Research Article

A Local and Global Search Combined Particle Swarm Optimization Algorithm and Its Convergence Analysis

Weitian Lin,¹ Zhigang Lian,² Xingsheng Gu,¹ and Bin Jiao²

¹ Key Laboratory of Advanced Control and Optimization for Chemical Process, Ministry of Education, Shanghai 200237, China

² Shanghai Dianji University, Shanghai 201306, China

Correspondence should be addressed to Xingsheng Gu; xsgu@ecust.edu.cn

Received 4 December 2013; Accepted 6 February 2014; Published 16 March 2014

Academic Editor: Huaicheng Yan

Copyright © 2014 Weitian Lin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Particle swarm optimization algorithm (PSOA) is an advantage optimization tool. However, it has a tendency to get stuck in a near optimal solution especially for middle and large size problems and it is difficult to improve solution accuracy by fine-tuning parameters. According to the insufficiency, this paper researches the local and global search combine particle swarm algorithm (LGSCPSOA), and its convergence and obtains its convergence qualification. At the same time, it is tested with a set of 8 benchmark continuous functions and compared their optimization results with original particle swarm algorithm (OPSOA). Experimental results indicate that the LGSCPSOA improves the search performance especially on the middle and large size benchmark functions significantly.

1. Introduction

Swarm intelligence (SI) techniques, including particle swarm optimization (PSO) and ant colony optimization (ACO), use swarm behavior to solve the problem, and they use the concept of neighborhood intelligence along with individual intelligence. Particle swarm was originally formulated to study the interesting concept of a social behavior (bird flocking or fish schooling) in a simplified manner in simulation. However, very soon the potential of this technique was realized to develop into a powerful optimization tool which can be successfully applied in the fields of both continuous and discrete optimization problems such as function optimization, scheduling and hence received many attentions since it is originated.

The particle swarm concept originated as a simulation of simplified social system, and this technique has been developing rapidly. Eberhart and Kennedy in [1, 2] originally formulated particle swarm in 1995, and recently, several investigations have been undertaken to improve the performance of original PSO. There are many improved PSO algorithm such as Shi and Eberhart who introduced an inertia weight making the influence of the previous velocity on the

new velocity in [3–5] and Kennedy and Mendes use cluster analysis improving PSO's performance in [6]. Angelina presented the model of improved PSO algorithm called (HPSO) in [7] which uses selection operation of evolution computing. Literature [8] Suganthan sets up the model of PSO with adjacent filed operation. Jiao et al. present a dynamic inertia weight particle swarm optimization algorithm in paper [9] in which the dynamic inertia weight that decreases according to iterative generation increasing is used. Many experts also studied others properties of PSO such as Clerc and Kennedy in [10, 11] who have researched the particle swarm explosion stability and convergence in a multidimensional complex space. Eberhart and Shi have investigated PSO developments, applications, and resources in [12] and parameter selection in [13] and dimension selection methods in [13].

PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where GA can be applied. Parsopoulos and Vrahatis approach global optimization problems using PSO in [14, 15] and Abido uses PSO optimizing power flow in [16]. PSO technique have been used in assignment problem in [17], reactive power and voltage control considering voltage security assessment in [18], and so forth.

PSO algorithm is widely used, such as the paper [19, 20] that introduces a Hybrid Particle Swarm algorithm with artificial Immune Learning (HPSIL) for solving Fixed Charge Transportation Problem (FCTP). Hamta et al. in paper [21] presented a hybrid PSO algorithm for a multiobjective assembly line balancing problem with flexible operation times, sequence-dependent setup times, and learning effect. Chou [22] used a particle swarm optimization with cocktail decoding method for hybrid flow shop scheduling problems with multiprocessor tasks. This study searches for a number of solid decoding methods that can be incorporated into the cocktail decoding method. Then, it develops a particle swarm optimization (PSO) algorithm that can be combined with the cocktail decoding method. PSO has particularly gained prominence due to its relative ease of operation and capability to quickly arrive at an optimal/near-optimal solution. However it was pointed out in our studies that original PSO might frequently tend to get stuck in a near optimal solution in reaching optimum solutions especially for middle or large size function optimization problems. According to the shortcoming of original PSO, a local and global search combine particle swarm optimization algorithm is presented in this paper, and the simulation results show that it is efficacious. At the same time, this paper rigorously analyzes new PSO algorithm convergence and obtains its convergence qualification.

This work differs from the existing ones at least in four aspects: firstly, it proposes a local and global search combine particle swarm optimization (LGSCPSO) algorithm iterative formula, in which all particles share the best information of the local particles, global particles and neighborhood particles. Secondly, it finds the best combine parameters of LGSCPSO for different size optimization problems. Thirdly, it is to compare the original PSO with LGSCPSO and shows that the latter is more efficacious for optimization problems. Fourthly, it strictly analyzes the PSO algorithm's convergence and obtains its convergence qualification. The rest of the paper is organized as follows: The next section introduces the original PSO model. The iteration formulation of LGSCPSO algorithm is presented in Section 3. In Section 4, it analyzes the PSO algorithm's convergence and finds its convergence qualification. Then, it describes the test functions, experimental settings, and compares experimental results of original PSO with LGSCPSO algorithm. Finally, Section 5 summarizes the contribution of this paper and conclusions.

