Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 69 (2006) 2396–2401
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Letters

A novel adaptive sequential niche technique for multimodal function optimization

Jun Zhang[a,b,c,*], De-Shuang Huang[a], Tat-Ming Lok[d], Michael R. Lyu[e]

[a]Intelligent Computing Lab, Institute of Intelligent Machines, Chinese Academy of Sciences, P.O. Box 1130, Hefei, Anhui 230031, China
[b]Department of Automation, University of Science and Technology of China, Hefei, Anhui, China
[c]School of Electronic Science and Technology, Anhui University, Hefei, Anhui, China
[d]Information Engineering Department, The Chinese University of Hong Kong, Shatin, Hong Kong
[e]Computer Science & Engineering Department, The Chinese University of Hong Kong, Shatin, Hong Kong

Received 4 January 2006; received in revised form 10 February 2006; accepted 10 February 2006
Communicated by R.W. Newcomb
Available online 14 June 2006

Abstract

This paper proposes a novel adaptive sequential niche particle swarm optimization (ASNPSO) algorithm, which uses multiple sub-swarms to detect optimal solutions sequentially. In this algorithm, the hill valley function is used to determine how to change the fitness of a particle in a sub-swarm run currently. This algorithm has strong and adaptive searching ability. The experimental results show that the proposed ASNPSO algorithm is very effective and efficient in searching for multiple optimal solutions for benchmark test functions without any prior knowledge.

Keywords: Genetic algorithm; Niche technique; Particle swarm optimization; Penalty function; Multimodal function optimization

1. Introduction

The stochastic search algorithms are widely used in evolving artificial neural network (ANN) architecture and weights [5,3]. As a rule, the best weights or architecture of an ANN are not exclusive. In fact, the different architecture of an ANN is very useful in different situations. However, the ordinary stochastic search algorithm only finds one solution. The niche methods for stochastic search algorithms are techniques that can maintain a stable subpopulation for multiple solutions. In practical application, there are two categories of niche techniques, the parallel and sequential niche methods. Since Beasley et al. [1], however first proposed a sequential niche technique, over the years, this method has been ignored by researchers. Thus, there is little progress on it. On the contrary, the parallel techniques have attained more attention in recent years. Mahfound [7] had pointed out that the sequential technique has some disadvantages, while the parallel niche technique is generally faster than the sequential one. Nevertheless, the sequential technique still has its unique advantages. Especially the sequential technique can be integrated with the parallel ones [12]. So a good sequential method can also improve the performance of the entire niche technique.

Currently, most niche techniques need some extra tunable parameters, where the most important parameter is niche radius. An inappropriate radius will generally make a niche algorithm performance worse. In fact, the determination of the niche radius more or less depends on some prior knowledge from a special problem. These situations will often prevent this technique from being widely applied to practical application.

This paper presents a novel adaptive sequential niche technique, which can ensure that most extra niche parameters including niche radius are not needed. In this paper, we combine the particle swarm optimization (PSO)

*Corresponding author. Intelligent Computing Lab, Institute of Intelligent Machines, Chinese Academy of Sciences, P.O. Box 1130, Hefei, Anhui 230031, China. Tel.: 86 0551 5591103.
E-mail address: zhangjun@iim.ac.cn (J. Zhang).

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2006.02.016

[6,9] algorithm with our technique to achieve this goal. The proposed algorithm uses multi-sub-swarm to detect multi-optimal solutions sequentially. In addition, the hill valley function proposed in the literature [10] is used in this algorithm to determine how to change the fitness of a particle in the currently running sub-swarm.

This paper is organized as follows. In Section 2, we shall give a brief overview of the hill valley function and PSO algorithm. In Section 3, we present the adaptive sequential niche particle swarm optimization (ASNPSO) algorithm and how it is implemented. Section 4 gives the experimental results for a set of test functions. Section 5 draws some conclusions.

