Hindawi Publishing Corporation
Journal of Optimization
Volume 2013, Article ID 356420, 8 pages
http://dx.doi.org/10.1155/2013/356420

Research Article

Double Flight-Modes Particle Swarm Optimization

Wang Yong,[1] Li Jing-yang,[1] and Li Chun-lei[2]

[1] *College of Information Science and Engineering, Guangxi University for Nationalities, Nanning 530006, China*
[2] *Nanning Power Supply Bureaus, Nanning, Guangxi 530031, China*

Correspondence should be addressed to Wang Yong; wangygxnn@sina.com

Received 20 February 2013; Revised 29 September 2013; Accepted 30 September 2013

Academic Editor: Jein-Shan Chen

Getting inspiration from the real birds in flight, we propose a new particle swarm optimization algorithm that we call the double flight modes particle swarm optimization (DMPSO) in this paper. In the DMPSO, each bird (particle) can use both rotational flight mode and nonrotational flight mode to fly, while it is searching for food in its search space. There is a King in the swarm of birds, and the King controls each bird's flight behavior in accordance with certain rules all the time. Experiments were conducted on benchmark functions such as Schwefel, Rastrigin, Ackley, Step, Griewank, and Sphere. The experimental results show that the DMPSO not only has marked advantage of global convergence property but also can effectively avoid the premature convergence problem and has good performance in solving the complex and high-dimensional optimization problems.

1. Introduction

Particle swarm optimization (PSO) was developed by Kennedy and Eberhart in 1995 [1], based on the swarm behavior of birds in searching for food. Since then, PSO has got more and more attention from the researchers in the domain of information and has generated much wider interests, because of its simplicity of implementation, and less domain knowledge required. However, the original PSO still has the phenomenon of the premature convergence problem, which exists in most of the stochastic optimization algorithms. In order to improve the performance of the PSO, many scholars have proposed various approaches to improve the performance of the PSO such as listed in the paper [2–22]. The methods presented by the authors mentioned in the paper [2–22] can be summed up into two strategies. The first strategy is to add the group quantity of information through increasing the population size of swarm, in order to achieve the purpose of improving the performance of algorithm. However, this strategy cannot fundamentally overcome the premature convergence problem and will certainly lead to the increase in running time of computation. The second strategy is, under the condition of not increasing the population size of swarm, to excavate or to increase every particle's latent capacity to achieve the goal of improving the performance of algorithm. Although these approaches mentioned in the paper [2–22] can improve the performance of the PSO to some extent but cannot fundamentally solve the premature convergence problem which exists in the original PSO.

In this paper, we intend to present a new particle swarm optimization, namely, the double flight modes particle swarm optimization (DMPSO for short), based on the flight characteristics of birds. The rest of this paper is organized as follows. In Section 2, we briefly introduce the original PSO, the PSO-W, and the CLPSO. Section 3 describes the double flight-modes particle swarm optimization. In Section 4, we conduct our simulation experiments on some test functions for each algorithm and compare the performance of the DMPSO with that of the original PSO, with that of the PSO-W, and with that of the CLPSO. We give our conclusions in Section 5.

2. Particle Swarm Optimizers

2.1. The Original PSO. Particle swarm optimization (PSO) was developed by Kennedy and Eberhart [1]. PSO emulates the swarm behavior and the individuals are treated as points in the D-dimensional search space. Each individual is named as a "particle" which represents a potential solution to a problem. The position and the velocity of the ith particle are represented as $X_i(t) = (x_{i1}(t), \ldots, x_{iD}(t))$, and

$V_i(t) = (v_{i1}(t), \ldots, v_{iD}(t))$ respectively. The best previous position (the position yielding the best fitness value) of the ith particle is represented as $P_{\text{best}_i}(t) = (p_{i1}(t), \ldots, p_{iD}(t))$. The best position discovered by the whole population is represented as $G_{\text{best}}(t) = (g_{b1}(t), \ldots, g_{bD}(t))$. Then the vector $V_i(t+1)$ and the position $X_i(t+1)$ of the ith particle are updated according to the following equation [1]:

$$\begin{aligned} v_{ij}(t+1) &= v_{ij}(t) + c_1 r_1 \left(p_{ij}(t) - x_{ij}(t)\right) \\ &\quad + c_2 r_2 \left(g_j(t) - x_{ij}(t)\right), \quad j = 1, \ldots, D \qquad (1) \\ x_{ij}(t+1) &= x_{ij}(t) + v_{ij}(t+1), \quad j = 1, \ldots, D, \end{aligned}$$

where c_1 and c_2 are the acceleration coefficients reflecting the weighting of stochastic acceleration terms that pull each particle toward G_{best} and P_{best} positions, respectively, and r_1 and r_2 are two random numbers in the range $[0, 1]$.

