 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

ascertain receding horizon control parameters of formation flight for unmanned aerial vehicles [8].

However, similar to other swarm intelligence algorithms such as ant colony optimization (ACO) [9], particle swarm optimization (PSO) [10], and differential evolution (DE) [11], the BSO algorithm exhibits a fundamental issue in that it is apt to achieve premature convergence when handling complicated issues. Consequently, avoiding premature convergence while achieving good convergence speed are important issues for the BSO algorithm. To enhance BSO performance, several BSO variants have been proposed over the past few years.

Zhan et al. proposed a modified brain storm optimization (MBSO) algorithm that uses a simple grouping method (SGM) to improve calculation efficiency and introduces the idea difference strategy (IDS) to avoid premature convergence [12]. Similarly, Li and Duan developed a simplified brain storm optimization (SBSO) algorithm for optimizing an automatic carrier landing system by applying the SGM strategy and simplifying the individual-generating operation to achieve good convergence speed [13]. To reduce the computational cost of the BSO, Sun et al. developed three variants of closed-loop brain storm optimization (CLBSO) algorithms by introducing new operators [7]. Zhou et al. developed a dynamic step size and individual-updating strategy using a batch pattern to avoid local minima and strengthen the convergence ability [14]. In [15], two partial reinitializing strategies were adopted to avoid premature convergence by enhancing the population diversity of the BSO algorithm.

Furthermore, hybrid algorithms play a crucial role in avoiding premature convergence and accelerating convergence. To optimize a DC brushless motor, Duan et al. proposed a predator-prey BSO (PPBSO) algorithm that introduces a predator-prey operator to enrich swarm diversity and avoid premature convergence [16]. Krishnanand et al. incorporated the teaching-learning-based algorithm into the BSO algorithm to obtain a self-evolving feature in the entire iterative process [17]. Inspired by the quantum-behaved PSO (QPSO) algorithm [18], [19], Duan and Li presented a quantum-behaved BSO (QBSO) algorithm to handle Loney's solenoid issue by incorporating a quantum mechanism into each idea to improve population diversity and avoid local optima [20]. Both the differential evolution strategy and a new step-size control strategy for the BSO were employed in [21] to achieve an effective balance between avoiding premature convergence and accelerating convergence. Yang et al. developed an advanced discussion-mechanism-based brain storm optimization (ADMBSO) algorithm by creating a new discussion pattern to maintain a trade-off global search ability and convergence speed [22]. Chen et al. proposed an improved BSO algorithm that adopts an affinity propagation (AP) technique with an individual updating pattern to adaptively transform the number of clusters and enhance global search ability [23]. Cao et al. presented an improved BSO in which a stochastic grouping technique is designed to improve the time complexity, and a dynamic step-size parameter is used to balance the global and local search capabilities of the algorithm [24]. In addition, Cao et al. proposed an improved BSO algorithm that used a dynamic clustering strategy (BSO-DCS) to reduce the computational time complexity of the BSO [25]. Jia et al. developed a new hybrid BSO algorithm that integrates the simulated annealing (SA) technique to avoid falling into local optima [26].

Although the aforesaid BSO variants achieved acceptable results, they still tend to fall into local optima when applied for solving increasingly complicated optimization problems such as the Popular CEC2013 test suit in [27]. A crucial reason behind the problems with these BSO variants is that their individual clustering strategies and individual updating strategies are unable to maintain a rational balance between global and local search capabilities. In addition, these BSO variants use complicated strategies such as K-means clustering, resulting in excessive computational burdens.

To address the abovementioned problems, this paper proposes a novel BSO variant named the simple brain storming optimization with periodic quantum learning strategy (SBSO-PQLS), for which a new individual clustering strategy, a new individual updating mechanism, and a quantum-behaved individual updating with periodic learning strategy are developed. The three strategies work cooperatively to avoid premature convergence and reduce the computational burden of SBSO-PQLS. First, a new individual clustering strategy, called the simple individual clustering (SIC) strategy, is developed, which sorts all individuals according to their fitness values instead of the distances between them and then reasonably allocates all individuals into different clusters. The SIC strategy not only effectively reduces the computational burden of SBSO-PQLS but also provides a reasonable improvement regarding premature convergence. Second, a new individual updating mechanism, called the simple individual updating (SIU) strategy, is presented, which integrates the difference strategy proposed in [12]. The SIU strategy both enriches the new individual-generating pattern and reduces its redundancy, which efficiently preventes new individuals from becoming trapped in local optima and accelerates convergence. Third, a quantum-behaved individual updating with periodic learning (QBIU-PL) strategy is presented. The QBIU-PL includes a quantum-behaved mechanism that provides new momentum, enabling individuals to jump out of local optima. However, too-frequent utilization of the quantum-behaved mechanism resulting in excessive states, over-expands the search range for individuals, and may cause the SBSO-PQLS algorithm to conduct meaningless and purposeless exploration, reducing its search efficiency. Therefore, the periodic learning strategy is integrated with the quantum-behaved mechanism to achieve rational utilization of the quantum-behaved mechanism throughout the iterative process. In addition, in the QBIU-PL strategy, a new individual is selected through the fitness evaluation mechanism rather than via a probability mechanism. Only new individuals with better fitness values

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

should be kept to ensure good solutions across the entire swarm.

To demonstrate its superiority, SBSO-PQLS is compared with BSO [2], MBSO [12], SBSO [13], CLBSO [7], QBSO [20], BSODE [21], and ADMBSO [22]. These are representative BSO algorithms with good performance. Furthermore, we also compare SBSO-PQLS with PSO [10] and DE [11] to further evaluate its performance. The CEC2013 benchmark functions [27] are used to evaluate and verify the superiority of SBSO-PQLS.

II. RELATED WORKS

A. HUMAN BRAINSTORMING

Brainstorming has been broadly adopted to promote creative thinking. The concept was first presented by Osborn in 1939 [3]. The brainstorming procedure is an exercise in creativity in which a group of people with different backgrounds gather together and spontaneously contribute their best ideas to address a specific problem. In humans, the brainstorming steps can be described as follows [3]:

Step 1 People with backgrounds as varied as possible gather to create new ideas based on four rules, called Osborn’s rules.

Step 2 Several clients act as issue owners, and one idea per owner is selected as the most useful answer for the issue.

Step 3 The ideas from Step 2 are then used as cues to spark more ideas.

Step 4 Some ideas are randomly selected as cues to produce yet more ideas.

Step 5 The issue owners select several of the best ideas from Steps 3 and 4.

Step 6 Execute Steps 2–6 repeatedly until a sufficiently effective solution is achieved for the issue.

For the brainstorming procedure to work properly, the participants follow four general rules of brainstorming, called Osborn’s rules, which are described as follows [1]–[3]:

Rule 1 Focus on quantity: The first rule is based on the assumption that producing a larger number of ideas increases the probability of achieving an effective solution. This rule focuses on facilitating problem solving by using a "quantity implementing quality" approach.

Rule 2 Avoid criticism: Participants in the brainstorming procedure should concentrate on generating ideas rather than criticizing ideas already produced. Criticism is reserved for the next stage to enable the participants to be unrestricted so that they can produce unusual and creative ideas and improve the probability of a satisfactory result.

Rule 3 Welcome unusual ideas: Participants can contribute unusual ideas by viewing the issues from different perspectives and suspending their assumptions. These solutions are welcomed to achieve a good and sufficiently large series of ideas.

Rule 4 Cross-fertilize: By modeling the budding ideas through associations, the fourth rule assumes that better ideas can be achieved by integrating good ideas.

B. BRAIN STORM OPTIMIZATION ALGORITHM

As proposed in [1] and [2], the BSO algorithm procedure can be described as follows.

(1) Population Initialization

In the BSO algorithm, each individual, termed an idea, is regarded as a candidate solution vector for the problem solution. For $i \in \{1,2,\cdots,N\}$, the ith idea is described as vector $P_i^k = [p_{i1}^k, p_{i2}^k, \cdots, p_{iD}^k]$, N represents the population size, D denotes the dimension of the search space, and k is the current iteration index. Here, p_{id}^k is the dth dimension of the ith idea P_i^k in the range $[l_{d\_min}, u_{d\_max}]$, where d represents a dimension that satisfies $d \in \{1,2,\cdots,D\}$, and l_{d_min} and u_{d_max} are defined as the minimum and maximum boundaries of the dth dimension of the search space, respectively. Then, the dth dimension of the ith idea is randomly initialized based on the uniform distribution in the search space as follows:

$$p_{id} = l_{d\_min} + r_d(u_{d\_max} - l_{d\_min}), \tag{1}$$

where r_d is a random number uniformly distributed in the range $[0,1]$. Subsequently, each idea is evaluated using a fitness function $F(P_i^k)$ to determine that idea’s fitness value.

(2) Individual Clustering

In each generation, all N ideas in the entire swarm are divided into M clusters according to the K-means clustering strategy [1], [2]. The best idea $C_m^k = [c_{m1}^k, c_{m2}^k, \cdots, c_{mD}^k]$, $m \in \{1,2,\cdots,M\}$ in the mth cluster is selected as the mth cluster center, after which the M cluster centers can be described as $\{C_1^k, C_2^k, \cdots, C_M^k\}$.

(3) Cluster Center Disruption

In addition, the jth cluster center $C_j^k = [c_{j1}^k, c_{j2}^k, \cdots, c_{jD}^k]$, $j \in \{1,2,\cdots,M\}$ is randomly selected from the M cluster centers $\{C_1^k, C_2^k, \cdots, C_M^k\}$ and is replaced by a random idea $Q = [q_1, q_2, \cdots, q_D]$ when the uniformly distributed random number r_0 is smaller than the pre-determined probability p_{r0}; otherwise, it is not replaced. The cluster center C_j^k is updated as follows:

$$c_{jd}^k = \begin{cases} q_d, & r_0 < p_{r0} \\ c_{jd}^k, & r_0 \geq p_{r0} \end{cases} \tag{2}$$

where c_{jd}^k, $d \in \{1,2,\cdots,D\}$ is the dth dimension of the cluster center C_j^k, r_0 is a uniformly distributed random number in the range $[0,1]$, and q_d, $d \in \{1,2,\cdots,D\}$ is the dth dimension of the random idea Q, randomly initialized in the dth search space according to (1).

(4) Individual Updating

The new idea is updated based on either a single old idea from one cluster, or a combination of two old ideas from two different clusters.

1) Generating new ideas from one cluster

When the uniform distribution random number r_1 is less

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

than the pre-determined probability p_{r1}, the new idea ($P_i^{k+1} = [p_{i1}^{k+1}, p_{i2}^{k+1}, \cdots, p_{iD}^{k+1}]$, $i \in \{1, 2, \cdots, N\}$) is updated based on one old idea $U^k = [u_1^k, u_2^k, \cdots, u_D^k]$ from one cluster of M cluster as follows:

$$p_{id}^{k+1} = u_d^k + \eta_d(\mu, \sigma)\xi(\mathrm{k}),\ r_1 < p_{r1} \tag{3}$$

where p_{id}^{k+1} is the dth dimension of the updated idea P_i^{k+1}, and u_d^k is the dth dimension of the vector U^k. Here, η_d is a Gaussian random number, μ is its mean, σ is its variance, and $\xi(k)$ denotes the step size function for regulating the convergence speed.

Suppose the old idea U^k comes from the jth selected cluster. The jth cluster center C_j^k is selected as the old idea U^k if the uniform distribution random number r_{11} is less than the pre-determined probabilityp_{r11}; otherwise, an idea $\Psi^k = [\psi_1^k, \psi_2^k, \cdots, \psi_D^k]$ is randomly selected from the jth cluster as the U^k. The u_d^k can be formulated as

$$u_{id}^k = \begin{cases} c_{jd}^k, & r_{11} < p_{r11} \\ \psi_d^k, & r_{11} \geq p_{r11} \end{cases}, \tag{4}$$

where c_{jd}^k is the dth dimension of the jth cluster center C_j^k, and ψ_d^k is the dth dimension of the randomly selected idea Ψ^k from the jth cluster.