2. Hill valley function and PSO

The determination of the niche radius is generally a hard work existing in most niche methods. However, if we have a method that can determine whether or not two points of search space belong to a peak of the multimodal function, then the niche radius is not needed in this situation. Ursem's hill valley function is the first method proposed in the literature [10], which can be described in Fig. 1, where i_p and i_q are any two points in search space. Fig. 2 just shows one-dimensional (1D) function case. In fact, it can be easily extended to the case including arbitrary dimensions. Generally speaking, the function returns 0 if the fitness of all the interior points is greater than the minimal fitness of i_p and i_q, otherwise it returns 1. With this function, the algorithm is able to determine whether i_p and i_q belong to one hill or not.

The sample array is generally used to calculate the interior points where the hill valley function computes the fitness of these samples. The points i_{interior} can be calculated as

$$i_{\text{interior}} = i_p + (i_q - i_p) \cdot \text{samples}\,[j], \tag{1}$$

where j is jth entry in the array. The upper boundary of j is the length of the samples, which is very important for the hill valley function. We refer to the length as sample rate (SR).

PSO is a population-based stochastic search algorithm. The algorithm was firstly developed by Dr. Eberhart and Dr. Kennedy in 1995 [6], inspired by social behavior of bird flocking or fish schooling. The equations for the manipulation of the swarm can be written as

$$V_{id} = W\,V_{id} + C1\,rand1(*)(P_{id} - X_{id}) + C2\,rand2(*)(P_{gd} - X_{id}), \tag{2}$$

$$X_{id} = X_{id} + V_{id}, \tag{3}$$

where $i = 1,2, \ldots, N$, and W is called as inertia weight. $C1$ and $C2$ are positive constants, referred to as cognitive and social parameters, and $rand1$ (*) and $rand2$ (*) are random numbers, respectively, uniformly distributed in [0..1]. The ith particle of the swarm can be represented by the D dimensional vector X_{id}, and the best particle in the swarm denoted by the index g, the best previous position of the ith particle is recorded and represented as P_{id} and the velocity of the ith particle is as V_{id}.

```
Hill_valley(ip,iq,samples)
minfit=min(fitness(ip),fitness(iq))
for j=1 to samples.length
   Calculate point iinterior on the line between the points ip and iq
   If (minfit> fitness(iinterior))
      return 1
   end if
end for
return 0
```

Fig. 1. Pseudo code of the hill valley function.

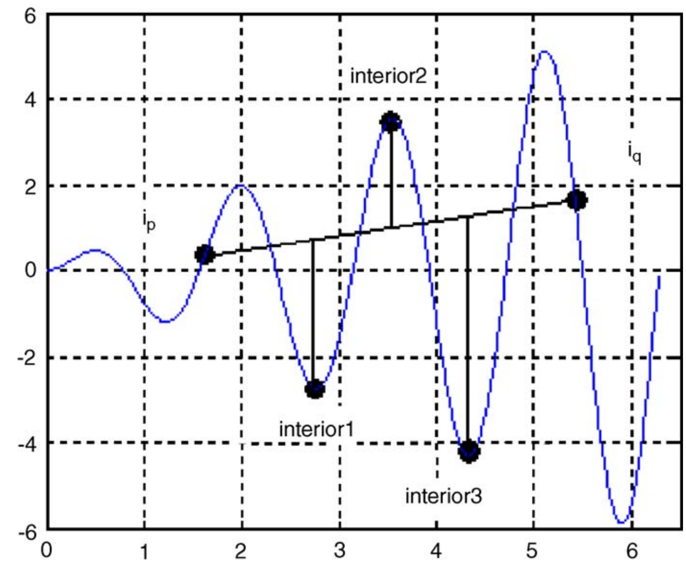


Fig. 2. The scheme for hill valley function.

3. The ASNPSO algorithm

3.1. Basic principles

The adaptive sequential niche technique is essentially an add-on technique, which can be used together with any stochastic search algorithm. Hereby, we choose the PSO algorithm to implement it. ASNPSO consists of several sub-swarms. Each sub-swarm can detect one optimal solution. Because of the algorithm using multi-sub-swarms to detect different solutions sequentially, in order to avoid all sub-swarms converging to one or several certain optimal solutions, the algorithm must be able to modify the fitness function of the particles of the sub-swarm run later. Hereby we introduce the penalty function [11] by means of a constrained optimization problem. Assume that a simple death penalty function is used in our algorithm. Then it can make the particle of the moving sub-swarm locate in the same hill of the optimal solutions found before losing influence. In this paper, the modified fitness function of the

```
Repeat
    Run a new sub-swarm
    N=N+1;
    For k=1 to N-1
      If Hill_valley(x^i,x_k)==0
        Change the fitness of xi
      Else
        Keep the original fitness
      End
    Next
    Train the sub-swarm until the halt window of sub-swarm is larger than a given value
Until N is greater than a given value or reach as a maximum iteration number
```