2.2. Some Variants PSO. Since PSO was introduced by Kennedy and Eberhart [1], many researchers have worked on improving its performance in various ways and deriving many interesting variants. One of the variant PSOs [2] introduces a parameter called inertia weight w into the original PSO as follows:

$$\begin{aligned} v_{ij}(t+1) &= w v_{ij}(t) + c_1 r_1 \left(p_{ij}(t) - x_{ij}(t)\right) \\ &\quad + c_2 r_2 \left(g_j(t) - x_{ij}(t)\right), \qquad (2) \\ x_{ij}(t+1) &= x_{ij}(t) + v_{ij}(t+1), \quad j = 1, \ldots, D, \qquad (3) \end{aligned}$$

in which the inertia weight w plays the role of balancing the global and local search. A large inertia weight facilitates a global search, while a small inertia weight facilitates a local search. In (2), if its inertia weight w is a linearly decreasing weight over the course of search, then the variant PSO [2] is usually represented as PSO-W.

Another variant PSO [16], called the comprehensive learning particle swarm optimizer (CLPSO), presents a new learning strategy. In the CLPSO, the velocity updating equation is changed to

$$V_i^d \longleftarrow w * V_i^d + c^* \text{rand}_i^d * \left(p_{\text{best}\,f_i(d)}^{\,d} - X_i^d\right), \qquad (4)$$

in which $f_i = [f_i(1), \ldots, f_i(M)]$ defines which particles' p_{best}s the ith particle should follow. $p_{\text{best}\,f_i(d)}^{\,d}$ can be the corresponding dimension of any particle's p_{best} (including its own p_{best}), and the decision depends on probability P_c, referred to as the learning probability, which can take different values for different particles. We first generate a random number for each dimension of the ith particle. If this random numberis larger than P_{c_i}, then the corresponding dimension will learn from its own p_{best}; otherwise, it will learn from another particle's p_{best}.

3. The Double Flight Modes Particle Swarm Optimization

3.1. The Flight Characteristics of Birds. Through careful observation, we have found that (1) most of birds have superb flight-skills. They can use various flight modes, such as rotational flight mode and non-rotational flight mode, to fly in their search space, can avoid the attack of their natural enemies and various obstacles and can avoid themselves being immersed into blind alley and (2) there is usually a King of birds (a flight commander) in most of the swarms of birds; the King controls or directs every bird's flight mode and flying direction while the swarms of birds are searching for food in the search space. Therefore, we think if a bird uses only one flight mode to fly in its search space all the time, it will be unable to avoid the attack of its natural enemies and various obstacles and will easily be immersed into blind alley. If there is not a King of birds controlling the flying direction of the swarm, then the swarm will be fallen into being scattered and disunited. In most cases, if a bird has superb flight skills, it usually can find more food when it is foraging for food in its search space.

For the sake of simplicity, we use the following idealized rules.

(1) Each bird only uses rotational flight mode and non-rotational flight mode to fly while it is searching for food in its search space.

(2) There is a King of birds among a swarm of birds. The King controls or directs every bird's flight behavior in accordance with certain rules and directs each bird's flight mode and flying direction while a swarm of birds is searching for food in its search space.

(3) The flight speed of a bird has something to do with the distance between the bird and its flying destination. We can say to a certain degree that the farther the distance between the bird and its flying destination, the faster the speed flying to the destination.

If we idealize the flight characteristics of a swarm of birds according to the previous description, then we can redevelop a new particle swarm optimization inspired by the real birds in flight. In simulations, we use virtual birds (particles) naturally.