Note that the jth cluster is selected from the M cluster centers $\{C_1^k, C_2^k, \cdots, C_M^k\}$ based on the probability p_C, which is acquired using roulette wheel selection [2] as follows:

$$p_C = F(C_j^k) / \textstyle\sum_{m=1}^{M} F(C_m^k), \tag{5}$$

where C_j^k represents the selected center of the jth cluster, $F(C_j^k)$ stands for the fitness value of the center C_j^k, C_m^k denotes any of M cluster centers $\{C_1^k, C_2^k, \cdots, C_M^k\}$, and $F(C_m^k)$ represents any fitness value from the M cluster centers.

In addition, the step size function is defined as

$$\xi(k) = r_d \mathrm{logsig}[(0.5 \times K - k)/\mathrm{c}], \tag{6}$$

where r_d is a uniform distribution random number in the range $[0, 1]$, K is the maximum iterative number, k is the current iterative number, and c is a regulating factor for switching the slope of the step size function $\xi(k)$ and improving the convergence speed of the algorithm.

2) Generating new ideas from two clusters

When the uniform distribution random number r_1 is greater than the pre-determined probabilityp_{r1}, the new idea ($P_i^{k+1} = [p_{i1}^{k+1}, p_{i2}^{k+1}, \cdots, p_{iD}^{k+1}]$, $i \in \{1, 2, \cdots, N\}$) can be updated based on one vector $V^k = [v_1^k, v_2^k, \cdots, v_D^k]$ as follows:

$$p_{id}^{k+1} = v_d^k + \eta_d(\mu, \sigma)\xi(\mathrm{k}), r_1 \geq p_{r1}, \tag{7}$$

where p_{id}^{k+1} is the dth dimension of the updated idea P_i^{k+1}, v_d^k is the dth dimension of the vector V^k, and d represents a dimension that satisfies $d \in \{1, 2, \cdots, D\}$.

Note that unlike the vector U^k from (4), the vector V^k from (7) is a combination of two different ideas from two different M clusters. Suppose that the jth cluster and the hth cluster are two different clusters randomly selected from M clusters, and C_j^k and C_h^k are used to described their cluster centers, respectively. In addition, Ψ^k and Φ^k are two different old ideas randomly selected from the jth and hth clusters, respectively. If the uniform distribution random number r_{12} is less than the pre-determined probability p_{r12}, the vector V^k is created by a combination of the centers, C_j^k and C_h^k; otherwise, the vector V^k is created by combining two ideas, Ψ^k and Φ^k. Thus, v_d^k can be formulated as

$$v_d^k = \begin{cases} r_d c_{jd}^k + (1 - r_d) C_{hd}^k, r_{12} < p_{r12} \\ r_d \psi_d^k + (1 - r_d) \phi_d^k, r_{12} \geq p_{r12} \end{cases}, \tag{8}$$

where c_{jd}^k and c_{hd}^k are the dth dimension of the centers C_j^k and C_h^k, respectively, ψ_d^k and ϕ_d^k are the dth dimension of the two different ideas Ψ^k and Φ^k from the jth and hth clusters, respectively, and r_d is the uniform distribution random number in the range $[0,1]$.

(5) Individual selection

The selection strategy is utilized to preserve the competitive solutions in all individuals. The fitness value of the new idea P_i^{k+1}, $i \in \{1, 2, \cdots, N\}$ is compared with that of the old idea P_i^k. When the fitness value of the new idea P_i^{k+1} is better than that of the old idea P_i^k, the new idea P_i^{k+1} is preserved for next iterative updating process; otherwise, the old idea P_i^k is preserved for the next iterative updating process. Without loss of generality, suppose that the considered fitness value is for minimization. The individual selection procedure is as follows:

$$P_i^{k+1} = \begin{cases} P_i^{k+1}, & F(P_i^{k+1}) < F(P_i^k) \\ P_i^k, & F(P_i^{k+1}) \geq F(P_i^k) \end{cases}, \tag{9}$$

where $F(P_i^{k+1})$ is the fitness value of the new idea P_i^{k+1}, and $F(P_i^k)$ is the fitness value of the old idea P_i^k.

When N ideas have been generated, the termination conditions of the BSO algorithm are checked, and if the termination conditions are matched, the BSO algorithm terminates and outputs the results. Otherwise, the BSO algorithm continues to run until the termination conditions are met.

C. QUANTUM-BEHAVED BRAIN STORM OPTIMIZATION

Recently, the quantum mechanism has been exploited to enhance the global search capabilities of intelligent swarm techniques. In [18] and [19], Sun et al. introduced quantum behavior and proposed a quantum-behaved PSO (QPSO) algorithm. The quantum behavior generates new states and offers new momentums for each individual in the entire swarm, effectively improving the global search capability of each individual.

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

Inspired by the QPSO, Duan et al. proposed a QBSO algorithm by introducing quantum behavior into the BSO algorithm to enhance its global search capabilities. In the QBSO, each idea in the swarm is updated based on quantum behavior. The new idea with quantum states is formulated as follows

$$q_{id}^{k+1} = \begin{cases} q_{id+}^{k+1}, & r_d < 0.5 \\ q_{id-}^{k+1}, & r_d \geq 0.5 \end{cases}, \quad (10)$$

where q_{id}^{k+1} is the dth dimension of the new quantum-behaved idea $Q_i^{k+1} = [q_{i1}^{k+1}, q_{i2}^{k+1}, \cdots, q_{iD}^{k+1}]$, $i \in \{1, 2, \cdots, N\}$ and has two different states: q_{id+}^{k+1} and q_{id-}^{k+1}, defined as

$$q_{id+}^{k+1} = m_{id}^k + \ln(1/r^d)\left(b\left|\bar{c}_d^k - u_d^k\right|\right) + \eta_d(\mu,\sigma)\xi(\mathrm{k}) \quad (11)$$

and

$$q_{id-}^{k+1} = m_{id}^k - \ln(1/r^d)\left(b\left|\bar{c}_d^k - u_d^k\right|\right) + \eta_d(\mu,\sigma)\xi(\mathrm{k}), \quad (12)$$

respectively. The m_{id}^k can be expressed as

$$m_{id}^k = r^d g_d^k + (1 - r^d)c_{jd}^k, \quad (13)$$

where g_d^k is the dth dimension of the best idea from the entire swarm in the kth iteration, and c_{jd}^k is the dth dimension of the cluster center C_j^k from cluster j, to which the idea P_i^k belongs. The $\bar{c}_d^k$ from (11) or (12) is the dth dimension of the mean value of the M cluster centers $\{C_1^k, C_2^k, \cdots, C_M^k\}$, defined as

$$\bar{c}_d^k = \sum_{i=1}^{M} c_{id}^k / M, \quad (14)$$

where c_{id}^k represents the dth dimension of the ith cluster center, and i satisfies $i \in \{1, 2, \cdots, M\}$. The parameter b from (11) or (12) decreases linearly from 1 to 0.5, according to the formula

$$b = 1 - b_0 * k/K, \quad (15)$$

where k is the current iteration index, K is the maximum number of iterations, and b_0 is a constant.

In addition, the new quantum-behaved idea Q_i^{k+1}, when crossed with an existing idea P_i^k generates two new ideas as follows:

$$x_{id}^{k+1} = r^d q_{id}^{k+1} + (1 - r^d)p_{id}^k \quad (16)$$

and

$$y_{id}^{k+1} = (1 - r_d)q_{id}^{k+1} + r^d p_{id}^k, \quad (17)$$

respectively. Here, x_{id}^{k+1} and y_{id}^{k+1} are the dth dimension of the new ideas X_i^{k+1} and Y_i^{k+1} in the crossover operation, respectively, and p_{id}^k is the dth dimension of the existing idea P_i^k. Subsequently, the fitness values of four individuals Q_i^{k+1}, P_i^k, X_i^{k+1}, and Y_i^{k+1} are evaluated, and the individual with the best fitness is selected as the new idea for the next iteration.

Note that, except for the individual updating and selection operations, the other procedures of the QBSO algorithm are the same as those of the BSO algorithm.

III. PROPOSED ALGORITHM

In the SBSO-PQLS algorithm, three new strategies, the SIC strategy, the SIU strategy, and the QBIU-PL strategy, are developed to improve premature convergence and reduce the computational cost.

A. SIMPLE INDIVIDUAL CLUSTERING STRATEGY

The original BSO algorithm used basic K-means clustering to gather similar individuals into small clusters in each iteration. In the original BSO algorithm, the purpose of clustering is to refine a search area and accelerate convergence. However, the K-means clustering algorithm causes a heavier computational burden because the K-means algorithm must compute the distances between all individuals. Hence, to reduce the computational burden, some BSO variants have adopted the simple grouping method (SGM) [12]. However, the SGM strategy also involves computing the distances among individuals, which causes the computational burden of BSO variants to remain high.

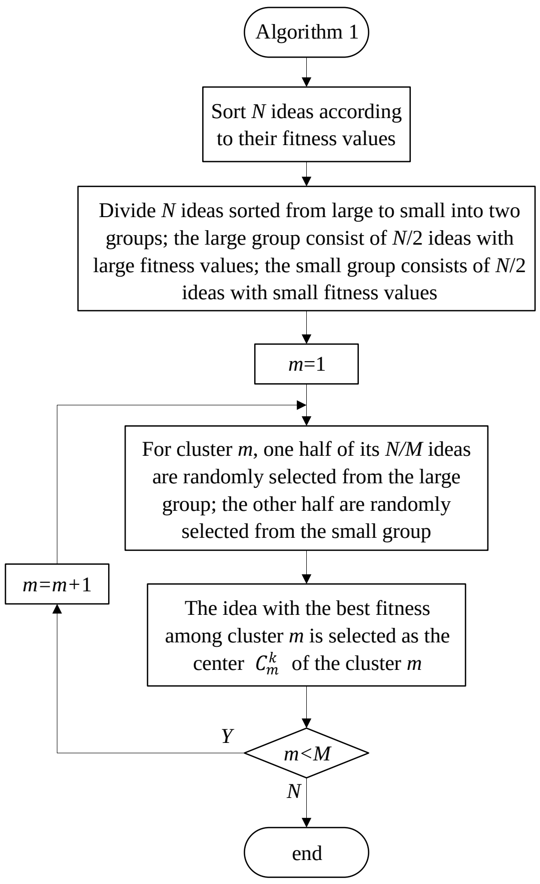

Algorithm 1
Sort N ideas according to their fitness values
Divide N ideas sorted from large to small into two groups; the large group consist of N/2 ideas with large fitness values; the small group consists of N/2 ideas with small fitness values
m=1
For cluster m, one half of its N/M ideas are randomly selected from the large group; the other half are randomly selected from the small group
m=m+1
The idea with the best fitness among cluster m is selected as the center C_m^k of the cluster m
Y
m<M
N
end

FIGURE 1. **Flowchart of SIC Strategy (Algorithm 1).**

From the analysis above, we develop a simple individual clustering (SIU) strategy, which sorts all individuals in

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

accordance with their fitness values and then reasonably allocates all individuals into different clusters. Because the SIC strategy does not need to calculate the distances between all individuals, it effectively reduces the computational burden. Furthermore, the SIC strategy can generate a reasonably diverse population and improve premature convergence of the SBSO-PQLS algorithm. The pseudocode for the SIC strategy is shown in Algorithm 1, and its flowchart is displayed in Fig. 1.

Algorithm 1: SIC Strategy

```
1: Sort all $N$ ideas $\{P_1^k, P_2^k, \cdots, P_N^k\}$ in the swarm from large to small according to their fitness values, $\{F(P_1^k), F(P_2^k), \cdots, F(P_N^k)\}$;
2: Divide the sorted $N$ ideas into two groups: a large group and a small group. The large group consists of the $N/2$ ideas with larger fitness values; the small group includes the rest of the $N/2$ ideas, which have smaller fitness values;
3: Distribute the $N$ ideas from the two groups into $M$ clusters so that each cluster contains $N/M$ ideas. For each cluster $m \in \{1, 2, \cdots, M\}$, half of its $N/M$ ideas are randomly selected from the large group, and the other half are randomly selected from the small group;
4: For each cluster $m \in \{1, 2, \cdots, M\}$, the idea with the best fitness value is selected as the cluster center $C_m^k$, $C_m^k = [c_{m1}^k, c_{m2}^k, \cdots, c_{mD}^k]$; thus, all $M$ cluster centers can be described by $\{C_1^k, C_2^k, \cdots, C_M^k\}$.
```

B. SIMPLE INDIVIDUAL UPDATING STRATEGY

In the original BSO algorithm, individual updating is based on two different cases. In one case, the new idea is generated from the center or from any idea in one cluster to refine a search area and strengthen the local search capability. In contrast, in the other case, the new idea is generated by combining two centers from two clusters or by combining two ideas from two clusters. This approach maintains population diversity and enhances global search capability.