2. Original Particle Swarm Optimization Algorithm

2.1. Description of Original Particle Swarm Optimization Algorithm. Particle swarm optimization (PSO) is an evolutionary computation technique, and its concept originated as a simulation of a simplified social system. The major difference is that the evolution computing techniques use genetic operators whereas swarm intelligence techniques use the physical movements of the individuals in the swarm. PSO is distinctly different from other evolutionary-type methods

in a way that it does not need complex encoding and decoding processes and does not use the operation (such as crossover and/or mutation), and it takes real numbers as particles in the aspects of representation solution. From the procedure, one can learn that PSO shares many common points with GA; for example, they start with a neighborhood of a randomly generated population, evaluate the population with fitness values, update the population and search for the optimum with random techniques, and do not guarantee optimal. Similar to other population-based algorithms, PSO as an optimization tool can solve a variety of difficult optimization problems. Compared to GA, one of the important attractive factors of PSO is simple that there are very few parameters to adjust. It can achieve the optimal or near-optimal solutions in a rather short time without enormous iterative computations in digital implementation.

Swarm intelligence simulates a social behavior such as bird flocking to a promising position for certain objectives in a multidimensional space. It is initialized with a population of random solutions and searches for optima by updating generations. PSO system combines local search method (through self-experience) with global search methods (through neighboring experience), attempting to balance exploration. Like evolutionary algorithm, PSO search uses a population (called swarm) of individuals (called particles) that are updated from iteration to iteration. Each particle represents a candidate position (i.e., solution) to the problem at hand, resembling the chromosome of GA. A particle is treated as a point in an M -dimension space, and the status of a particle is characterized by its position and velocity. Initialized with a swarm of random particles, PSO is achieved through particle flying along the trajectory that will be adjusted based on the best experience or position of the one particle (called local best) and ever found by all particles (called global best). PSO updates a population of particles with the internal velocity and attempts to profit from the discoveries of themselves and previous experiences of all other companions.

2.2. The Model of Original Particle Swarm Optimization Algorithm. In every search iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far, and this value is called $pBest$. Another "best" value obtained so far by any particle in the population, and this best value is a global best and called $gBest$. After finding the two best values, the particle updates its velocity and positions with following formulas (1) and (2):

$$v_{id}(k+1) = wv_{id}(k) + c_1r_1(p_{id}(k) - x_{id}(k)) + c_2r_2(p_{gd}(k) - x_{id}(k)), \quad (1)$$

$$x_{id}(k+1) = x_{id}(k) + v_{id}(k+1) \quad (i = 1, 2, \dots, m; d = 1, 2, \dots, D). \quad (2)$$

In (1), $P_i = (p_{i1}, p_{i2}, \dots, p_{iD})$ is the best previous position of the i th particle (also known as $pBest$). According to the different definitions, there are two different versions of PSO, with local version, and particles only have information of their own and their neighbors' bests, rather than that of

the entire neighborhood. If $P_g = (p_1, p_2, \dots, p_D)$ is the best position among all the particles in the swarm (also known as $gBest$), such a version is called the global version. If P_g is taken from some smaller number of adjacent particles of the population (also known as $lBest$), such a version is called the local version. In (1) and (2) k represents the iterative number and D is dimension of particles, and range of particles are all determined by the problem to be optimized. Variables c_1, c_2 are learning factors, usually $c_1 = c_2 = 2$, which represent the weighting of the stochastic acceleration terms that pull each particle towards $pBest$ and $gBest$ positions. Thus, adjustment of these constants changes the amount of “tension” in the system. $r_1 \sim U(0, 1)$, $r_2 \sim U(0, 1)$, and w are an inertia weight, which is initialized typically in the range of $[0, 1]$. Inertia weight controls the impact of previous historical values of particle velocity on its current one. A larger inertia weight pressures towards global exploration (searching new area), while a smaller inertia weight pressures toward fine-tuning the current search area. Particle’s velocities on each dimension are confined to a maximum velocity $V_{\max} \sim x$ which is a parameter specified by the user. If the iteration formulas would cause the velocity on that dimension to exceed $V_{\max} \sim x$, then the velocity on that dimension is limited to $V_{\max}$. The termination criterion for the iterations is determined according to whether the max generation or a designated value of the fitness of P_g is reached.

The main disadvantage of the above original PSO is that it is difficult to keep the diversity of population and to balance local and global search and hence it may result in local optimal solutions. Besides, their search rates are commonly lower and sometimes need more computation when solving some difficult optimization problems. The original PSO frequently gets into the local solution, especially when the problem size is middle or large. In the following a local and global search combine particle swarm optimization algorithm is proposed to deal with the above disadvantages of original PSO in solving optimization problems.

3. A Local and Global Search Combined Particle Swarm Optimization Algorithm

3.1. The Model of a Local and Global Search Combined Particle Swarm Optimization Algorithm. Because both local and global version have their own advantages and disadvantage, respectively, one can use them both in the algorithm, while global version is used to get quick result and local version can refine the search space. In original PSO algorithm, the information of individual best and global best was shared by next generation particles, in which it tends to get stuck in a near optimal solution especially for middle and large size problems. A local and global combine particle swarm optimization (LGSCPSO) algorithm is presented in this paper, in which the particles of next generation share the best information of one particle itself, the best of particles in neighborhood population (local best particles), and the best of all particles in the swarm (global best particles). Among the neighborhood population, the particles are constituted by the best particles of every generation with different proportion,

which is called $lBest$. As the generation increases, neighborhood population diversity will grow larger and larger. LGSCPSO algorithm improves upon the original PSO variation to increase accuracy of solution without sacrificing the speed of solution significantly, and its detailed information will be given in following.

