Available online at www.sciencedirect.com

ScienceDirect

Neurocomputing 70 (2007) 672–679

NEUROCOMPUTING

www.elsevier.com/locate/neucom

# Center particle swarm optimization

Yu Liu[a,b,*], Zheng Qin[a,b], Zhewen Shi[b], Jiang Lu[b]

[a]*Department of Computer Science and technology, Tsinghua University, Beijing 100084, P.R.China*
[b]*Department of Computer Science, Xi'an Jiaotong University, Xi'an 710049, P.R.China*

Available online 13 October 2006

**Abstract**

Center particle swarm optimization algorithm (CenterPSO) is proposed where a center particle is incorporated into linearly decreasing weight particle swarm optimization (LDWPSO). Unlike other ordinary particles in LDWPSO, the center particle has no explicit velocity, and is set to the center of the swarm at every iteration. Other aspects of the center particle are the same as that of the ordinary particle, such as fitness evaluation and competition for the best particle of the swarm. Because the center of the swarm is a promising position, the center particle generally gets good fitness value. More importantly, due to frequent appearance as the best particle of swarm, it often attracts other particles and guides the search direction of the whole swarm. CenterPSO and LDWPSO are extensively compared on three well-known benchmark functions with 10, 20, 30 dimensions. Experimental results show that CenterPSO achieves not only better solutions but also faster convergence. Furthermore, CenterPSO and LDWPSO are compared as neural network training algorithms. The results show that CenterPSO achieves better performance than LDWPSO.
© 2006 Elsevier B.V. All rights reserved.

*Keywords:* Particle swarm optimization; Neural networks; Evolutionary computation

## 1. Introduction

Particle swarm optimization (PSO) [11] is an emerging evolutionary computation technique, inspired by social behavior simulations of bird flocking and fish schooling. Since it has fast convergence and promising performance on nonlinear function optimization, PSO has received much attention. Many researchers have devoted to improving its performance in various ways and developed many interesting variations. Most variations can be roughly grouped into the following categories.

(1) The improvement depends on incorporating the new coefficient into the velocity and position equations of the PSO algorithm or rationally selecting the values of the coefficients. Angeline [1] pointed that the original version of PSO had poor local search ability. In order to overcome this disadvantage, Shi and Eberhart [23] proposed linearly decreasing weight particle swarm optimization (LDWPSO) where a linearly decreasing inertia factor was introduced into the velocity update equation of the original PSO. Because the inertia factor effectively balances the global and local search abilities of the swarm, performance of PSO is significantly improved. Clerc [5] presented constriction PSO where a constriction factor was introduced into PSO to control the magnitude of velocities. It was mathematically proved that the resulting algorithm could guarantee the convergence even without clamping the velocity. However, if the strategy of clamping the velocity was combined into the algorithm, the performance could be improved further [6]. The constriction PSO has faster convergence than LDWPSO, but it is prone to be trapped in the local optima when multi-modal functions are presented.

(2) A key feature of PSO algorithms is social sharing information among the neighborhood. Therefore, various information sharing mechanisms were proposed to improve the performance. Kennedy [12] investigated the impacts of various neighborhood topologies and pointed out that the von Neumann topology results in superior performance. Suganthan [26] introduced a variable neighborhood operator. During the initial stages of the optimization, the neighborhood will be an individual particle itself. As the number of generations increases, the

*Corresponding author.
*E-mail address:* particleswarm@126.com (Y. Liu).

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2006.10.002

neighborhood will be gradually extended to include all particles. Mohais [19] proposed dynamically adjusted neighborhood where randomly generated, directed structures were used as the topology of the initial population and then edges of the structures were randomly migrated from one source node to another during the course of run. Liang et al. [15] presented a new learning strategy to make particles have different learning exemplars for different dimensions. Van den Bergh and Engelbrecht [27] proposed to split the solution vector into several sub-vectors which were then allocated their own swarms. Peram et al. [20] proposed to utilize the additional information of the nearby higher fitness particle that was selected according to fitness-distance-ratio (FDR) that denoted the ratio of fitness improvement over the respective distance. Baskar and Suganthan [3] proposed a novel concurrent PSO algorithm where modified PSO and FDR-PSO algorithms were simulated concurrently with frequent message passing between them. Janson [10] used dynamic hierarchy to define the neighborhood structure. He [8] introduced an additional component, passive congregation, into the velocity update equation.

(3) The operators of other evolutionary algorithms were combined with PSO. Angeline [2] used selection operator to improve the performance of the PSO algorithm. Løvbjerg et al. [17] combined the PSO algorithm with the idea of breeding and subpopulations. Poli et al. extended PSO via genetic programming [21]. Zhang and Xie [28] introduced differential evolution operator into PSO. Krink and Løvbjerg [13] combined PSO, genetic algorithms and hill climbers. Various mutation operators were also incorporated into PSO [7,9,25].