However, we discovered that the individual updating in the original BSO algorithm still needs further improvement. For instance, a new idea generated from one cluster does not consider a combination of any two ideas from one cluster. Similarly, a new idea generated from two clusters does not consider the combination of a center from one cluster and any idea from the other cluster. Furthermore, in the original BSO algorithm, for each new idea, its individual updating equations that contain some redundant information can be simplified. Moreover, the updating equation of the original BSO algorithm uses a logarithmic sigmoid mechanism, which is disadvantageous for two reasons. The first is that the logarithmic sigmoid function is constrained in the range [-4, 4], which may result in an invalid search when the solution space is sufficiently large [12]. The second is that the logarithmic sigmoid function does not contain any feedback information and thus, it may not achieve some effective search features [7].

From the above analysis, we developed an simple individual updating (SIU) strategy based on the original BSO algorithm. The SIU strategy uses a combination of any two ideas from one cluster to improve population diversity. Subsequently, a combination of any two ideas from two clusters is adopted to improve population diversity and decrease the redundant information in the pattern for generating new individuals. Furthermore, the difference strategy from [12] was introduced to replace the logarithmic sigmoid function. The use of the SIU strategy improves premature convergence and reduces the computational burden of SBSO-PQLS. The corresponding details are formulated as follows.

(1) Generating new ideas from one cluster

When the uniform distribution random number r_1 is less than the pre-determined probability p_{r1}, the new idea $P_i^{k+1} = [p_{i1}^{k+1}, p_{i2}^{k+1}, \cdots, p_{iD}^{k+1}]$, $i \in \{1, 2, \cdots, N\}$ can be updated based on one vector $U^k = [u_1^k, u_2^k, \cdots, u_D^k]$ from one cluster of M clusters as follows:

$$p_{id}^{k+1} = u_d^k + \zeta(\mathrm{k}), \; r_1 < p_{r1}, \tag{18}$$

where $\zeta(\mathrm{k})$ is the step-size function that regulates the convergence speed, d satisfies $d \in \{1, 2, \cdots, D\}$, and u_d^k is the dth dimension of the vector U^k.

Assume that the vector U^k is generated from the jth cluster randomly selected from the M clusters based on the roulette wheel selection [2]. If the uniform distribution random number r_{11} is less than the pre-determined probability p_{r11}, the jth cluster center C_j^k is used as the vector U^k; otherwise, two ideas $\Psi^k = [\psi_1^k, \psi_2^k, \cdots, \psi_D^k]$ and $\Phi^k = [\phi_1^k, \phi_2^k, \cdots, \phi_D^k]$ are randomly selected from the jth cluster, and then combined into vector U^k. Thus, u_d^k is defined as

$$u_d^k = \begin{cases} c_{jd}^k, r_{11} < p_{r11} \\ r_d\psi_d^k + (1 - r_d)\phi_d^k, r_{11} \geq p_{r11}, \end{cases} \tag{19}$$

where c_{jd}^k is the dth dimension of the jth cluster center C_j^k, ψ_d^k and ϕ_d^k is the dth dimension of the ideas Ψ^k and Φ^k, r_d is the uniform distribution random number in the range $[0, 1]$, and d represents a dimension that satisfies $d \in \{1, 2, \cdots, D\}$.

Note that u_d^k is equal to $r_d\psi_d^k + (1 - r_d)\phi_d^k$ when $r_{11} \geq p_{r11}$, indicating that u_d^k can achieve different results given the uniform distribution of the random number r_d in the range [0, 1]. For instance, when r_d = 1 is true, u_d^k will be equal to ψ_d^k, and when r_d = 0 is true, u_d^k will be equal to ϕ_d^k. Finally, when $0 < r_d < 1$ is true, u_d^k is an arbitrary combination of ψ_d^k and ϕ_d^k. Thus, the u_d^k from (19) has more possible combinations than does the u_d^k from (4), implying that the new idea of SBSO-PQLS from one cluster has a richer diversity of the population compared with that of the original BSO.

(2) Generating new ideas from two clusters

When the uniform distribution random number r_1 is greater than the pre-determined probability p_{r1}, the new idea $P_i^{k+1} = [p_{i1}^{k+1}, p_{i2}^{k+1}, \cdots, p_{iD}^{k+1}]$, $i \in \{1, 2, \cdots, N\}$ can be updated based on one vector, $V^k = [v_1^k, v_2^k, \cdots, v_D^k]$, which is a combination of two different ideas from two clusters among the M clusters. Then, the new idea P_i^{k+1}

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

can be updated by

$$p_{id}^{k+1} = v_{\mathrm{d}}^{k} + \zeta(\mathrm{k}),\ r_1 \geq\ p_{r1}, \tag{20}$$

where p_{id}^{k+1} is the dth dimension of the new idea P_i^{k+1}, v_d^k is the dth dimension of vector V^k, ζ(k) is the step size function for regulating the convergence speed, and d satisfies $d \in \{1, 2, \cdots, D\}$.

To improve the diversity of the population in SBSO-PQLS, the vector V^k should contain as many combinations of two old ideas from two clusters as possible. Therefore, to describe all possible combinations that can be created by combining two ideas from two clusters, an effective combination for vector V^k is

$$v_d^k = r_d\psi_d^k + (1 - r_d)\phi_d^k, \tag{21}$$

where v_d^k is the dth dimension of the vector V^k, r_d is the uniform distribution random number in the range [0, 1], ψ_d^k and ϕ_d^k are the dth dimension of the two different ideas $\Psi^k = [\psi_1^k, \psi_2^k, \cdots, \psi_D^k]$ and $\Phi^k = [\phi_1^k, \phi_2^k, \cdots, \phi_D^k]$ from any two clusters, respectively, and r_d is a uniform distribution random number in the range [0, 1].

Note that the ideas Ψ^k and Φ^k can be randomly selected from the two clusters from the M clusters, respectively. Hence, both Ψ^k and Φ^k have two possible cases. One is that each of the ideas Ψ^k and Φ^k is selected from any cluster center of the M cluster centers. The other is that each of them is selected from any idea in the selected cluster other than the center of the selected cluster. Hence, v_d^k contains four possible combinations according to (21). This implies that p_{id}^{k+1} also has four possible combinations according to (20) and (21). From the above analysis, the new idea-generating pattern in SBSO-PQLS from two clusters can enhance the diversity of the population compared to the idea-generating pattern used by the standard BSO.

(3) Step size function

The step size function in SBSO-PQLS incorporates the difference strategy from [12] to improve the search range and provide feedback on the search information. The step size function is

$$\zeta(\mathrm{k}) = \begin{cases} \mathcal{M} - u_d^k,\ r_1 <\ p_{r1}, r_2 <\ p_{r2} \\ \mathcal{M} - v_d^k,\ r_1 \geq\ p_{r1}, r_2 <\ p_{r2}\,. \\ r_d\left(p_{sd}^k - p_{td}^k\right),\ r_2 \geq\ p_{r2} \end{cases} \tag{22}$$

Here, u_d^k and v_d^k can be computed according to (19) and (21), respectively; p_{sd}^k and p_{td}^k correspond to the dth dimension of two different ideas, P_s^k and P_t^k, from all current ideas, respectively; the indices s and t are mutually exclusive integers randomly selected from the range [1, N]; $\mathcal{M}$ is written as

$$\mathcal{M} = l_{d\_min} + r_d\left(u_{d\_max} - l_{d\_min}\right), \tag{23}$$

where u_{d_max} and l_{d_min} denote the maximum and minimum boundaries of the dth dimension of the search space, respectively.

Generally, the SIU strategy can both enrich the diversity and decrease the redundancy when generating new individuals, which efficiently avoids premature convergence. The new individual updating pattern is illustrated in Algorithm 2 with the flowchart shown in Fig. 2.

Algorithm 2: SIU Strategy

```
1:  Generate a random number $r_1$;
2:  for each new idea $P_i^{k+1}$, $i \in \{1, 2, \cdots, N\}$ do
3:    if $r_1 < p_{r1}$ then
4:      Select a cluster $j$, $j \in \{1, 2, \cdots, M\}$ from $M$ cluster according
        to the probability $p_C$ in the (5);
5:      Update the new idea $P_i^{k+1} = [p_{i1}^{k+1}, p_{i2}^{k+1}, \cdots, p_{iD}^{k+1}]$ using (18),
        (19), and (22);
6:    else if $r_1 \geq p_{r1}$ then
        Randomly select two clusters $j$ and $h$, $j(h) \in \{1, 2, \cdots, M\}$
        from the $M$ clusters;
7:      Update new idea $P_i^{k+1} = [p_{i1}^{k+1}, p_{i2}^{k+1}, \cdots, p_{iD}^{k+1}]$ by using (20),
        (21), and (22).
8:    end if
9:  end for
```

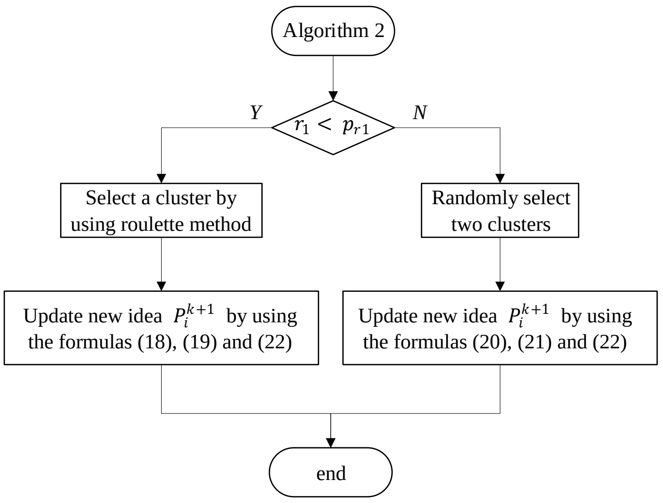

Algorithm 2
$r_1 < p_{r1}$
Y
N
Select a cluster by using roulette method
Randomly select two clusters
Update new idea P_i^{k+1} by using the formulas (18), (19) and (22)
Update new idea P_i^{k+1} by using the formulas (20), (21) and (22)
end

FIGURE 2. Flowchart of SIU Strategy (Algorithm 2).

C. QUANTUM-BEHAVED INDIVIDUAL UPDATING WITH PERIODIC LEARNING STRATEGY

The new strategies for clustering and generating individuals improve population diversity and help SBSO-PQLS avoid becoming trapped into local optima to some extent. However, these strategies may be less effective when the individuals of SBSO-PQLS become similar. This is because that these similar individuals lack new momentums; therefore, it is difficult for similar individuals to escape a local optimum and find promising search spaces.

In [18] and [19], Sun et al. proposed a quantum-behaved PSO (QPSO) algorithm by introducing quantum theory into PSO to generate new momentum for the individuals, which enhances QPSO's global search capabilities and avoids premature convergence. Inspired by QPSO, Duan et al. proposed a quantum-behaved BSO (QBSO) algorithm to enhance the performance of the BSO algorithm by introducing quantum behavior into the BSO algorithm.

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

Although QBSO generates new momentum and causes extreme expansion of individuals' search range, such a range, it may cause QBSO to carry out excessive exploration with sketchy exploitation and, thus, affect the algorithm's search efficiency. To overcome this defect of QBSO and improve its search efficiency, we develop a quantum-behaved individual updating with periodic learning (QBIU-PL) strategy as follows.