Suppose that the searching space is D -dimensional and m particles form the colony. The i th particle represents a D -dimensional vector $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})$ ($i = 1, 2, \dots, m$). It means that the i th particle locates at X_i in the searching space and the position of each particle is a potential solution. We could calculate the particle’s fitness by putting its position into a designated objective function. When the fitness is lower, the corresponding X_i is “better.” The i th particle’s “flying” velocity is also a D -dimensional vector, denoted as $V_i = (v_{i1}, v_{i2}, \dots, v_{iD})$. Denote the best position of the i th particle as $P_i = (p_{i1}, p_{i2}, \dots, p_{iD})$, the best position of the local as $P_l = (p_{l1}, p_{l2}, \dots, p_{lD})$, and the best position of the global as $P_g = (p_1, p_2, \dots, p_D)$, respectively. After finding the three best values, the particle of LGSCPSO updates its velocity and positions with the following formulas:

$$v_{id}(k+1) = wv_{id}(k) + c_1r_1(\alpha * (p_{id}(k) - x_{id}(k)) + (1 - \alpha) * (p_{ld}(k) - x_{id}(k))) + c_2r_2(p_g(k) - x_{id}(k)), \quad (3)$$

$$x_{id}(k+1) = x_{id}(k) + v_{id}(k+1), \quad (4)$$

where w is an inertia weight, which is initialized typically in the range of $[0, 1]$. The neighborhood particles of k generation are composed by the best neighborhood particle (local the particle) of k generation. $\alpha \in [0, 1]$ is weight index that is chosen according to different optimization problem, which reflects relatively important degree of the best position of the i th particle and the best position of the k th generation neighborhood particle. The variables c_1, c_2 are acceleration constant, which control how far a particle will move in a single iteration, and others parameters are same as in (1) and (2).

The searching is a repeat process, and the stop criteria are that the maximum iteration number is reached or the minimum error condition is satisfied. The stop condition depends on the problem to be optimized. In LGSCPSO algorithm, each particle of the swarm shares mutual information globally and benefits from discoveries and previous experiences of all other colleagues during the search process.

3.2. Pseudocode of LGSCPSO (See Pseudocode 1). Because PSO has very deep intelligent background, it is suitable for science computation and engineering applications. A thorough mathematical foundation for the methodology was not developed so far with the step of the algorithm; however, it has been proven to be very effective for application.

3.3. Convergence Analysis of LGSCPSO. In this section the LGSCPSO algorithm’s convergence is studied.

Step 1. Let parameters; including swarm size PS , maximum of generation $endgen$ and others parameters will be used in LGSCPSO algorithm.

Step 2. Iteration process

- (i) Generate stochastically initialization population and velocity;
- (ii) Evaluate each particle's fitness;
- (iii) Initialize $gBest$ position with the lowest fitness particle in the whole swarm;
- (iv) Initialize $pBest$ position with a copy of particle itself;
- (v) Initialize $lBest$ position with the best particle of initializing population;
- (vi) $k := 0$;

While (the maximum $endgen$ of generation is not met)

{

- (i) $k := k + 1$;
- (ii) Generate next swarm by (3), (4);
- (iii) Evaluate swarm;
 - { (a) Compute each particle's fitness in the swarm;
 - (b) Find new P_g , P_l of the swarm and P_i of each particle by comparison, and update P_i and P_g ;
 - (c) Update P_l using the best particle of k generation colony;}

}

Step 3. Output optimization results.

PSEUDOCODE 1

Theorem 1. In LGSCPSO algorithm in which its recurrence equation is (3) and (4), when parameters relation $c_1 r_1 + c_2 r_2 \leq 1 + w - 2\sqrt{w}$ or $1 + w + 2\sqrt{w} \leq c_1 r_1 + c_2 r_2 < 2w + 2$ is satisfied, it is convergent.

Proof. Let $R_1 = c_1 r_1$ and $R_2 = c_2 r_2$; from (3) and (4) we have

$$\begin{aligned} x_{id}(k+1) = & (1 - R_1 - R_2) x_{id}(k) + w v_{id}(k) \\ & + R_1 \alpha p_{id}(k) + R_1 (1 - \alpha) p_{ld}(k) \\ & + R_2 p_{gd}(k). \end{aligned} \quad (5)$$

From (4) we can get $v_{id}(k) = x_{id}(k) - x_{id}(k-1)$, and combining (5) we have

$$\begin{aligned} x_{id}(k+1) - (1 + w - R_1 - R_2) x_{id}(k) + w x_{id}(k-1) \\ = R_1 \alpha p_{id}(k) + R_1 (1 - \alpha) p_{ld}(k) + R_2 p_{gd}(k). \end{aligned} \quad (6)$$

LGSCPSO algorithm's recurrence equation is linearity constant coefficient nonhomogeneous, and the secular equation of corresponding homogeneous recurrence equation is as follows:

$$x^2 - (1 + w - R_1 - R_2)x + w = 0. \quad (7)$$

Latent roots of above secular equation are as follows:

$$\begin{aligned} \mu_1 = & \frac{1 + w - R_1 - R_2 + \sqrt{(1 + w - R_1 - R_2)^2 - 4w}}{2}, \\ \mu_2 = & \frac{1 + w - R_1 - R_2 - \sqrt{(1 + w - R_1 - R_2)^2 - 4w}}{2}. \end{aligned} \quad (8)$$

According to the relations of recurrence equation and its special solution, let the special solution of LGSCPSO

algorithm's recurrence equation (6) $H^*(k) = S$ substitute into (6), and we can solve

$$S = \frac{R_1 \alpha p_{id}(k) + R_1 (1 - \alpha) p_{ld}(k) + R_2 p_{gd}(k)}{R_1 + R_2}. \quad (9)$$

