Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2013, Article ID 384125, 7 pages
http://dx.doi.org/10.1155/2013/384125

Research Article

Convergence Analysis of Particle Swarm Optimizer and Its Improved Algorithm Based on Velocity Differential Evolution

Hongtao Ye, Wenguang Luo, and Zhenqiang Li

School of Electrical and Information Engineering, Guangxi University of Science and Technology, Liuzhou 545006, China

Correspondence should be addressed to Hongtao Ye; yehongtao@126.com

Received 22 April 2013; Revised 28 July 2013; Accepted 4 August 2013

Academic Editor: Yuanqing Li

This paper presents an analysis of the relationship of particle velocity and convergence of the particle swarm optimization. Its premature convergence is due to the decrease of particle velocity in search space that leads to a total implosion and ultimately fitness stagnation of the swarm. An improved algorithm which introduces a velocity differential evolution (DE) strategy for the hierarchical particle swarm optimization (H-PSO) is proposed to improve its performance. The DE is employed to regulate the particle velocity rather than the traditional particle position in case that the optimal result has not improved after several iterations. The benchmark functions will be illustrated to demonstrate the effectiveness of the proposed method.

1. Introduction

Algorithms to tackle optimization problems include not only classical techniques such as dynamic programming, branch-and-bound, and gradient-based methods, but also more recent techniques such as metaheuristics [1]. Among the existing metaheuristic algorithms, the particle swarm optimization (PSO) algorithm is a population-based optimization technique developed by Kennedy and Eberhart in 1995 [2]. The PSO has resulted in a large number of variants of the standard PSO. Some variants are designed to deal with specific applications [3–6], and others are generalized for numerical optimization [7–10]. A hierarchical version of PSO (H-PSO) has been proposed by Janson and Middendorf [10]. In H-PSO, all particles are arranged in a tree that forms the hierarchy. A particle is influenced by its own best position and the best position particle in its neighborhood. It was shown that H-PSO performed very well compared to the standard PSO on unimodal and multimodal test functions [10, 11]. H-PSO presents the advantage of being conceptually very simple and requiring low computation time. However, the main disadvantage of H-PSO is the risk of a premature search convergence, especially in complex multiple peak search problems.

A number of algorithms combined various algorithmic components, often originating from algorithms of other research areas on optimization. These approaches are commonly referred to as hybrid meta-heuristics [12]. The surveys on hybrid algorithms that combine the PSO and differential evolution (DE) [13] were presented recently [14, 15]. These PSO-DE hybrids usually employ DE to adjust the particle position. But the convergence performance is dependent on the particle velocity. Limiting the velocity can help the particle to get out of local optima traps [16, 17]. In this paper, we will combine these two optimization algorithms and propose the novel hybrid algorithm H-PSO-DE. The DE is employed to regulate the particle velocity rather than the traditional particle position in case that the optimal result has not improved after several iterations. The hybrid algorithm aims to aggregate the advantages of both algorithms to efficiently tackle the optimization problem.

The remainder of this paper is organized as follows. Section 2 briefly describes the basic operations of the PSO, H-PSO, and DE algorithms. Section 3 presents an analysis of the relationship of particle velocity and convergence. Section 4 provides the hybrid optimization method: H-PSO-DE. Section 5 reveals the simulations and analysis of H-PSO-DE in solving unconstrained optimization problems. Finally, conclusions are given in Section 6.

2. The PSO, H-PSO, and DE Algorithms

2.1. The PSO Algorithm. The PSO [18–20] is a stochastic population-based optimization approach. Each particle is a D-dimensional vector, and it consists of a position vector x_n, which represents a candidate solution of the optimization problem, a velocity vector v_n, and a memory vector y_n, which is the best candidate solution encountered by the particle. The velocity and position of the particle are updated in every dimension d $(1 \leqslant d \leqslant D)$ by

$$\begin{aligned} v_{n,d}(t+1) = {} & w v_{n,d}(t) + c_1 r_1 \left(y_{n,d}(t) - x_{n,d}(t)\right) \\ & + c_2 r_2 \left(y_{p(n),d}(t) - x_{n,d}(t)\right), \end{aligned} \tag{1}$$

$$x_{n,d}(t+1) = x_{n,d}(t) + v_{n,d}(t+1), \tag{2}$$

where w is the inertia weight, which determines how much of the previous velocity the particle is preserved. c_1 and c_2 are positive constants. r_1 and r_2 are randomly chosen numbers uniformly distributed in the interval $[0, 1]$. $y_{p(n)}$ represents the best position achieved by any member of the population.