Fig. 3. Pseudo code of ASNPSO algorithm.

sub-swarm currently running is to satisfy the following equation:

$$eval(x_n^i) = \begin{cases} f(x_n^i) & \text{if } hill\_valley(x_n^i, x_k) = 1, \\ f(x_n^i) - p(x_n^i) & \text{otherwise.} \end{cases} \quad (4)$$

where ${x^i}_n$ is the ith particle of the nth sub-swarm; n is the number of sub-swarm run currently; $f({x^i}_n)$ is a raw fitness function, x_k is the best position found by the sub-swarm launched before, k can be selected from 1 to $n-1$; $p({x^i}_n)$ can be a very large value for real world problem and makes the altered fitness smaller than all local optima.

3.2. Sub-swarm termination conditions

The termination condition for each sub-swarm is an important task for sequential niche technique. In practical applications, if the termination condition is satisfied, we must halt the running PSO and switch to run a new sub-swarm. In this paper, Beasley's [1] halt window technique is adopted as a termination condition. The technique is to record the population average fitness over a halting window of h generations, and terminate the run if the fitness of any generation is not greater than that of the population h generations earlier. In our experiments, assume that the halt window is set to 20.

3.3. Algorithm

The ASNPSO algorithm uses a hill valley function to determine whether the position of a running particle and the best converging position of the swarm launched before belong to one hill or not. The procedure for the algorithm can be described in Fig. 3.

The number of sub-swarms used for the algorithm depends on the number of optimal solutions what we want to get.

4. Experimental results

In this section, there are three-benchmark functions with different complexities chosen to test our algorithm. In experiments, the SR is set as 5, and the sample array is defined as [0.02,0.25,0.5,0.75,0.98]. The halt windows are set as 20. Other experimental parameters are similar to the ordinary PSO algorithm. Assume that the inertia weight of every sub-swarm used in an experiment is set to 0.729, C1 and C2 are set to 1.49445, and $V_{\max}$ is set to the maximum range $X_{\max}$. Shi and Eberhart [4] concluded that this parameter setup can ensure better convergence and have better performance than others. In addition, let the maximum iterative number for each sub-swarm be 10,000. We shall vary the population size of the sub-swarms in our experiments every time to show the efficiency and effectiveness for our proposed algorithm. In particular, for each of the four functions used, assuming that 10 experiments are done with the same population size of the sub-swarms, we shall investigate and evaluate the average number of the optimal solutions found by ASNPSO algorithm in the following experiments.

4.1. One-dimensional Shubert function

This function is just a simple 1D Shubert function [8], which is defined by

$$f1(x) = \sum_{i=1}^{5} i\cos[(i+1)x + i], \quad (5)$$

where $-10 \leqslant x \leqslant 10$. As shown in Fig. 4, this function has 3 global maxima and 16 local maxima. To obtain all 19 solutions of the function in our experiments; the number of sub-swarms run sequentially is set to 19. The red circle represents a solution found by a sub-swarm in the following function demo chart. The experimental results are shown in Fig. 5.

The experimental results show that all the solutions for Shubert 1D function can be easily found by using ASNPSO algorithm, especially when the population size of each sub-swarm is large enough.