According to the relations of recurrence equation's general solution, special solution, and its latent roots, we can obtain the general solution of LGSCPSO algorithm's recurrence equation (6) as follows:

$$x(k) = S + c_1'' \mu_1^k + c_2'' \mu_2^k. \quad (10)$$

The initial solution and velocity of LGSCPSO algorithm recurrence equation is stochastically generated; suppose first and second solution is $x_{id}(1)$ and $x_{id}(2)$, respectively, and according to (10), we can obtain that

$$\begin{aligned} c_1'' = & \frac{\mu_2 (S - x(1)) + x(2) - S}{\mu_1 (\mu_1 - \mu_2)}, \\ c_2'' = & \frac{\mu_1 (x(1) - S) - x(2) + S}{\mu_2 (\mu_1 - \mu_2)}. \end{aligned} \quad (11)$$

To use (11) substitution into (10) simplification, we can obtain the general solution of LGSCPSO recurrence equation (6) as follows:

$$\begin{aligned} x(k) = & S + \frac{\mu_2 (S - x(1)) + x(2) - S}{\mu_1 - \mu_2} * \mu_1^{k-1} \\ & + \frac{\mu_1 (x(1) - S) - x(2) + S}{\mu_1 - \mu_2} * \mu_2^{k-1}. \end{aligned} \quad (12)$$

If the limit of above $x(k)$ exists when $k \rightarrow \infty$, the LGSCPSO algorithm is convergent.

Let

$$f(x) = x^2 - (1 + w - R_1 - R_2)x + w. \quad (13)$$

So the Δ of above function is as follows:

$$\Delta = (1 + w - R_1 - R_2)^2 - 4w. \quad (14)$$

Evidently, when

$$\begin{aligned} f(1) &> 0, \\ f(-1) &> 0, \\ \Delta &\geq 0 \end{aligned} \quad (15)$$

hold, we can obtain the function (13) latent roots $-1 < \mu_1, \mu_2 < 1$.

Namely, when $0 < c_1 r_1 + c_2 r_2 \leq 1 + w - 2\sqrt{w}$ or $1 + w + 2\sqrt{w} \leq c_1 r_1 + c_2 r_2 < 2w + 2$ is satisfy, we can obtain $-1 < \mu_1, \mu_2 < 1$.

From LGSCPSO algorithm theory, we obviously get that $p_{id}(k)$ and $p_{gd}(k)$ are descending with k increase, and S is also descending with k increase.

When $0 < c_1 r_1 + c_2 r_2 \leq 1 + w - 2\sqrt{w}$ or $1 + w + 2\sqrt{w} \leq c_1 r_1 + c_2 r_2 < 2w + 2$ is satisfied, namely, $-1 < \mu_1, \mu_2 < 1$ hold, we have

$$\begin{aligned} \lim_{k \rightarrow \infty} x(k) &= \lim_{k \rightarrow \infty} \left(S + \frac{\mu_2 (S - x(1)) + x(2) - S}{\mu_1 - \mu_2} * \mu_1^{k-1} \right. \\ &\quad \left. + \frac{\mu_1 (x(1) - S) - x(2) + S}{\mu_1 - \mu_2} \mu_2^{k-1} \right) \\ &= \lim_{k \rightarrow \infty} \frac{R_1 \alpha p_{id}(k) + R_1 (1 - \alpha) p_{ld}(k) + R_2 p_{gd}(k)}{R_1 + R_2}. \end{aligned} \quad (16)$$

Because $(R_1 \alpha p_{id}(k) + R_1 (1 - \alpha) p_{ld}(k) + R_2 p_{gd}(k)) / (R_1 + R_2)$ is monotone decreasing and have lower bound which is its optimum solution, $\lim_{k \rightarrow \infty} (R_1 \alpha p_{id}(k) + R_1 (1 - \alpha) p_{ld}(k) + R_2 p_{gd}(k)) / (R_1 + R_2)$ exist, namely, the endless particle sequence brought about by LGSCPSO algorithm's recurrence equation is limited. So we can obtain that the LGSCPSO algorithm is convergent when its parameters relations $0 < c_1 r_1 + c_2 r_2 \leq 1 + w - 2\sqrt{w}$ or $1 + w + 2\sqrt{w} \leq c_1 r_1 + c_2 r_2 < 2w + 2$ is satisfied. $\square$

4. Numerical Simulation

4.1. Test Functions. To illustrate the effectiveness and performance of LGSCPSO algorithm for optimization problems, a set of 8 representative benchmark functions with different dimensions were employed to evaluate it in comparison with

original PSO. Many authors tested algorithm using them widely.

Sphere Model. Consider $f_1(x) = \sum_{i=1}^n x_i^2$, where $x_i \in [-100, 100]$.

Schwefel's Problem 1.2. Consider $f_2(x) = \sum_{i=1}^n (\sum_{j=1}^i x_j)^2$, where $x_i \in [-100, 100]$.

Schwefel's Problem 2.21. Consider $f_3(x) = \max_i \{|x_i|, 1 \leq i \leq n\}$, where $x_i \in [-100, 100]$.

Generalized Rosenbrock's Function. Consider $f_4(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$, where $x_i \in [-100, 100]$.

Ackley's Function. Consider $f_5(x) = -20 \exp(-0.2 \sqrt{(1/n) \sum_{i=1}^n x_i^2}) - \exp((1/n) \sum_{i=1}^n \cos 2\pi x_i) + 20 + e$, where $x_i \in [-32, 32]$.