3.2. The Flight Modes of Birds. Let $X_i(t) = (x_{i1}(t), \ldots, x_{iD}(t))$ and $V_i(t) = (v_{i1}(t), \ldots, v_{iD}(t))$ be the position and the velocity of particle i, respectively, $P_{\text{best}_i}(t) = (p_{i1}(t), \ldots, p_{iD}(t))$ be the best previous position yielding the best fitness value for the ith particle, and $G_{\text{best}} = (g_{b1}(t), \ldots, g_{bD}(t))$ be the best position discovered by the whole population.

We first give the conceptions of rotational flight mode and non-rotational flight mode, respectively.

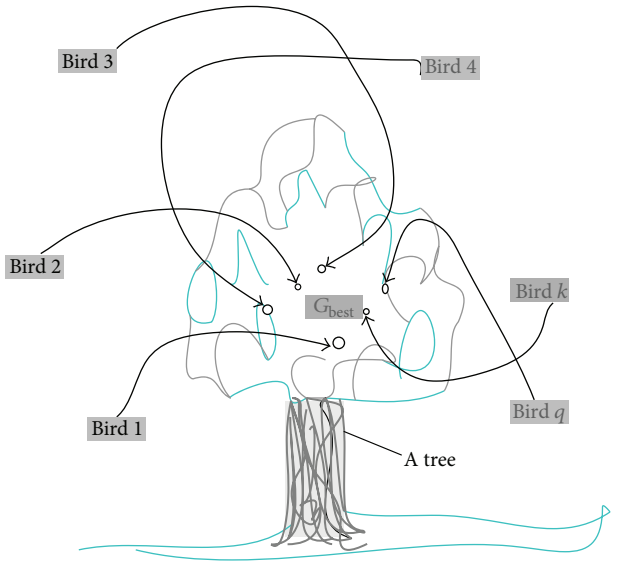

FIGURE 1: A diagrammatic sketch of a swarm of birds using *rotational flight mode* to fly to the G_{best}.

Definition 1. Let $X_i(t) = (x_{i1}(t), \ldots, x_{iD}(t))$ be the position of particle i at the time instant t and $G_{best}(t) = (g_1(t), \ldots, g_D(t))$ be the best position discovered by the whole population.

(1) We call that particle i uses *rotational flight mode* to fly to the position $G_{best}(t)$, if particle i flies to the position $G_{best}(t)$ according to the following equation:

$$x_{ij}(t+1) = g_k(t), \quad j = 1, \ldots, D, \tag{5}$$

where the number k is a random integer in the set $\{1, \ldots, D\}$.

We can use a diagrammatic sketch to depict that a group of birds is using *rotational flight mode* to fly to the G_{best} as in Figure 1.

We can foresee if a group of birds is using *rotational flight mode* to fly to the position $G_{best}(t)$ at the time instant t, then to some extent, the group will gather around the position $G_{best}(t)$ at the time instant $t+1$.

(2) We call that particle i uses *non-rotational flight mode* to fly to the position $G_{best}(t)$, if particle i flies to the position $G_{best}(t)$ according to the following equation:

$$\begin{aligned} v_{ij}(t+1) &= v_{ij}(t) + c_1 r_1 \left(p_{ij}(t) - x_{ij}(t)\right) \\ &\quad + c_2 r_2 \left(g_j(t) - x_{ij}(t)\right); \\ x_{ij}(t+1) &= x_{ij}(t) + w(t) v_{ij}(t+1), \quad j = 1, \ldots, D, \end{aligned} \tag{6}$$

where $w(t)$ is an increasing function about the variable $|g_j(t) - x_{ij}(t)|$ (the distance between the jth component of $X_i(t)$ and the jth component of $G_{best}(t)$), c_1 and c_2 are the acceleration coefficients, and both r_1 and r_2 are two uniformly distributed random numbers in the range $[0, 1]$.

In our simulations of this paper, we select $w(t) = \mu(1 - \exp(-\rho|g_j(t) - x_{ij}(t)|))$ as the increasing function in (6), where $\mu = 0.8$ and $\rho = 5$, $j = 1, \ldots, D$.

