RANK-BASED SELF-ADAPTIVE INERTIA WEIGHT SCHEME TO ENHANCE THE PERFORMANCE OF NOVEL BINARY PARTICLE SWARM OPTIMIZATION

Faryal Amin,
Yasir Mehmood*,
Marium Sadiq,
Samina Khalid,
Iftikhar Ahmad

Department of CS&IT,
Mirpur University of Science and Technology (MUST),
Mirpur, AJK, Pakistan

Abstract:
Inertia weight is a significant parameter of the particle swarm optimization (PSO) algorithm. Its controllers the search capabilities of PSO and provides a balance between exploration and exploitation. There are a plethora of studies on inertia weight variants of continuous PSO (CPSO). However, a few numbers of studies have been presented for binary PSO (BPSO). In existing BPSO variants, despite different positions of particles, every individual is treated equally by ignoring the dispersion of particles in the search space. To deal with each particle according to its fitness value, we have proposed a Rank-based Self-adaptive Inertia Weight to enhance the performance of the Novel BPSO (NBPSO). The proposed algorithm controls the movement of particles by defining the ranks of each particle based on their fitness. The performance of the proposed algorithm is evaluated on four benchmark test functions. The experimental results show that the proposed method performs better than the compared algorithms in terms of convergence speed.

Keywords:
PSO, fitness rank, self-adaptive, inertia weight, convergence speed.

INTRODUCTION

The continuous particle swarm optimization (CPSO) [1] is a nature-inspired algorithm proposed by Kennedy and Eberhart in 1995. The algorithm is motivated by the social behavior of bird's flock and fish schooling. Its quick convergence, simple implementation, and non-complex computations have made it a widely accepted algorithm to solve many real-world optimization problems. This basic version of CPSO is utilized for the real number spaces and continuous problems [2].

To address binary optimization problems, Kennedy and Eberhart developed the Binary PSO (BPSO) in 1997 [3]. In BPSO each particle changes its position by either selecting 0 or 1. To enhance the performance of BPSO, several improved variants have been proposed. Khanesar et al. [2] proposed the NBPSO by presenting a new definition of velocity vector that is the rate of changing particle bits. The NBPSO also addressed the issue of selecting a proper value of inertia weights introduced in [4].

Correspondence:
Yasir Mehmood

e-mail:
yasir.mehmood@must.edu.pk

DOI: 10.15308/Sinteza-2021-63-69

Inertia weight provides the balance between exploration and exploitation capabilities of CPSO [5]. In literature, various inertia weight schemes have been presented. These schemes are classified into three classes [6]. First is a constant [4] or random inertia weight [7], in which the value of inertia weight is constant or random.

The second class is time-varying inertia weight, in which the value of inertia weight changes in every iteration step. These include linearly decreasing inertia weight [8, 9], non-linearly decreasing inertia weight [10], and Chaotic inertia weight [11]. The third class is adaptive inertia weight, which uses feedback parameters to set the value of inertia weight and monitor the algorithm's state. This class includes adaptive inertia weight [12], dynamic adaptive inertia weight [7], and rank-based inertia weight [13].

In this paper, we have proposed a rank-based self-adaptive inertia weight scheme to enhance the performance of NBPSO [2]. In the proposed scheme, an adaptive inertia weight strategy [13, 14] is incorporated to enhance the convergence speed. The velocity of each particle is directly controlled by their fitness such that the particle with high fitness gets the high rank and the particle with low fitness gets the lower rank. The movement of each particle is controlled directly by its fitness so that the particle with a low rank moves with high velocity. The proposed RIW-NBPSO enhanced the performance of NBPSO in terms of convergence speed.

The paper is structured as follows: Section II includes a Binary PSO and its variants. Section III presents the Novel BPSO. The proposed algorithm is presented in IV and in section V simulation results are presented. Finally, the paper is concluded in section VI.