Each new quantum-behaved individual Q_i^{k+1}, $i \in \{1, 2, \cdots, N\}$ has two quantum states: Q_{i+}^{k+1} and Q_{i-}^{k+1}. The QBIU-PL strategy is written as follows:

$$q_{id}^{k+1} = \begin{cases} q_{id+}^{k+1}\Delta(T) \\ q_{id-}^{k+1}\Delta(T) \end{cases}, \tag{24}$$

where q_{id}^{k+1} is the dth dimension of the new quantum-behaved idea $Q_i^{k+1} = \left[q_{i1}^{k+1}, q_{i2}^{k+1}, \cdots, q_{iD}^{k+1}\right]$, $i \in \{1, 2, \cdots, N\}$, $q_{id+}^{k+1}$and q_{id-}^{k+1} are the dth dimension of two different states Q_{i+}^{k+1} and Q_{i-}^{k+1} of the quantum-behaved idea Q_i^{k+1}, and $\Delta(T)$ is the time periodic learning strategy, defined as

$$\Delta(T) = \begin{cases} 1, k = nT \\ 0, k \neq nT \end{cases}, n \in \{1,2 \cdots, K/T\}, \tag{25}$$

where T is the time period, k is the current iteration index, K is the maximum number of iterations, and n is a positive integer. This indicates that all the ideas in the whole swarm will periodically conduct a quantum-behaved individual updating operation in the n th iteration. Moreover, two different quantum-behaved states Q_{i+}^{k+1} and Q_{i-}^{k+1} can be improved as follows:

$$q_{id+}^{k+1} = \left[m_{id}^k + \ln(1/r^d)\left(b\left|\bar{c}_d^k - p_{id}^k\right|\right) + \zeta(\mathrm{k})\right] \tag{26}$$

and

$$q_{id-}^{k+1} = \left[m_{id}^k - \ln(1/r^d)\left(b\left|\bar{c}_d^k - p_{id}^k\right|\right) + \zeta(\mathrm{k})\right], \tag{27}$$

respectively. Here, m_{id}^k and $\bar{c}_d^k$ are obtained from (13) and (14), respectively; p_{id}^k is the dth dimension of the existing idea P_i^k; $\zeta(\mathrm{k})$ is formulated by (22). The parameter b decreases from 1 to $1 - b_0$ linearly:

$$b = 1 - b_0 * k/K, \tag{28}$$

where k is the current iteration index, K is the maximum number of iterations, and b_0 is a certain constant. Unlike (11) and (12), (26) and (27) use p_{id}^k and $\zeta(\mathrm{k})$ instead of u_{id}^k and $\eta_d(\mu, \sigma)\xi(\mathrm{k})$.

In particular, for $i \in \{1, 2, \cdots, N\}$, the fitness values of the new quantum-behaved idea Q_i^{k+1} with two different states are compared with that of the new idea P_i^{k+1}, and the best idea among them is preserved as a new idea for the next iterative updating procedure. By using the QBIU-PL strategy, our proposed BSO algorithm can avoid excessive exploration and achieves an effective balance between exploration and exploitation.

The pseudocode for the QBIU-PL strategy is listed in Algorithm 3. The corresponding flowchart is displayed in Fig. 3.

Algorithm 3: QBIU-PL Strategy

```
1:  $Q_i^{k+1}$, $i \in \{1, 2, \cdots, N\}$ is composed of two quantum states: $Q_{i+}^{k+1}$ and $Q_{i-}^{k+1}$;
2:  for each new idea $Q_i^{k+1}$, $i \in \{1, 2, \cdots, N\}$ do
3:    if $\Delta(T) = 1$ then
4:      Generate and update quantum-behaved idea $Q_i^{k+1}$ ($Q_{i+}^{k+1}$ and $Q_{i-}^{k+1}$) using (24)–(28);
6:    end if
7:    Evaluate the fitness values $F(P_i^{k+1})$, $F(Q_{i+}^{k+1})$, and $F(Q_{i-}^{k+1})$;
8:    if $F(Q_{i+}^{k+1}) < F(P_i^{k+1})$ then
9:      $P_i^{k+1} = Q_{i+}^{k+1}$;
10:   else if $F(Q_{i-}^{k+1}) < F(P_i^{k+1})$
11:     $P_i^{k+1} = Q_{i-}^{k+1}$.
12:   end if
13: end for
```

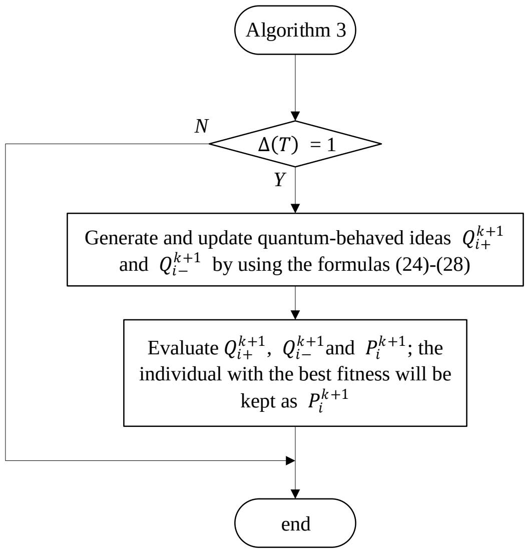

Algorithm 3
$\Delta(T) = 1$
N
Y
Generate and update quantum-behaved ideas Q_{i+}^{k+1} and Q_{i-}^{k+1} by using the formulas (24)-(28)
Evaluate Q_{i+}^{k+1}, $Q_{i-}^{k+1}$and P_i^{k+1}; the individual with the best fitness will be kept as P_i^{k+1}
end

FIGURE 3. Flowchart of QBIU-PL Strategy (Algorithm 3).

D. PROCEDURES OF PROPOSED ALGORITHM

The main steps of SBSO-PQLS are listed in Algorithm 4 with its flowchart shown in Fig. 4. First, the N ideas of the entire swarm are randomly initialized in the search space, and the corresponding fitness values are evaluated. Second, the SIC strategy is conducted for the entire swarm based on Algorithm 1. Third, cluster center disruption is carried out for the cluster centers of SBSO-PQLS. Fourth, the SIU strategy is conducted for each idea in the entire swarm. Fifth, when the QBIU-PL strategy is activated, each idea of the entire swarm is updated using the QBIU-PL strategy. Sixth, the individual selection strategy is used to discover the most promising ideas in the entire swarm. Finally, the idea with the best corresponding fitness value is achieved.

In addition, we use the same constraint mechanism used in the PSO algorithm to constrain the search range of SBSO-PQLS as follows:

$$p_{id}^k = min\left\{u_{d\_max}, max\left\{l_{d\_max}, p_{id}^k\right\}\right\}. \tag{29}$$

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

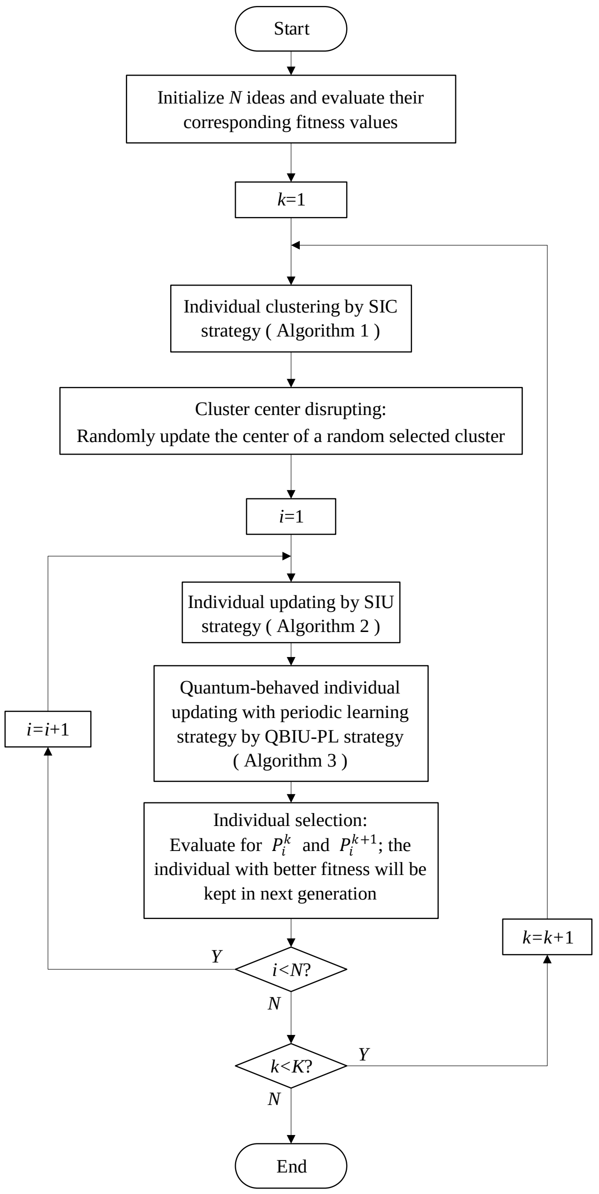

Start
Initialize N ideas and evaluate their corresponding fitness values
k=1
Individual clustering by SIC strategy (Algorithm 1)
Cluster center disrupting: Randomly update the center of a random selected cluster
i=1
Individual updating by SIU strategy (Algorithm 2)
Quantum-behaved individual updating with periodic learning strategy by QBIU-PL strategy (Algorithm 3)
Individual selection: Evaluate for P_i^k and P_i^{k+1}; the individual with better fitness will be kept in next generation
i=i+1
k=k+1
Y
i<N?
N
k<K?
Y
N
End

FIGURE 4. **Flowchart of SBSO-POLS Algorithm (Algorithm 4).**

Algorithm 4: SBSO-PQLS Algorithm

```
1: Initialization: Randomly generate $N$ ideas, $\{P_1^k, P_2^k, \cdots, P_N^k\}$; each
   idea denotes a potential solution in a $D$-dimensional search space,
   described as $P_i^k = [p_{i1}^k, p_{i2}^k, \cdots, p_{iD}^k]$, $i \in \{1, 2, \cdots, N\}$. Evaluate the
   $N$ ideas, $\{P_1^k, P_2^k, \cdots, P_N^k\}$; the corresponding fitness values are
   described as $\{F(P_1^k), F(P_2^k), \cdots, F(P_N^k)\}$; $k$ and $K$ are the current
   iteration index and maximum iteration number, respectively.
2: While (stop condition is not satisfied) do
3:     New Individual Clustering: Algorithm 1;
4:     Cluster Center Disruption: a cluster center is randomly selected
       from the centers of the $M$ clusters. The selected cluster center is
       replaced by a randomly generated idea if $r_0 < p_{r0}$, where the
       random number $r_0$ is uniformly distributed from 0 to 1; $p_{r0}$ is
       the pre-determined probability;
5:     New Individual Updating: Algorithm 2;
6:     Quantum-behaved Individual Updating with Periodic
       Learning Strategy: Algorithm 3;
7:     Individual Selection: Evaluate each new idea $P_i^{k+1}$, $i \in$
       $\{1, 2, \cdots, N\}$, select and keep the best idea according to (9).
8: end While
```

IV COMPARISONS WITH OTHER ALGORITHMS

A. BENCHMARK FUNCTIONS

The evaluations of SBSO-PQLS were carried out using a popular test suite, called the CEC2013 benchmark [27]. Section S-I of the supplementary material describes the CEC2013 test suite, which is composed of 28 benchmark functions, including shifted and rotated functions for real parameter optimization in complicated and difficult cases. These shifted and rotated functions are separated into three types according to their features. The first type contains 5 unimodal benchmark functions, F1–F5. The second type consists of 15 multimodal benchmark functions, F6–F20. The third type consists of the composition benchmark functions, F21–F28. For each function from the 28 CEC2013 benchmark functions, the dimension D is set to 30; each dimension is initialized within [-100, 100], and the search range of each dimension is set to [-100, 100].

