International Journal of Electrical and Computer Engineering (IJECE)
Vol. 7, No. 3, June 2017, pp. 1611~1619
ISSN: 2088-8708, DOI: 10.11591/ijece.v7i3.pp1611-1619

Transitional Particle Swarm Optimization

Nor Azlina Ab Aziz[1], Zuwairie Ibrahim[2], Marizan Mubin[3], Sophan Wahyudi Nawawi[4], Nor Hidayati Abdul Aziz[5]
[1,3] Faculty of Engineering, University of Malaya, Malaysia
[1,5] Faculty of Engineering & Technology, Multimedia University, Malaysia
[2] Faculty of Electrical & Electronic Engineering, Universiti Malaysia Pahang, Malaysia
[4] Faculty Electrical Engineering, Universiti Teknologi Malaysia, Malaysia

Article Info

Article history:

Received Feb 5, 2017
Revised Apr 8, 2017
Accepted Apr 24, 2017

Keyword:

Asynchronous update
Iteration strategy
Particle swarm optimization
Synchronous update

ABSTRACT

A new variation of particle swarm optimization (PSO) termed as transitional PSO (T-PSO) is proposed here. T-PSO attempts to improve PSO via its iteration strategy. Traditionally, PSO adopts either the synchronous or the asynchronous iteration strategy. Both of these iteration strategies have their own strengths and weaknesses. The synchronous strategy has reputation of better exploitation while asynchronous strategy is stronger in exploration. The particles of T-PSO start with asynchronous update to encourage more exploration at the start of the search. If no better solution is found for a number of iteration, the iteration strategy is changed to synchronous update to allow fine tuning by the particles. The results show that T-PSO is ranked better than the traditional PSOs.

Corresponding Author:

Nor Azlina Ab Aziz,
Faculty of Engineering & Technology,
Multimedia University,
75450 Melaka, Malaysia.
Email: azlina.aziz@mmu.edu.my

1. INTRODUCTION

In particle swarm optimization (PSO), optimization problem is solved by swarm of particles that move around a search space looking for optimal solution. Each of the particles has velocity and position. The particles move within the search space by iteratively updating these two values. There are two choices of update mechanisms/ iteration strategies; synchronous or asynchronous [1]. The synchronous update is the more popular approach. In synchronous PSO (S-PSO) the particles change their velocities and positions after the fitness of the whole swarm is evaluated. On the other hand, in asynchronous PSO (A-PSO), a particle is able to update its velocity and position as soon as it completes its fitness evaluation without the need to wait for the other particles to complete their fitness evaluation.

The synchronous update is stronger in exploitation, while asynchronous update is better in exploration [2]. Exploration and exploitation are important in ensuring the search space is effectively explored and information obtained is efficiently exploit. The exploration and exploitation improve performance and efficiency of an algorithm. Typically, higher exploration is favorable in the early stage while stronger exploitation is expected towards the end.

Here, a new variation of PSO algorithm that combines both update methods is proposed. The proposed algorithm is known as transitional PSO (T-PSO). The particles in T-PSO start with asynchronous update mechanism and then transit to synchronous update at a later state. The asynchronous update is adopted at the start of the search process so that exploration is encouraged. Once no improvement is achieved after S iterations, the particles transited to synchronous update mechanism and focus on exploitation of the information.

The proposed T-PSO is tested using CEC2014's benchmark functions. Its performance is then benchmarked with S-PSO and A-PSO. The findings show that T-PSO is able to achieve better rank than S-PSO and A-PSO. In the following section, both S-PSO and A-PSO algorithms are reviewed. The T-PSO algorithm is discussed in section 3. Next, the results of the experiment conducted are presented in section 4. Lastly this work is concluded in section 5.

2. PARTICLE SWARM OPTIMIZATION AND ITS TRADITONAL ITERATION STRATEGIES

PSO is a nature inspired algorithm. It emulates the social behavior as seen in nature, like birds flocking and fishes schooling. These organisms search for food source by moving in group without any centralized control. The search is performed through information sharing and individual effort.

This social behavior is copied in PSO, where the search for the solution of an optimization problem is carried by swarm of particles. Each particle has its position;

$$X_i(t) = \left\{x_i^1(t), x_i^2(t), x_i^3(t), \dots, x_i^d(t), \dots, x_i^D(t)\right\} \quad (1)$$

and velocity;

$$V_i(t) = \left\{v_i^1(t), v_i^2(t), v_i^3(t), \dots, v_i^d(t), \dots, v_i^D(t)\right\} \quad (2)$$

Where

$$i = \{1,2,3, \dots, N\} \quad (3)$$

is particle number and N is the size of the swarm i.e. number of particles, t is the iteration number, d is dimension number and D is the size of the problem's dimension. The particles look for optimal solution by updating their velocity and position.