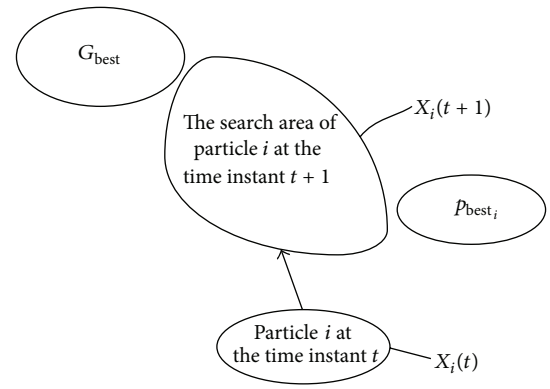

FIGURE 2: A diagrammatic sketch of particle i using *non-rotational flight mode* to fly to the G_{best}.

We can use a diagrammatic sketch to depict that particle i is using *non-rotational flight mode* to fly to the position G_{best} as in Figure 2.

3.3. The Flight-Control Approach of the King. Since the King of birds controls each bird's flight behavior in accordance with certain rules; therefore, we look at the King as a flight commander, and we think that every bird's flight behavior is controlled by the King. Following that, we will set up a flight-command rule for the King, and the King uses this rule to control each bird's flight behavior. We first give the conception of *flight command* as follows.

Definition 2. Let M be the population size of the swarm and $s_i(t)$ be the order of particle i in the swarm according to the ascending sort of the fitness value at the time instant t. Then the conception of *flight command* is defined as the following equation:

$$\mathrm{FC}_i(t) = \frac{s_i(t) - 1}{M - 1}, \tag{7}$$

in which, if $s_i(t) = 1$ ($\mathrm{FC}_i(t) = 0$ accordingly), then the fitness value of particle i will be the best one in the swarm at the time instant t. Meanwhile, if $s_i(t) = M$ ($\mathrm{FC}_i(t) = 1$ accordingly), then the fitness value of particle i will be the worst one in the swarm at the time instant t.

The King controls each particle's flight mode according to the following approach.

Step 1. The King first gives an instruction δ randomly, where δ is a normal random distribution in the range $[0, 1]$.

Step 2. Each particle chooses its flight mode according the following *rule*:

$$\begin{aligned} &\text{particle } i \text{ chooses the formula (6) to fly,} \quad \text{if } \mathrm{FC}_i > \delta, \\ &\text{particle } i \text{ chooses the formula (5) to fly,} \quad \text{if } \mathrm{FC}_i \le \delta. \end{aligned} \tag{8}$$

Objective function $f(X)$, $X = (x_1, \ldots, x_D)$
Initialize each particle's position X_i and velocity V_i randomly and assign X_i to the P_{best_i} at the same time $(i = 1, \ldots, M)$
while (The stop criterion is not satisfied) do {
For $i = 1 : M$, **do**
{
if $(i \le M)$
calculate the fitness value of particle i $f(X_i(t))$
end **if**
}
Rank the swarm according to the ascending sort of the fitness value and get $s_i(t), FC_i(t), G_{\text{best}}(t)$
Assign each particle's flight-mode according to the ***Rule*** (8)
Update $P_{\text{best}}(t)$
}
$t = t + 1$
end while
output $G_{\text{best}}, f(G_{\text{best}})$

Procedure 1: Double flight-modes particle swarm optimization.

That is to say, if $s_i(t) > \delta(M-1)+1$, then particle i will choose *non-rotational flight mode* to fly to next step. Otherwise, particle i will choose *rotational flight mode* to fly to next step.

The procedure of the DMPSO can be simply described as in Procedure 1.

4. Validation and Comparison

In order to test the performance of the DMPSO, we have tested it against the original PSO [1], the PSO-W [2], and the CLPSO [16]. For the ease of visualization, we have implemented our simulations using Matlab for various test functions.

4.1. Benchmark Functions. For the sake of having a fair and reasonable comparison between the DMPSO and the PSO, the DMPSO and the PSO-W, or the DMPSO and the CLPSO, we have chosen six well-known high-dimensional functions as our optimization simulation tests. All functions are tested on 50 dimensions. The properties and the formula of these functions are presented as follows.

