A Study of Different Transfer Functions for Binary Version of Particle Swarm Optimization

Article · April 2012

CITATIONS
11

READS
203

5 authors, including:

Seyedeh Zahra Mirjalili
Torrens University Australia Sydney campus
21 PUBLICATIONS **3,163** CITATIONS

SEE PROFILE

A Study of Different Transfer Functions for Binary Version of Particle Swarm Optimization

S. Mirjalili[1], S.Z. Mohd Hashim[1], G. Taherzadeh[2], S.Z. Mirjalili[3], and S. Salehi[1]
[1]Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia, 81300 Skudai, Johor Bahru, Malaysia
[2]Faculty of Information Technology, Multimedia University, Selangor, Malaysia
[3]Faculty of Mathematics, Sharif University of Technology, Tehran, Iran

Abstract - *Particle Swarm Optimization (PSO) is one of the most widely used heuristic algorithms. The simplicity and inexpensive computational cost make this algorithm very popular and powerful in solving wide ranges of problems. However, PSO suffers two problems of trapping in local minima and slow convergence speed. Binary version of this algorithm has been introduced for solving binary problems. Because BPSO uses the same concepts of PSO, it also undergoes the same problems. The main part of the binary version is the transfer function. There is not enough study in the literature focusing on the transfer function. In this study, eight new transfer functions dividing into two families (s-shape and v-shape) for binary particle swarm optimization are introduced and evaluated. Four benchmark optimization problems are employed in order to evaluate these transfer functions in terms of avoiding local minima, convergence speed, and accuracy of results. The results prove that the new introduced v-shape family of transfer functions could improve the performance of original binary PSO based on the above-mentioned drawbacks.*

Keywords: Optimization Algorithm (MOA), Magnetic field theory; Function optimization; Transfer function

1 Introduction

Particle Swarm Optimization (PSO) is one of the most widely used evolutionary algorithms inspired from social behavior of animals [1,2]. The simplicity and inexpensive computational cost make this algorithm very popular. Due to above-mentioned advantages, PSO has been applied to many domains such as medical detecting [3], grid scheduling [4], robot path planning [5], and video abstraction [6]. In spite of these advantages, like all other population-based algorithm, trapping in local minima and slow convergence rate are two unavoidable problems for PSO. With the increase of problems' dimension, these two problems become more complex. PSO is capable of solving problems which have continuous search space. However, some problems have different search spaces.

There are many optimization problems, which have discrete binary search spaces. They need binary algorithms to be solved. Binary version of PSO was proposed by Kennedy and Eberhart in 1997 [7]. Like PSO, binary version of PSO has the problems of trapping in local minima and slow convergence speed because of using the same concepts for solving problems [8]. The only different element between these two algorithms is transfer function that is used to map continuous search space to the binary one. Transfer function is the most important part of binary PSO [9]. In the literature, there is not enough study about the transfer function. In This study, eight different transfer functions for binary version of PSO are introduced and evaluated. The effectiveness of employing these new transfer functions are investigated in terms of avoiding local minima, convergence speed, and accuracy of results.

The rest of the paper is organized as follow. Section II presents a brief introduction to PSO. Section III discusses the basic principles of binary version of PSO. The experimental results are demonstrated in section IV. Finally, section V concludes the work and suggests some researches for future works.

2 The Particle Swarm Optimization

PSO is an evolutionary computation technique which is proposed by Kennedy and Eberhart [10,11]. The PSO was inspired from social behavior of bird flocking. It uses a number of particles (candidate solutions) which fly around in the search space to find best solution. Meanwhile, they all look at the best particle (best solution) in their paths. In other words, particles consider their own best solutions as well as the best solution has found so far.

Each particle in PSO should consider the current position, the current velocity, the distance to *pbest*, and the distance to *gbest* to modify its position. PSO was mathematically modeled as follow:

$$v_i^{t+1} = wv_i^t + c_1 \times rand \times (pbest_i - x_i^t) + c_2 \times rand \times (gbest - x_i^t) \quad (1)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (2)$$

where v_i^t is the velocity of particle i at iteration t, w is a weighting function, c_j is a weighting factor, rand is a random number between 0 and 1, x_i^t is the current position of particle i at iteration t, $pbest_i$ is the *pbest* of agent i at iteration t, and *gbest* is the best solution so far.