The velocity is influenced by particle's experience and information shared within the swarm and updated using the following equation;

$$v_i^d(t) = \omega \times v_i^d(t-1) + c_1 \times {r_1}_i^d \times \left(p_i^d(t) - x_i^d(t-1)\right) + c_2 \times {r_2}_i^d \times \left(g^d(t) - x_i^d(t-1)\right) \quad (4)$$

In equation (4), ω is inertia weight it controls the momentum of the search. Typically, linearly decreasing inertia is used to encourage exploration in earlier phase of the search and facilitates fine tuning at the later stage. Two learning factors, c_1 and c_2 are used in the equation. Both learning factors are usually set to same value so that the importance of particle's own experience and social influence is equally weighted. The particle's own experience is represented by $P_i(t) = \left\{p_i^1(t), p_i^2(t), p_i^3(t), \dots, p_i^d(t), \dots, p_i^D(t)\right\}$, where this is the best solution that has been encountered by the particle since the start of the search up to the t^{th} iteration. Whereas, the best solution found by the swarm till t^{th} iteration is; $G(t) = \{g^1(t), g^2(t), g^3(t), \dots, g^d(t), \dots, g^D(t)\}$. PSO is a stochastic algorithm, where ${r_1}_i^d$ and ${r_2}_i^d$ are two independent random number ranging from $[0,1]$.

The position is updated using;

$$x_i^d(t) = x_i^d(t-1) + v_i^d(t) \quad (5)$$

Normally $x_i^d(t)$ is bounded according to the search space.

S-PSO and A-PSO are differentiated by the order a particle updates its velocity and position with respect to the swarm fitness evaluation. This can be seen in the flowchart for S-PSO and A-PSO, Figure 1 and Figure 2 respectively.

In S-PSO, the whole population need to be evaluated first. This is followed by identification of the particles' best, $P_i(t)$ and population's best $G(t)$. Next the whole population's new velocities and positions are calculated.

On the other hand, in A-PSO a particle does not need to wait for the whole population to be evaluated first before its new velocity and position is updated. After its own fitness is evaluated, the particle checks whether the current fitness contributes to new $P_i(t)$ or $G(t)$ and update the best values accordingly. Then the particle's new velocity and position are immediately calculated. The flowchart in Figure 2 shows sequential implementation of A-PSO. A-PSO is suitable for parallel implementation [3]–[5]

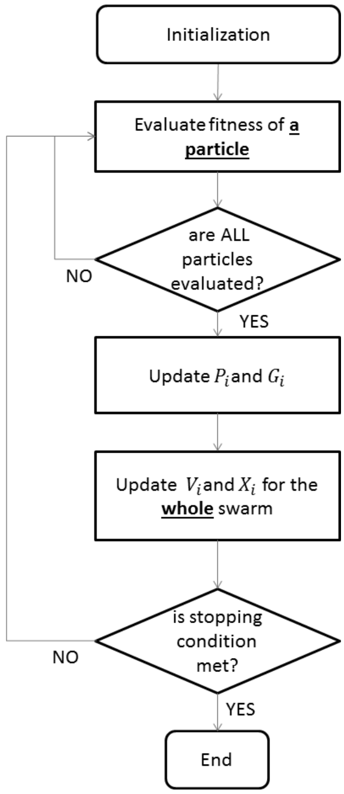

Figure 1. S-PSO Algorithm

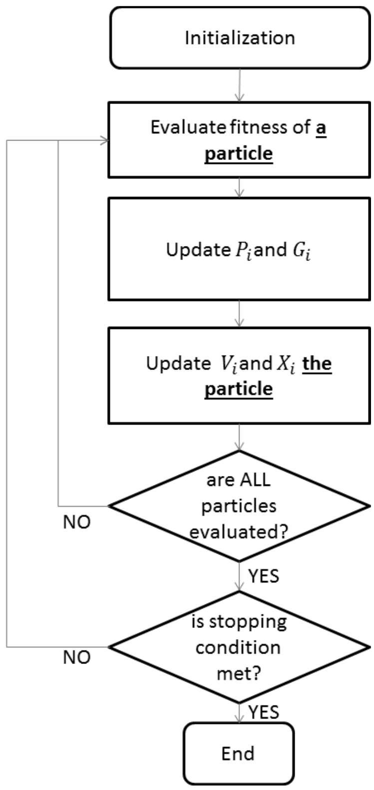

Figure 2. A-PSO Algorithm

3. TRANSITIONAL PSO

Many research had been conducted towards better performing PSO. For example, the inertia weight is introduced to the velocity update equation of PSO so that exploration and exploitation is balanced and better performance is achieved [6]. Ever since its introduction, inertia weight had become part of the standard PSO [7]. Constriction factor had been introduced as an additional parameter in PSO's velocity update equation [8]. Similar to inertia weight, it is used to control the exploration and exploitation of the particle.

