The 3rd International Conference of Reliable Information and Communication Technology 2018 (IRICT 2018), Putrajaya, Malaysia, July 2018.

Novel Multi-Swarm Approach for Balancing Exploration and Exploitation in Particle Swarm Optimization

Sinan Q. Salih[1,2] , AbdulRahman A. Alsewari[1], Bellal Al-Khateeb[2], Mohamad Fadli Zolkipli[1],

[1] Faculty of Computer Systems & Software Engineering, Universiti Malaysia Pahang, 26300, Gambang, Pahang, Malaysia

[2] Computer Science Department, College of Computer Science and Information Technology, University of Anbar, Ramadi, Iraq.

Abstract. Several metaheuristic algorithms and improvements to the existing ones have been presented over the years. Most of these algorithms were inspired either by nature or the behavior of certain swarms, such as birds, ants, bees, or even bats. These algorithms have two major components, which are exploration and exploitation. The interaction of these components can have a significant influence on the efficiency of the metaheuristics. Meanwhile, there are basically no guiding principles on how to strike a balance between these two components. This study, therefore, proposes a new multi-swarm-based balancing mechanism for keeping a balancing between the exploration and exploitation attributes of metaheuristics. The new approach is inspired by the phenomenon of the leadership scenario among a group of people (a group of people being governed by a selected leader(s)). These leaders communicate in a meeting room, and the overall best leader makes the final decision. The simulation aspect of the study considered several benchmark functions and compared the performance of the suggested algorithm to that of the standard PSO (SPSO) in terms of efficiency.

Keywords. Swarm Intelligence, Exploration, Exploitation, Metaheuristics, Optimization, Computational Intelligence.

1 Introduction

Over the past 2 decades, nature-inspired metaheuristics have attracted much attention due to their efficiency in establishing accurate solutions to complex industrial and engineering problems, especially the NP-complete problems. Most nature inspired metaheuristics are classified as stochastic techniques. These stochastic algorithms randomly pick a set of solutions and improve them based on the algorithmic mechanism. The solutions are constantly improved until a set stopping criterion is met. Stochastic techniques are classified as random searches but guided to the next iteration by heuristics. In the last few years, many stochastic algorithms have been proposed due to their great success in finding best solutions to science and engineering problems [1–4].

The Particle Swarm Optimizer (PSO) is one of the most popular algorithms first introduced by Kennedy and Eberhard [5, 6]. The PSO solves optimization problems by emulating the flocking behavior of birds; where each bird is regarded as a solution. The advantage of the PSO when compared to the evolution-based frameworks like the Genetic algorithm, lies in its ease of implementation and in requiring just a few parameters to be adjusted [7, 8]. The PSO has successfully been applied in several instances such as function optimization, fuzzy systems, artificial neural network training, and feature selection [9–17]. It can also be applied in other areas where GA can be employed.

In the original PSO or simple PSO (SPSO), a major difficulty lies in maintaining the balancing between exploration (searching for the global optimum) and exploitation (searching for the local optimum). Although the SPSO can converge quickly in the initial iterations towards an optimum, its problems lie in reaching a near optimal solution. This problem has attracted several attentions in terms of ways to enhance the performance of SPSO, including a proposal for hybrid models [8, 18–20].

In the literature, several multi-swarm attempts have earlier been proposed. The balancing between local search 'exploration' and global search 'exploitation' in PSO using a master-slave approach has been proposed [21]. This approach is a cooperative scheme made up of a superior swarm known as a master swarm and several other inferior swarms known as slave swarms. These slave swarms provide the master swarm with new promising particles (new positions with the best fitness value) as the evolution continues. The state of these new particles is updated by the master swarm with respect to the best position so far discovered by both itself and the slave swarms.

This study proposed a new multi-swarm cooperative scheme for balancing the exploration and exploitation of PSO. The proposed scheme consists of several swarms called 'clans'; each clan has its own leader. The leader of each clan is the best solution in the clan and represents the local best. All the clan leaders meet periodically to select the best among themselves who will represent the global best solution. This best leader has control over all the other clan leaders. The interaction between the selected best leader (the global best) and the individual clan leaders (the local best) has an influence on the balance between their exploratory and local search performances, and maintain a suitable population diversity even when approaching the global best solution, thereby, minimizing the risk of convergence or being trapped to the local sub-optimal.