(4) Some mechanisms were designed to increase the diversity in order to prevent premature convergence to local minimum. Silva et al. [24] presented a predator–prey model to maintain population diversity. Zhang et al. [29] proposed to re-initialize the velocities of all particles at a predefined extinction interval, which simulated natural process of mass extinction in the fossil record. Krink [14] proposed several collision strategies to avoid crowding of the swarm. Løvbjerg [16] used self-organized criticality (SOC) to add diversity. Riget et al. [22] introduced attractive and repulsive PSO where the two phases, attraction and repulsion, alternated during the search according to a diversity measure.

Although variations of PSO have applied different strategies and parameters, all of them follow the same principle of swarm intelligence. Therefore, all variations appear similar features of social behavior. Within a swarm, individuals are relatively simple, but their collective behavior becomes quite complex. A group of particles in a swarm move around in the defined search space to find the optimum. Each particle relies on direct and indirect interaction and cooperation with other particles to determine the next search direction and step-size, so the swarm will move around and gradually converge toward the candidates of global optima or local optima. Thus the center of the swarm is probably near to the optimum. While this position changes during the search process, it can supply very useful information for capturing the optimum.

In this paper, a center particle, whose position is updated with the center of the swarm at every iteration, is incorporated into the PSO algorithm. Among variations of PSO, LDWPSO has been regarded as the most representative one, so LDWPSO is selected to demonstrate the effectiveness of the center particle. After a center particle is introduced into LDWPSO, it frequently becomes the best particle of the swarm during the search and then guides the whole search process. Consequently, the center particle greatly influences the performance of the PSO algorithm.

The rest of this paper is organized as follows. In Section 2, the center PSO is described in detail, and then the effect of the center particle on the performance of PSO is also discussed. In Section 3, firstly center particle swarm optimization (CenterPSO) and LDWPSO are compared on function optimization problems, and then they are compared on neural network training. Finally, Section 4 gives conclusions.

## 2. Center particle swarm optimization

### 2.1. LDWPSO

A swarm consists of $N$ particles moving around in a $D$-dimensional search space. The $i$th particle at the $t$th iteration has a position $X_i^{(t)} = (x_{i1}, x_{i2}, \ldots, x_{iD})$, a velocity $V_i^{(t)} = (v_{i1}, v_{i2}, \ldots, v_{iD})$, the best solution achieved so far by itself (*pbest*) $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$. The best solution achieved so far by the whole swarm (*gbest*) is represented by $P_g = (p_{g1}, p_{g2}, \ldots, p_{gD})$. The position of the $i$th particle at the next iteration will be calculated according to the following equations:

$$V_{id}^{(t+1)} = w \cdot V_{id}^{(t)} + c_1 \cdot rand() \cdot (P_{id} - X_{id}^{(t)}) + c_2 \cdot rand() \cdot (P_{gd} - X_{id}^{(t)}), \tag{1}$$

$$X_{id}^{(t+1)} = X_{id}^{(t)} + V_{id}^{(t+1)}, \tag{2}$$

where $c_1$ and $c_2$ are two positive constants, called cognitive learning rate and social learning rate, respectively; $rand()$ is a random function in the range $[0, 1]$ ; $w$ is inertia factor which linearly decreases from 0.9 to 0.4 through the search process. In addition, the velocities of the particles are confined within $[V_{\min}, V_{\max}]^D$. If an element of velocities exceeds the threshold $V_{\min}$ or $V_{\max}$, it is set equal to the corresponding threshold.

### 2.2. CenterPSO

The motivation for developing CenterPSO derives from the observation on search behavior of the swarm. From formula (1), it can be seen that the velocities of particles are determined by their previous velocities, cognitive learning, and social learning. Due to social learning (the third part of the formula), all the particles are attracted by *gbest* and move toward it. While the other two parts, previous velocities and cognitive learning, are corresponded to the autonomy property, which makes particles keep their own information. Therefore, during the search all particles move in the certain region where *gbest* locates, but their positions are usually different and approximately around *gbest*. For convenient observation, two-dimensional function optimization is illustrated. Figs. 1(a)–(f) show the distribution of the particles at the first, 10th, 40th, 60th, 80th, 100th iterations when the two-dimensional Rastrigrin function was optimized with LDWPSO. The signs: point, circle, star, and diamond denote the positions of the particle, optimum, *gbest*, and the center of the swarm, respectively. As shown in these figures, the following fact can be seen. The region of swarm activity is constantly changing. The *gbest* is near the center of the swarm, but it is not exactly the center of the swarm for reasons of many stochastic factors. In order to prove that the above fact happens in general optimization problems, 30-dimensional Rastrigrin function optimization is conducted. Fig. 2(a) shows the distances between *gbest* and the optimum, the center and the optimum at every iteration during optimization process. Fig. 2(b) shows the distance between *gbest* and the center of the swarm. From Fig. 2(a), it appears that the center is usually closer to the optimum during the search than the *gBest*. On the other hand, Fig. 2(b) indicates that at early iterations the center of the swarm and the *gBest* locate in different positions while at the end of iterations they converge to the same position. To sum up, the center of the swarm is a very important position and the swarm statistically shrinks to it through the iterations, but it is not explicitly visited in previous PSO algorithms.