(a) Schwefel's function:

$$f_1(X) = \sum_{i=1}^{D} |x_i| + \prod_{i=1}^{D} |x_i|, \quad -10 \le x_i \le 10, \ D = 50. \tag{9}$$

It is a unimodal function and has a global minimum $f_{\min} = 0$ at $(0, \ldots, 0)$. The complexity of Schwefel's function is due to its deep local optima being far from the global optimum. It will be hard to find the global optimum if many particles fall into one of the deep local optima.

(b) Rastrigin's function:

$$f_2(X) = 10D \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i)\right], \quad -5.12 \le x_i \le 5.12, \ D = 50. \tag{10}$$

The function is a complex multimodal function, and the number of its local minima shows an exponential increase with the problem dimension and its peak shape is up and down in jumping. When attempting to solve Rastrigin's function, algorithms may easily fall into a local optimum. Therefore, an algorithm capable of maintaining a larger diversity is likely to yield better results, so Rastrigin is viewed as a typical function used for testing global search performance of algorithm. It has a global minimum $f_{\min} = 0$ at $(0, \ldots, 0)$.

(c) Step function:

$$f_3(X) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2, \quad -100 \le x_i \le 100, \ D = 50, \tag{11}$$

in which $y = \lfloor x \rfloor$ is a bracket function (Gaussian function) and $x - 1 < \lfloor x \rfloor \le x < \lfloor x \rfloor + 1, \ \forall x \in \mathbf{R}$. Step function is a

Table 1: Experimental results.

f	Algorithms	BFV	WFV	Mean	MFEs ± STDEV
f_1	DMPSO	0	1.125465e − 009	6.434905e − 011	7350 ± 1.771972e − 010 (56%)
	PSO	9.736494	61.567035	31.111268	15000 ± 11.065242 (0%)
	CLPSO	8.174606e − 005	0.001325	5.125738e − 021	15000 ± 2.702365e − 004 (0%)
	PSO-w	4.429573	1.994109e + 002	80.584602	15000 ± 58.997786 (0%)
f_2	DMPSO	0	49.747953	3.979836	11130 ± 13.496281 (64%)
	PSO	1.790843e + 002	4.039120e + 002	2.847614e + 002	15000 ± 49.430124 (0%)
	CLPSO	21.153304	47.516825	31.762105	15000 ± 6.732226 (0%)
	PSO-w	1.416269e + 002	2.890130e + 002	2.1243064e + 002	15000 ± 43.169379 (0%)
f_3	DMPSO	0	0	0	1470 ± 0 (100%)
	PSO	351	11160	3.083180e + 003	15000 ± 3.937885e + 003 (0%)
	CLPSO	0	122	4.22	10440 ± 17.105894 (48%)
	PSO-w	45	311	1.405400e + 002	15000 ± 54.365809 (0%)
f_4	DMPSO	8.881784e − 016	3.286260e − 014	1.140421e − 014	60000 ± 8.672115e − 015 (0%)
	PSO	0.079333	14.476725	3.692153	60000 ± 4.976905 (0%)
	CLPSO	1.289395e − 010	2.200675	1.082862	60000 ± 0.692050 (0%)
	PSO-w	0.003600	11.916679	0.656150	60000 ± 1.737976 (0%)
f_5	DMPSO	0	1.729170e − 069	3.458545e − 071	11130 ± 2.420835e − 070 (28%)
	PSO	1.236611e − 004	26.217801	4.196887	60000 ± 9.707225 (0%)
	CLPSO	2.700980e − 038	9.595381e − 036	9.748685e − 037	60000 ± 1.924460e − 036 (0%)
	PSO-w	5.651214e − 021	1.681261e − 017	2.451834e − 018	60000 ± 3.861006e − 018 (0%)
f_6	DMPSO	0	3.330669e − 016	3.774758e − 017	35430 ± 9.049510e − 017 (82%)
	PSO	0.04331	1.810140e + 002	20.290841	60000 ± 41.921578 (0%)
	CLPSO	1.787459e − 014	0.092264	0.014715	60000 ± 0.021761 (0%)
	PSO-w	1.760398e − 004	90.751966	1.826534	60000 ± 12.832618 (0%)

discontinuous function and has a global minimum $f_{\min} = 0$ in the domain $\{X \in \mathbf{R}^{50} \mid -0.5 \le x_i \le 0.5\}$.