The remaining part of this study is organized thus: Section 2 described the original PSO and its variants, while Section 3 described the motivation for the proposed approach and provided the algorithmic pseudo-code as well. In Section 4, a description of the benchmark continuous optimization problems for benchmarking the performance of the algorithm was provided, followed by the discussions of the study results. Section 5 provided a brief conclusion of the study.

2 Particle Swarm Optimization (PSO)

The standard version of particle swarm optimization PSO is a well-known optimization algorithm, the swarm is initialized with a random population of solutions. The PSO searches for the best positions by updating its component generations. The generated particles in the PSO (which are the solutions) fly in a D-dimensional search space at a velocity dynamically adjusted based on both their own respective experiences and the experience of their neighbors.

The *i*th particle in the PSO is denoted in the D-dimensional space as $x_i = (x_{i1}, x_{i2}, x_{i3}, \dots, x_{iD})$, where $x_{id} \in [LB_d, UB_d]$, $d \in [1, D]$, LB_d, UB_d respectively represents the minimum and maximum limits of the dth dimension. The velocity of particle *i* is given as $v_i = (v_{i1}, v_{i2}, v_{i3}, \dots, v_{iD})$, which is maintained at a maximum user-specified velocity V_{max}. The particles, at each time step *t*, are manipulated based on the following relations:

$$v_i(t+1) = v_i(t) + r_1 c_1 \big(P_i - x_i(t)\big) + r_2 c_2 \left(P_g - x_i(t)\right) \quad (1)$$

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

where r_1 and $r_2$represents the random values in the range of 0 and 1. $c_1$and c_2 represents the acceleration constants that governs the extent a particle can move within a given iteration. The previous best position of the *i*th particle is represented by P_i.

Based on the several definitions of P_g, there are 2 variants of the PSO. A global version of PSO is achieved when P_g represents the position of the best particle among the other particles in the same population (also referred to as the as$gbest$). But if P_g is derived from a few number of adjacent particles of a population (called $lbest$), a local version of PSO is achieved. An inertia term w was later introduced by Shi and Eberhard [22] via a modification of equation (1) into:

$$v_i(t+1) = w \times v_i(t) + r_1 c_1 \big(P_i - x_i(t)\big) + r_2 c_2 \left(P_g - x_i(t)\right) \quad (3)$$

They suggested that a proper balance between global and local explorations can be achieved through a proper selection of w, thus, requiring averagely less iterations to establish an optimal solution. The w, as originally developed, is set using the following equation:

$$w = w_{max} - \frac{w_{max} - w_{min}}{itr_{max}} \times itr \quad (4)$$

Where w_{max} represents the initial weight, w_{min} represents the final weights, itr_{max} is the highest number of allowable iterations, and itr represents the present number of iterations.

This version of PSO is in this study, henceforth referred to as a linearly decrease inertia weight method (LPSO).

In addition to LPSO, a random inertia weight factor for dynamic systems tracking has also been suggested [23]. The inertia weight factor in this development is set to randomly change based on the following relation:

$$w = 0.5 - \frac{rand()}{2} \tag{5}$$

where $rand()$ represents a uniformly distributed random number in the range of $0\ and\ 1$.

The acceleration coefficients were suggested to be maintained at 1.494. This method is henceforth referred to as random weight method (RPSO) in the remaining part of this paper.

3 Multi-Swarm PSO Algorithm

The core idea of the multi-swarm is the interaction between several groups while searching for a solution. Many multi-swarm-based schemes have been proposed, each being inspired by a natural behavior. In this paper, a new cooperative multi-swarm scheme inspired by the human social behavior (the interaction between a group of people known as 'Clan' and their leaders) was proposed. The proposed scheme consists of several swarms called clans; each clan consists of several solutions represented by the group members. The best member of each clan is the clan leader and has control over the members of its clan in terms of the time to move and where they are moving to. Figure 1 showed the structure of the individual swarm.