In present paper, a center particle is proposed to explicitly visit the center of swarm at every iteration. After $N-1$ particles update their positions as the usual PSO algorithms at every iteration, a center particle is updated according to the following formula:

$$X_{cd}^{(t+1)} = \frac{1}{N-1} \sum_{i=1}^{N-1} X_{id}^{(t+1)}. \tag{3}$$

Unlike other particles, the center particle has no velocity, but it is involved in all operations the same as the ordinary particle, such as fitness evaluation, competition for the best particle, except for the velocity calculation. The resulting algorithm is called CenterPSO. The pseudo-code of CenterPSO is as follows:

```
BeginPSO
    Initialize();
    for $t = 1$ to max iteration
        $Fitness_c^{(t)} = EvaluationFitness(X_c^{(t)})$;
        if needed, update $P_i$ and $P_g$;
        for $i = 1 : N-1$
            $Fitness_i^{(t)} = EvaluationFitness(X_i^{(t)})$;
            $UpdateVelocity(V_i^{(t+1)})$ according to formula (1);
            $LimitVelocity(V_i^{(t+1)})$;
            $UpdatePosition(X_i^{(t+1)})$ according to formula (2);
            if needed, update $P_i$ and $P_g$;
        End
        Update the position of center particle according to
      formula (3);
        Terminate if $P_g$ meets problem requirements;
    End
EndPSO
```

The center particle has potential to get good solutions, as mentioned above. If the effect of center particle is just to try good positions, it will not influence the performance greatly. One particle is too few in contrast to a swarm of ordinary particles. More importantly, the center particle has more opportunities to become the *gbest* of the swarm. Hence it can guide the whole swarm to promising region and accelerate convergence. Fig. 3 shows the ratio of the number of iterations (where a particle serves as *gbest*) to the predefined maximum iteration in every run. One hundred runs of CenterPSO with 19 ordinary particles and one center particle, maximum iteration of 2000 were conducted on 30-dimensional Rastrigrin function. The center particle and a randomly selected ordinary particle were recorded. It is clear that the center particle has higher probability to be *gBest*. Therefore, the center particle often guides the search process. Although it is only one particle, it imposes great effect on the swarm.

## 3. Experiments

### 3.1. Function optimization

Three well-known benchmark functions (all minimization) were used in our experiments.

The first function is the Rosenbrock function:

$$f_1(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2) \quad (-100 \leqslant x_i \leqslant 100). \tag{4}$$

The second function is the generalized Rastrigrin function:

$$f_2(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10) \quad (-10 \leqslant x_i \leqslant 10). \tag{5}$$

The third function is the generalized Griewank function:

$$f_3(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad (-600 \leqslant x_i \leqslant 600). \tag{6}$$