2. THE BINARY PSO (BPSO) AND ITS VARIANTS

In contrast with CPSO, each particle in BPSO is represented with a bit string. A particle decides to pick a value of either 0 or 1. The particle updates the position by shifting values between 0 and 1. A particle's velocity is the probability change of taking 0 or 1, so the velocity of a particle must be bounded within the range [0, 1]. The sigmoid transfer function (sig) is used to represent and bound all the real number velocities within the range of [0 1] as:

$$v_{ik} = sig\left(v_{ik}(t)\right) = \frac{1}{1+e^{-vik(t)}} \tag{1}$$

Based on the above $sig(v_{ik}(t))$, the new position will be computed as:

$$x_{ik}(t+1) = \begin{cases} 1, if\ rand_{ik} < sg\left(v_{ik}(t+1)\right) \\ 0, otherwise \end{cases} \tag{2}$$

where $rand_{ik}$ is a random number between [0, 1]. In standard BPSO velocity update is based on the following rule:

$$v_{ik} = wv_{ik} + c_1 r_{1_k}\left(p_{ik} - x_{ik}\right) + c_2 r_{2_k}\left(g_k - x_{ik}\right) \tag{3}$$

where w is the inertia weight: $0 < w < 1$. p_{ik} is the kth bit of ith particle's personal best and gk is the kth bit of the best particle among all of the particles (global best position). c1 and c2 are the acceleration coefficients such that $c1, c2 > 0$, r1k, and r2k are random uniform distributions within 0 and 1.

Several BPSO variants have been proposed to achieve better performance in solving various problems. Beheshti et al. [15] improved BPSO to Memetic BPSO (MBPSO) depends on the hybridization of global and local topologies in PSO. Chaung et al. [16] proposed a Chaotic BPSO (CBPSO) by embedded chaotic maps in BPSO to solve feature selection problems. Khanesar et al. [2] proposed Novel BPSO by representing a new definition of the velocity vector and also addressed the issue of selecting a value of inertia weight in existing BPSO. A quantum computing-inspired BPSO (QBPSO) was proposed by Jeong et al. [17] that addressed the premature convergence of original BPSO and applied it in unit commitment problems for power systems. A modification was made by Afshinmanesh et al. [18] in BPSO based on the negative selection in the Artificial Immune system. Liao et al. [19] extended the basic discrete PSO to solve flow shop scheduling problems by redefining the particles and their velocities. An improved BPSO was proposed by El-Maleh et al. [20] that overcome the drawbacks of the original BPSO and solved the issues of state assignment in sequential circuit synthesis targeting area optimization. Wei and Jing [21] presented a Novel BPSO to solve the heliotype inference problem. A new modified BPSO for solving knap-sack problems [22] is proposed. A new probability function is inserted that maintains the diversity of the swarm. A modification of BPSO was presented by Vieira et al. [23] to predict mortality of septic patients using SVM. A modification was made by Yang et al. [24] in which different transfer functions were used along with a new procedure to update position to search for best task allocation solution for wireless sensor network. Lin et al. [25] proposed a

high-utility item-set mining (HUIM-BPSO) by using BPSO to find HUI efficiently. In [26], the theoretical as well as empirical analysis of effect of inertia weight strategies on the performance of BPSO have been presented. In [14], the value of acceleration coefficients was modified base on the fitness of each particle to improve the convergence speed. Ji et al [28] proposed an effective approach named Improved BPSO to address the formulated problems in feature selection and improve its accuracy. Too et al [29] presented a new co-evolution BPSO by utilizing different inertia weight strategies to solve feature selection problems. Mafarja et al [30] proposed a feature selection approach by using BPSO with a time-varying inertia weight strategy to reduce the processing time.

3. THE NOVEL BPSO

A Novel BPSO (NBPSO) [2] was proposed to address the difficulties of standard BPSO and also solved the issues of selecting a proper value of inertia weight. In NBPSO personal best position *pbest* and global best position *gbest* are updated the same as the standard BPSO equations. The definition of velocity is different in this novel version. Two velocity vectors V0 and V1 were introduced for each particle such that V0 holds a chance of a particle's bits to change to 0, while V1 holds a chance of particle's bits to change to 1. V0 and V1 are computed as:

$$V_{ik}^{1} = wv_{ik}^{1} + q_{1}^{1} + q_{2}^{2} \tag{4}$$

$$V_{ik}^{0} = wv_{ik}^{0} + q_{1}^{0} + q_{2}^{0} \tag{5}$$

w is the inertia weight and q1, q0 are temporary values. If k_{th} bit in gbest and *pbest* is zero, V_{ik}^{0} will grow and the chance of changing to one will be decreased to zero. And if the kth bit in *gbest* is one, V_{ik}^{1} will be increased and V_{ik}^{0} will decrease. Based on the above description, the following rules are elicited:

$$\text{If } P_{ibest}^{k} = 1 \text{ then } g_{ik,1}^{1} = \text{c1r1 and } g_{ik,1}^{0} = \text{-c2r2}$$
$$\text{If } P_{ibest}^{k} = 0 \text{ then } g_{ik,1}^{0} = \text{c1r1 and } g_{ik,1}^{1} = \text{-c2r2}$$
$$\text{If } P_{gbest}^{k} = 1 \text{ then } g_{ik,2}^{1} = \text{c2r2 and } g_{ik,2}^{0} = \text{-c2r2}$$
$$\text{If } P_{gbest}^{k} = 0 \text{ then } g_{ik,2}^{0} = \text{c2r2 and } g_{ik,2}^{1} = \text{-c2r2}$$

where r1 and r2 are random variables within (0, 1) and are updated after each iteration. c1 and c2 are the acceleration coefficients. After V^0 and V^1 are updated, the velocity of change is computed as:

$$V_{c} = \begin{cases} V^{1}, if \;\; x = 0 \\ V^{0}, if \;\; x = 1 \end{cases} \tag{6}$$

Normalization is performed using a sigmoid function in equation (1). The next position of the particle is computed as:

$$x_{ik}(t+1) = \begin{cases} x_{ik}(t), if \; r_{ik} < V_{ik}^{'} \\ x_{ik}(t), if \; r_{ik} < V_{ik}^{'} \end{cases} \tag{7}$$

The performance of NBPSO was compared with other versions of binary PSO. Experiments were performed on four test functions.

4. THE PROPOSED ALGORITHM

To improve the performance of NBPSO, we have proposed a RIW-NBPSO algorithm. Instead of a fixed value for inertia weight in NBPSO, an adaptive inertia weight strategy based on fitness rank is introduced. The proposed algorithm works the same as the NBPSO. The velocity vectors are evaluated using equation (3) where the value of w is computed as in equation (8). In the proposed algorithm fitness of each particle is computed. All particles are then sorted based on their fitness. Then rank is assigned to every particle for their corresponding fitness. In RIW-NBPSO a particle with a high fitness value gets the first rank and the value of w for this particle will be minimum which speeds up the convergence rate, while a particle with a low fitness value, gets the lowest rank, and w for this particle will be maximum which improves search abilities so the particle with low fitness can move with the high velocity. This improves the convergence speed.

$$w_{1} = w_{min} + \left(\frac{w_{max} - w_{min}}{Popuilation(n)} \right) \cdot FR_{i} \tag{8}$$

where FR_i is the fitness rank of each particle. w_{min} is 0.4 and w_{max} is 0.9. As an important parameter of CPSO, it is important to set a proper value of inertia weight. This parameter highly affects the performance of the algorithm [23]. In the proposed RIW-NBPSO the employment of adaptive w has served well and better than NBPSO in terms of fast convergence.

I. The population is initialized with random positions of particles within the hypercube (particles are selected randomly from binary values 0, 1).
II. Compute the fitness for an individual particle by using its current position.
III. Find the personal best position of each particle by comparing every particle's fitness to its best fitness. Set the current place as the best place if

fitness at the current place is better than its best place.

IV. Find the global best position from all the particles by comparing the individual's fitness to its best fitness within the population. Set the current position as the best position if the fitness at the current place is better than its best place.

V. Sort and rank all the particles with respect to their fitness.

VI. Calculate the inertia weight for each particle using equation (8), so that the movement of each particle is commanded by its fitness.

VII. Update the velocity of particle V^0 and V^1 according to equations (4) and (5).

VIII. Compute the velocity of change of bits according to equation (6).

IX. Generate a random variable r within range (0, 1) to move each particle to a new place according to Equation (7).

X. Go to I, repeat till the convergence.

4.1. EXPERIMENTAL SETUP & RESULTS

To evaluate the performance of RIW-NBPSO, four test functions were selected and shown in equation (9) to equation (12) for Sphere, Rosenbrock, Griewangk, and Rastrigin [27] respectively. The comparison of the improved performance of the proposed RIW-NBPSO with NBPSO and other algorithms is provided in tables. The experiments are conducted on the minimization of test functions.

$$f_{1(x)} = \sum_{i=1}^{N} x_i^2 \tag{9}$$

$$f_{2(x)} = \sum \left(100 \left(x_{i+1} - x_i^2 \right)^2 + \left(x_i - 1 \right)^2 \right) \tag{10}$$

$$f_3(x) = \frac{1}{4000} \sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos \frac{x_i}{\sqrt{i}} + 1 \tag{11}$$

$$f_4(x) = \sum_{i=1}^{N} \left(x_i^2 - 10\cos\left(2\pi x_i\right) + 10 \right) \tag{12}$$

In the above-mentioned benchmark functions, N represents the dimensions of search space. The population size of 100 is carried out for a maximum number of iterations: 1000 within range of [-50, 50]. Real numbers are represented using 20 bits binary values. Three different dimensions are tested: 3, 5, and 10. The experimental results in Table (I-IV) show the improved performance of RIW-NBPSO in terms of fast convergence for all four test functions.