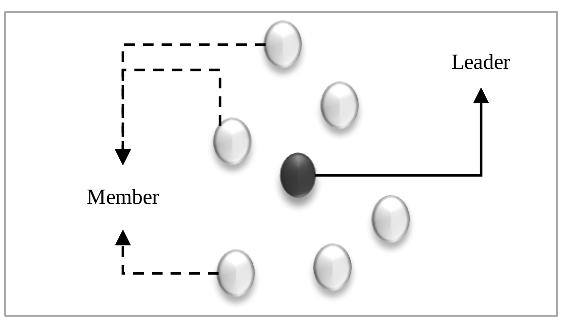

Fig 1. The structure of the individual swarm

In each generation, the leaders often meet in one room to select an overall best leader who will update the position of the other leaders based on his new-found position. This behavior of knowledge sharing helps to balance the exploration stage with the searching process of the PSO, which represents the exploitation stage. The new multi-swarm approach is called a 'Meeting Room Approach' (MRA). Figure 2 showed the MRA model, where each member in the clan represents a particle in the swarm, and its position and velocity is updated based on the steps of PSO algorithm. Once the new generation of each clan has been set, a new clan leader (the best leader) is elected and sent to the meeting room. The best among the leaders will be selected as the overall best leader (global best) in the meeting room. The newly-selected overall best leader shares his positional information with the other leaders using the following relation:

$$w^{Ln} = \left(\frac{w^{Lg} - w^{Ln}}{Itr}\right) \times rand() \tag{6}$$

$$v_i{}^{Ln}(t+1) = w^{Ln} \times v_i{}^{Ln}(t) + rc\left(P_g{}^L - P_n{}^L(t)\right) \tag{7}$$

$$x_i{}^{Ln}(t+1) = x_i{}^{Ln}(t) + v_i{}^{Ln}(t) \tag{8}$$

where Ln represents the normal leaders, Lg represents the overall best leader, $x_i{}^L$ represents the position of the normal leaders, $v_i{}^{Ln}$ represents the velocity of the normal leaders, w^{Lg} and w^{Ln} represent the inertia weight of the overall best leader and the normal leaders, respectively.