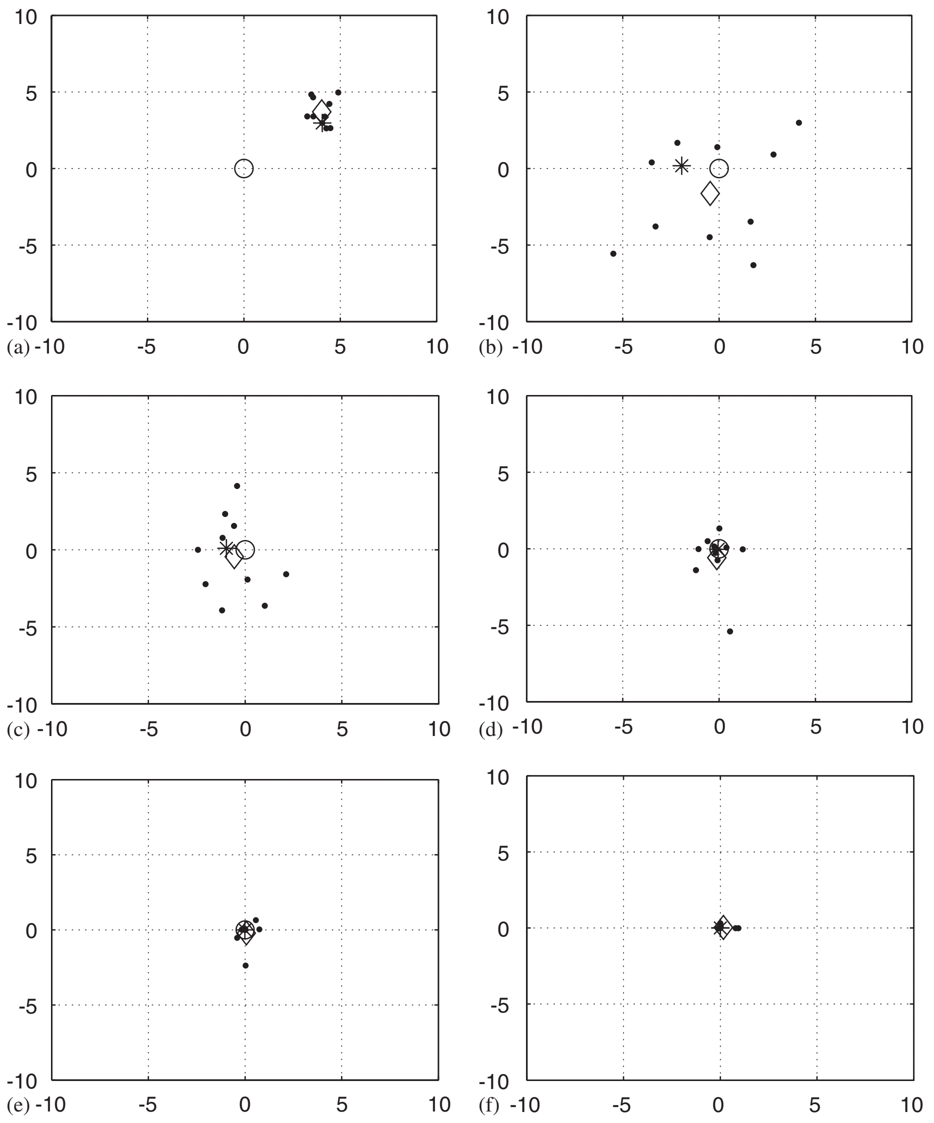

Fig. 1. The distribution of particles: (a) at the first iteration; (b) at the 10th iteration; (c) at the 40th iteration; (d) at the 60th iteration; (e) at the 80th iteration; and (f) at the 100th iteration.

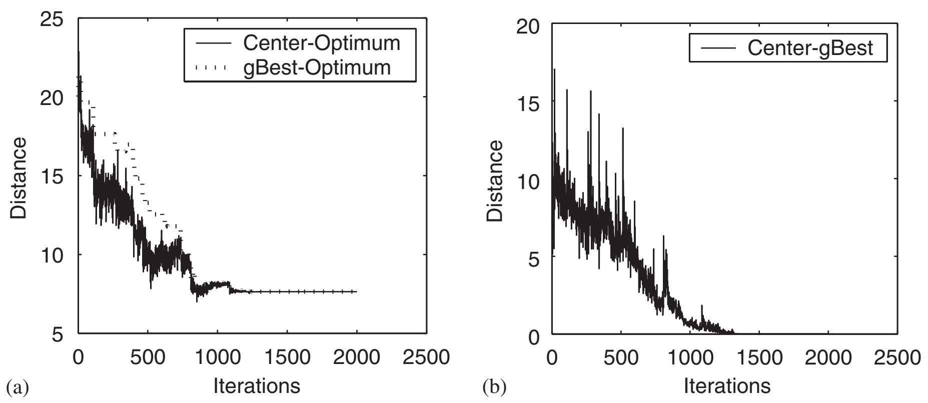

Fig. 2. The distances: (a) between the center of the swarm, *gbest* and optimum; (b) between the center of the swarm and *gbest*.

In our experiments, LDWPSO and CenterPSO were compared. The same set of parameters was assigned for two algorithms: inertia weight $w$ linearly decreased from 0.9 to 0.4; the learning rates were $c_1 = c_2 = 2$; $V_{\min}$ was equal to $X_{\min}$; $V_{\max}$ equal to $X_{\max}$. For each function three dimensions were tested: 10, 20 and 30; correspondingly, the

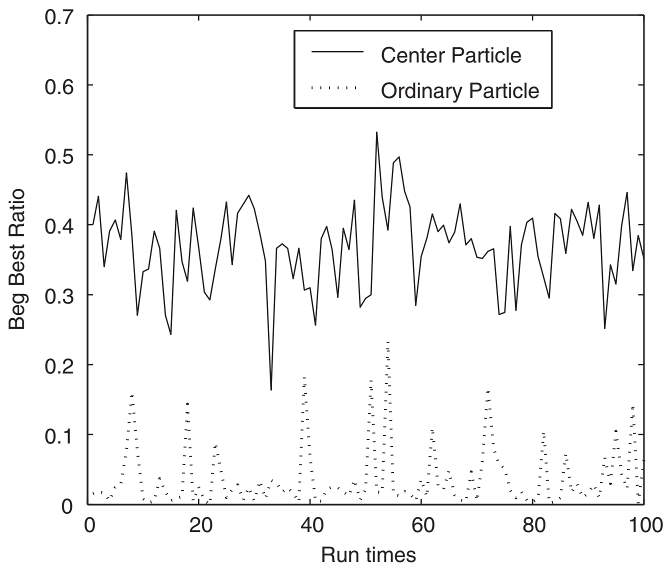

Fig. 3. The ratio of the number of iterations where a particle is the *gbest* to the predefined maximum iteration.