2.2. The H-PSO Algorithm. In H-PSO [21], all particles are arranged in a hierarchy. The hierarchy is defined by the *height* h, the *branching degree* bd, and the *total number of nodes* tnn of the corresponding tree.

In H-PSO, the iteration starts with the evaluation of the objective function of each particle at its current position. Then, the new velocity vectors and the new positions for the particles are determined. This means that for particle n, the value of $y_{p(n)}$ in (1) equals y_m, with m being the particle in the parent node of the node of particle n. H-PSO uses $y_{p(n)} = y_n$ only when particle n is in the root. If the function value of a particle n is better than the function value at its personal best position so far, then the new position is stored in y_n. For each particle n in a node of the tree, its own best solution is compared to the best solution found by the particles in the child nodes $S(n)$. If the best of these particles m is better than particle n, then particles n and m swap their places within the hierarchy.

2.3. The DE Algorithm. The DE [11, 13, 22] is a stochastic parallel direct search method. More specifically, DE's basic strategy can be summarized as follows.

Initialization. DE begins with a randomly initiated population of N D-dimensional parameter vectors $x_{i,g}$, $i = 1, 2, \ldots, N$ as a population for each generation g. The initial population $(g = 0)$ of the jth parameter of the ith vector is

$$x_{j,i,0} = x_{j,\min} + \text{rand}_{i,j}[0, 1] \cdot \left(x_{j,\max} - x_{j,\min}\right), \tag{3}$$

where $x_{j,\min}$ and $x_{j,\max}$ indicate the lower and upper bounds, respectively. $\text{rand}_{i,j}[0, 1]$ is a uniformly distributed random number lying between 0 and 1.

Mutation. DE mutates and recombines the population to produce a population of N trial vectors. Specifically, for each individual $x_{i,g}$, a mutant vector $v_{i,g}$ is generated according to

$$v_{i,g} = x_{r_1^i,g} + F \cdot \left(x_{r_2^i,g} - x_{r_3^i,g}\right), \tag{4}$$

where F, commonly known as scale factor, is a positive real number. Three other random individuals $x_{r_1^i,g}$, $x_{r_2^i,g}$, and $x_{r_3^i,g}$ are sampled randomly from the current population such that $r_1^i, r_2^i, r_3^i \in \{1, 2, \ldots, N\}$, and $i \neq r_1^i \neq r_2^i \neq r_3^i$.

Crossover. DE crosses each vector with a mutant vector:

$$u_{j,i,g} = \begin{cases} v_{j,i,g}, & \text{if } \left(\text{rand}_{i,j}[0, 1] \leq C_r \text{ or } j = j_{\text{rand}}\right), \\ x_{j,i,g}, & \text{otherwise}, \end{cases} \tag{5}$$

where C_r is called the crossover rate.

Selection. To decide whether or not it should become a member of generation $g+1$, the trial vector $v_{i,g}$ is compared to the target vector $x_{i,g}$ using the greedy criterion. The selection operation is described as

$$x_{i,g+1} = \begin{cases} u_{i,g}, & \text{if } f\left(u_{i,g}\right) \leq f\left(x_{i,g}\right), \\ x_{i,g}, & \text{otherwise}, \end{cases} \tag{6}$$

where $f(x)$ is the objective function to be minimized.

3. Relationship of Particle Velocity and Convergence

This section presents an analysis of the relationship of particle velocity and convergence.