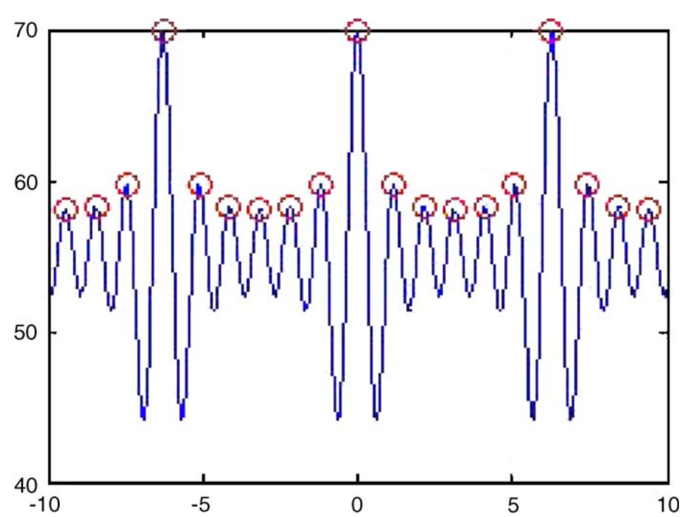


Fig. 4. Shubert 1D function, where 19 sub-swarms were run sequentially.

4.2. Scekel's Foxhole function

Scekel's Foxhole function was first introduced in De Jong's dissertation [2]. It is a two-dimensional (2D) function with 25 peaks of different heights. Let $a(i) = 16[(i \bmod 5) - 2]$ and $b(i) = 16(\lfloor i/5 \rfloor - 2)$, the function can be defined as

$$f3(x, y) = 500 - \frac{1}{0.002 + \sum_{i=0}^{24} 1/(1 + i + (x - a(i))^6 + (y - b(i))^6)}, \quad (6)$$

where $-65.536 \leqslant x, y \leqslant 65.536$. Fig. 6 shows the function, and Fig. 7 shows the experimental results.

Obviously, this function is more complex than the above one. With the population size of each sub-swarm increasing, however, all 25 optimal solutions can be found by our algorithm.

4.3. Two-dimensional Shubert function

This is a very interesting function [8], which can be defined as

$$f4(x1, x2) = \prod_{i=1}^{2} \sum_{j=1}^{5} j \cos[(j + 1)xi + j], \quad (7)$$

where $-10 \leqslant xi \leqslant 10$ for $i = 1, 2$. It has 760 local minima, 18 of which are global minima with function value of -186.73. Moreover, the global optima are unevenly distributed. In our experiments, we converted the minima to maxima through multiplying by a negative sign the raw fitness function.

Fig. 8 shows the converted function and Fig. 9 shows the experimental results.

When the number of the sub-swarms sequentially run is equal to the solution number, the experimental results show that the algorithm cannot find all 18 optimal solutions. Fig. 10 shows that it is not enough to solve the

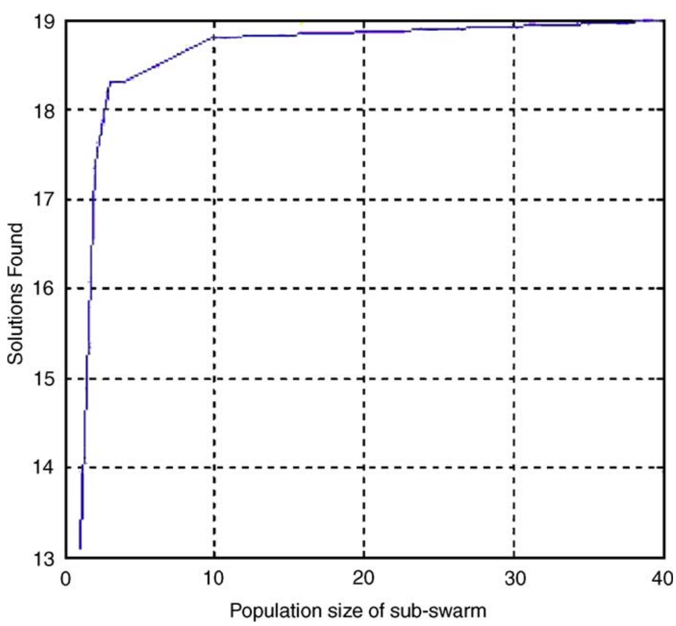


Fig. 5. The average results for 10 runs of experiment.