B. BENCHMARK FUNCTIONS PARAMETER CONFIGURATIONS FOR COMPARED ALGORITHMS

The SBSO-PQLS algorithm was compared with seven BSO variants: the original BSO (BSO) [1], the modified BSO (MBSO) [12], the closed-loop brain storm optimization using random selection (CBSO-RS) [7], the simplified BSO (SBSO) [13], the BSO with a differential evolution strategy and a new step size method (BSODE) [21], the quantum-behaved BSO (QBSO) [20], and the advanced discussion mechanism-based BSO (ADMBSO) [22]. The above seven BSO algorithms are typical BSO algorithms that have achieved good performances in the literature. As listed in Table I, their parameter configurations are set following the original references, except for the population size and the maximum fitness evaluations (FES).

Furthermore, the proposed BSO algorithm was compared with the DE using the DE/rand/1/bin strategy [11] and the global version of PSO (GPSO) [10] to determine whether the proposed algorithm could surpass the widely used GPSO and DE algorithms (their parameter configurations are also listed in Table I). In addition, the parameters T and b_0 of the SBSO-PQLS algorithm were set to 100 and 0.9, respectively.

TABLE I
ALGORITHM PARAMETERS CONFIGURATIONS

Algorithm	Year	Parameter Settings	Reference
BSO	2011	p_{r0}=0.2, p_{r1}=0.8, p_{r11}=0.4, p_{r21}=0.5, c=25, M=5	[1]
MBSO	2012	p_{r0}=0.2, p_{r1}=0.8, p_{r11}=0.4, p_{r21}=0.5, c=25, M=5, p_r= 0.005	[12]
CLBSO-RS	2013	p_{r0}=0.2, p_{r1}=0.8, p_{r11}=0.4, p_{r21}=0.5, M=5	[7]
SBSO	2015	p_{r0}=0.2, p_{r1}=0.8, p_{r11}=0.4, p_{r21}=0.5, c=25, M=5	[13]
BSODE	2015	p_{r0}=0.2, p_{r1}=0.8, p_{r11}=0.4, p_{r21}=0.5, M=5, CR=0.5, F=0.5	[21]
QBSO	2015	p_{r0}=0.2, p_{r1}=0.8, p_{r11}=0.4, p_{r21}=0.5, M=5, b_0=0.5	[20]
ADMBSO	2015	p_{cen}=0.7, p_{ind}=0.2, p_{rnd}=0.1, p_{cens}=0.7, p_{low}=0.2, p_{high}=0.2, m=5	[22]
DE	1997	CR=0.5, F=0.5	[11]
PSO	1999	ω: 0.9-0.4, c1=2, c2=2	[10]
SBSO-PQLS		p_{r0}=0.2, p_{r1}=0.8, p_{r11}=0.4, p_{r21}=0.5, p_{r2}=0.005, M=5, T=100, b_0=0.9	

The above algorithms were all implemented in MATLAB R2014b and executed on a PC with an Intel Core (TM) CPU i7-4790 CPU @ 3.60 GHz with 16 GB RAM. Each of these

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

algorithms was independently executed 30 times on each of 28 CEC2013 benchmark functions using the same maximum fitness evaluation (FES), 400,000. The maximum number of iterations was set to 8000, and the population size was set to 50. The dimensions of each function were set to 30.

C. COMPARISION WITH BSO VARIANTS, PSO AND DE

The SBSO-PQLS algorithm was compared with seven other BSO variants, DE, and PSO on 28 CEC2013 benchmark functions. We have evaluated the error mean values between the best solution found by the algorithms listed in Table I and the global optimum solution over 30 runs on each function of 28CEC 2013 functions. The error mean value is described as

$$\text{Mean} = \sum_{k=1}^{30}\left[F_k(P) - F(P^*)\right]/30, \tag{30}$$

where $F(P)$ and $F(P^*)$ denote the best result found by the algorithms listed in Table I and the global optimum solution on each function of the 28 functions, respectively. In addition, the standard deviation value of each function is defined as

$$\text{STD} = \sqrt[2]{\sum_{k=1}^{30}\left[F_k(P) - \text{Mean}\right]^2/N-1}\ . \tag{31}$$

Overall evaluation: Table II shows the error mean and standard deviation values on the 28 functions resulting from each algorithm. The best result of each function among all algorithms is marked in bold. We can observe that SBSO-PQLS achieves the best results on 4 of the 28 CEC2013 benchmark functions, the second-best results on 11, and the third best results on 5. However, it performed the worst or the second worst result on only 1. In addition, it performed third-worst on 2 of the 28 CEC2013 benchmark functions. By evaluating the error mean and standard deviation values of the 28 functions, we can obtain the average and final rank for each algorithm, shown the last two rows of Table II. Although SBSO-PQLS did not achieve the best result on any of the 28 CEC2013 benchmark functions, it achieves the best average rank value among all the tested algorithms, 3.46 on the 28 CEC 2013 benchmark functions. This indicates that SBSO-PQLS algorithm obtained the best overall performance on the 28 CEC2013 benchmark functions among all ten tested algorithms.

Unimodal functions F1-F5: Table II shows that SBSO, MBSO, and BSODE achieved the best result on unimodal functions F1, F2 and F4, respectively; DE achieved the best results on the unimodal functions F3 and F5. However, QBSO had the worst results on the unimodal functions F1, F3, and F5, SBSO was worst on the unimodal function F4, and DE was worst on the unimodal function F2.

Furthermore, Table II shows that SBSO-PQLS achieved the second-best result on function F5, the third best results on functions F1 and F3, and the fourth best results on functions F2 and F4. Interestingly, SBSO-PQLS did not achieve the best result on any of the unimodal functions, but it also did not have the worst result on any unimodal function. Similarly, BSO, CLBSO, ADMBSO, and PSO did not achieve the best results on any of the unimodal functions, nor did they have the worst results among all the unimodal functions.

The average rank for each algorithm on all unimodal functions is listed in Table III, which shows that SBSO-PQLS achieved the best average rank, 3.25, on all the unimodal functions. This indicates that the proposed algorithm has steady performance on five of the unimodal functions, and thus achieved the best overall performance on all five.

Multimodal functions F6-F20: Table II shows that SBSO-PQLS achieved the best results on the multimodal functions F9, F12, and F13. BSODE achieved the best results on the multimodal functions F10, F15, and F16. CLBSO achieved the best results on the multimodal functions F6, F18, and F20. BSO achieved the best result on the multimodal function F8. PSO achieved the best results on the multimodal functions F11, F14, F17, and F19. DE achieved the best result on multimodal function F7.

However, QBSO was the worst on the multimodal functions F6, F7, F10, and F17-F19. BSO performed the worst on the multimodal functions F11–F14. ADMBSO performed the worst on the multimodal function F16. DE was in last place on the multimodal functions F9 and F15, and PSO performed the worst on the multimodal function F20. SBSO-PQLS algorithm was the worst on the multimodal function F8; however, as Table II shows, its performance on that function was only slightly worse than the best result of the BSO algorithm on that function. From the above analysis, SBSO-PQLS has the best overall performances on all the multimodal functions of all the tested algorithms listed in Table III.

Composition functions F21-F28: The composition functions consist of various basic unimodal and multimodal benchmark functions with a randomly situated global optimum and several randomly situated deep local optima. Table II shows that SBSO-PQLS achieves the best result on the composition function F28. ADMBSO achieves the best results on the composition functions F25 and F26. MBSO achieves the best result on the composition function F23. DE achieves the best results on the composition functions F24, F27, and F28, and PSO achieves the best results on composition functions F21 and F22. However, QBSO performs the worst on the composition functions F21, F22, F24, F27, and F28. DE performs the worst on the composition function F23. BSO performs the worst on the composition function F25, and PSO performs the worst on the composition function F26. Table III shows the average rank for each algorithm on all the composition functions. ADMBSO achieves the best average rank, 3.13, on all the composition functions, and MBSO is in second place, with 3.5, on all the composition functions. The SBSO-PQLS algorithm achieves the third-best average rank, 3.75, on all the composition functions.