Substituting (1) into (2) results in

$$\begin{aligned} x_{n,d}(t+1) = {} & x_{n,d}(t) + w v_{n,d}(t) + c_1 r_1 \left(y_{n,d}(t) - x_{n,d}(t)\right) \\ & + c_2 r_2 \left(y_{p(n),d}(t) - x_{n,d}(t)\right). \end{aligned} \tag{7}$$

From (2), it is known that

$$v_{n,d}(t) = x_{n,d}(t) - x_{n,d}(t-1). \tag{8}$$

Substituting (8) into (7) results in

$$\begin{aligned} x_{n,d}(t+1) = {} & \left(1 + w - c_1 r_1 - c_2 r_2\right) x_{n,d}(t) - w x_{n,d}(t-1) \\ & + c_1 r_1 y_{n,d}(t) + c_2 r_2 y_{p(n),d}(t). \end{aligned} \tag{9}$$

Step 1. Initialize particles in swarm. Initialize location x_n, and velocity v_n of each particle n. Best position $y_n = x_n$.
Step 2. Evaluate objective function $f(x_n)$ and update personal best.
for each particle n, do
if $f(x_n) < f(y_n)$ then
$y_n = x_n$
end if
end for
Step 3. Swap particles.
for each particle n, do
Determine the best successor m = arg min $\{f(y_q) \mid q \in S(n)\}$, where are the successors of n.
if $f(y_m) < f(y_n)$ then
Swap particles n and m
end if
end for
Step 4. Update the position and velocity of the nth particle.
for each particle n, do
Update the velocity v_n in each dimension d:
$$v_{n,d} = wv_{n,d} + c_1 r_1 (y_{n,d} - x_{n,d}) + c_2 r_2 (y_{p(n),d} - x_{n,d})$$
Move the particle: $x_{n,d} = x_{n,d} + v_{n,d}$
end for
Step 5. Judge the evolution process of H-PSO.
if ($g_0 == G_0$), then
goto Step 6
else goto Step 7
end if
Step 6. Update the velocity and position of the particle according to (16).
if $f(x_n') < f(x_n)$ then $x_n = x_n'$
end if
Step 7. If a stopping criterion is met, then output the global best position and stop; otherwise, repeat Step 2–Step 6.

Algorithm 1: Procedure for the H-PSO-DE.

This recurrence relation can be written as a matrix-vector product, so that

$$\begin{bmatrix} x_{n,d}(t+1) \\ x_{n,d}(t) \\ 1 \end{bmatrix} = \begin{bmatrix} 1 + w - c_1 r_1 - c_2 r_2 & -w & c_1 r_1 y_{n,d}(t) + c_2 r_2 y_{p(n),d}(t) \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_{n,d}(t) \\ x_{n,d}(t-1) \\ 1 \end{bmatrix}. \tag{10}$$

The characteristic polynomial of the matrix in (10) is $(1 - \lambda)(w - \lambda(1 + w - c_1 r_1 - c_2 r_2) + \lambda^2)$, which has a trivial root of $\lambda = 1$ and two other solutions

$$\alpha = \frac{(1 + w - c_1 r_1 - c_2 r_2 + \gamma)}{2}, \qquad \beta = \frac{(1 + w - c_1 r_1 - c_2 r_2 - \gamma)}{2}, \tag{11}$$

where $\gamma = \sqrt{(1 + w - c_1 r_1 - c_2 r_2)^2 - 4w}$.

Note that α and β are both eigenvalues of the matrix in (10). The explicit form of the recurrence relation (9) is then given by

$$x_{n,d}(t) = k_1 + k_2 \alpha^t + k_3 \beta^t, \tag{12}$$

where k_1, k_2, and k_3 are constants determined by the initial conditions of the system.

Substituting (12) into (8) results in

$$v_{n,d}(t) = h_1 \alpha^t + h_2 \beta^t, \tag{13}$$

where $h_1 = k_2(1 - 1/\alpha)$, $h_2 = k_3(1 - 1/\beta)$.