Generalized Griewank Function. Consider $f_6(x) = (1/4000) \sum_{i=1}^n (x_i - 100)^2 - \prod_{i=1}^n \cos((x_i - 100)/\sqrt{i}) + 1$, where $x_i \in [-600, 600]$.

Generalized Penalized Functions. Consider

$$\begin{aligned} f_7(x) &= \frac{\pi}{n} \left\{ 10 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 \right. \\ &\quad \left. \times [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\} \\ &\quad + \sum_{i=1}^n u(x_i, 10, 100, 4), \\ f_8(x) &= 0.1 \left\{ \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 \right. \\ &\quad \left. \times [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\} \\ &\quad + \sum_{i=1}^n u(x_i, 10, 100, 4), \end{aligned} \quad (17)$$

where

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \leq x_i \leq a \\ k(-x_i - a)^m, & x_i < -a \end{cases} \quad (18)$$

$$y_i = 1 + \left(\frac{1}{4}\right)(x_i + 1), \quad x_i \in [-50, 50].$$

They can be divided into unimodal (function f_1 - f_3) and multimodal functions (function f_4 - f_8), where the number of local minima increases exponentially with the problem dimension.

4.2. Experimental Results and Comparison. The experimental results of original PSO and LGSCPSO algorithm on

TABLE 1: The comparison results of the PSO algorithm and the LGSPSO algorithm.

Min/Max/Average ($e \sim n$ represents $\times 10^6$)					
a	LGC	0.1	0.2	0.3	0.4
w					
F_{un}					0.5
f_1	1	79247/798.136/6/119.2735	4.8684/788.0086/137.7727	3.4298/466.135/129.4039	6.2134/108.3578/34.9678
Dim	2	1.0216/328.5215/36.3621	0.2372/39.35/15.2925	0.2165/36.7093/9.1546	0.336/297.5513/35.9433
150,	3	0.1788/3.6762/0.9580	0.0516/72.579/18.8086	0.0446/0.9215/0.3245	0.0931/2.9157/0.6956
$Best$	4	0.0276/1.0302/0.1732	0.0024/16.9442/0.2085	0.0081/5.1882/0.6059	0.2738/12.3747/2.6693
0;	5	0.0015/0.2004/0.0345	0.0011/6.0067/0.7585	0.0021/15.3744/3.103	0.0354/0.9538/0.2124
PS	6	6.199e - 4/20.4397/2.1839	0.1109/326.9967/38.2332	9.091e - 4/50.1812/9.9065	0.0088/0.3742/0.0659
150,	7	8.7194e - 4/32.076/3.5118	217.0564/2.5137e + 3/922.9063	0.1802/416.0314/88.5176	1.5877e - 4/2.5125/0.3145
EG	8	0.0027/1.0443/0.3794	131.8696/1.0093e + 3/605.7608	22.395/2.7571e + 3/760.49	4.9801e - 5/0.0052/8.812e - 4*
2500	9	0.0301/3.1973e + 5/5.7818e + 4	3.5023/3.5197e + 5/75222e + 4	3.3725/3.5151e + 5/3.3331e + 4	9.7207e - 5/390.2679/40.2488
w					
PSO		3.5545e + 5/4.2006e + 5/3.9996e + 5	3.6563e + 5/4.1904e + 5/3.9654e + 5	3.8295e + 5/4.1465e + 5/4.0243e + 5	3.834e + 5/4.2439e + 5/4.0552e + 5
LGC	0.1		0.2	0.3	0.4
F_{un}					0.5
f_2	1	1.5314e - 5/0.0012/2.3471e - 4	3.4819e - 8/9.7461e - 7/2.8081e - 7	1.6392e - 9/8.0260e - 8/2.4483e - 8	1.9438e - 8/4.5893e - 6/1.0199e - 6