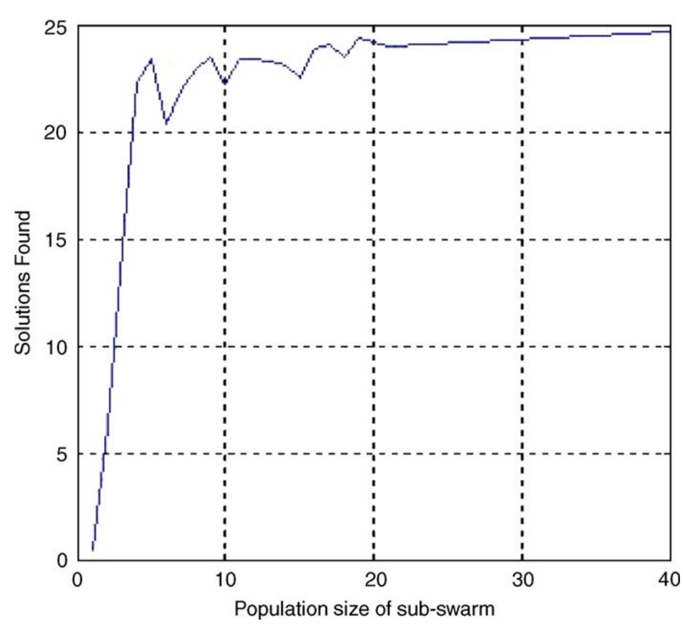


Fig. 7. The average results for 10 runs of experiments.

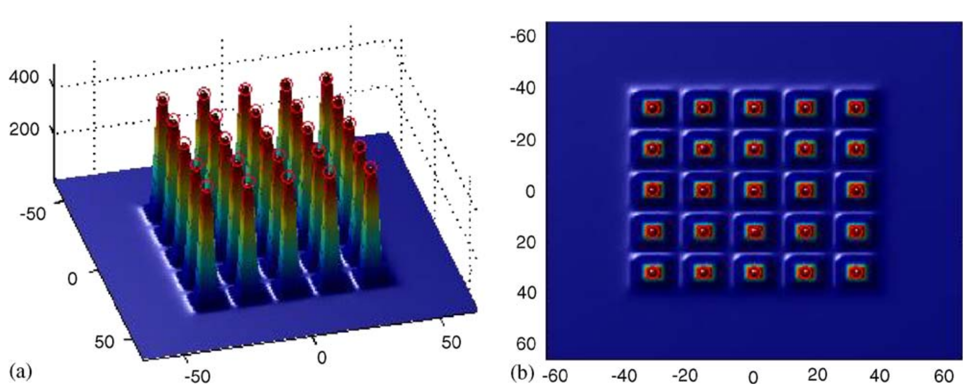


Fig. 6. Scekel's Foxhole function where 25 sub-swarms were run sequentially: (a) is 3D graph; (b) is contour graph.

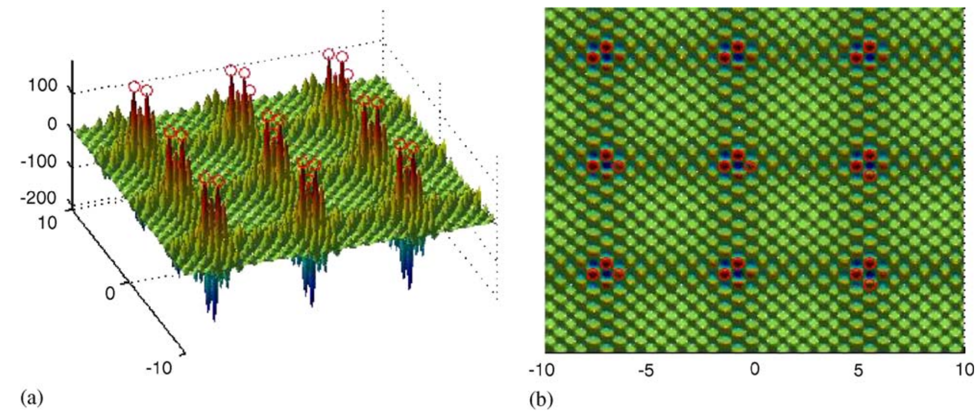


Fig. 8. 2D Shubert function where 23 sub-swarms were run sequentially: (a) is 3D graph, (b) is contour graph.