Dim	RIW-NBPSO	NBPSO [4]	BPSO [4]	BPSO [4]
3	**6.821*10-9**	6.821*10-9	0.0561	0.154
5	**1.136*10-8**	1.921*10-6	7.9578	224.404
10	**1.682*10-7**	0.112	213.606	394.706

Table 1 Results of Sphere function

Table 1- presents the experiments of the mean of the best *gbest* carried out on Sphere function. In which RIW-NBPSO outperforms the NBPSO and other algorithms for different dimensions: 3, 5, and 10 in terms of fast convergence.

Dim	RIW-NBPSO	NBPSO [4]	BPSO [4]	BPSO [4]
3	**0.031**	0.093	0.938	0.864
5	**1.366**	2.247	1406	3746.5
10	**8.724**	32.831	1.309*106	1.523*106

Table 2 Results of Rosenbrock function

Table 2- shows the experiments for the mean of best *gbest* conducted on Rosenbrock function. The improved convergence of RIW-NBPSO as compared to NBPSO and other algorithms are listed with different dimensions: 3, 5, and 10.

Dim	RIW-NBPSO	NBPSO [4]	BPSO [4]	BPSO [4]
3	**2.08*10-9**	2.08*10-9	0.1716	0.2025
5	**2.59*10-9**	7.4*10-3	0.5824	0.6574
10	**0.0230**	0.0579	1.3864	1.4333

Table 3 Results of Griewangk function

Table 3- demonstrates the results for the mean of best *gbest* on the Griewangk function. RIW-NBPSO increases the convergence speed than the NBPSO and other algorithms for different dimensions: 3, 5, and 10.

Dim	RIW-NBPSO	NBPSO [4]	BPSO [4]	BPSO [4]
3	**4.5109*10-9**	1.353*10-6	2.669	3.7127
5	**4.5109*10-9**	0.0034	25.875	51.3154
10	**4.5109*10-9**	10.392	490.	539.337

Table 4 Results of Rastrigin function

Table 4- presents the results for the mean of best *gbest* on Rastrigin function. It is cleared that RIW-NBPSO performed better than NBPSO and other algorithms for all dimensions 3, 5, and 10 in 1000 iterations.

It is cleared from Table (1-4) that the proposed RIW-NBPSO using Rank-based inertia weight significantly improved the convergence speed. The proposed scheme outperforms as compared with NBPSO and other algorithms in terms of quick convergence.

4.2. ADDITIONAL EXPERIMENTS & DISCUSSION

To validate the improved performance in terms of fast convergence of RIW-NBPSO, we have performed further experiments by reducing the number of iterations from 1000 to 500, 200, 100, and 50. When the iterations are 1000, the particles get a higher chance to search the space so for 1000 iterations our proposed RIW-NBPSO quickly converges.

We have performed experiments by reducing the iterations. When the iterations are reduced to 500 RIW-NBPSO still provided better convergence results. We further reduced the iterations to 200 to check the convergence speed of the proposed RIW-NBPSO, it gives better results here too. The number of iterations further reduced to 100 and still the particles converge quickly which shows the proposed algorithm provided better results. We then reduced the iterations to 50, RIW-NBPSO showed quick convergence. It is validated from these experiments that the proposed algorithm accelerates convergence.

Table 5- evaluated the experimental results of RIW-NBPSO on Sphere function for different iterations tested on dimensions: 3, 5, and 10. The results demonstrated that the RIW-NBPSO particles quickly converged for 1000 iterations, when the iterations are reduced to 500 the algorithm still provided a better convergence. For 200, 100 and 50 iterations algorithm performed well.