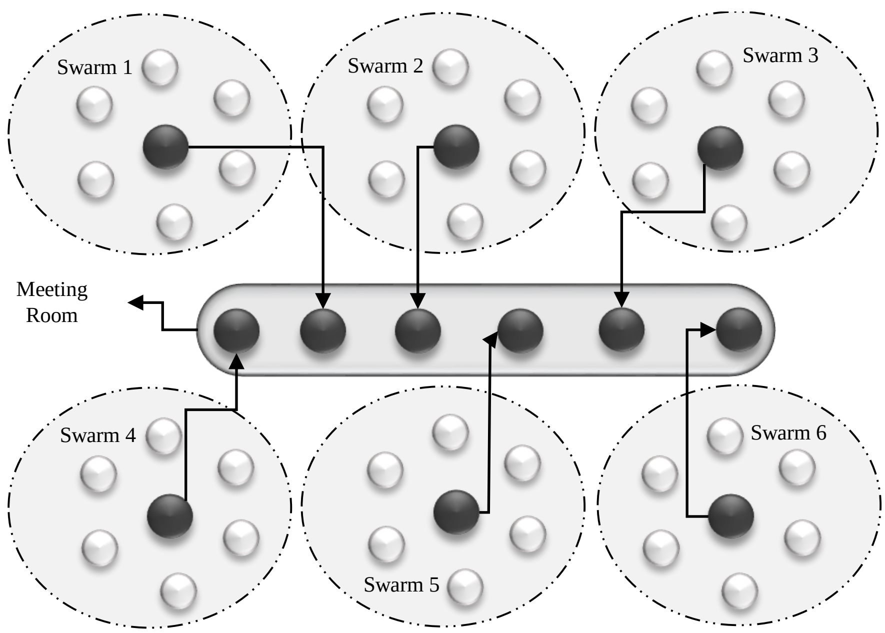

Fig 2. Meeting Room Approach

After each generation, a new leader is selected for each swarm because the positions of the members are changed or updated during the meeting. The new equation of the inertia in the meeting room controls the exploration of the search algorithm. The pseudo-code for the MPSO algorithm is listed in Figure 3.

```
Algorithm MPSO
Input:
    #Swarm, #P, c_1, c_2, #Dim, #MaxGen
Output:
    Best Solution (Leader)
Procedure:
    Start
    |   Initialized the Swarms
    |   Evaluate the fitness of each particle in Swarms
    |   While (itr ≤ #MaxGen)
    |   |   For each s in Swarm
    |   |   |   For each member in Swarm s
    |   |   |   |   Update the velocity of the member via eq.3
    |   |   |   |   Update the position of the member via eq.2
    |   |   |   Next
    |   |   |   Select the local best as a Leader_s
    |   |   Next
    |   |   Select the best Leader among all leaders
    |   |   Update the inertia weight of the clan via eq.6
    |   |   Update the velocity of the Leader_s via eq.7
    |   |   Update the position of the Leader_s via eq.8
    |   |   Select the Best Leader ever as the global best.
    |   Loop
    |   Return Best Leader
    Stop
```

Fig 3. MPSO Pseudo-Code

4 Results and Discussion

This section presents the detailed description of the nonlinear benchmark functions commonly used in the evolutionary literature [24]. Each test function varies in terms of modality (unimodal and multimodal) and the number of dimensions (fixed and dynamic). Table 1 showed the different test functions and their basic characteristics.

The performance of the proposed MPSO was evaluated by benchmarking with two established algorithms (the original PSO (SPSO)[25] and the Master-Slave PSO (MCPSO)[21]). The parameters used for the SPSO have earlier been recommended by [25] with asymmetric initialization method and a linearly decreasing *w* which was changed from

0.9 to 0.4. Several swarms of SPSO were involved in the MPSO and MCPSO as clans and slaves respectively, to optimize the listed benchmark function. Each of them has the same parameter settings as SPSO1. To investigate the performance of MPSO, different population sizes were employed with different dimensions for each function. The maximum number of iterations was set at 500, which corresponds to 50 dimensions. For each experimental setting, a total of 30 runs were conducted. Table 2 presents the parameters setting for all the algorithms used in this study.

Table 1. Benchmark test functions

Name	Function	Range	Opt.
Sphere Unimodal	$f_1 = \sum_{i=1}^{D} X^2$	-100,100	0
Griewank Unimodal	$f_2 = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	-600,600	0
Rastrigin Multimodal	$f_3 = \sum_{i=1}^{D} (x_i^2 - 10\ \cos(2\pi x_i) + 10\ )$	-5.12,5.12	0
Ackley Multimodal	$f_4 = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2 0}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + e$	-32,32	0

Table2. Parameters Setting

Algorithm	Parameter	Value
SPSO	W	0.9 – 0.4
	No. of Swarms	1
	Swarm Size	50
	c_1,c_2	1.5
MCPSO	W	0.9-0.6
	No. of Slaves	5
	Swarm Size	50
	c_1,c_2,c_3	1.5
MPSO	w^{Ln}	0.8 – 0.5
	w^{Lg}	0.9 – 0.7
	c_1,c_2	1.5
	No. of Clans	5
	Clan Size	10

Table 3 presents the best and mean fitness values of the particle after 30 runs and 4 benchmark functions. Based on the table, MPSO performed better than the other algorithms in almost all the studied cases. A general analysis of the table showed that MPSO had 5 swarms, each consisting of 10 particles, and only 5 particles interacted in the meeting room. It may be concluded that MPSO required less computational complexity, and yet, had a better performance in terms of finding the best solution. Figures 4(a and b) illustrate the sustainability of the MPSO to evolve even when the other algorithms were almost stagnated.

Table3. Results for benchmark test functions

f_n	Swarm	Algorithm	Best	Mean	S.D
f_1	50	SPSO	2.5457521	2.7647845	0.0784516
		MCPSO	0.9854126	1.0154784	0.0014784
		MPSO	**0.0007845**	**0.0008748**	**0.0000184**
f_2	50	SPSO	0.0884741	0.0964587	0.0078478
		MCPSO	0.0078414	0.0087789	0.0009874
		MPSO	**0.0000897**	**0.0000997**	**0.0000658**
f_3	50	SPSO	21.695847	27.947512	0.0847896
		MCPSO	2.0018977	2.6647845	0.0078487
		MPSO	**0.0004687**	**0.0045214**	**0.0000144**
f_4	50	SPSO	16.4875218	26.110161	0.0238484
		MCPSO	1.99847	2.5869124	0.0084578
		MPSO	**0.0002648**	**0.0017636**	**0.0000584**