Consider

$$\lim_{t \to \infty} v_{n,d}(t) = \lim_{t \to \infty} \left(h_1 \alpha^t + h_2 \beta^t\right), \tag{14}$$

$$\lim_{t \to \infty} v_{n,d}(t) = \begin{cases} 0, & \text{if } \max(\|\alpha\|, \|\beta\|) < 1, \\ h_1 \text{or } h_2 \text{ or } h_1 + h_2, & \text{if } \max(\|\alpha\|, \|\beta\|) = 1. \end{cases} \tag{15}$$

Equation (15) implies that if the PSO algorithm is convergent, the velocity of the particles will decrease to zero or stay unchanged until the end of the iteration.

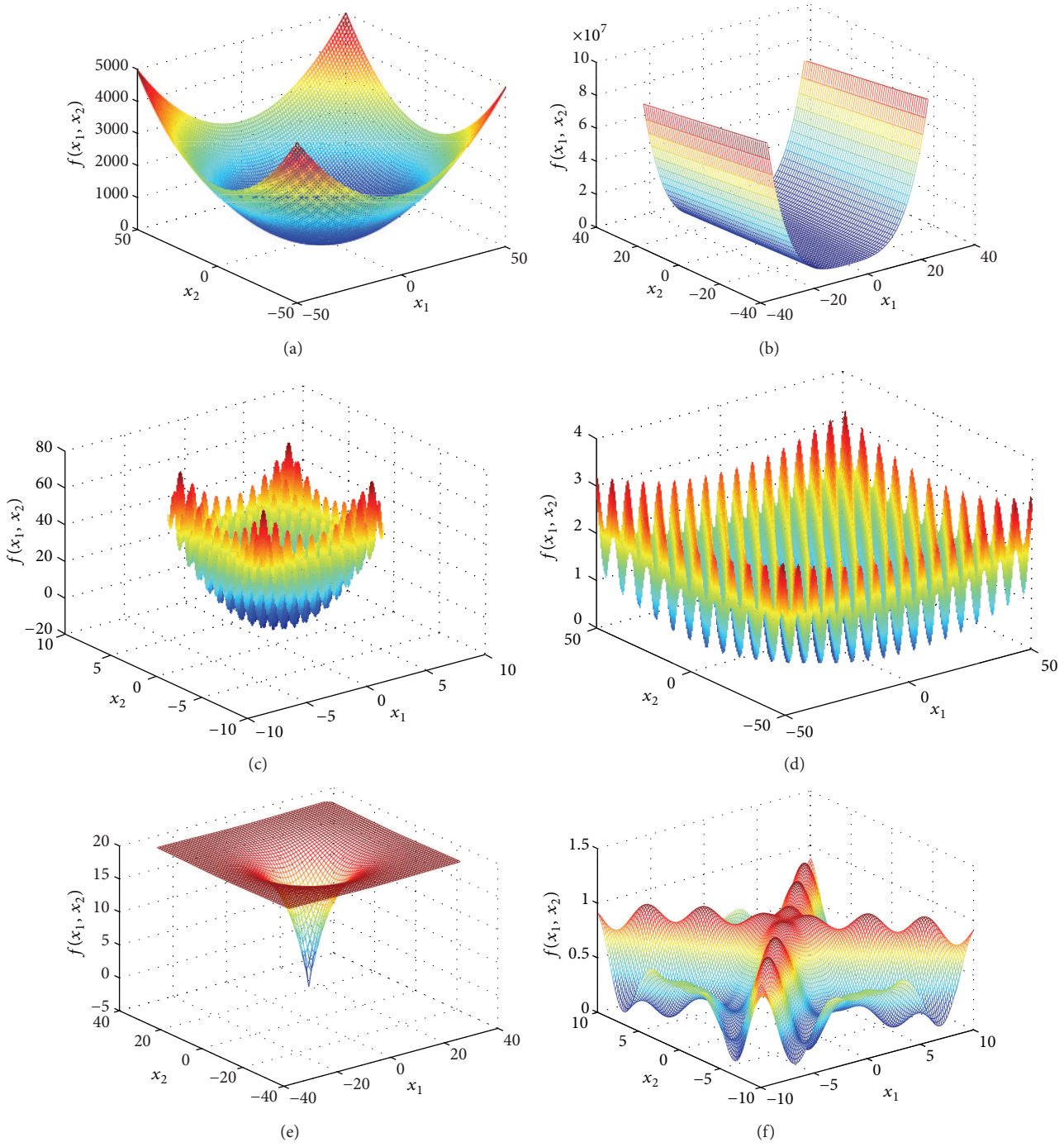

Figure 1: An illustration for 2-dimensional landscapes of the test functions. (a) Sphere function; (b) Rosenbrock function; (c) Rastrigin function; (d) Griewank function; (e) Ackley function; and (f) Schaffer's F6.

4. The Proposed H-PSO-DE Algorithm

The main idea of the hybrid H-PSO-DE algorithm is to employ the DE to regulate the particle velocity rather than the traditional particle position in case that the optimal result has not improved after several iterations. If the swarm is going to be in equilibrium, the evolution process will be stagnated as time goes on. To prevent the trend, if the stagnating step of evolution process g_0 is larger than threshold value G_0, the particle velocity performs mutation operators. The velocity and position of the particles are updated as follows.

If (rand() $< C_r$ or $d == k, k \in [1, D]$), then

$$v'_{n,d} = v_{n,d} + F \cdot (v_{1,d} - v_{2,d}),$$
$$x'_{n,d} = x_{n,d} + v'_{n,d}, \tag{16}$$