Dim	1000	500	200	100	50
3	**6.821*10-9**	6.821*10-9	6.821*10-9	6.821*10-9	6.821*10-9
5	**1.1369*10-8**	1.136*10-8	1.136*10-8	6.93*10-6	0.010
10	**1.682*10-7**	6.472*10-5	0.007	0.616	8.236

Table 5 Results of Sphere function

Table 6- showed the results on the Rosenbrock function for dimensions: 3, 5, and 10. The results demonstrated that the RIW-NBPSO particles quickly converged for 1000 iterations, when the iterations are reduced to 500 the algorithm still provided a better convergence. For 200, 100 and 50 iterations algorithm performed well.

Dim	1000	500	200	100	50
3	**0.031**	1.771	0.286	0.392	0.070
5	**1.366**	3.980	2.879	3.136	3.446
10	**8.724**	17.422	56.395	105.485	5.75*102

Table 6 Results of Rosenbrock function

Table 7- demonstrated the results on the Griewangk function for dimensions: 3, 5, and 10. The results made it clear that the RIW-NBPSO particles quickly converged for 1000 iterations, when the iterations are reduced to 500 the algorithm still provided a better convergence. For 200, 100 and 50 iterations algorithm performed well.

Dim	1000	500	200	100	50
3	**2.08*10-9**	2.086*10-9	2.086*10-9	2.086*10-9	0.0074
5	**2.59*10-9**	0.0075	0.0148	0.0300	0.0311
10	**0.0230**	0.0124	0.0160	0.1531	0.2238

Table 7 Results of Griewangk function

Table 8- presented the results on the Rastrigin function for dimensions: 3, 5, and 10. The results demonstrated that the RIW-NBPSO particles quickly converged for 1000 iterations, when the iterations are reduced to 500 the algorithm still provided a better convergence. For 200, 100 and 50 iterations algorithm performed well.

Dim	1000	500	200	100	50
3	**4.5109*10-9**	4.5109*10-7	4.5109*10-7	4.5109*10-7	4.5109*10-7
5	**4.5109*10-9**	4.5109*10-7	4.5109*10-7	4.5109*10-7	4.5109*10-7
10	**4.5109*10-9**	4.5109*10-7	4.5109*10-7	4.5109*10-7	4.5109*10-7

Table 8 Results of Rastrigin function

5. CONCLUSION

In this work, a Rank-based Self-adaptive inertia weight scheme is introduced in NBPSO to enhance its convergence speed. Unlike BPSO where all the particles with distinct positions are equally considered, the proposed RIW-NBPSO uses rank-based inertia weight that controls the movement of particles by assigning ranks to the particles based on their fitness value. The experiments are performed on four benchmark test functions to evaluate the performance of RIW-NBPSO. The findings affirmed the improved performance of the proposed RIW-NBPSO than the compared algorithms in terms of convergence speed. To validate the improved convergence speed of RIW-NBPSO, additional experiments are executed on four test functions for different iteration. The additional results demonstrated that RIW-NBPSO performed better, not just for 1000 iterations, it also performed better when the iterations are reduced from 1000 to 500, 200, 100, and 50. Hence the proposed RIW-NBPSO based on fitness rank enhances the convergence speed.

REFERENCES

[1] K. James and E. Russell, "Particle swarm optimization," in *Proceedings of 1995 IEEE International Conference on Neural Networks,* 1995, pp. 1942-1948.

[2] M. A. Khanesar, M. Teshnehlab, and M. A. Shoorehdeli, "A novel binary particle swarm optimization," in *Control & Automation, 2007. MED'07. Mediterranean Conference on,* 2007, pp. 1-6.

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

[4] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference* on, 1998, pp. 69-73.

[5] A. Rathore and H. Sharma, "Review on Inertia Weight Strategies for Particle Swarm Optimization," in *Proceedings of Sixth International Conference on Soft Computing for Problem Solving*, 2017, pp. 76-86.

[6] M. Taherkhani and R. Safabakhsh, "A novel stability-based adaptive inertia weight for particle swarm optimization," *Applied Soft Computing*, vol. 38, pp. 281-295, 2016.

[7] X. Shen, Z. Chi, J. Yang, and C. Chen, "Particle swarm optimization with dynamic adaptive inertia weight," in Challenges in *Environmental Science and Computer Engineering (CESCE), 2010 International Conference* on, 2010, pp. 287-290.

[8] M. A. Arasomwan and A. O. Adewumi, "On the performance of linear decreasing inertia weight particle swarm optimization for global optimization," *The Scientific World Journal*, vol. 2013, 2013.