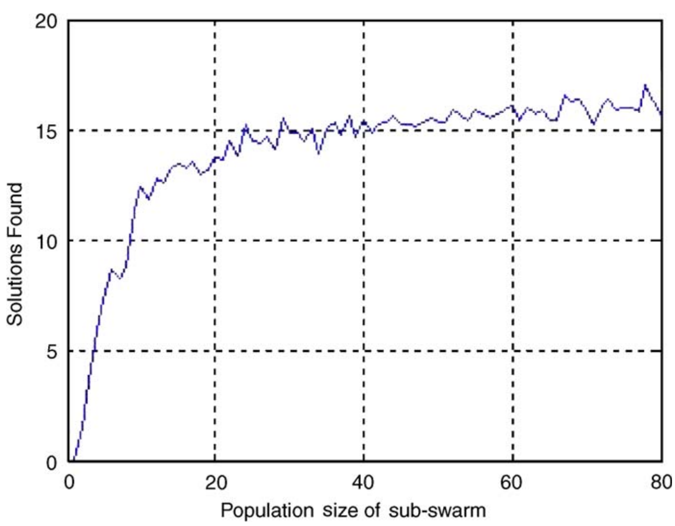


Fig. 9. The average results for 10 runs of experiments.

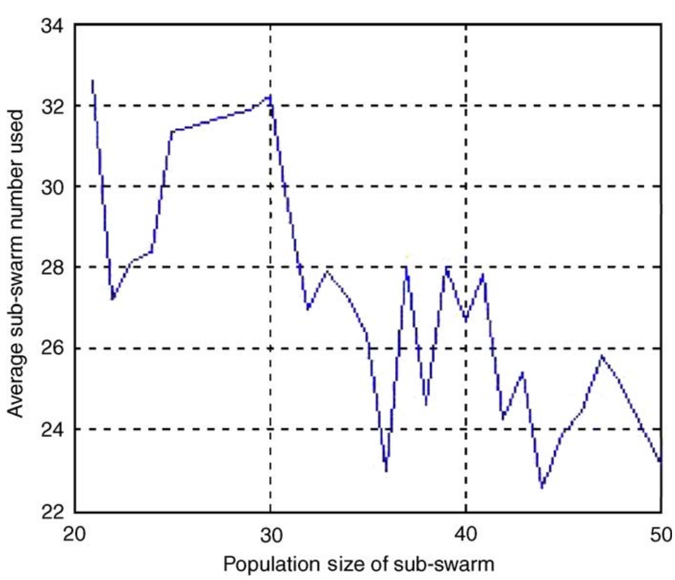


Fig. 10. The average results for 10 runs of experiments where every time 18 optimal solutions were found.

problem by simply increasing the population size of the sub-swarms for a complex function. In fact, the algorithm can use more sub-swarms to solve the problem. So, another experiment using more sub-swarms is conducted, the algorithm was run until it found all 18 optimal solutions. In these experiments, 20–50 particles were used in each sub-swarm. Assume that the algorithm was run 10 times for every population size. The experimental results are shown in Fig. 9.

In Fig. 10, it can be seen that with the population size of each sub-swarm increasing, the average sub-swarm number used in the algorithm also decreases.

In all experiments conducted above, if we set the number of sub-swarms too few for a test function, the algorithm cannot find all optima. In contrast, the number of sub-swarms set is too many, then there is no notable impact on the experimental results. The experimental results show that the number of the sub-swarms must be set larger than the number of optima of a complex function.

5. Conclusions

In this paper, we proposed a novel adaptive sequential niche algorithm for multimodal function optimization. The algorithm uses hill valley function to determine whether two particles of search space belong to one hill. If a particle and the best solution found before are to locate in the same hill, through changing the particle's fitness, the algorithm will bring the particle that lost its influence into a sub-swarm. By these means, this algorithm does not need the niche radius, and at the same time, no prior knowledge is required. The experimental results showed that the proposed algorithm is very efficient and effective since all optimal solutions for multimodal function can be found sequentially.

Acknowledgment

This work was supported by the National Science Foundation of China (Nos. 60472111 and 30570368).

References

[1] D. Beasley, D.R. Bull, R.R. Martin, A sequential niching technique for multimodal function optimization, Evol. Comput. 1 (2) (1993) 101–125.