Table 1: Comparing the mean value of H-PSO-DE with respect to the other state-of-the-art algorithms.

Function	Mean value of the solution			
	H-PSO-DE	H-PSO	DE	PSO-DE
f_1	$2.06e-8$	$6.13e-7$	$7.29e-5$	$6.80e-7$
f_2	$3.23e-7$	$5.14e-7$	$2.14e-6$	$7.95e-7$
f_3	$1.77e-6$	$3.26e-5$	$5.21e-3$	$8.03e-5$
f_4	$1.08e-7$	$1.19e-6$	$5.61e-5$	$1.01e-6$
f_5	$2.01e-9$	$3.15e-9$	$4.21e-7$	$8.17e-9$
f_6	$3.09e-10$	$7.14e-10$	$7.26e-9$	$9.50e-10$

Table 2: Comparing the maximum value of H-PSO-DE with respect to the other state-of-the-art algorithms.

Function	Maximum value of the solution			
	H-PSO-DE	H-PSO	DE	PSO-DE
f_1	$6.35e-8$	$8.74e-7$	$6.20e-3$	$4.81e-6$
f_2	$5.94e-7$	$6.61e-7$	$7.24e-4$	$1.25e-6$
f_3	$7.12e-6$	$4.91e-5$	$6.16e-2$	$2.73e-4$
f_4	$4.25e-7$	$3.46e-6$	$4.91e-3$	$2.51e-6$
f_5	$4.34e-9$	$5.24e-9$	$7.61e-5$	$9.42e-9$
f_6	$5.32e-10$	$1.81e-9$	$2.81e-8$	$1.05e-9$

Table 3: Comparing the minimum value of H-PSO-DE with respect to the other state-of-the-art algorithms.

Function	Minimum value of the solution			
	H-PSO-DE	H-PSO	DE	PSO-DE
f_1	$9.34e-9$	$5.21e-7$	$5.28e-7$	$5.94e-8$
f_2	$1.56e-7$	$4.26e-7$	$3.92e-7$	$2.63e-7$
f_3	$8.35e-7$	$2.74e-5$	$1.97e-4$	$6.91e-6$
f_4	$8.61e-8$	$7.54e-7$	$3.71e-6$	$3.08e-7$
f_5	$5.97e-10$	$2.01e-9$	$8.87e-8$	$9.86e-10$
f_6	$1.23e-10$	$3.51e-10$	$6.24e-10$	$2.37e-10$

where $x_{n,d} = r \cdot y_{n,d} + (1-r) \cdot y_{p(n),d}$, r is a random number in the interval $[0, 1]$, and $v_{1,d}$ and $v_{2,d}$ are sampled randomly from v_n.