The first part of (1), wv_i^t, provides exploration ability for PSO. The second and third parts, $c_1 \times rand \times (pbest_i - x_i^t)$ and $c_2 \times rand \times (gbest - x_i^t)$, represent private thinking and collaboration of particles respectively. The PSO starts with randomly placing the particles in a problem space. In each iteration, the velocities of particles are calculated using (1). After defining the velocities, the position of masses can be calculated as (2). The process of changing particles' position will continue until meeting an end criterion.

3 Binary Version of Particle Swarm Optimization

Generally, there are many problems which have intrinsic discrete binary search space like feature selection and dimensionality reduction [12,13]. In addition, the problems with continuous real search space can be converted into binary problems. However, a binary search space has its own structure with some limitations.

A binary search space can be considered as a hypercube. The agents of a binary optimization algorithm can only shift to nearer and farther corners of the hypercube by flipping various numbers of bits [7]. Hence, for designing binary version of PSO, some basic concepts such as velocity and position updating process had been modified.

In the original PSO, particles can move around the search space because of having position vectors with continuous real domain. Consequently, the concept of position updating can be easily implemented for particles adding velocities to positions using (2). However, the meaning of position updating is different in a discrete binary space. In binary space, due to dealing with only two numbers ("0" and "1"), the position updating process cannot be done using (2). Therefore, we have to find a way to use velocities to change agents' positions from "0" to "1" or vice versa. In other words, we have to find a link between velocity and position, as well as revise (2).

Basically, in discrete binary space, the position updating means a swithcing between "0" and "1" values. This switching should be done based on velocities of agents. The question here is that how the concept of velocity in real space should be employed in order to update positions in binary space. According to [7,14,15], the idea is to change position of an agent with the probability of its velocity. In order to do this, a transfer function is needed to map the velocities values to probability values for updating the positions.

As mentioned above, transfer functions define the probability of changing position vector's elements from "0" to "1" and vice versa. Transfer functions force agents to move in a binary space. According to [14], some concepts should be taken into account for selecting a transfer function in order to map velocity values to probability values.

The transfer function should be able to provide a high probability of changing the position for a large absolute value of the velocity. It should also present a small probability of changing the position for a small absolute value of the velocity. Moreover, the range of a transfer function should be bounded in the interval [0,1] and increased with the increasing of velocity. The function that have been used in [7] are presented as (3). This function is also depicted in Fig.1.

$$S\left(v_{i,j}^k(t)\right) = \frac{1}{1+e^{-v_{i,j}^k(t)}} \tag{3}$$

The above-mentioned transfer function and the new introduced transfer functions in this work are listed in Table I. These transfer functions are also visualized in Fig.1 and Fig.2. We call the first and second groups s-shape and v-shape family transfer functions respectively. According to [7], for the transfer function in Fig.1, we use (4) in order to update position vectors. According to [14], for the transfer function in Fig.2, we use (5) to update position vectors based on velocities. It should be noticed that these transfer functions satisfy all aforementioned concepts.

$$x_{i,j}^k(t+1) = \begin{cases} 0 & \text{If } rand < S\left(v_{i,j}^k(t+1)\right) \\ 1 & \text{If } rand \geq S\left(v_{i,j}^k(t+1)\right) \end{cases} \tag{4}$$

$$x_{i,j}^k(t+1) = \begin{cases} complement\left(x_{i,j}^k(t)\right) & \text{If } rand < S\left(v_{i,j}^k(t+1)\right) \\ x_{i,j}^k(t) & \text{If } rand \geq S\left(v_{i,j}^k(t+1)\right) \end{cases} \tag{5}$$

TABLE I. TRANSFER FUNCTIONS

No	Transfer Functions
1	$S(x) = \frac{1}{1+e^{-2x}}$
2 [7]	$S(x) = \frac{1}{1+e^{-x}}$
3	$S(x) = \frac{1}{1+e^{\frac{-x}{2}}}$
4	$S(x) = \frac{1}{1+e^{\frac{-x}{3}}}$
5	$S(x) = \left\lvert \text{erf}\left(\frac{\sqrt{\pi}}{2}x\right)\right\rvert = \left\lvert \frac{\sqrt{2}}{\pi}\int_0^{\frac{\sqrt{\pi}}{2}x} e^{-t^2}\,dt \right\rvert$
6 [14]	$S(x) = \lvert \tanh(x) \rvert$
7	$S(x) = \left\lvert \frac{x}{\sqrt{1+x^2}} \right\rvert$
8	$S(x) = \left\lvert \frac{2}{\pi}\arctan\left(\frac{\pi}{2}x\right) \right\rvert$