(d) Ackley's function:

$$f_4(X) = -20\exp\left[-\frac{1}{5}\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right] - \exp\left[\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right] + 20 + e, \quad -30 \le x_i \le 30,\ D = 50. \tag{12}$$

The function has one narrow global optimum basin and many minor local optima and has a global optimum $f_{\min} = 0$ at $(0, \ldots, 0)$.

(e) Sphere function:

$$f_5(X) = \sum_{i=1}^{D}x_i^2, \quad -5.12 \le x_i \le 5.12,\ D = 50. \tag{13}$$

It is a unimodal function, and its global minimum $f_{\min} = 0$ at $(0, \ldots, 0)$.

(f) Griewank's function:

$$f_6(X) = \frac{1}{4000}\sum_{i=1}^{D}x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1, \quad -600 \le x_i \le 600,\ D = 50. \tag{14}$$

The search space is relatively large in this optimization problem; Griewank's function has a $\prod_{i=1}^{D}\cos(x_i/\sqrt{i})$ component causing linkages among variables, thereby making it difficult to reach the global optimum. Therefore, it is generally regarded as a complex multimodal problem that is hard to optimize. The function has a global minimum $f_{\min} = 0$ at $(0, \ldots, 0)$.

4.2. Comparison of Experimental Results and Discussions. There are many means to carry out the comparison of algorithm performance. We can use such ways to compare the number of function evaluations (FEs) for a given accuracy or to compare their accuracies for a fixed number of function evaluations, and so forth. In our simulations, we use the two