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

TABLE II
COMPARISON OF EXPERIMENTAL RESULTS AMONG TEN ALGORITHMS ON 30 DIMENSIONS CEC 2013 TEST FUNCTIONS

Function	Evaluation Criteria	ADMBSO	BSO	BSODE	SBSO	MBSO	QBSO	CLBSO-RS	DE	PSO	SBSO-PQLS
F1	Mean	2.88E-13	1.29E-13	1.89E-13	**0**	2.96E-13	2.25E-02	2.05E-13	7.58E-15	4.55E-13	1.06E-13
	Std	2.53E-13	1.15E-13	8.62E-14	0.00E+00	1.22E-13	9.71E-03	6.94E-14	4.15E-14	1.58E-13	1.15E-13
	rank	7	4	5	**1**	8	10	6	2	9	3
F2	Mean	1.06E+05	1.04E+06	2.82E+05	1.85E+06	**9.48E+04**	2.11E+07	1.20E+05	5.69E+07	1.26E+07	1.37E+05
	Std	1.57E+05	3.54E+05	9.57E+04	3.02E+05	5.67E+04	6.86E+06	5.32E+04	1.18E+07	9.47E+06	7.15E+04
	rank	2	6	5	7	**1**	9	3	10	8	4
F3	Mean	1.47E+07	5.19E+07	3.35E+07	5.51E+07	1.07E+07	5.58E+09	5.58E+06	**9.37E+02**	1.17E+08	6.00E+06
	Std	3.68E+07	6.79E+07	4.40E+07	7.21E+07	1.67E+07	3.48E+09	1.11E+07	3.93E+03	1.22E+08	1.05E+07
	rank	5	7	6	8	4	10	2	**1**	9	3
F4	Mean	1.05E+02	1.57E+04	**2.22E+00**	3.23E+04	1.19E+02	2.56E+04	3.35E+03	1.86E+04	3.43E+03	5.00E+02
	Std	2.49E+02	6.77E+03	3.40E+00	7.28E+03	9.88E+01	3.82E+03	7.19E+03	2.86E+03	7.83E+02	7.27E+02
	rank	2	7	**1**	10	3	9	5	8	6	4
F5	Mean	3.64E-13	1.06E-02	1.11E-03	6.20E-03	4.40E-13	2.50E+00	3.37E-13	**1.14E-13**	4.36E-13	1.63E-13
	Std	1.78E-13	2.25E-03	2.07E-04	1.59E-03	2.56E-13	5.05E+00	1.85E-13	1.28E-29	1.16E-13	5.73E-14
	rank	4	9	7	8	6	10	3	**1**	5	2
F6	Mean	2.23E+01	3.35E+01	2.26E+01	4.44E+01	2.16E+01	9.02E+01	**1.26E+01**	1.38E+01	6.61E+01	1.74E+01
	Std	2.19E+01	2.73E+01	2.12E+01	2.99E+01	1.94E+01	3.58E+01	6.61E+00	5.42E+00	3.40E+01	1.56E+01
	rank	5	7	6	8	4	10	**1**	2	9	3
F7	Mean	3.35E+01	1.21E+02	2.59E+02	1.02E+02	3.24E+01	6.47E+02	3.09E+01	**1.91E-01**	3.44E+01	1.97E+01
	Std	2.17E+01	4.29E+01	3.39E+02	2.84E+01	1.81E+01	1.31E+03	1.55E+01	5.08E-01	1.19E+01	1.68E+01
	rank	5	8	9	7	4	10	3	**1**	6	2
F8	Mean	2.0935E+01	**2.0883E+01**	2.0890E+01	2.0890E+01	2.0915E+01	2.0933E+01	2.0924E+01	2.0927E+01	2.0929E+01	2.0949E+01
	Std	4.67E-02	8.60E-02	6.34E-02	6.24E-02	6.85E-02	7.12E-02	6.06E-02	4.11E-02	7.20E-02	5.29E-02
	rank	9	**1**	3	2	4	8	5	6	7	10
F9	Mean	2.63E+01	3.26E+01	3.02E+01	2.70E+01	2.19E+01	3.72E+01	2.34E+01	3.84E+01	2.27E+01	**1.68E+01**
	Std	5.04E+00	2.71E+00	3.67E+00	4.18E+00	3.94E+00	2.75E+00	4.23E+00	1.18E+00	3.33E+00	4.61E+00
	rank	5	8	7	6	2	9	4	10	3	**1**
F10	Mean	2.86E-01	2.74E-01	**6.08E-03**	4.78E-01	2.00E-01	1.40E+01	1.43E-01	7.15E-03	2.22E-01	1.32E-01
	Std	1.59E-01	2.58E-01	4.65E-03	2.91E-01	1.19E-01	5.63E+00	6.62E-02	3.90E-03	1.49E-01	5.15E-02
	rank	8	7	**1**	9	5	10	4	2	6	3
F11	Mean	4.14E+01	4.15E+02	2.88E+02	3.14E+02	6.32E+01	4.14E+02	5.40E+01	8.14E+01	**2.34E+01**	3.06E+01
	Std	1.20E+01	8.35E+01	7.88E+01	6.27E+01	2.07E+01	7.83E+01	1.71E+01	7.27E+00	8.12E+00	7.66E+00
	rank	3	10	7	8	5	9	4	6	**1**	2
F12	Mean	5.05E+01	4.19E+02	3.30E+02	3.08E+02	6.56E+01	4.16E+02	5.13E+01	1.79E+02	7.61E+01	**3.68E+01**
	Std	1.29E+01	8.49E+01	8.23E+01	5.13E+01	1.72E+01	8.02E+01	1.46E+01	1.04E+01	3.28E+01	1.22E+01
	rank	2	10	8	7	4	9	3	6	5	**1**
F13	Mean	1.07E+02	4.883E+02	4.32E+02	4.16E+02	1.33E+02	4.880E+02	1.18E+02	1.81E+02	1.54E+02	**8.78E+01**
	Std	3.48E+01	6.94E+01	8.08E+01	5.11E+01	3.38E+01	9.58E+01	2.78E+01	9.47E+00	3.79E+01	2.37E+01
	rank	2	10	8	7	4	9	3	6	5	**1**
F14	Mean	3.32E+03	4.29E+03	3.62E+03	3.74E+03	2.88E+03	4.18E+03	2.88E+03	3.92E+03	**8.85E+02**	2.41E+03
	Std	7.41E+02	4.02E+02	6.34E+02	8.38E+02	7.43E+02	5.60E+02	7.55E+02	2.18E+02	2.54E+02	7.82E+02
	rank	5	10	6	7	3	9	4	8	**1**	2
F15	Mean	6.38E+03	4.393E+03	**4.387E+03**	4.42E+03	4.84E+03	4.59E+03	5.17E+03	7.20E+03	6.71E+03	6.68E+03
	Std	1.44E+03	5.43E+02	6.49E+02	6.09E+02	1.57E+03	6.70E+02	1.44E+03	1.63E+02	7.62E+02	1.16E+03
	rank	7	2	**1**	3	5	4	6	10	9	8
F16	Mean	2.48E+00	1.69E-01	**3.57E-02**	6.11E-01	2.36E+00	1.61E+00	2.451E+00	2.450E+00	2.29E+00	2.454E+00
	Std	2.61E-01	4.74E-02	2.06E-02	4.89E-01	3.30E-01	4.52E-01	1.90E-01	2.41E-01	3.26E-01	3.07E-01
	rank	10	2	**1**	3	6	4	8	7	5	9
F17	Mean	7.23E+01	4.35E+02	3.98E+02	3.31E+02	8.78E+01	5.29E+02	7.95E+01	1.16E+02	**5.60E+01**	6.29E+01
	Std	1.25E+01	9.32E+01	1.32E+02	6.40E+01	1.64E+01	1.10E+02	1.54E+01	8.02E+00	1.78E+01	1.10E+01
	rank	3	9	8	7	5	10	4	6	**1**	2
F18	Mean	1.94E+02	3.86E+02	3.54E+02	3.97E+02	2.00E+02	5.61E+02	**1.87E+02**	2.09E+02	2.35E+02	1.90E+02
	Std	2.78E+01	7.26E+01	8.45E+01	5.16E+01	1.20E+01	1.08E+02	4.28E+01	7.42E+00	2.72E+01	8.65E+00
	rank	3	8	7	9	4	10	**1**	5	6	2
F19	Mean	7.41E+00	7.93E+00	9.63E+00	1.43E+01	6.69E+00	9.21E+01	5.33E+00	1.19E+01	**3.42E+00**	3.63E+00
	Std	6.05E+00	1.81E+00	1.73E+00	2.22E+00	1.83E+00	2.54E+01	1.36E+00	7.25E-01	8.45E-01	1.16E+00
	rank	5	6	7	9	4	10	3	8	**1**	2
F20	Mean	1.143E+01	1.45E+01	1.41E+01	1.31E+01	1.15E+01	1.45E+01	**1.11E+01**	1.22E+01	1.46E+01	1.144E+01
	Std	6.64E-01	1.20E-01	5.73E-01	9.55E-01	6.92E-01	3.17E-01	6.99E-01	2.28E-01	1.01E+00	5.66E-01
	rank	2	8	7	6	4	9	**1**	5	10	3
F21	Mean	3.05E+02	3.42E+02	3.21E+02	3.52E+02	3.00E+02	3.54E+02	2.85E+02	2.88E+02	**2.81E+02**	3.21E+02
	Std	8.57E+01	1.01E+02	1.15E+02	9.02E+01	9.55E+01	6.94E+01	9.23E+01	4.78E+01	7.22E+01	9.70E+01
	rank	5	8	6	9	4	10	2	3	**1**	7
F22	Mean	3.23E+03	5.51E+03	4.74E+03	4.05E+03	2.77E+03	5.94E+03	3.21E+03	4.31E+03	**9.77E+02**	2.13E+03
	Std	1.02E+03	8.42E+02	7.64E+02	9.58E+02	9.78E+02	8.52E+02	7.45E+02	3.30E+02	2.38E+02	7.22E+02
	rank	5	9	8	6	3	10	4	7	**1**	2
F23	Mean	5.16E+03	5.50E+03	5.62E+03	5.64E+03	**4.66E+03**	5.94E+03	5.06E+03	7.39E+03	6.42E+03	6.05E+03
	Std	1.36E+03	7.89E+02	8.03E+02	7.69E+02	1.16E+03	1.10E+03	9.95E+02	2.88E+02	1.11E+03	1.49E+03
	rank	3	4	5	6	**1**	7	2	10	9	8
F24	Mean	2.53E+02	3.09E+02	2.95E+02	3.11E+02	2.59E+02	3.18E+02	2.54E+02	**2.00E+02**	2.67E+02	2.47E+02
	Std	9.97E+00	1.77E+01	1.41E+01	1.59E+01	9.39E+00	1.22E+01	8.27E+00	7.32E-04	7.40E+00	9.71E+00
	rank	3	8	7	9	5	10	4	**1**	6	2
F25	Mean	**2.69E+02**	3.53E+02	3.16E+02	3.51E+02	2.80E+02	3.29E+02	2.78E+02	2.98E+02	2.84E+02	2.69E+02
	Std	1.19E+01	1.63E+01	1.19E+01	1.60E+01	1.09E+01	1.52E+01	1.36E+01	1.93E+01	8.66E+00	9.81E+00
	rank	**1**	10	7	9	4	8	3	6	5	2
F26	Mean	**2.00E+02**	2.34E+02	2.07E+02	3.07E+02	2.29E+02	2.78E+02	2.56E+02	2.04E+02	3.14E+02	2.50E+02
	Std	1.01E-02	6.60E+01	3.10E+01	7.72E+01	5.98E+01	9.61E+01	7.49E+01	9.47E-01	7.03E+01	6.74E+01
	rank	**1**	5	3	9	4	8	7	2	10	6
F27	Mean	7.85E+02	1.12E+03	1.11E+03	1.03E+03	7.81E+02	1.31E+03	8.42E+02	**3.02E+02**	8.82E+02	7.01E+02
	Std	1.33E+02	1.19E+02	1.13E+02	8.77E+01	1.18E+02	7.76E+01	1.27E+02	8.35E+00	7.36E+01	8.20E+01
	rank	4	9	8	7	3	10	5	**1**	6	2
F28	Mean	3.36E+02	3.96E+03	3.52E+03	3.34E+03	3.40E+02	4.13E+03	3.71E+02	**3.00E+02**	4.12E+02	**3.00E+02**
	Std	1.96E+02	5.71E+02	6.32E+02	3.16E+02	2.21E+02	4.77E+02	3.00E+02	0.00E+00	3.67E+02	0.00E+00
	rank	3	9	8	7	4	10	5	**1**	6	**1**
Overall	Average rank	4.25	7.18	5.79	6.93	4.04	8.96	3.75	5.04	5.57	**3.46**
Overall	Final rank	4	9	7	8	3	10	2	5	6	**1**

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

TABLE III
AVERAGE AND FINAL RANK AMONG TEN ALGORITHMS ON UNIMODAL FUNCTIONS F1-F5, MULTIMODAL FUNCTIONS F6-F20, AND COMPOSITION FUNCTIONS F21-F28 WITH 30 DIMENSION

Function	Evaluation Criteria	ADMBSO	BSO	BSODE	SBSO	MBSO	QBSO	CLBSO-RS	DE	PSO	SBSO-PQLS
Unimodal Functions F1-F5	Average rank	4.00	6.60	4.80	6.80	4.40	9.60	3.80	4.40	7.40	**3.20**
Unimodal Functions F1-F5	Final rank	3	7	6	8	4	10	2	4	9	**1**
Multimodal Functions F6-F20	Average rank	4.93	7.07	5.73	6.53	4.20	8.67	3.60	5.87	5.00	**3.40**
Multimodal Functions F6-F20	Final rank	4	9	6	8	3	10	2	7	5	**1**
Composition Functions F21-F28	Average rank	**3.13**	7.75	6.50	7.75	3.50	9.13	4.00	3.88	5.50	3.75
Composition Functions F21-F28	Final rank	**1**	8	7	8	2	10	5	4	6	3

D. STATISTICAL ANALYSIS

To compare SBSO-PQLS with each of the nine compared algorithms on each benchmark function of the 28 CEC2013 benchmark functions at a statistical hypothesis level of 0.05, non-parametric Wilcoxon signed-rank tests were conducted, and the results are listed in Table IV. Each result is described using a mathematical symbol (">", "=", or "<"), denoting that SBSO-PQLS achieved a significantly better, equal, or significantly worse performance than the compared algorithm, respectively. For instance, the statistical results for the comparison between SBSO-PQLS and ADMBSO are listed in the first column of Table IV, where 16 ">" symbols denote that SBSO-PQLS achieved significantly better performance than ADMBSO on the benchmark functions F1, F5, F7, F9-F14, F17-F19, F22, F24, F27, and F28; 8 "=" symbols mean that SBSO-PQLS was statistically equal to ADMBSO on the benchmark functions F3, F6, F8, F15, F16, F20, F21, and F25; 4 "<" symbols mean that the SBSO-PQLS algorithm performed significantly worse than the ADMBSO algorithm on the benchmark functions F2, F4, F23, and F26.