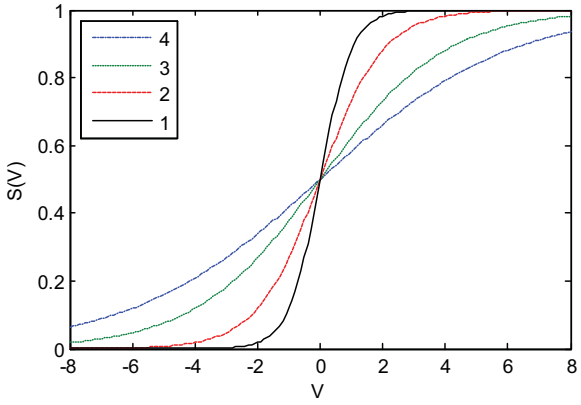

Figure 1. S-shape family transfer functions

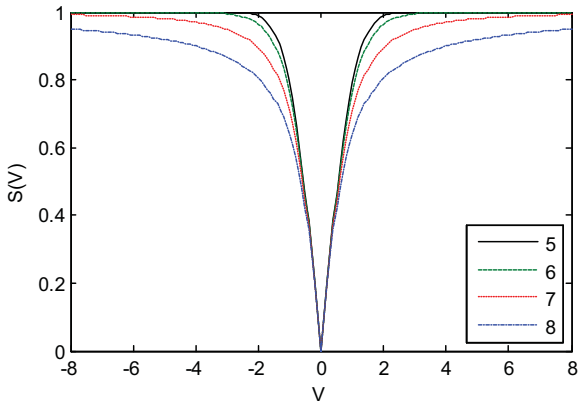

Figure 2. V-shape family transfer functions

The general steps of Binary PSO are as follows.

a) All particles are initialized with random values
b) Repeat steps c-e until the meeting of the end condition
c) For all particles, velocities are defined using (1)
d) Calculate probabilities for changing elements of position vectors based on transfer function's formula.
e) Update the elements of position vectors based on the rules in (4) or (5) based on the type of transfer function

4 Experimental results and discussion

In order to evaluate the performance of new binary versions of PSO called BPSO with different transfer functions, 4 standard benchmark functions are employed [16]. Table II lists down these benchmark functions and the range of their search spaces. Fig.3, Fig.4, Fig.5, and Fig.6 illustrate them, Spherical, Rastrigin, Rosenbrock, and Griewank functions, respectively. Furthermore, function's dimension is set to 5 (m=5). To represent each continuous variable, 15 bits are used. It should be noticed that one bit is reserved for the sign of each functions' dimension. Therefore, the dimension of agents are 75 (Dim=m×15).

TABLE II. BENCHMARK FUNCTIONS

Function	Range
$F_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100,100]^m$
$F_2(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30,30]^m$
$F_3(x) = \sum_{i=1}^{n} [x_i^2 - 10cos(2\pi x_i) + 10]$	$[-5.12,5.12]^m$
$F_4(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600,600]^m$

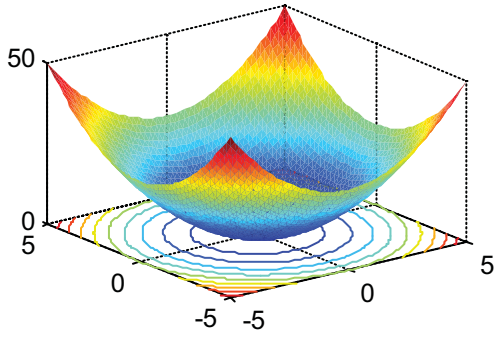

Figure 3. Spherical function (F1)

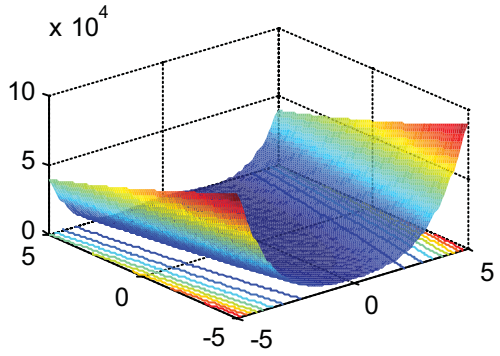

Figure 4. Rastrigin function (F2)