The procedure for H-PSO-DE algorithm is presented in Algorithm 1.

5. Simulations and Results

In this section, we present a simulation study to validate the proposed H-PSO-DE algorithm. A set of test functions that are commonly used in the field of continuous function optimization is listed in the appendix. They are a set of curvilinear functions for difficult unconstrained minimization problems. For illustration, the landscapes of two-dimensional versions of the six functions are depicted in Figure 1. The first two functions (Sphere and Rosenbrock) are unimodal functions, and they have a single local optimum that is also the global optimum. The remaining functions are multimodal, and they have several local optima. Note that the dimensional increase of these scalable functions does not change their basic features.

In our experiments, the H-PSO uses the parameter values $w = 0.729$, and $c_1 = c_2 = 1.494$ as suggested in [23] for a faster convergence rate. The population size that has been used is $N = 21$. The maximal number of generations uses $G = 5000$. The remainder parameters are set as $C_r = 0.5$, $F = 0.6$, $r = 0.5$, $G_0 = 8$, $h = 3$, and $bd = 4$. Thirty independent runs were carried out. The convergence behavior of the H-PSO is shown in Figure 2. For comparison purpose, the H-PSO-DE is also given in the same figure. As shown in Figure 2, the convergence performance of the H-PSO-DE is better than the H-PSO. H-PSO-DE is compared with H-PSO, DE, and PSO-DE [1] in terms of the selected performance metrics, such as the mean, maximum, and minimum values. In DE, we use DE/rand/1/bin strategy ($C_r = 0.5$, $F = 0.6$). As shown in Tables 1, 2, and 3, the H-PSO-DE outperforms H-PSO, DE, and PSO-DE. The H-PSO-DE is quite competitive when compared with the other existing methods.

6. Conclusions

In this paper, a new method named H-PSO-DE is proposed to solve optimization problems, which improves the performance of the H-PSO by incorporating DE. In H-PSO-DE, when the evolution process is stagnated for several generations, all the particles may lose the ability of finding a better solution. Then, the DE is employed to regulate the particle velocity to avoid wasting too much calculation time for vain search, so the searching efficiency of the H-PSO-DE is improved greatly. The H-PSO-DE is compared on test functions with H-PSO, DE, and PSO-DE. It is shown that H-PSO-DE performs significantly better.

Appendix

Benchmark Functions

Sphere:

$$f_1(x) = \sum_{i=1}^{30} x_i^2, \quad -50 \le x_i \le 50, \qquad \min(f_1) = f_1(0, \ldots, 0) = 0. \tag{A.1}$$

Rosenbrock:

$$f_2(x) = \sum_{i=1}^{29} \left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right), \quad -30 \le x_i \le 30, \qquad \min(f_2) = f_2(1, \ldots, 1) = 0. \tag{A.2}$$