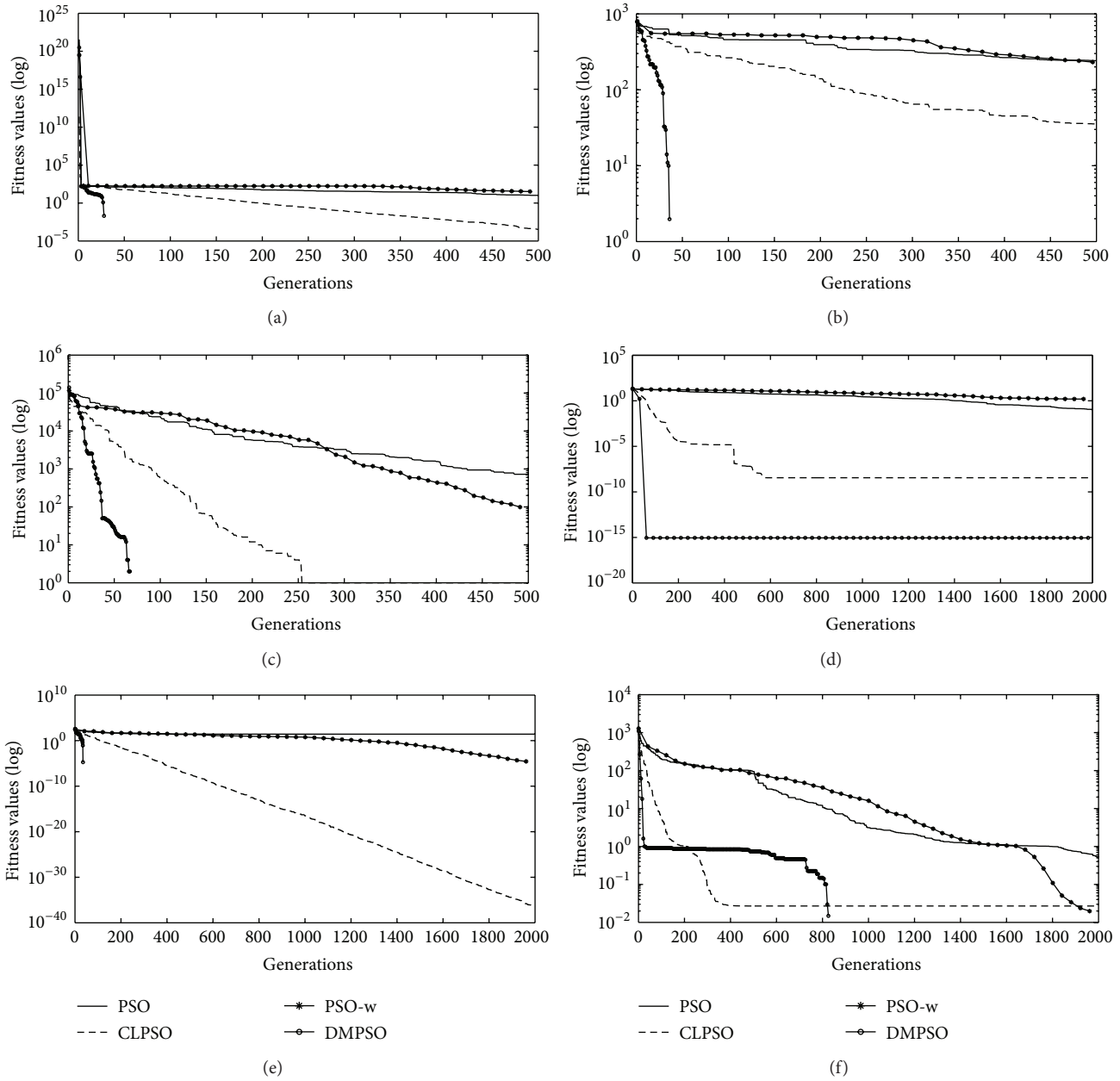

Figure 3: The median convergence characteristics of 50D test functions: (a) Schwefel's function, (b) Rastrigin's function, (c) Step function, (d) Ackley's function, (e) Sphere function, and (f) Griewank's function.

ways mentioned previously, and we set up a running-stopping condition for each algorithm. If a run has found the global optimal solution of optimization problem or has reached a fixed number of function evaluations, then it will stop running. We run each algorithm for 50 times so that we can do reasonable and meaningful analysis.

In order to ensure the comparability of the experimental results, the parameters settings are set as consistent as possible for the DMPSO, the PSO, the PSO-W, and the CLPSO. The parameters settings are listed as follows: the population size $M = 30$ and the acceleration coefficients $c_1 = c_2 = 2$ for all simulations. As for the test functions $f_1(X)$, $f_2(X)$, and $f_3(X)$, we set the maximum iterations at 500 (or the maximum number of function evaluations at 15000 correspondingly). As for the test functions $f_4(X)$, $f_5(X)$, and $f_6(X)$, we set the maximum iterations at 2000 (or the maximum number of function evaluations at 60000 correspondingly). On the other hand, we set the inertia weight $w = 0.628$ for the CLPSO, and set $w(t) = 0.8(1 - \exp(-5|g_j(t) - x_{ij}(t)|))$ for the DMPSO. As for the PSO-W, the linearly decreasing inertia weight w is used which starts at 0.9 and ends at 0.4, and $v_{\max} \leq 20\%(u_d - l_d)$.

In our experiments, we choose the best fitness value (BFV), the worst fitness value (WFV), the mean value (Mean), and the number of function evaluations in the form of mean (MFEs) ± standard deviation (STDEV) (success rate

of algorithm in finding the global optima) as the evaluation indicators of optimization ability for the four algorithms mentioned previously. These evaluation indicators cannot only reflect the optimization ability but also indicate the computing cost. We have got the experimental results being listed in Table 1.

In order to more easily contrast the convergence characteristics of the four algorithms, Figure 3 presents the convergence characteristics in term of the best fitness value of the mean run of each algorithm for each test function.

Discussions. From the results listed in Table 1 and the convergence curve simulation diagram in Figure 3, we can see that (1) the DMPSO performs much better than the original PSO, the CLPSO, and the PSO-W and (2) the DMPSO is much superior to the original PSO, the CLPSO, and the PSO-W in terms of global convergence property, accuracy, and efficiency. So, we conclude that the performance of the DMPSO is much better than that of the original PSO, that of the CLPSO, and that of the PSO-W.

5. Conclusions

This paper presents a novel particle swam optimization with double flight modes that we call the double flight modes particle swam optimization (DMPSO). In the optimization algorithm, each bird (particle) can use both rotational flight mode and non-rotational flight mode to fly while it is foraging for food in the search space. By using its superb flight skills, each bird (particle) has much improved its searching efficiency. From the experiments we conduct on some benchmark functions such as Schwefel, Rastrigin, Ackley, Step, Griewank, and Sphere, we can conclude that the DMPSO not only has marked advantage of global convergence property but also can effectively avoid the premature convergence problem to some extent and is one of good choices for use to solve the complex and high-dimensional optimization problems, although the DMPSO is not necessarily the best choice for solving various real-world optimization problems.