Table 1
Search space and initialization range

| Function | Search space | Initialization range |
|---|---|---|
| $f_1(x)$ | $-100 \leqslant x_i \leqslant 100$ | $15 \leqslant x_i \leqslant 30$ |
| $f_2(x)$ | $-10 \leqslant x_i \leqslant 10$ | $2.56 \leqslant x_i \leqslant 5.12$ |
| $f_3(x)$ | $-600 \leqslant x_i \leqslant 600$ | $300 \leqslant x_i \leqslant 600$ |

maximum numbers of iterations were set to 1000, 1500 and 2000. For investigating the scalability of algorithms, four population sizes ($N = 20, 40, 80, 160$) were used for each function with different dimensions. CenterPSO included one center particle and $N-1$ ordinary particles. For each experimental setting, 100 runs of the algorithm were performed. In order to give right indications of relative performance, an asymmetric initialization was adopted according to literature [1]. The search space and initialization range for each of the test functions were listed in Table 1.

Tables 2–4, respectively, listed the mean fitness value and standard deviation of the best solutions averaged over 100 trails on Rosenbrock, Rastrigrin, and Griewank functions with each experimental setting. As shown in these tables, it is clear that CenterPSO algorithm outperforms LDWPSO algorithm on all the benchmark problems on which we have conducted experiments so far. CenterPSO achieves the smaller fitness value and standard deviation than LDWPSO. Figs. 4–6 show fitness evolution curve of CenterPSO and LDWPSO with 20 particles on three 30-dimensional functions. The results in each figure were averaged over 100 trials. Here, we can see that faster convergence is achieved and an improvement in the best solution is found by CenterPSO. It can be concluded that the introduction of center particle not only accelerates convergence but also improves the solution quality.

Table 2
The best fitness values for the Rosenbrock function $f_1(x)$

| Size | Dim | Max iteration | CenterPSO | LDWPSO |
|---|---|---|---|---|
| 20 | 10 | 1000 | $52.4964 \pm 99.3027$ | $149.6743 \pm 321.0806$ |
| | 20 | 1500 | $111.5968 \pm 173.3808$ | $198.2874 \pm 364.0810$ |
| | 30 | 2000 | $131.9323 \pm 135.8345$ | $271.7280 \pm 368.5546$ |
| 40 | 10 | 1000 | $25.1575 \pm 65.3048$ | $63.0347 \pm 117.2544$ |
| | 20 | 1500 | $59.7008 \pm 57.2410$ | $169.0082 \pm 307.6407$ |
| | 30 | 2000 | $87.1757 \pm 63.6501$ | $234.9583 \pm 342.3578$ |
| 80 | 10 | 1000 | $17.6040 \pm 25.2435$ | $34.5463 \pm 67.8022$ |
| | 20 | 1500 | $61.0385 \pm 56.2370$ | $118.6109 \pm 235.0387$ |
| | 30 | 2000 | $62.3455 \pm 59.4016$ | $221.7500 \pm 392.2834$ |
| 160 | 10 | 1000 | $15.5263 \pm 25.4743$ | $20.5899 \pm 39.8360$ |
| | 20 | 1500 | $47.6311 \pm 46.4689$ | $79.3686 \pm 140.9803$ |
| | 30 | 2000 | $42.9959 \pm 44.9927$ | $104.3028 \pm 194.7433$ |

Table 3
The best fitness values for the generalized Rastrigrin function $f_2(x)$

| Size | Dim | Max iteration | CenterPSO | LDWPSO |
|---|---|---|---|---|
| 20 | 10 | 1000 | $4.5308 \pm 2.2246$ | $5.3124 \pm 2.7026$ |
| | 20 | 1500 | $17.5823 \pm 6.4463$ | $23.0965 \pm 6.9425$ |
| | 30 | 2000 | $33.5934 \pm 9.5629$ | $48.6432 \pm 11.1707$ |
| 40 | 10 | 1000 | $2.9592 \pm 1.7284$ | $3.9407 \pm 2.0244$ |
| | 20 | 1500 | $12.5039 \pm 4.5840$ | $16.5678 \pm 5.5229$ |
| | 30 | 2000 | $26.6870 \pm 7.7643$ | $38.8859 \pm 10.1198$ |
| 80 | 10 | 1000 | $1.8696 \pm 1.1424$ | $2.4810 \pm 1.3439$ |
| | 20 | 1500 | $10.6870 \pm 3.8770$ | $13.1075 \pm 4.2326$ |
| | 30 | 2000 | $22.7680 \pm 6.7589$ | $29.3049 \pm 7.4000$ |
| 160 | 10 | 1000 | $1.0558 \pm 0.9584$ | $1.5337 \pm 1.1121$ |
| | 20 | 1500 | $8.2121 \pm 2.2916$ | $9.3754 \pm 3.1535$ |
| | 30 | 2000 | $21.4185 \pm 5.9499$ | $23.9643 \pm 5.7055$ |