Dim	2	7.544e - 6/4.1057e - 4/1.2064e - 4	3.6274e - 9/2.252e - 7/6.7068e - 8	4.3649e - 11/1.9808e - 8/3.0528e - 9	6.9751e - 9/4.4344e - 6/5.5416e - 7
30,	3	3.3468e - 6/2.5205e - 4/8.7442e - 5	1.7849e - 10/6.8322e - 8/1.3459e - 8	4.3800e - 11/8.3452e - 9/2.5903e - 9	3.0993e - 10/2.669e - 8/6.5853e - 5
$Best$	4	2.2108e - 5/3.3094e - 4/9.2148e - 5	9.6468e - 11/1.3930e - 7/2.0447e - 8	2.9752e - 12/5.6342e - 10/1.6065e - 10*	1.1121e - 6/2.0099e - 5/6.5348e - 6
0;	5	1.4607e - 5/3.3855e - 4/9.0081e - 5	7.8029e - 11/9.93e - 9/4.5009e - 9	7.1437e - 12/4.1282e - 9/8.6320e - 10	1.5491e - 6/1.6945e - 5/6.7154e - 6
PS	6	1.4749e - 4/0.0097/0.0016	3.0876e - 9/4.2278e - 7/5.6384e - 8	2.0796e - 11/5.0013e - 9/1.0295e - 9	4.0553e - 10/3.5928e - 8/5.5263e - 9
150,	7	2.2105e - 4/0.0705/0.0232	1.8076e - 7/1.0146e - 5/2.1222e - 6	1.1123e - 10/2.6655e - 8/8.6717e - 9	6.2429e - 9/6.853e - 7/1.8691e - 7
EG	8	0.0879/0.9873/0.4732	1.437e - 6/6.4389e - 4/2.8104e - 4	8.3474e - 8/7.0419e - 6/2.2156e - 6	9.0305e - 7/1.0765e - 4/1.9495e - 5
2000	9	1.3288/157.24/5.0923	0.0042/0.1077/0.0241	7.8655e - 5/0.0055/0.002	2.6371e - 4/0.0057/0.0024
w					
PSO		364.253/2.4939e + 4/1.0134e + 4	1.3765/4.6947e + 3/481.8464	0.0691/2.5876/0.7283	17.3828/984.8965/188.1513
LGC	0.1		0.2	0.3	0.4
F_{un}					0.5
f_3	1	2.3621e - 4/0.0013/7.311e - 4	1.3902e - 5/2.0080e - 4/8.8198e - 5	1.6884e - 5/3.3877e - 4/1.194e - 4	0.001/0.0262/0.0055
Dim	2	6.0977e - 5/4.2674e - 4/2.1957e - 4	1.1701e - 6/2.8489e - 5/8.0878e - 6	7.4212e - 7/4.1010e - 6/1.8869e - 6	1.1206e - 5/5.1122e - 4/1.9353e - 4
30,	3	5.5341e - 5/7.0551e - 4/2.1313e - 4	3.8603e - 7/8.6707e - 6/1.7948e - 6	5.1647e - 8/1.5398e - 6/3.7869e - 7	3.9851e - 8/2.4747e - 6/7.2311e - 7
$Best$	4	4.9251e - 5/3.4147e - 4/1.4638e - 4	1.4e - 7/5.6791e - 6/1.3055e - 6	1.5353e - 9/1.8772e - 7/6.9333e - 8*	3.4271e - 8/2.4.2062e - 7/1.0447e - 7
0;	5	5.2844e - 5/0.0019/4.5952e - 4	1.6044e - 7/1.6259e - 6/5.7405e - 7	4.7963e - 9/3.1684e - 7/6.3918e - 8	6.6274e - 9/8.7745e - 7/1.04709e - 7
PS	6	4.8841e - 4/0.0053/0.0029	1.7168e - 7/6.7027e - 6/2.6332e - 6	2.0730e - 9/6.1577e - 7/9.0325e - 8	2.0358e - 9/4.6960e - 8/1.9028e - 8
150,	7	0.0046/0.0476/0.0191	9.6091e - 6/7.5094e - 5/2.8845e - 5	3.4465e - 8/7.9999e - 6/1.0596e - 6	3.9048e - 8/3.9274e - 7/1.5350e - 7
EG	8	0.0569/0.5819/0.2266	8.2486e - 5/0.0012/3.9755e - 4	1.1441e - 6/1.3502e - 5/5.7652e - 6	7.5274e - 7/1.2135e - 5/3.6172e - 6
2000	9	0.4589/4.2785/1.3995	0.001/0.0436/0.0158	2.4963e - 5/0.0027/6.1308e - 4	0.0016/0.0311/0.0073
w					
PSO		0.9442/7.2642/3.7178	0.1348/0.5193/0.2436	0.005/0.0675/0.0212	0.2007/0.9216/0.4999