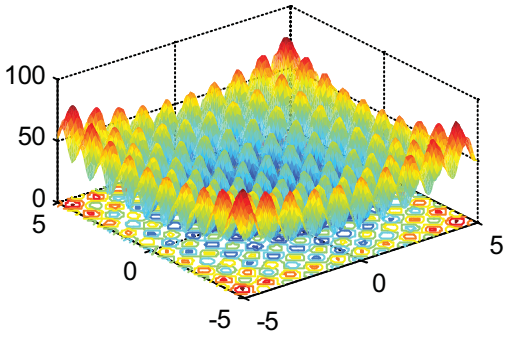

Figure 5. Rosenbrock function (F3)

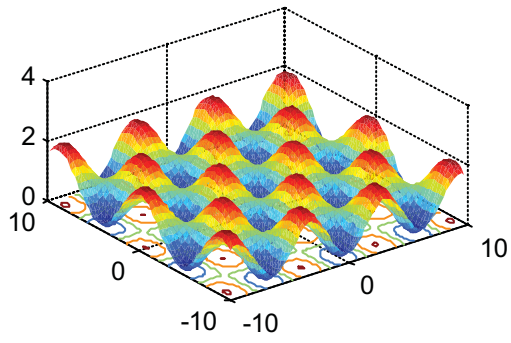

Figure 6. Griewank function (F4)

In this paper, our objective is minimization. The global minimum values for all appeared functions in table I are 0. The number of particles is 30, C_1 and C_2 are set to 2, W is linearly decreased from 0.9 to 0.4, maximum velocity is set to 6, maximum iteration is set to 500, and the stopping criteria is the meeting of maximum number of iteration. According to [14], to achieve a good convergence rate, the velocity should be limited. So, the maximum velocities for all versions of BPSO in this work are set to 6.

The experimental results are presented in Table III. The results are averaged over 30 independent runs, and the best results are indicated in bold type.

TABLE III. MINIMIZATION RESULTS OF 4 BENCHMARK FUNCTIONS OVER 30 INDEPENDENT RUNS

F	Algorithm	ABSF [a]	STDV [b]	MBSF [c]	Best [d]
F1	*BPSO1*	0.4921	0.4165	0.3822	0.0492
	BPSO2	5.2965	2.7657	4.6684	1.7044
	BPSO3	33.3306	17.0770	30.58	2.5258
	BPSO4	71.0918	37.9921	59.313	18.2828
	BPSO5	0.7844	2.9905	0.0044	**0**
	BPSO6	0.3514	0.8997	0.0179	**0**
	BPSO7	0.2663	0.7691	**0.0063**	**0**
	BPSO8	**0.0977**	**0.2725**	0.0064	**0**
F2	*BPSO1*	20.2467	26.1441	8.366	2.0515
	BPSO2	65.3304	79.3444	39.838	5.8852
	BPSO3	437.3886	347.783	310.043	68.5187
	BPSO4	744.8748	623.8992	612.7769	57.498
	BPSO5	26.7199	45.3441	8.5981	1.5156
	BPSO6	21.1831	34.2671	6.7977	07392
	BPSO7	14.8538	33.188	3.2592	0.8658
	BPSO8	**7.3941**	**19.4263**	**2.9353**	**0.1093**
F3	*BPSO1*	2.2054	1.0149	2.2792	0.2207
	BPSO2	4.1528	1.6081	3.6184	2.23
	BPSO3	8.2714	3.3343	8.6903	1.7784
	BPSO4	12.4642	3.2046	12.6385	6.1475
	BPSO5	1.9923	0.8618	1.9932	0.995
	BPSO6	2.0495	0.8105	1.995	0.995
	BPSO7	1.8899	1.2806	**1.7407**	**0.0242**
	BPSO8	**1.6945**	**0.7212**	1.9903	0.9952
F4	*BPSO1*	0.2598	0.0868	0.2776	0.0558
	BPSO2	0.3985	0.1088	0.3753	0.1961
	BPSO3	0.6403	0.1612	0.6409	0.2682
	BPSO4	0.7989	0.1465	0.8471	0.4665
	BPSO5	0.1684	0.1068	0.161432	0.0272
	BPSO6	0.1762	0.1016	0.1623	0.0144
	BPSO7	0.1351	0.0904	0.1154	0.0330
	BPSO8	**0.1032**	**0.0655**	**0.0958**	**0.0154**

a. Indicates average best so far solution over 30 runs in the last iteration
b. Indicates standard deviation of the best so far solution over 30 runs in the last iteration
c. Indicates median best so far solution over 30 runs in the last iteration
c. Indicates the best solution over 30 runs in all iteration

For functions F1 and F2, BPSO8 reaches better results than other algorithms for ABSF and STDV variables in Table II. The functions F1 and F2 belong to family of unimodal functions which are monotonous functions without any local solution. As shown in the Fig.3 and Fig.4, there is only one global solution for these kinds of functions. Hence, the results of the aforementioned statistical variables show BPSO8 improve the ability of exploiting the global minima in original BPSO with its new transfer function.