Table 4
The best fitness values for the Griewank function $f_3(x)$

| Size | Dim | Max iteration | CenterPSO | LDWPSO |
|---|---|---|---|---|
| 20 | 10 | 1000 | $0.0831 \pm 0.0466$ | $0.0996 \pm 0.0568$ |
| | 20 | 1500 | $0.0258 \pm 0.0266$ | $0.0312 \pm 0.0293$ |
| | 30 | 2000 | $0.0120 \pm 0.0165$ | $0.0186 \pm 0.0197$ |
| 40 | 10 | 1000 | $0.0856 \pm 0.0440$ | $0.0882 \pm 0.0511$ |
| | 20 | 1500 | $0.0276 \pm 0.0239$ | $0.0314 \pm 0.0265$ |
| | 30 | 2000 | $0.0088 \pm 0.0119$ | $0.0166 \pm 0.0193$ |
| 80 | 10 | 1000 | $0.0798 \pm 0.0370$ | $0.0746 \pm 0.0326$ |
| | 20 | 1500 | $0.0326 \pm 0.0266$ | $0.0348 \pm 0.0334$ |
| | 30 | 2000 | $0.0093 \pm 0.0120$ | $0.0118 \pm 0.0132$ |
| 160 | 10 | 1000 | $0.0597 \pm 0.0277$ | $0.0634 \pm 0.0288$ |
| | 20 | 1500 | $0.0295 \pm 0.0279$ | $0.0301 \pm 0.0253$ |
| | 30 | 2000 | $0.0120 \pm 0.0168$ | $0.0122 \pm 0.0150$ |

## *3.2. Neural network training*

Neural network training is a complex optimization problem. Usually, the objective is the mean sum of squared

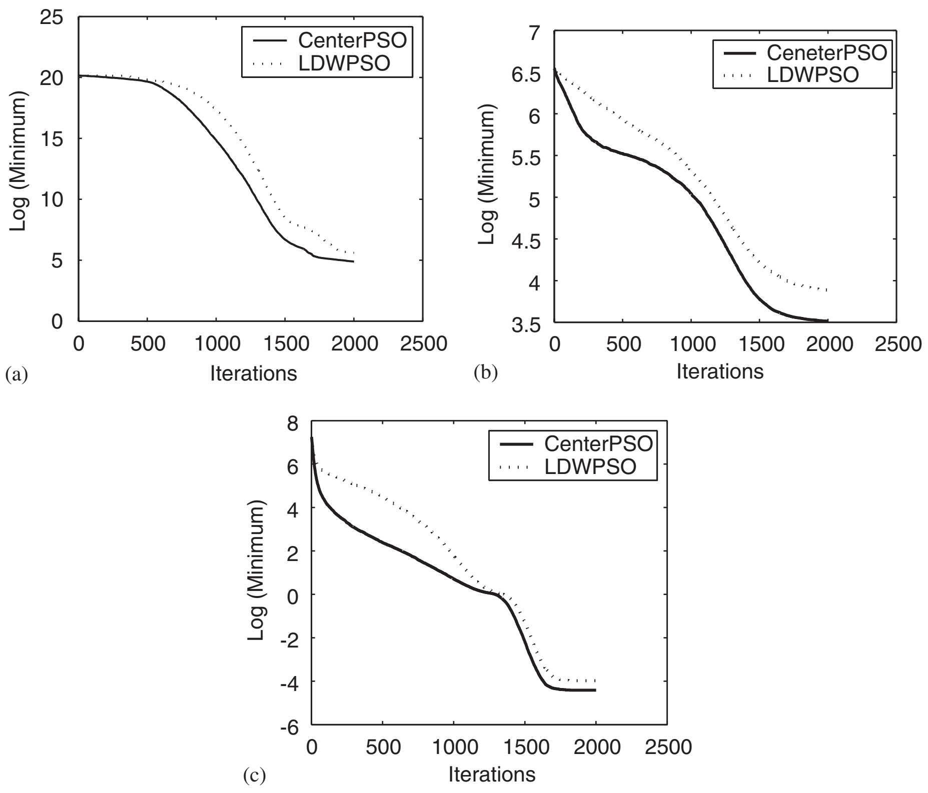

Fig. 4. Performance on 30-dimensional functions: (a) Rosenbrock; (b) Rastrigrin; and (c) Griewank.

errors (MSE) over all training patterns. The variables consist of neural network weights and biases. Suppose a standard network architecture with $D$ input units, $M$ hidden units and $C$ output units, the number of the variables is $W = (D+1) \times M + (M+1) \times C$. In brief, neural network training is a high dimensional optimization problem with many local minima.