[9] Y. Mehmood and W. Shahzad, "An Accelerated Convergent Particle Swarm Optimizer (ACPSO) of Multimodal Functions," 2018.

[10] W. Liao, J. Wang, and J. Wang, "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," in *International Conference in Swarm Intelligence*, 2011, pp. 80-85.

[11] Y. Feng, G.-F. Teng, A.-X. Wang, and Y.-M. Yao, "Chaotic inertia weight in particle swarm optimization," in I*nnovative Computing, Information and Control, 2007. ICICIC'07. Second International Conference* on, 2007, pp. 475-475.

[12] K. Lei, Y. Qiu, and Y. He, "A new adaptive well-chosen inertia weight strategy to automatically harmonize global and local search ability in particle swarm optimization," in *Systems and Control in Aerospace and Astronautics, 2006. ISSCAA 2006. 1st International Symposium* on, 2006, pp. 4 pp.-980.

[13] B. Panigrahi, V. R. Pandi, and S. Das, "Adaptive particle swarm optimization approach for static and dynamic economic load dispatch," *Energy conversion and management*, vol. 49, pp. 1407-1415, 2008.

[14] Y. Mehmood, M. Sadiq, W. Shahzad, and F. Amin, "Fitness-based acceleration coefficients to enhance the convergence speed of novel binary particle swarm optimization," in *2018 International Conference on Frontiers of Information Technology (FIT)*, 2018, pp. 355-360.

[15] Z. Beheshti, S. M. Shamsuddin, and S. Hasan, "Memetic binary particle swarm optimization for discrete optimization problems," *Information Sciences,* vol. 299, pp. 58-84, 2015.

[16] L.-Y. Chuang, C.-H. Yang, and J.-C. Li, "Chaotic maps based on binary particle swarm optimization for feature selection," *Applied Soft Computing,* vol. 11, pp. 239-248, 2011.

[17] Y.-W. Jeong, J.-B. Park, S.-H. Jang, and K. Y. Lee, "A new quantum-inspired binary PSO: application to unit commitment problems for power systems," *IEEE Transactions on Power Systems*, vol. 25, pp. 1486-1495, 2010.

[18] F. Afshinmanesh, A. Marandi, and A. Rahimi-Kian, "A novel binary particle swarm optimization method using artificial immune system," in *Computer as a Tool, 2005. EUROCON 2005. The International Conference* on, 2005, pp. 217-220.

[19] C.-J. Liao, C.-T. Tseng, and P. Luarn, "A discrete version of particle swarm optimization for flowshop scheduling problems," *Computers & Operations Research,* vol. 34, pp. 3099-3111, 2007.

[20] A. H. El-Maleh, A. T. Sheikh, and S. M. Sait, "Binary particle swarm optimization (BPSO) based state assignment for area minimization of sequential circuits," *Applied Soft Computing*, vol. 13, pp. 4832-4840, 2013.

[21] B. Wei and J. Zhao, "Haplotype inference using a novel binary particle swarm optimization algorithm," *Applied Soft Computing*, vol. 21, pp. 415-422, 2014.

[22] J. C. Bansal and K. Deep, "A modified binary particle swarm optimization for knapsack problems," *Applied Mathematics and Computation*, vol. 218, pp. 11042-11061, 2012.

[23] S. M. Vieira, L. F. Mendonça, G. J. Farinha, and J. M. Sousa, "Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients," *Applied Soft Computing,* vol. 13, pp. 3494-3504, 2013.

[24] J. Yang, H. Zhang, Y. Ling, C. Pan, and W. Sun, "Task allocation for wireless sensor network using modified binary particle swarm optimization," *IEEE Sensors Journal*, vol. 14, pp. 882-892, 2014.

[25] J. C.-W. Lin, L. Yang, P. Fournier-Viger, T.-P. Hong, and M. Voznak, "A binary PSO approach to mine high-utility itemsets," *Soft Computing*, vol. 21, pp. 5103-5121, 2017.

[26] J. Liu, Y. Mei, and X. Li, "An analysis of the inertia weight parameter for binary particle swarm optimization," *IEEE Transactions on Evolutionary Computation,* vol. 20, pp. 666-681, 2016.

[27] M. Jamil and X.-S. Yang, "A literature survey of benchmark functions for global optimisation problems," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, pp. 150-194, 2013.