TABLE I: Continued.

a	Min./Max./Average ($e \sim n$ represents $\times 10^6$)				
	LGC	0.1	0.2	0.3	w
<i>Fun</i>	LGC	0.1	0.2	0.3	0.4
<i>f₄</i>	1	1.8571/1.229e + 3/135.8807	6.2873/3.2863e + 3/709.8222	0.2013/285.8471/44.28	0.5944/144.3777/23.7188
<i>Dim</i>	2	0.1568/2.5441e + 3/388.2091	0.0113/789.7377/115.475	4.0693/4.9375e + 3/562.5381	1.8021/937.5994/153.7682
25,	3	0.6066/799.8443/169.6422	2.3034/796.4533/123.7322	3.3983e - 4/302.2135/32.9758	0.0455/207.5395/40.3132
<i>Best</i>	4	0.2485/181.0596/25.4415	9.6966e - 5/68529/2.0791*	0.2062/179.2151/21.4179	3.4459/1962.925/115.1339
0;	5	0.0237 /40.8918/11.1441	0.0069/23.873/4.9974	3.5267e - 4/3.8396/0.8494	0.0125 /423.4865/75.3354
PS	6	0.0253/276.2887/57.0108	2.547e - 4/9.0536/4.042	3.6341e - 4/6.6633/3.6567	0.0129/26.4581/10.2862
150,	7	0.0916/228.4057/30.7965	0.0702/14.2596/6.6697	0.0372/152.4195/22.0972	0.1671/672.3137/106.7018
EG	8	0.4745/435.3273/56.9634	0.089/16.2452/9.9379	0.1941/238.9312/38.1225	0.6944/213.5931/30.5417
2500	9	4.3957/341.7015/679295	3.2415/838.6487/128.4819	0.7361/334.4369/49.1472	0.2055/11812e + 3/161.6274
<i>Fun</i>	PSO	4.3221e + 4/5.1189e + 4/4.6655e + 4	3.9971e + 4/5.4303e + 4/4.6593e + 4	2.876e + 4/5.5191e + 4/4.7462e + 4	3.937e + 4/5.2589e + 4/4.6619e + 4
<i>f₅</i>	LGC	0.1	0.2	0.3	0.4
<i>Dim</i>	1	7.9936e - 15/4.3521e - 14/2.256e - 14	7.9936e - 15/7.9936e - 15/7.9936e - 15	4.4409e - 15/7.9936e - 15/7.2831e - 15	7.9936e - 15/1.5099e - 14/9.4147e - 15
30	2	7.9936e - 15/7.9936e - 15/7.9936e - 15	4.4409e - 15/7.9936e - 15/5.1514e - 15	4.4409e - 15/7.9936e - 15/5.1514e - 15	4.4409e - 15/7.9936e - 15/7.2831e - 15
<i>Best</i>	3	4.4409e - 15/7.9936e - 15/7.2831e - 15	4.4409e - 15/7.9936e - 15/5.5067e - 15*	4.4409e - 15/1.5017/0.1502	4.4409e - 15/7.9936e - 15/5.5067e - 15*
0;	4	4.4409e - 15/7.9936e - 15/5.1514e - 15	4.4409e - 15/1.6462/0.2801	4.4409e - 15/2.019/0.3167	4.4409e - 15/2.5795/0.3735
PS	5	4.4409e - 15/7.9936e - 15/5.8620e - 15	4.4409e - 15/2.3162/0.5967	4.4409e - 15/1.6462/0.7128	4.4409e - 15/2.0133/0.5
100,	6	4.4409e - 15/2.3162/0.2316	4.4409e - 15/2.3162/1.0934	4.4409e - 15/3.6807/1.3585	4.4409e - 15/1.3404/0.2272
EG	7	4.4409e - 15/1.1908e - 10	7.9936e - 15/2.887/1.1089	4.4409e - 15/2.4949/1.2564	4.4409e - 15/1.3404/0.134
2000	8	6.9209e - 5/2.2201/0.7328	0.0021/3.0268/1.7571	7.9936e - 15/2.0133/1.2372	4.4409e - 15/2.0133/0.5
	9	4.8998e - 5/1.6462/0.3991	0.0043/2.579/1.7115	7.9936e - 15/2.6602/0.8856	4.4409e - 15/1.1551/0.1155
<i>Fun</i>	PSO	2.4655e - 5/2.2315/1.1133	1.6387e - 5/2.8859/0.7401	7.9936e - 15/2.4959/0.6931	4.4409e - 15/7.9936e - 15/7.2831e - 15
<i>f₆</i>	LGC	0.1	0.2	0.3	0.4
<i>Dim</i>	1	0.9416/1.4223/1.1635	0.5782/1.452/0.9452	0.3887/8.8918/1.7237	0.9863/3.3689/1.3278
150,	2	0.2941/0.9446/0.4842	0.0804/2.7225/0.5028	0.0397/2.1197/0.4602	0.3404/0.6563/0.4877
<i>Best</i>	3	0.0383/0.2148/0.0994	0.0036/0.8091/0.1571	0.0055/4.4603/0.5501	0.0084/0.2726/0.1332
0;	4	0.002/0.337/0.0945	6.671e - 4/1.0183/0.1775	8.199e - 4/0.4379/0.122	0.0013/0.2891/0.1343
PS	5	4.6926e - 4/0.2611/0.0958	8.1220e - 5/2.1079/0.4195	2.5102e - 4/0.5375/0.3158	0.0044/0.9635/0.1852
150,	6	3.7473e - 5/0.1647/0.0541	0.2266/6.0141/2.1203	0.0023/3.2548/0.7676	3.0448e - 4/1.2816/0.2687
EG	7	1.572e - 4/0.0664/0.0194	1.6879/24.077/11.2293	0.2953/4.2413/1.1668	0.0074/0.5053/0.1469
2500	8	1.1609e - 4/0.2322/0.0541	1.927/2170006/71286	1.3151/78887/9.7229	1.3352e - 5/0.061/0.0226
	9	0.0129/3.4498e + 3/639.0998	0.2582/3.7637e + 3/668.9854	0.0979/12.3986/3.8033	5.0879e - 6/1.1096/0.1934*
		3.1058e + 3/3.9496e + 3/3.6383e + 3	2.766e + 3/3.9524e + 3/3.5857e + 3	0.9187/3.9622e + 3/2.9803e + 3	0.916/3.9592e + 3/2.8647e + 3
					3.5377e + 3/3.9083e + 3/3.7889e + 3

Min./Max./Average ($e \sim n$ represents $\times 10^n$)

Remark : In Table 1, F_{un} and Dim denote function and its dimension, respectively; the *Best* is this function's optimum value; *PS* and *EG* indicate the population size and algorithm terminate generation. *PSO* and *LGCG* denote the original *PSO* and *LGSCPSO* algorithm. The better solutions and the parameters found in *LGSCPSO* are illustrated with bold letters, and the best solutions are shown in data with asterisk.

FIGURE 1: Continued.

FIGURE 1: Convergence figure of LGSCPSO comparing with original OPSO for f_1-f_8 .

each test function are listed in Table 1. To get the average performance of the LGSCPSO algorithm ten runs on each problem instance were performed and the solution quality was averaged. The comparisons results of LGSCPSO and Original PSO algorithm are shown in Table 1 of the appendix.

From Table 1 one can observe that $w \in [0.2, 0.4]$ have the highest performance since using them has smaller minimum and smaller arithmetic mean in relation to the solutions obtained by the others, especially $w \approx 0.4$ has the better search efficiency. In LGSCPSO algorithm, various parameter α should be selected for optimizing different problems, as a whole, it is better when $\alpha \in [0.3, 0.9]$, especially $\alpha \approx 0.4$. From simulation results we can obtain that the LGSCPSO algorithm is clearly better than the original PSO for continuous nonlinear function optimization problem. The convergence figures of most effective LGSCPSO comparing with original PSO for 8 instances are as follows.