LDWPSO and CenterPSO were compared as neural network training algorithms. Two benchmark data sets: the Australian credit card assessment problem and the diabetes problem were used. Both data sets are available from the UCI repository of machine learning databases [4]. The following is a brief description of each data set. The Australian credit card assessment problem is to assess applications for credit cards based on a number of attributes. This data set contains 690 patterns with 14 attributes; Six of them are number and eight are discrete. The output has two classes. One is to award the credit card, and the other is not. The diabetes data set is a two-class problem that has 500 examples of class 1 and 268 of class 2. There are eight attributes for each example. The objective is to test if a patient has a diabetes or not.

For neural network training, *n*-fold cross-validation technique [18] was used where the data are randomly divided into *n* mutually exclusive data groups of equal size. One data group is selected as the testing set, and the other groups become the training set. In our experiment *n* is set to 10. Three-layer feedforward neural networks with sigmoidal transfer functions were adopted for classification tasks. For credit card, the neural network was set to 14 input units, five hidden units, and two output units, therefore, the dimension of each particle was 87. For diabetes, the neural network was set to eight input units, five hidden units, and two output units, thus the dimension of each particle was 57. The parameters of CenterPSO and LDWPSO were the same as those in above experiments for function optimization except that $V_{\min} = -2$, $V_{\max} = 2$; the maximum number of iteration was set to 1000; population size was set to 20.

The values of MSE of two algorithms on training sets are listed in Table 5. The results were averaged on 10-fold cross-validation. The measures Mean, SD, Min, and Max indicate the mean value, standard deviation, minimum, and maximum value, respectively. The values of mean accuracy rates averaged on 10-fold cross-validation are listed in Table 6. Tables 5 and 6 show that CenterPSO achieves better performance than LDWPSO as neural network training algorithms. This further indicates the effectiveness of the center particle.

## 4. Conclusions

In this paper, we proposed CenterPSO algorithm where a center particle was introduced into LDWPSO. The position of the center particle was updated with the center of swarm that consists of ordinary particles at every

Table 5
The MSE of two algorithms on training sets

| | Credit card | | Diabetes | |
|---|---|---|---|---|
| | LDWPSO | CenterPSO | LDWPSO | CenterPSO |
| Mean | 0.2904 | 0.2361 | 0.3969 | 0.3918 |
| Std | 0.0616 | 0.0601 | 0.0159 | 0.0124 |
| Min | 0.1893 | 0.1818 | 0.3752 | 0.3747 |
| Max | 0.3620 | 0.3558 | 0.4379 | 0.4157 |

Table 6
The accuracy rates on two benchmark data sets

| | Training set | | Test set | |
|---|---|---|---|---|
| | LDWPSO | CenterPSO | LDWPSO | CenterPSO |
| Credit card | 0.8059 | 0.8526 | 0.7439 | 0.7900 |
| Diabetes | 0.6971 | 0.7110 | 0.6645 | 0.6703 |

iteration. Due to characteristic of swarm activity, all particles oscillate around the center of the swarm and gradually converge toward it. The center particle usually gets good position and often becomes the *gBest* of the swarm during the run. Therefore, despite only one center particle present, it has more opportunities to guide the search of the whole swarm, and influences the performance greatly. Experimental results show that CenterPSO achieves better performance than LDWPSO.

## Acknowledgment

This work was partially supported by the National Grand Fundamental Research 973 Program of China (No. 2004CB719401) and the National High Technology Development Program of China (No. 2003AA412020).