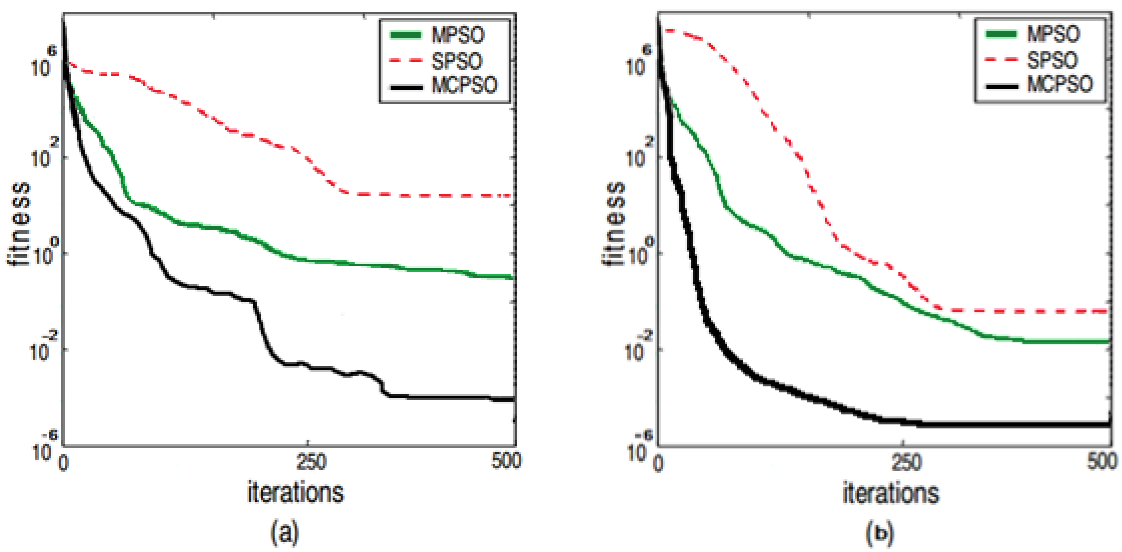

Fig 4: Convergence curve: a) Sphere Function b) Griewank Function

5 Conclusion

In this study, a social-inspired mechanism for improving the performance of the PSO was developed. The proposed mechanism simulates the social grouping behavior of human (existing as clans and interacting with their leaders). The proposed algorithm (MPSO) was able to control the balance between exploration and exploitation of PSO. During the simulation stage of the study, 4 benchmark functions were performed using different algorithms. The benchmarking in terms of the performance of the proposed MPSO showed that MPSO had a better performance than SPSO both in the quality and robustness of the solution. In the future works, more emphasis should be laid on applying the proposed MPSO into different swarm metaheuristics such as firefly algorithm, bat algorithm, and grey wolf optimizer.

Acknowledgement

This research is funded by UMP PGRS170338: Analysis System based on Technological YouTube Channels Reviews, and UMP RDU1603119 Grant: Modified Greedy Algorithm Strategy for Combinatorial Testing Problem with Constraints Supports. UMP RDU180367 Grant: Enhance Kidney Algorithm for IOT Combinatorial Testing Problem.

References

1. Slowik, A., Kwasnicka, H.: Nature Inspired Methods and Their Industry Applications – Swarm Intelligence Algorithms. IEEE Trans. Ind. Informatics. 14, 1004–1015 (2018).
2. Diao, R., Shen, Q.: Nature Inspired Feature Selection Meta-Heuristics. Artif. Intell. Rev. 44, 311–340 (2015).
3. Azrag, M.A.K., Kadir, T.A.A., Odili, J.B., Essam, M.H.A.: A Global African Buffalo Optimization. Int. J. Softw. Eng. Comput. Syst. 3, 138–145 (2017).
4. Odili, J.B., Kahar, M.N.M., Anwar, S.: African Buffalo Optimization: a Swarm-Intelligence Technique. Procedia Comput. Sci. 76, 443–448 (2015).
5. Eberhart, R., Kennedy, J.: A New Optimizer using Particle Swarm Theory. In: Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on. pp. 39–43 (1995).
6. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. Neural Networks, 1995. Proceedings., IEEE Int. Conf. 4, 1942–1948 vol.4 (1995).
7. Naik, B., Nayak, J., Behera, H.S.: A Novel FLANN with a Hybrid PSO and GA based Gradient Descent Learning for Classification. In: Advances in Intelligent Systems and Computing. pp. 745–754 (2014).
8. Shi, X.H., Liang, Y.C., Lee, H.P., Lu, C., Wang, L.M.: An Improved GA and a Novel PSO-GA-based Hybrid Algorithm. Inf. Process. Lett. 93, 255–261 (2005).
9. Chen, D., Chen, J., Jiang, H., Zou, F., Liu, T.: An Improved PSO Algorithm based on Particle Exploration for Function Optimization and the Modeling of Chaotic Systems. Soft Comput. 19, 3071–3081 (2015).