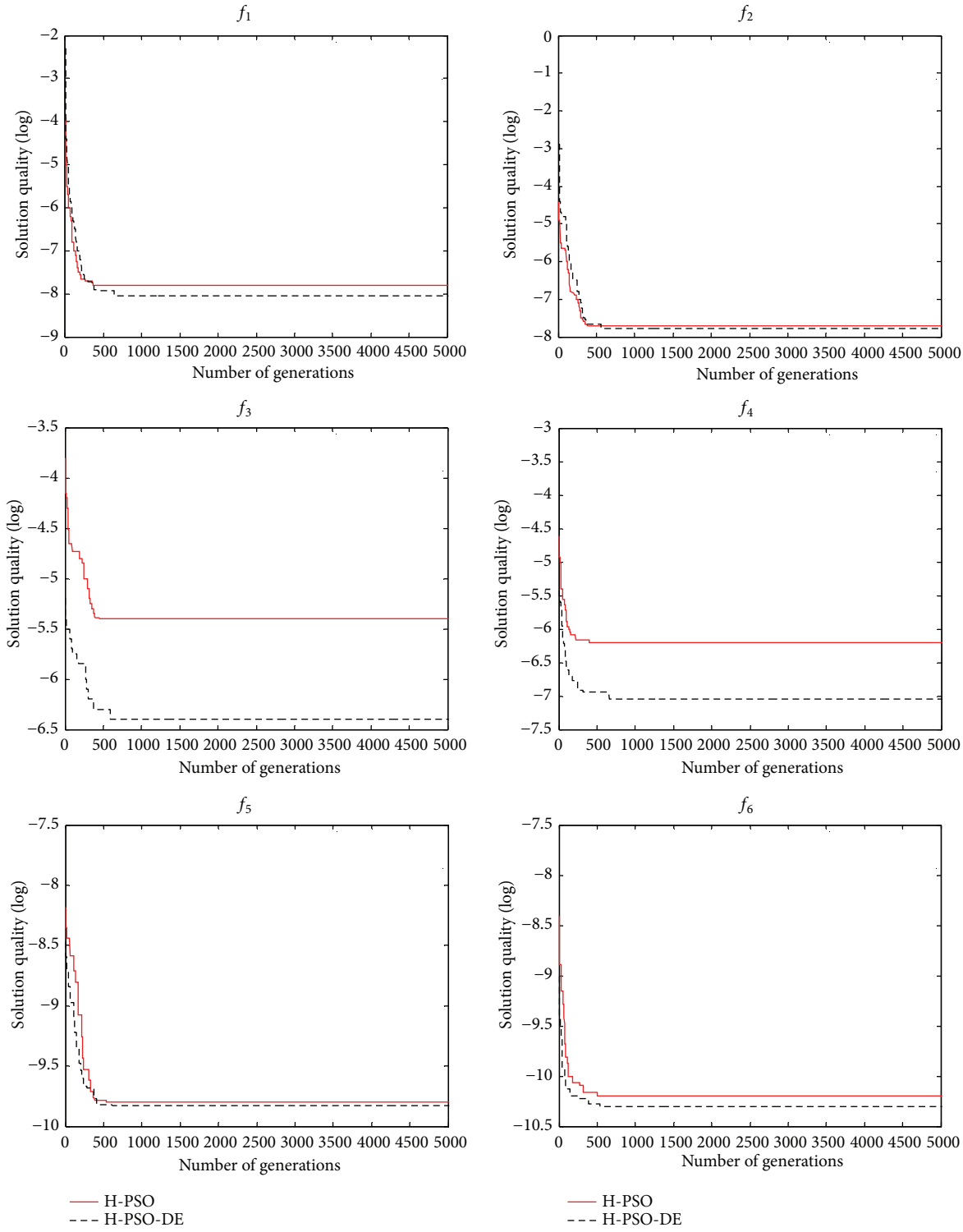

FIGURE 2: Convergence graph of the H-PSO and the H-PSO-DE for f_1–f_6.

Rastrigin:

$$f_3(x) = \sum_{i=1}^{30} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right), \quad -5.12 \le x_i \le 5.12, \quad \min(f_3) = f_3(0,\ldots,0) = 0. \tag{A.3}$$

Griewank:

$$f_4(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \Pi_{i=1}^{30}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1, \quad -600 \le x_i \le 600, \quad \min(f_4) = f_4(0,\ldots,0) = 0. \tag{A.4}$$

Ackley:

$$f_5(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30}x_i^2}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^{30}\cos 2\pi x_i\right) \quad (A.5)$$
$$-32 \le x_i \le 32,$$
$$\min(f_5) = f_5(0,\ldots,0) = 0.$$

Schaffer's F6:

$$f_6(x) = 0.5 + \frac{\left(\sin\sqrt{x_1^2 + x_2^2}\right) - 0.5}{\left(1 + 0.001\left(x_1^2 + x_2^2\right)\right)^2}, \quad (A.6)$$
$$-100 \le x_i \le 100,$$
$$\min(f_6) = f_6(0,0) = 0.$$

Acknowledgments

This work was supported by the Key Project of Chinese Ministry of Education (no. 212135), the Guangxi Natural Science Foundation (no. 2012GXNSFBA053165), the project of Education Department of Guangxi (no. 201203YB131), and the Doctoral Initiating Project of Guangxi University of Science and Technology (no. 11Z09).