## References

[1] P.J. Angeline, Evolutionary optimization versus particle swarm optimization: philosophy and performance differences, Lecture Notes in Computer Science, vol. 1447, Springer, Berlin, 1998, pp. 601–610.
[2] P.J. Angeline, Using selection to improve particle swarm optimization, Proceedings of the IEEE Conference on Evolutionary Computation, 1998, pp. 84–89.
[3] S. Baskar, P. Suganthan, A novel concurrent particle swarm optimization, Proceedings of the Congress on Evolutionary Computation, 2004, pp. 792–796.
[4] C. Blake, C.J. Merz, UCI repository of machine learning databases, ⟨http://www.ics.uci.edu/~mlearn/MLRepository.html⟩, 1998.
[5] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evol. Comput. 6 (2002) 58–73.
[6] R. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, Proceedings of the IEEE Conference on Evolutionary Computation, 2000, pp. 84–88.
[7] S.C. Esquivel, C.A. Coello Coello, On the use of particle swarm optimization with multimodal functions, Proceedings of the Congress on Evolutionary Computation, 2003, pp. 1130–1136.
[8] S. He, Q.H. Wu, J.Y. Wen, J.R. Saunders, R.C. Paton, A particle swarm optimizer with passive congregation, Biosystems 78 (1–3) (2004) 135–147.
[9] N. Higashi, H. Iba, Particle swarm optimization with Gaussian mutation, Proceedings of the 2003 IEEE Swarm Intelligence Symposium, 2003, pp. 72–79.
[10] S. Janson, M. Middendorf, A hierarchical particle swarm optimizer and its adaptive variant, IEEE Trans. Syst. Man Cybern. Part B 35 (6) (2005) 1272–1282.
[11] J. Kennedy, R. Eberhart, Particle swarm optimization, Proceeding of the IEEE International Conference on Neural Networks, 1995, pp. 1942–1947.
[12] J. Kennedy, R. Mendes, Population structure and particle swarm performance, Proceedings of the Congress on Evolutionary Computation, 2002, pp. 1671–1676.
[13] T. Krink, M. Løvbjerg, The lifecycle model: combining particle swarm optimisation, genetic algorithms and hillclimbers, Proceedings of Parallel Problem Solving from Nature, vol. VII, 2002, pp. 621–630.
[14] T. Krink, J.S. Vesterstrøm, J. Riget, Particle swarm optimisation with spatial particle extension, Proceedings of the Congress on Evolutionary Computation, 2002, pp. 1474–1479.
[15] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Particle swarm optimization algorithms with novel learning strategies, Proceedings of IEEE Conference on Systems, Man and Cybernetics, 2004, pp. 3659–3664.
[16] M. Løvbjerg, T. Krink, Extending particle swarm optimisers with self-organized criticality, Proceedings of the Congress on Evolutionary Computation, 2002, pp. 1588–1593.
[17] M. Løvbjerg, T.K. Rasmussen, T. Krink, Hybrid particle swarm optimiser with breeding and subpopulations, Proceedings of the Third Genetic and Evolutionary Computation Conference, 2001, pp. 469–476.
[18] D. Michie, D.J. Spiegelhalter, C.C. Taylor, Machine Learning, Neural and Statistical Classification, Ellis Horwood, New York, 1994.
[19] A. Mohais, C. Ward, C. Posthoff, Randomized directed neighborhoods with edge migration in particle swarm optimization, Proceedings of the IEEE Congress on Evolutionary Computation, 2004, pp. 548–555.
[20] T. Peram, K. Veeramachaneni, C.K. Mohan, Fitness-distance-ratio based particle swarm optimization, Proceedings of the IEEE Swarm Intelligence Symposium, 2003, pp. 174–181.
[21] R. Poli, W.B. Langdon, O. Holland, Extending particle swarm optimisation via genetic programming, Proceedings of the Eighth European Conference on Genetic Programming, 2005, pp. 291–300.
[22] J. Riget, J.S. Vesterstrøm, A diversity-guided particle swarm optimizer—the ARPSO, Technical Report 2002-02, EVALife, Department of Computer Science, University of Aarhus, 2002.
[23] Y. Shi, R. Eberhart, A modified particle swarm optimizer, Proceedings of the IEEE Conference on Evolutionary Computation, 1998, pp. 69–73.
[24] A. Silva, A. Neves, E. Costa, An empirical comparison of particle swarm and predator prey optimisation, Lecture Notes in Computer Science, vol. 2464, Springer, Berlin, 2002, pp. 103–110.
[25] A. Stacey, M. Jancic, I. Grundy, Particle swarm optimization with mutation, Proceedings of the Congress on Evolutionary Computation, 2003, pp. 1425–1430.
[26] P.N. Suganthan, Particle swarm optimiser with neighbourhood operator, Proceedings of the Congress on Evolutionary Computation, 1999, pp. 1958–1962.
[27] F. van den Bergh, A.P. Engelbrecht, A cooperative approach to particle swarm optimization, IEEE Trans. Evol. Comput. 8 (3) (2004) 225–239.
[28] W.J. Zhang, X.F. Xie, Depso: hybrid particle swarm with differential evolution operator, Proceedings of IEEE International Conference on Systems, Man and Cybernetics, 2003, pp. 3816–3821.

[29] W.J. Zhang, X.F. Xie, Z.L. Yang, Hybrid particle swarm optimizer with mass extinction, International Conference on Communication, Circuits and Systems, 2002, pp. 1170–1173.

**Yu Liu**, born in 1977, received his Ph.D. degree in computer science and technology from Xi'an Jiaotong University. He is currently a research assistant at Tsinghua University. His main research interests include computational intelligence, parallel computation, and data mining.

**Zheng Qin** is a professor of Tsinghua University. He received his Doctor degree in Northwestern Polytechnical University. Now he is also director of E-Commerce Institute of Xi'an Jiaotong University. His researches mainly include computer system integration, E-Commerce and intelligent decision.

**Zhewen Shi**, born in 1982, now is studying as a master candidate at department of computer science of Xi'an Jiaotong University. Her research interests mainly involve artificial intelligence and graphic retrieval.

**Jiang Lu**, born in 1982, now is studying as a master candidate at department of computer science of Xi'an Jiaotong University. Her research interests mainly involve artificial intelligence and graphic retrieval.