The results for Best variable in Table III prove that BPSO8 also owns best result accuracy among the other algorithm. Moreover, Fig.7 and Fig.8 prove that BPSO8 possesses good convergence rate in the last iterations.

According to Table III, for functions F3 and F4, BPSO8 outperform other algorithms in all statistical variables. Functions F3 and F4 are multimodal functions that have many local solutions in comparison with unimodal functions. Hence, it can be said that BPSO8 with its new transfer function could enhance the ability of original BPSO to avoid local minima.

The results of Best variable in Table III for BPSO8 also insist on having more accurate results than the other algorithms. Fig.9 and Fig.10 prove that BPSO8 has better convergence speed in multimodal functions for the last iterations as well.

To summarize, results prove that family of s-shape transfer functions with their method of updating position are not suitable for binary version of PSO. In contrary, the new introduced v-shape family of transfer functions with their special method of updating positions is useful for binary version of PSO in terms of avoiding local minima, convergence speed, and accuracy of results. It can be concluded that the new introduced family of transfer function has merit to use in binary algorithms.

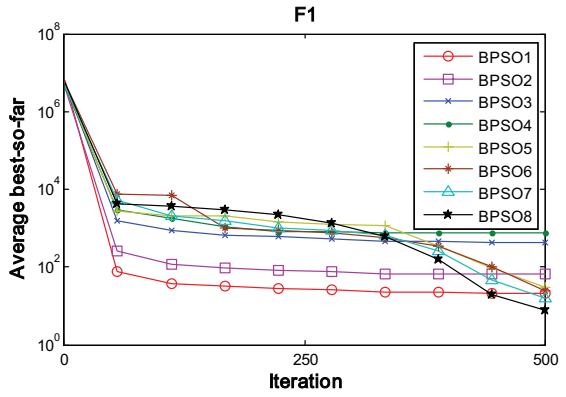

Figure 7. Comparison between BPSOs with diffetent transfer functions on fuction F1

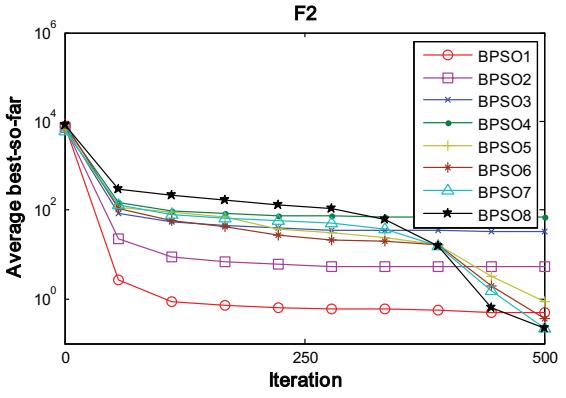

Figure 8. Comparison between BPSOs with diffetent transfer functions on fuction F2

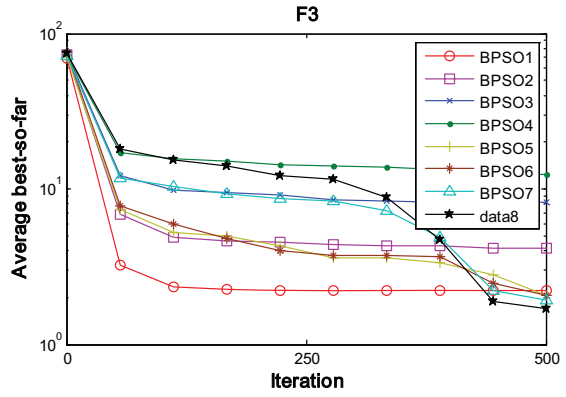

Figure 9. Comparison between BPSOs with diffetent transfer functions on fuction F3