TABLE IV
RESULTS OF WILCOXON SIGNED RANK TESTS FOR SBSO-PQLS AND NINE ALGORITHMS

	Pairwise Comparison: SBSO-PQLS Versus								
Function	ADMBSO	BSO	BSODE	SBSO	MBSO	QBSO	CLBSO	DE	PSO
F1	>	=	>	<	>	>	>	<	>
F2	<	>	>	>	<	>	=	>	>
F3	=	>	>	>	=	>	=	<	>
F4	<	>	<	>	<	>	>	>	>
F5	>	>	>	>	>	>	>	<	>
F6	=	>	=	>	=	>	=	=	>
F7	>	>	>	>	>	>	>	<	>
F8	=	<	<	<	<	=	=	=	=
F9	>	>	>	>	>	>	>	>	>
F10	>	>	<	>	>	>	=	<	>
F11	>	>	>	>	>	>	>	>	<
F12	>	>	>	>	>	>	>	>	>
F13	>	>	>	>	>	>	>	>	>
F14	>	>	>	>	>	>	>	>	<
F15	=	<	<	<	<	<	<	>	=
F16	=	<	<	<	=	<	=	=	<
F17	>	>	>	>	>	>	>	>	=
F18	>	>	>	>	>	>	=	>	>
F19	>	>	>	>	>	>	>	>	=
F20	=	>	>	>	=	>	=	>	>
F21	=	=	=	=	=	>	=	=	=
F22	>	>	>	>	>	>	>	>	<
F23	<	<	=	<	<	=	<	>	=
F24	>	>	>	>	>	>	>	<	>
F25	=	>	>	>	>	>	>	>	>
F26	<	=	=	>	=	=	=	=	>
F27	>	>	>	>	>	>	>	<	>
F28	>	>	>	>	>	>	>	>	>
">"/ "=" /"<"	**16/8/4**	**21/3/4**	**19/4/5**	**22/1/5**	**17/6/5**	**23/3/2**	**16/10/2**	**16/5/7**	**18/6/4**

The statistical results from Table IV show that the SBSO-PQLS algorithm can surpass the other algorithms. The number of CEC2013 benchmark functions on which SBSO-PQLS achieved significantly better performance than the other nine algorithms, is greater than the number on which the SBSO-PQLS algorithm performed significantly worse than the other nine algorithms. Therefore, the results further show that our SBSO-PQLS algorithm has significantly better overall performance than the other nine algorithms on the CEC2013 benchmark functions, indicating that the SBSO-PQLS algorithm has the best global search capability among all ten tested algorithms.

V EXPERIMENTAL INVESTIGATION ON PROPOSED ALGORITHM

To fully understand the effects of the SIC strategy, the SIU strategy, and the QBIU-PL strategy of SBSO-PQLS, experiments were conducted on the 28 CEC2013 benchmark functions [27] in 30 dimensions with 30 independent runs. These experiments were also implemented in MATLAB R2014b, and executed on a PC with an Intel Core (TM) CPU i7-4790 CPU @ 3.60 GHz with 16 GB RAM. The maximum number of iterations was set to 8000, and the population size was set to 50. The maximum number of fitness evaluations (FES) was set to 400,000.

A. SIC STRATEGY INVESTIGATION

To clearly distinguish the effectiveness of SIC strategy, we compared SBSO-PQLS with the following variants: SBSO-PQLS with K-means (SBSO-PQLS-11), and SBSO-PQLS with SGM (SBSO-PQLS-12) on the 28 CEC2013 benchmark functions. In other words, while the SBSO-PQLS algorithm adopts the SIU strategy as the individual clustering strategy, the SBSO-PQLS-11 and the SBSO-PQLS-12 algorithm adopt the K-means strategy and the SGM strategy to replace the SIU strategy, respectively. This can provide a fair comparison between the SIC, K-means, and SGM strategies. The parameter configurations of the above algorithms were set the same as these of the SBSO-PQLS algorithm, shown in Table I. In addition, Table S-II of Section S-II of the supplementary material shows the error mean and standard deviation values of the 28 functions on each algorithm. Based on the error mean and standard deviation values of the 28 functions, we can achieve the average and final rank for each algorithm listed in the Table V, where the SBSO-PQLS algorithm achieved the best average rank compared with the other two algorithms. This indicates that the SIC strategy performs

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

better than the K-means and the SGM on the 28 CEC2013 benchmark functions.

TABLE V
COMPARISON BETWEEN SBSO-PQLS, AND SBSO-PQLS-11 AND SBSO-PQLS-12 FOR EVALUATING SIC STRATEGY

Function	Evaluation Criteria	SBSO-PQLS	SBSO-PQLS-11	SBSO-PQLS-12
Overall	Average rank	**1.75**	2.18	2.04
Overall	Final rank	**1**	3	2
Overall	Average computational time	**1.79E+01**	5.03E+01	1.45E+02

Moreover, Table S-II of Section S-II of the supplementary material shows the mean values of computational time (t-mean) among three algorithms on each of the 28 CEC2013 benchmark functions over 30 runs in units of seconds. From these results, we define the mean value of the sum of the mean values of computational time on all the 28 functions for each algorithm as its average computational time for evaluating the computational time cost shown in Table V. It can be observed that SBSO-PQLS achieves the lowest average computational time on all the 28 CEC2013 functions among the three algorithms.

Therefore, the SIC strategy not only improves global search capabilities, but also has the lowest computational time cost compared with the K-means and the SGM strategies.

B. SIU STRATEGY INVESTIGATION

To clearly understand the effectiveness of the SIU strategy, SBSO-PQLS is compared with SBSO-PQLS using the individual updating from BSO (SBSO-PQLS-2) on the 28 CEC2013 benchmark functions. The SBSO-PQLS algorithm adopts the SIU strategy as its individual updating strategy and SBSO-PQLS-2 adopts the individual updating strategy from the BSO algorithm in place of the SIU strategy. The parameters of the SBSO-PQLS-2 algorithm were set to the same values as those of the SBSO-PQLS algorithm shown in Table I.

Table S-III of Section S-II of the supplementary material shows the error mean and standard deviation values of the 28 functions from each algorithm. By evaluating the mean error and standard deviation values of the 28 functions, we can obtain the average and final rank for each algorithm, listed in Table VI. The results show that SBSO-PQLS achieved a higher average rank than SBSO-PQLS-2, indicating that the SIU strategy performs better than the individual updating strategy from BSO algorithm on most of the benchmark functions.

In addition, Table S-III of Section S-II of the supplementary material also shows the mean value of the computational time (t-mean) on each of the 28 CEC2013 benchmark functions in 30 runs for each algorithm in units of seconds. To evaluate the computational time cost, we define the mean value of the sum of the mean values of computational time on all the 28 functions for each algorithm as its average computational time. As listed in Table VI, the results concerning the average computational time show that SBSO-PQLS provide a lower computational time cost than does SBSO-PQLS-2 on all the 28 CEC2013 benchmark functions. This is due to the fact that SBSO-PQLS adopts the SIU strategy.

From the above comparisons, the SIU strategy not only enhances global search capabilities but also has the lowest computational time cost compared with the individual updating strategy from the BSO algorithm.

TABLE VI
COMPARISON BETWEEN SBSO-PQLS AND SBSO-PQLS-2 FOR EVALUATING SIU STRATEGY

Function	Evaluation Criteria	SBSO-PQLS	SBSO-PQLS-2
Overall	Average rank	**1.25**	1.75
Overall	Final rank	**1**	2
Overall	Average computational time	**1.79E+01**	2.22E+01

C. QBIU-PL Strategy Investigation

In the previous experiments, SBSO-PQLS adopts T=100, and b_0=0.9 in the QBIU-PL strategy; however, different T and b_0 parameter values can affect the performance of SBSO-PQLS. Here, we investigate parameters T and b_0 through experiments on the 28 CEC2013 benchmark functions.

(1) Configuration for Parameter T

First, parameter T is varied. Using different T parameter values for the SBIU-PL strategy can affect the global search capability of SBSO-PQLS. To evaluate the effects of different values of T, we tested a series of values $T \in \{0, 1, 50, 100, 200, 500, 1000\}$ when $b_0 = 0.9$. Table S-IV of Section S-II of the supplementary material shows the error mean error and standard deviation values of the 28 functions from each $T \in \{0, 1, 50, 100, 200, 500, 1000\}$. According to the error mean and standard deviation values of the 28 functions, we can evaluate the average and final ranks for $T \in \{0, 1, 50, 100, 200, 500, 1000\}$, listed in Table VII.

In Table VII, the QBIU-PL strategy with the parameter T = 500 achieves the highest overall rank on the 28 CEC2013 benchmark functions when $T \in \{0, 1, 50, 100, 200, 500, 1000\}$, and the QBIU-PL strategy with the parameter T = 1 achieves the worst final overall rank. Specially, the QBIU-PL strategy with T = 1 means that the quantum-behaved individual updating is executed in every generation. It is generally believed that the quantum-behavior generates new momentum and causes individuals to search across an extremely large range. However, applying the quantum behavior in every generation causes the SBSO-PQLS algorithm to perform excessive exploration with sketchy exploitation, which affects the search efficiency and convergence speed. To avoid excessive exploration and achieve an effective balance between exploration and exploitation, the quantum-behaved individual updating with periodic learning strategy is used. Therefore, the QBIU-PL strategies using $T \in \{50, 100, 200, 500, 1000\}$ achieve higher rankings than the QBIU-PL strategy with T = 1. The QBIU-PL strategy with T = 500 achieves the best performance among the QBIU-PL strategies with $T \in \{1, 50, 100, 200, 500, 1000\}$.

In addition, note that when T = 0 the SBSO-PQLS does

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

not use the quantum-behaved individual updating strategy; however, the QBIU-PL strategy with T = 0 achieves a better overall ranking on the 28 CEC2013 benchmark functions than do the QBIU-PL strategies when $T \in \{1, 50, 100, 200, 1000\}$. This is because the quantum-behavior of the QBIU-PL strategies with $T \in \{1, 50, 100, 200, 1000\}$ causes the SBSO-PQLS algorithm to perform excessive exploration with sketchy exploitation to. On the other hand, the QBIU-PL strategy with T = 500 achieves better performance than the QBIU-PL strategy with T = 0, which indicates that the QBIU-PL strategy can effectively enhance the performance of the SBSO-PQLS algorithm when an appropriate T value is selected.

TABLE VII
COMPARISON OF DIFFERENT PARAMETER T WITH b_0=0.9 IN QBIU-PL STRATEGY

Function	Evaluation Criteria	T=0	T=1	T=50	T=100	T=200	T=500	T=1000
Overall	Average rank	3.68	5.07	4.50	4.04	3.75	**2.68**	3.75
Overall	Final rank	2	7	6	5	3	**1**	3

(2) Configuration for Parameter b_0

Next, we varied parameter b_0. Using different b_0 parameter values in the SBIU-PL strategy also influences the global search capability of SBSO-PQLS. To verify the effects of different b_0 parameter values we tested SBSO-PQLS with $b_0 \in \{0.1, 0.5, 0.9\}$ when T = 500. Table S-V of Section S-II of the supplementary material shows the error mean and standard deviation values of the 28 functions from each $b_0 \in \{0.1, 0.5, 0.9\}$. In terms of the error mean and standard deviation values of the 28 functions, we can compute the average and final ranks for $b_0 \in \{0.1, 0.5, 0.9\}$, listed in Table VIII. The results show the QBIU-PL strategy with b_0 = 0.9 achieved the best final rank on the 28 CEC2013 benchmark functions. Therefore, the SBIU-PL strategy with parameters T = 500 and b_0 = 0.9 enables the SBSO-PQLS algorithm to achieve its best performance on the 28 CEC2013 benchmark functions.

In particular, as shown in Table II, the SBSO-PQLS algorithm with T = 100 and b_0 = 0.9 achieved the best performance on the 28 CEC2013 benchmark functions compared with the other nine algorithms. However, in Table VII, the SBSO-PQLS algorithm with T = 100 and b_0 = 0.9 was always the fifth best overall performance on the 28 CEC2013 benchmark functions compared with the SBSO-PQLS algorithm with different the values of T and b_0 = 0.9, indicating that SBSO-PQLS with $T \in \{0, 200, 500, 1000\}$ and b_0 = 0.9 can perform better compared with the other nine algorithms listed in Table I.

TABLE VIII
COMPARISON OF DIFFERENT PARAMETER b_0 WITH T = 500 IN QBIU-PL STRATEGY

Function	Evaluation Criteria	b_0=0.1	b_0=0.5	b_0=0.9
Overall	Average rank	1.50	2.43	**1.36**
Overall	Final rank	2	3	**1**

VI DISCUSSION AND CONCLUSION

The BSO algorithm proposed by Shi in 2011 is a young and promising swarm intelligence optimization algorithm. Its fundamental principle is to simulate the human brainstorming process. Its major procedures consist of individual clustering, cluster center disruption, individual updating, and individual selection. However, most BSO algorithms tend to become trapped in local optima when solving complex optimization problems, such as the CEC2013 functions. In addition, most BSO algorithms include a complex individual clustering strategy and a redundant individual updating strategy, resulting in high computational costs. To overcome these defects, this study developed SBSO-PQLS. Compared with other BSO variants, the SBSO-PQLS algorithm has the following characteristics and advantages.