References

[1] C. Zhang, J. Ning, S. Lu, D. Ouyang, and T. Ding, "A novel hybrid differential evolution and particle swarm optimization algorithm for unconstrained optimization," *Operations Research Letters*, vol. 37, no. 2, pp. 117–122, 2009.

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

[3] J. Zhang, J. Wang, and C. Yue, "Small population-based particle swarm optimization for short-term hydrothermal scheduling," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 142–152, 2012.

[4] R. Bhattacharya, T. K. Bhattacharyya, and R. Garg, "Position Mutated hierarchical particle swarm optimization and its application in synthesis of unequally spaced antenna arrays," *IEEE Transactions on Antennas and Propagation*, vol. 60, no. 7, pp. 3174–3181, 2012.

[5] M. A. Cavuslua, C. Karakuzub, and F. Karakayac, "Neural identification of dynamic systems on FPGA with improved PSO learning," *Applied Soft Computing*, vol. 12, no. 9, pp. 2707–2718, 2012.

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

[7] F. Peng, K. Tang, G. Chen, and X. Yao, "Population-based algorithm portfolios for numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 5, pp. 782–800, 2010.

[8] X. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 210–224, 2012.

[9] M. Li, D. Lin, and J. Kou, "A hybrid niching PSO enhanced with recombination-replacement crowding strategy for multimodal function optimization," *Applied Soft Computing Journal*, vol. 12, no. 3, pp. 975–987, 2012.

[10] S. Janson and M. Middendorf, "A hierarchical particle swarm optimizer and its adaptive variant," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 35, no. 6, pp. 1272–1282, 2005.

[11] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximity-based mutation operators," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 99–119, 2011.

[12] C. Blum, J. Puchinger, G. R. Raidl, and A. Roli, "Hybrid metaheuristics in combinatorial optimization: a survey," *Applied Soft Computing Journal*, vol. 11, no. 6, pp. 4135–4151, 2011.

[13] K. V. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Springer, Berin, Germany, 2005.

[14] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Evolving cognitive and social experience in particle swarm optimization through differential evolution: a hybrid approach," *Information Sciences*, vol. 216, pp. 50–92, 2012.

[15] B. Xin and J. Chen, "A survey and taxonomy on hybrid algorithms based on particle swarm optimization and differential evolution," *Journal of Systems Science and Mathematical Sciences*, vol. 31, no. 9, pp. 1130–1150, 2011.

[16] H. Liu, X. Wang, and G. Tan, "Convergence analysis of particle swarm optimization and its improved algorithm based on chaos," *Control and Decision*, vol. 21, no. 6, pp. 636–645, 2006.

[17] S. Jiang, Q. Wang, and J. Jiang, "Particle swarm optimization algorithm based on velocity differential evolution," in *Proceedings of the Chinese Control and Decision Conference (CCDC '09)*, pp. 1860–1865, Guilin, China, June 2009.

[18] S. Ghosh, S. Das, D. Kundu, K. Suresh, and A. Abraham, "Inter-particle communication and search-dynamics of lbest particle swarm optimizers: an analysis," *Information Sciences*, vol. 182, no. 1, pp. 156–168, 2012.

[19] X. F. Xie, W. J. Zhang, and Z. L. Yang, "A dissipative particle swarm optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '02)*, pp. 1456–1461, Honolulu, Hawaii, USA, May 2002.

[20] W. Gao, S. Liu, and L. Huang, "Particle swarm optimization with chaotic opposition-based population initialization and stochastic search technique," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 11, pp. 4316–4327, 2012.

[21] S. Janson and M. Middendorf, "A hierarchical particle swarm optimizer for noisy and dynamic environments," *Genetic Programming and Evolvable Machines*, vol. 7, no. 4, pp. 329–354, 2006.

[22] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 61–106, 2010.

[23] I. C. Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information Processing Letters*, vol. 85, no. 6, pp. 317–325, 2003.