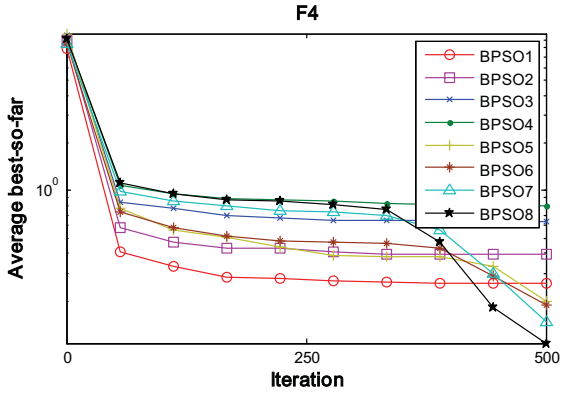

Figure 10. Comparison between BPSOs with diffetent transfer functions on fuction F4

5 Conclution

In this paper, some new versions of binary PSO are introduced utilizing different new transfer functions. Eight new transfer functions dividing into two families (s-shape and v-shape) are introduced and evaluated. In order to justify the performance of all versions, four benchmark functions are employed, and the results are compared together. The results prove that the new introduced v-shape family of transfer functions with their own method of updating positions vector can improve the performance of original binary version of PSO in terms of avoiding local minima, convergence rate, and results' accuracy. The results also show that new introduced v-shape family of transfer function has merit for binary algorithms.

For future studies, it is recommended to use the new introduced family transfer function for the other binary algorithms like Binary Gravitational Search Algorithm.

6 References

[1] R.C. Eberhart and J. Kennedy, "A new optimizer using particles swarm theory," in *Sixth Int. Symp. on Micro Machine and Human Science*, Nagoya, Japan, 1995, pp. 39–43.

[2] R.C. Eberhart and J. Kennedy, "Particle swarm optimization," in *IEEE Int. Conf. on Neural Network*, Perth, Australia, 1995, pp. 1942–1948.

[3] S. Chandra, R. Bhat, and H. Singh, "A PSO based method for detection of brain tumors from MRI," in *Nature & Biologically Inspired Computing*, Coimbatore, 2009, pp. 666 - 671.

[4] MP. Mathiyalagan, UR. Dhepthie, and SN. Sivanandam, "Grid Scheduling Using Enhanced PSO Algorithm," vol. 2, no. 2, pp. 140-145, 2010.

[5] E. Masehian and D. Sedighizadeh, "A multi-objective PSO-based algorithm for robot path planning," in *IEEE International Conference on Industrial Technology* , Vi a del Mar, 2010, pp. 465 - 470.

[6] MB. Fayk, HA. El Nemr, and MM. Moussa, "Particle swarm optimisation based video abstraction," *Journal of Advanced Research*, vol. 1, no. 2, pp. 163-167, 2010.

[7] J. Kennedy and R.C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *IEEE International Conference on Computational Cybernetics and Simulation*, vol. 5, 1997, pp. 4104 - 4108.

[8] K. Premalatha and A.M. Natarajan, "Hybrid PSO and GA for Global Maximization ," *Int. J. Open Problems Compt. Math.*, vol. 2, no. 4, pp. 597-608, 2009.

[9] Y. Xiaohui, Y. Yanbin, W. Cheng, and Z. Xiaopan, "An Improved PSO Approach for Profit-based Unit Commitment in Electricity Market," in *Transmission and Distribution Conference and Exhibition*, 2005, pp. 1-4.

[10] J. Kennedy and RC. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE international conference on neural networks*, vol. 4, 1995, pp. 1942–1948.

[11] Y. Shi and R.C. Eberhart, "A modified Particle SwarmOptimiser," in *IEEE International Conference on Evolutionary Computation, Anchorage*, Alaska, 1998.

[12] P. Avishek and J. Maitib, "Development of a hybrid methodology for dimensionality reduction in Mahalanobis-Taguchi system using Mahalanobis distance and binary particle swarm optimization," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1286-1293, March 2010.

[13] X. Zenga, Y. Li, and J. Qina, "A dynamic chain-like agent genetic algorithm for global numerical optimization and feature selection," *Neurocomputing*, vol. 72, no. 4-6, pp. 1214-1228, January 2009.

[14] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "BGSA: binary gravitational search algorithm," *Natural Computing*, vol. 9, no. 3, pp. 727-745, 2009.

[15] S. Mirjalili and S. Z. Mohd Hashim, "BMOA: Binary Magnetic Optimization Algorithm," in *2011 3rd International Conference on Machine Learning and Computing (ICMLC 2011)*, vol. 1, Singapore, 2011, pp. 201-206.

[16] Xin Yao, Yong Liu, and Guangming Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82 - 102, 1999.