From Figure 1 we can discover that the convergence rate of LGSCPSO algorithm is clearly faster than the OPSO on every benchmark function, especially it is more efficacious than OPSO for middle and large size optimization problem. Accordingly, we can state that the LGSCPSO algorithm is more effective than OPSO algorithm. For test function 5, the search capability of LGSCPSO algorithm is worse than that of OPSO at the beginning of the iteration. However, seen from the enlarged figure, the search capability of LGSCPSO is markedly enhanced along with the iteration increasing. In addition we can find that the simulation data from test function 5 has a strong advantage.

5. Conclusions and Perspectives

According to the shortcoming of OPSO algorithm that it frequently gets the local solution especially solving the middle

or large size problem, a LGSCPSO algorithm is presented, and simulations show that it is efficacious. The performance of the new approach is evaluated in comparison to OPSO algorithm for eight representative instances with different dimensions and obtained results show that LGSCPSO algorithm is efficacious for solving optimization problems.

The proposed LGSCPSO algorithm approach in this paper can be considered as effective mechanisms from this point of view. There are a number of research directions that can be considered as useful extensions of this research. Although the proposed algorithm is tested with eight representative instances, a more comprehensive computational study should be made to test the efficiency of proposed solution technique. In the future it is maybe used for solving other discrete combinatorial optimization problems such as traveling salesman problem and scheduling.

Appendix

See Table 1.

Conflict of Interests

The authors declare that they have no conflict of interests to this paper.

Acknowledgments

The authors are grateful to the anonymous reviewers for giving us helpful suggestions. This work is supported by National Natural Science Foundation of China (Grant nos. 61174040 and 61104178), Fundamental Research Funds for the Central Universities, Shanghai Commission of Science and Technology (Grant no. 12JC1403400), and Shanghai Dianji

University Computer Application Technology Disciplines (Grant no. 13XKJ01).

References

- [1] R. Eberhart and J. Kennedy, "New optimizer using particle swarm theory," in *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, pp. 39–43, Nagoya, Japan, October 1995.
- [2] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, December 1995.
- [3] Y. Shi and R. C. Eberhart, "A modified particle swarms optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 303–308, 1997.
- [4] Y. Shi and R. Eberhart, "Modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, May 1998.
- [5] H. Zhang, H. Yan, F. Yang, and Q. Chen, "Quantized control design for impulsive fuzzy networked systems," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 6, pp. 1153–1162, 2011.
- [6] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, pp. 1671–1676, IEEE Press, 2002.
- [7] P. J. Angeline, "Using selection to improve particle swarm optimization," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 84–89, Anchorage, Alaska, May 1998.
- [8] M. Lovbjerg, T. K. Rasmussen, and T. Krink, "Hybrid particle swarm optimization with breeding and subpopulations," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 1217–1222, San Diego, Calif, USA, 2000.
- [9] B. Jiao, Z. Lian, and X. Gu, "A dynamic inertia weight particle swarm optimization algorithm," *Chaos, Solitons and Fractals*, vol. 37, no. 3, pp. 698–705, 2008.
- [10] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.
- [11] H. Zhang, H. Yan, F. Yang, and Q. Chen, "Distributed average filtering for sensor networks with sensor saturation," *IET Control Theory & Applications*, vol. 7, no. 6, pp. 887–893, 2013.
- [12] R. C. Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," in *Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 81–86, May 2001.
- [13] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *Evolutionary Programming VII: Proceedings of the 7th Annual Conference on Evolutionary Programming*, pp. 591–600, Springer, New York, NY, USA, 1998.
- [14] F. van den Bergh and A. P. Engelbecht, "Cooperative learning in neural networks using particle swarm optimizers," *South African Computer Journal*, vol. 26, pp. 84–90, 2000.
- [15] H. C. Yan, Z. Z. Su, H. Zhang, and F. W. Yang, "Observer-based H_∞ control for discrete-time stochastic systems with quantisation and random communication delays," *IET Control Theory & Applications*, vol. 7, no. 3, pp. 372–379, 2013.
- [16] M. A. Abido, "Optimal power flow using particle swarm optimization," *International Journal of Electrical Power and Energy Systems*, vol. 24, no. 7, pp. 563–571, 2002.
- [17] A. Salman, I. Ahmad, and S. Al-Madani, "Particle swarm optimization for task assignment problem," *Microprocessors and Microsystems*, vol. 26, no. 8, pp. 363–371, 2002.
- [18] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, "A Particle swarm optimization for reactive power and voltage control considering voltage security assessment," *IEEE Transactions on Power Systems*, vol. 15, no. 4, pp. 1232–1239, 2000.
- [19] M. M. El-Sherbiny and R. M. Alhamali, "A hybrid particle swarm algorithm with artificial immune learning for solving the fixed charge transportation problem," *Computers & Industrial Engineering*, vol. 64, pp. 610–620, 2013.
- [20] H. Zhang, H. Yan, T. Liu, and Q. Chen, "Fuzzy controller design for nonlinear impulsive fuzzy systems with time delay," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 5, pp. 844–856, 2011.
- [21] N. Hamta, S. M. T. Fatemi Ghomi, F. Jolai, and M. Akbarpour Shirazi, "A hybrid PSO algorithm for a multi-objective assembly line balancing problem with flexible operation times, sequence-dependent setup times and learning effect," *International Journal of Production Economics*, vol. 141, no. 1, pp. 99–111, 2012.
- [22] F. D. Chou, "Particle swarm optimization with cocktail decoding method for hybrid flow shop scheduling problems with multiprocessor tasks," *International Journal of Production Economics*, vol. 141, pp. 137–145, 2013.