Acknowledgments

This work was supported by the key programs of the Institution of Higher Learning, Guangxi, China, under Grant (no. 201202ZD032), the Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis, the Natural Science Foundation of Guangxi, China, under Grant (no. 0832084), and the Natural Science Foundation of China under Grant (no. 61074185).

References

[1] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks. Part I*, pp. 1942–1948, Piscataway, NJ, USA, December 1995.

[2] Y. Shi and R. Eberhart, "Empirical study of particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation*, vol. 3, pp. 1945–1950, 1999.

[3] K. M. Rasmussen and T. Krink, "Hybrid particle swarm optimization with breeding and subpopulations," in *Proceedings of the 3rd Genetic and Evolutionary third Genetic and Evolutionary Computation Conference*, San Francisco, Calif, USA, 2001.

[4] Y. H. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation*, pp. 101–106, IEEE, Piscataway, NJ, USA, May 2001.

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

[6] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1671–1676, Honolulu, Hawaii, USA, 2002.

[7] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to participle swam optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[8] K. E. Parsopoulos and M. N. Vrahatis, "On the Computation of all global minimizers through particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 211–224, 2004.

[9] J. Sun and W. B. Xu, "A global search of quantum-behaved particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation*, pp. 325–331, IEEE Press, Washington, DC, USA, 2004.

[10] J. Sun, W. Xu, and J. Liu, "Parameter selection of quantum-behaved particle swarm optimization," Lecture Notes in Computer Science, Springer, Berlin, Germany.

[11] Z.-S. Lu and Z.-R. Hou, "Particle swarm optimization with adaptive mutation," *Acta Electronica Sinica*, vol. 32, no. 3, pp. 416–420, 2004 (Chinese).

[12] R. He, Y.-J. Wang, Q. Wang, J.-H. Zhou, and C.-Y. Hu, "An Improved particle swarm optimization based on self-adaptive escape velocity," *Journal of Software*, vol. 16, no. 12, pp. 2036–2044, 2005 (Chinese).

[13] L. Cong, Y.-H. Sha, and L.-C. Jiao, "Organizational evolutionary particle swarm optimization for numerical optimization," *Pattern Recognition and Artificial Intelligence*, vol. 20, no. 2, pp. 145–153, 2007 (Chinese).

[14] 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.

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

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

[17] X. F. Wang, F. Wang, and Y.-H. Qiu, "Research on a novel particle swarm algorithm with dynamic topology," *Computer Science*, vol. 34, no. 3, pp. 205–207, 2007 (Chinese).

[18] P. S. Shelokar, P. Siarry, V. K. Jayaraman, and B. D. Kulkarni, "Particle swarm and ant colony algorithms hybridized for improved continuous optimization," *Applied Mathematics and Computation*, vol. 188, no. 1, pp. 129–142, 2007.

[19] Q. Lu, S.-R. Liu, and X.-N. Qiu, "Design and realization of particle swarm optimization based on pheromone mechanism," *Acta Automatica Sinica*, vol. 35, no. 11, pp. 1410–1419, 2009.

[20] Q. Lu, X.-N. Qiu, and S.-R. Liu, "A discrete particle swarm optimization algorithm with fully communicated information," in *Proceedings of the Genetic and Evolutionary Computation Conference (GEC '09)*, pp. 393–400, ACM/SIGEVO, New York, NY, USA, June 2009.

[21] Q. Lu and S.-R. Liu, "A particle swarm optimization algorithm with fully communicated information," *Acta Electronica Sinca*, vol. 38, no. 3, pp. 664–667, 2010 (Chinese).

[22] Z.-Z. Shao, H.-G. Wang, and H. Liu, "Dimensionality reduction symmetrical PSO algorithm characterized by heuristic detection and self-learning," *Computer Science*, vol. 37, no. 5, pp. 219–222, 2010 (Chinese).