[2] K.A. De Jong, An analysis of the behavior of a class of genetic adaptive systems, Ph.D. Thesis, Department of Computer Science, University of Michigan, Ann Arbor, MI, USA, 1975.

[3] R.C. Eberhart, X. Hu, Human tremor analysis using particle swarm optimization, in: Proceedings of the Congress on Evolutionary Computation (CEC-99), Washington, DC, USA, 1999, pp. 1927–1930.

[4] R.C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in: Proceedings of the 2000 World Congress on Evolutionary Computation, San Diego, Southern CA, 2000, pp. 84–88.

[5] L. Guo, D.S. Huang, W.B. Zhao, Combining genetic optimisation with hybrid learning algorithm for radial basis function neural networks, IEE Electron. Lett. 39 (22) (2003) 1600–1601.

[6] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proceedings Of IEEE International Conference on Neural Networks (ICNN), vol. 4, Perth, Australia, 1995, pp. 1942–1948.

[7] S.W. Mahfoud, A comparison of parallel and sequential niching methods, in: Proceedings of the Sixth International Conference on Genetic Algorithms, San Mateo, California, 1995, pp. 136–143.

[8] Z. Michalewicz, Genetic Algorithm + Data Structures = Evolution Programs, Springer, New York, 1996.

[9] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proceedings of the 1998 IEEE International Conference on Evolutionary Computation, Piscataway, Anchorage, IEEE Press, Alaska, USA, NJ, USA, 1998, pp. 69–73.

[10] R.K. Ursem, Multinational evolutionary algorithms, in: Proceedings of Congress of Evolutionary Computation (CEC-99), vol. 3, Washington, DC, USA, 1999, pp. 1633–1640.

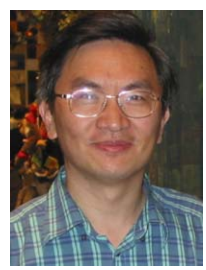
[11] Ö. Yeniay, Penalty function methods for constrained optimization with genetic algorithms, Math. Comput. Appl. 10 (1) (2005) 45–56.

[12] D. Zaharie, A multipopulation differential evolution algorithm for multimodal optimization, in: Proceedings of Mendel 2004, 10th International Conference on Soft Computing, Brno, Czechoslovakia, 2004, pp. 17–22.

Jun Zhang was born in Anhui province, China, in 1971. He received M.Sc. degree in Parttern Recognition & Intelligent System in 2004, from Institute of Intelligent Machines, Chinese Academy of Sciences. He is now in pursuit for Ph.D. degree in Pattern Recognition & Intelligent System in University of Science and Technology of China. His research interests include swarm intelligence, artificial neural networks, intelligent computing, and intelligent information processing.

De-Shuang Huang (SM'98) Professor and Ph.D. Advisor in the University of Science and Technology of China (USTC), Hefei, China, and Professor of Graduate School of the Institute of Intelligent Machines, Chinese Academy of Sciences (CAS). From Sept. 2000 to Mar. 2001, he worked as Research Associate in Hong Kong Polytechnic University. From Apr. 2002 to June 2003, he worked as Research Fellow in City University of Hong Kong. From Oct. to Dec. 2003, he worked as Research Fellow in Hong Kong Polytechnic University. From July to Dec. 2004, he worked as the University Fellow in Hong Kong Baptist University. Dr. Huang is currently a senior member of the IEEE.

Tat M. Lok received the B.Sc. degree in electronic engineering from the Chinese University of Hong Kong, and the M.Sc degree and the Ph.D. degree in Electrical Engineering from Purdue University. In 1996, he joined the Chinese University of Hong Kong, where he is currently an Associate Professor. His research interests include communication theory, signal processing for communications and CDMA systems.



Michael R. Lyu received his B.Sc. in Electrical Engineering from National Taiwan University in 1981, his M.Sc. in Computer Science from University of California, Santa Barbara, in 1985, and his Ph.D. in Computer Science from University of California, Los Angeles, in 1988. He is currently a Professor in the Computer Science and Engineering department of the Chinese University of Hong Kong. Dr. Lyu's research interests include software reliability engineering, distributed systems, fault-tolerant computing, web technologies, mobile networks, digital video library, multimedia processing, and video searching and delivery.