First, the SIC strategy is developed to improve the individual clustering strategy. Unlike the clustering strategies used in most BSO algorithms, such as K-means, the SIC strategy sorts the entire swarm based on the fitness values of all the individuals and then rationally assigns individuals to various clusters. Because the SIC strategy can generate a reasonable population diversity, SBSO-PQLS can avoid premature convergence and has an improved global search capability. In addition, because the SIC strategy does not need to compute the distances among all individuals, it effectively reduces the computational cost.

Second, the SIU strategy used in the SBSO-PQLS algorithm was developed to improve the individual updating strategy. In contrast to the individual updating of most BSO algorithms, the SIU strategy enriches the generating pattern for new individuals, reduces the redundant information, and improves the step size function. Therefore, the SIU strategy further enhances the global search capabilities of the SBSO-PQLS algorithm and reduces its computational cost.

Third, the QBIU-PL strategy for the SBSO-PQLS algorithm was developed by incorporating a quantum-behaved mechanism into SBSO-PQLS to generate new momentum and cause individuals to escape local optima. Furthermore, a periodic learning strategy is integrated with the quantum-behaved mechanism to avoid the problem of excessive exploration with sketchy exploitation caused by the quantum-behaved mechanism. In addition, fitness evaluation is adopted for new individuals in the QBIU-PL strategy to select the most competitive individuals for the next iteration instead of the probability selection mechanism of the QBSO algorithm in formula (10). Therefore, the QBIU-PL strategy effectively improves the global search capability and avoids premature convergence.

The results of the experiments on the CEC2013 benchmark functions confirm the contributions of the three

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access

new strategies to the SBSO-PQLS algorithm. By using the CEC2013 benchmark functions, we compared the SBSO-PQLS algorithm with seven popular BSO variants, PSO, and DE. The results demonstrate that SBSO-PQLS can effectively avoid premature convergence and that it achieves the best global search performance among all ten algorithms.

In general, the SIC strategy, the SIU strategy, and the QBIU-PL strategy cooperate in SBSO-PQLS to achieve its effective global search ability and reduce its computational burden when tackling complex optimizations.

Since swarm intelligence optimization algorithms share many similar features with evolution algorithms and are also treated as the evolution algorithm family [28], we will plan to incorporate evolution algorithms such as ACO [9], various DE strategies [11], and the imperialist competitive algorithm [29] into the BSO algorithm to further achieve an effective balance between the local exploitation and global exploration capabilities. Moreover, we will plan to exploit the SBSO-PQLS algorithm to optimize some specific science and engineering issues, such as the parameter estimation of the bio-impedance model of electro-tactile devices [30] and spectrum Management in cognitive radio Ad Hoc Networks [31].

REFERENCES

[1] Y. Shi, “Brain storm optimization algorithm,” in *Proc. 2nd Int. Conf. Swarm Intell.*, Chongqing, China, Jun. 12–15, 2011, pp. 303–309.

[2] Y. Shi, “An optimization algorithm based on brainstorming process,” *Int. J. Swarm Intell. Res.*, vol. 2, no. 4, pp. 35–62, Oct.–Dec. 2011.

[3] A. F. Osborn, “Applied Imagination: Principles and Procedures of Creative Problem Solving,” New York, NY, USA, 1963.

[4] J. Xue, Y. Wu, Y. Shi, and S. Cheng, “Multi-objective optimization based on brain storm optimization algorithm,” *Int Conf. on Advances in Swarm Intelligence, Springer-Verlag*, 2012, pp. 513-519.

[5] M. Arsuaga-Ríos and MA. Vega-Rodríguez. “Multi-objective energy optimization in grid systems from a brain storming strategy,” *Soft comput.*, vol. 19, no. 11, pp. 3159–3172, 2015.

[6] A. R. Jordehi, “Brainstorm optimization algorithm (BSOA): An efficient algorithm for finding optimal location and setting of FACTS devices in electric power systems,” *Int. J. Elec. Power*, vol. 69, pp. 48-57, 2015.

[7] C. Sun, H. Duan, and Y. Shi, “Optimal satellite formation reconfiguration based on closed-loop brain storm optimization,” *IEEE Comput. Intell. M.*, vol. 8, no. 4, pp. 39–51, 2013.

[8] H. Qiu and H. Duan, “Receding horizon control for multiple UAV formation flight based on modified Brain Storm Optimization,” *Nonlinear Dyn.*, vol.78, no.3, pp.1973–1988, 2014.

[9] M. Dorigo, “Optimization, Learning and Natural Algorithms,” Ph.D. thesis, Politecnico di Milano, Italy,1992.

[10] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, WA, Australia, 1995, pp. 1942–1948.

[11] R. Storn and K. Price, “Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces,” *J. Global Optim.*, vol. 11, pp. 341–359, 1997.

[12] Z. Zhan, J. Zhang, Y. H. Shi, and H. L. Liu, “A modified brain storm optimization,” in *Proc. IEEE Congr. Evol. Comput.*, Brisbane, Australia, Jun. 2012, pp. 1–8.

[13] J. Li, and H. Duan, “Simplified brain storm optimization approach to control parameter optimization in F/A-18 automatic carrier landing system,” *Aerosp. Sci. Technol.*, vol. 42, pp. 187-195, 2015.

[14] D. Zhou, Y. Shi, and S. Cheng, “Brain storm optimization algorithm with modified step-size and individual generation,” *Int. Conf. Adv. in Swarm Intell.*, Springer-Verlag, 2012, pp.243-252.

[15] S. Cheng, Y. Shi, Q. Qin, and Q. Zhang, and R. Bai, “Population diversity maintenance in brain storm optimization algorithm,” *J. Artif. Intell. Soft. Comput Res.*, vol.4, no. 2, pp. 83–97, 2014.

[16] H. Duan, S. Li, and Y. Shi, “Predator-prey based brain storm optimization for DC brushless motor,” *IEEE Trans. Magn.*, vol. 49, no. 10, pp. 5336–5340, 2013.

[17] K. R. Krishnanand, S. M. F. Hasani, B. K. Panigrahi, and S. K. Panda, “Optimal Power Flow Solution Using Self–Evolving Brain–Storming Inclusive Teaching–Learning–Based Algorithm,” Advances in Swarm Intelligence. Springer Berlin Heidelberg, pp. 338-345, 2013.

[18] J. Sun, B. Feng, and W. Xu, “Particle swarm optimization with particles having quantum behavior,” in *Proc. Congr. Evol. Comput.*, Portland, OR, USA, Jun. 2004, pp. 325–331.

[19] J. Sun, W. Xu, and B. Feng, “A global search strategy of quantum behaved particle swarm optimization,” in *Proc. IEEE Conf. Cybern. Intell. Syst.*, Singapore, Dec. 2004, vol. 1, pp. 111–116.

[20] H. Duan, and C. Li, “Quantum–behaved brain storm optimization approach to solving loney’s solenoid problem,” *IEEE Trans. Magn.*, vol. 51, no.1, pp.1–7, 2015.

[21] Z. Cao, L. Wang, X. Hei, Y. Shi, Rong X, “An improved brain storm optimization with differential evolution strategy for applications of ANNs,” *Math Problems Eng.*, pp. 1–18, 2015.

[22] Y. Yang, Y. Shi, and S. Xia, “Advanced discussion mechanism–based brain storm optimization algorithm,” *Soft. Computing*, vol.19, no.10, pp. 2997–3007, 2015.

[23] J. Chen, S. Cheng, Y. Chen, Y. Xie, and Y. Shi, “Enhanced Brain Storm Optimization Algorithm for Wireless Sensor Networks Deployment,” in *Proc. 6th Int. Conf. Adv. in Swarm & Comput. Intell.*, Springer-Verlag New York, Vol. 9140, pp. 373-381, 2016.

[24] Z. Cao, Y. Shi, X. Rong, B. Liu, Z. Du, and B. Yang, “Random Grouping Brain Storm Optimization Algorithm with a New Dynamically Changing Step Size,” *Int. Conf. Adv. in Swarm Intell.*, 2015, pp. 357–364.

[25] Z. Cao, X. Rong, and Z. Du, “An Improved Brain Storm Optimization with Dynamic Clustering Strategy,” *2016 the 3th Int. Conf. on Mec. & Mech. Eng.*, 2016, pp. 1–6.

[26] Z. Jia, H. Duan, Y. Shi, “Hybrid brain storm optimization and simulated annealing algorithm for continuous optimization problems,” *Int. J. Bio–Inspired Comput.*, vol.8, no.2, pp.109–121, 2016.

[27] J. J. Liang, B. Y. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, “Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization,” *Comput. Intell. Lab.*, Zhengzhou Univ., Zhengzhou, China, Tech. Rep. Dec. 2012, Jan. 2013.

[28] Y. H. Shi and R. C. Eberhart, “Comparison between genetic algorithms and particle swarm optimization,” *in Proc. 7th Int. Conf. Evolutionary Programming*, Mar. 1998, pp. 611–616.

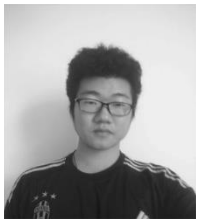
[29] E. Atashpaz-Gargari and C. Lucas, “Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition,” *2007 IEEE Congress on Evolutionary Computation*, Singapore, 2007, pp. 4661-4667.

[30] Y. Shen, J. Gregory, and N. Xi, “timulation Current Control for Load-aware Electrotactile haptic rendering: Modeling and Simulation,” *Robot Auton Syst*, 2014, vol. 62: 81~89.

[31] M. Parsapoor, “Spectrum Management in Cognitive Radio Ad Hoc Networks (CRAHNS) Using Bio-inspired Optimization Algorithms”, M.S. thesis, Dep. Comput. Electron. Eng., Halmstad Univ., Halmstad, Sweden, 2013.

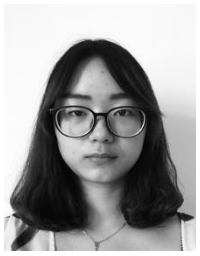
2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2776958, IEEE Access



ZHENSHOU SONG is currently working towards his B.Sc. degree in the School of Information Engineering, Nanchang University, Nanchang, China.

His current research interests include particle swarm optimization, genetic algorithms, and other computational intelligence techniques.



JIAQI PENG is currently working towards his B.Sc. degree in the School of Information Engineering, Nanchang University, Nanchang, China.

Her current research interests include particle swarm optimization, genetic algorithms, and other computational intelligence techniques.

CHUNQUAN LI received the B.Sc. M.Sc., and Ph.D. degrees from Nanchang University, Nanchang, China, in 2002, 2007, and 2015, respectively.

He has been with the School of Information Engineering, Nanchang University, since 2002, where he is currently an Associate Professor and a Young Scholar of Ganjiang River. He has published over 30 research articles.

He is also currently a Visiting Professor with the Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, Canada. His current interests include computing intelligence, haptics, virtual surgery simulation, robotics, and their applications to biomedical engineering.

PETER X. LIU (M'02–SM'07) received the B.Sc. and M.Sc. degrees from Northern Jiaotong University, Beijing, China, in 1992 and 1995, respectively, and the Ph.D. degree from the University of Alberta, Edmonton, AB, Canada, in 2002.

He has been with the Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, Canada, since 2002, where he is currently a Professor and the Canada Research Chair. He has published over 280 research articles. His current interests include computing intelligence, interactive networked systems and teleoperation, haptics, micro-manipulation, robotics, intelligent systems, context-aware intelligent networks, and their applications to biomedical engineering.

Dr. Liu was a recipient of the 2007 Carleton Research Achievement Award, the 2006 Province of Ontario Early Researcher Award, the 2006 Carty Research Fellowship, the Best Conference Paper Award of the 2006 IEEE International Conference on Mechatronics and Automation, and the 2003 Province of Ontario Distinguished Researcher Award. He serves as an Associate Editor for several journals including the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE/ASME TRANSACTIONS ON MECHATRONICS, the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, and the IEEE ACCESS. He is a licensed member of the Professional Engineers of Ontario and a fellow of the Engineering Institute of Canada.

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.