10. Palupi Rini, D., Mariyam Shamsuddin, S., Sophiyati Yuhaniz, S.: Particle Swarm Optimization: Technique, System and Challenges. Int. J. Comput. Appl. 14, 19–27 (2011).
11. Mendes, R., Cortez, P., Rocha, M., Neves, J.: Particle Swarms for Feedforward Neural Network Training. In: Proceedings of the 2002 International Joint Conference on Neural Networks IJCNN02 Cat No02CH37290. pp. 1895–1899 (2002).
12. Gudise, V.G., Venayagamoorthy, G.K.: Comparison of Particle Swarm Optimization and Backpropagation as Training Algorithms for Neural Networks. Proc. 2003 IEEE Swarm Intell. Symp. SIS'03 (Cat. No.03EX706). 2, 110–117 (2003).
13. Melin, P., Olivas, F., Castillo, O., Valdez, F., Soria, J., Valdez, M.: Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic. Expert Syst. Appl. 40, 3196–3206 (2013).
14. Niu, Q., Huang, X.: An improved fuzzy C-means clustering algorithm based on PSO. J. Softw. 6, 873–879 (2011).
15. Xue, B., Zhang, M., Browne, W.N.: Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms. Appl. Soft Comput. J. 18, 261–276 (2014).
16. Inbarani, H.H., Bagyamathi, M., Azar, A.T.: A Novel Hybrid Feature Selection Method Based on Rough Set and Improved Harmony Search. Neural Comput. Appl. 26, 1859–1880 (2015).
17. Huang, C.L., Dun, J.F.: A Distributed PSO-SVM Hybrid System with Feature Selection and Parameter Optimization. Appl. Soft Comput. J. 8, 1381–1391 (2008).
18. Mirjalili, S., Hashim, S.Z.M.: A New Hybrid PSOGSA Algorithm for Function Optimization. In: Proceedings of ICCIA 2010 - 2010 International Conference on Computer and Information Application. pp. 374–377 (2010).
19. Premalatha, K., Natarajan, a M.: Hybrid PSO and GA for Global Maximization. Int. J. Open Probl. Compt. Math. 2, 597–608 (2009).
20. Zhang, Y., Wu, L.: A Hybrid TS-PSO Optimization Algorithm. J. Converg. Inf. Technol. 6, 169–174 (2011).
21. Niu, B., Zhu, Y., He, X., Wu, H.: MCPSO: A Multi-Swarm Cooperative Particle Swarm Optimizer. Appl. Math. Comput. 185, 1050–1062 (2007).
22. Shi, Y., Eberhart, R.: A Modified Particle Swarm Optimizer. Evol. Comput. Proceedings, 1998. IEEE World Congr. Comput. Intell. 1998 IEEE Int. Conf. 69–73 (1998).
23. Eberhart, R. C. and Shi, Y.: Tracking and Optimizing Dynamic Systems with Particle Swarms. In: Proceedings of IEEE Congress on Evolutionary Computation. pp. 94–97. IEEE, Seoul, Korea (2001).
24. Jamil, M., Yang, X.S., Zepernick, H.J.D.: Test Functions for Global Optimization: A Comprehensive Survey. In: Swarm Intelligence and Bio-Inspired Computation. pp. 193–222 (2013).
25. Shi, Y., Eberhart, R.C.: Empirical Study of Particle Swarm Optimization. Evol. Comput. 1999. CEC 99. Proc. 1999 Congr. 3, 1945–1950 Vol. 3 (1999).