In other works, methods such as reinitialization [9]–[12] and relearning [13] are proposed to improve PSO. Other popular approach to improve performance of PSO is through the control of information sharing flow, such as in [14]. Hybridization of PSO with other optimization method has also been proposed and reported to be able to give a better performance [15],[16].

However, little is known on how the particle update strategy can be manipulated for improvement of PSO. Hence this work attempt to improve the performance of PSO via its iteration strategy. A PSO algorithm that transit from asynchronous strategy to synchronous strategy, transitional PSO (T-PSO), is proposed here. Exploration is favored during the early phase of the search. Therefore, T-PSO algorithm starts with asynchronous update to benefit from its strength in exploration. A counter, δ, is used in T-PSO. The counter is incremented;

$$\delta = \delta + 1 \tag{6}$$

if $G(t)$ is not changed from one iteration to the next; $G(t) = G(t-1)$. As the search progress and no new improved solution is detected for S iteration;

$$\delta > S \tag{7}$$

the swarm changes its update mechanism to synchronous strategy. In synchronous strategy, the information gathered is exploited and fine tuned so that better solution can be achieved. A standard PSO with inertia weight is used for T-PSO. This method does not introduce complex calculation or additionsl loop, therefore, the complexity of T-PSO similar to S-PSO and A-PSO. The T-PSO algorithm is shown in Figure 3 and its pseudocode is in Figure 4.

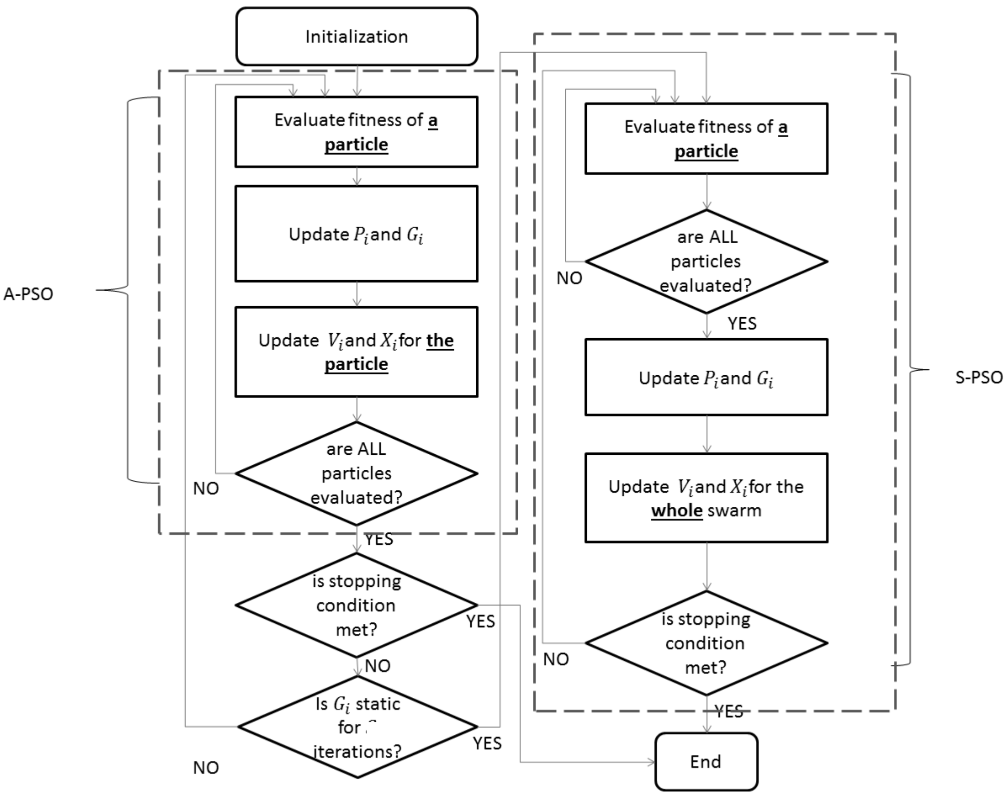

Figure 3. T-PSO Algorithm

```
Initialize the population;
δ = 0;
While not stopping condition
    While δ < S
        For each particle
            Evaluate fitness;
            Update P_i(t) & G(t);
            Update X_i(t) & V_i(t);
        End
        If G(t) = G(t − 1)
            δ = δ + 1;
        End
    End
    For each particle
        Evaluate fitness;
        Update P_i(t);
    End
    Update G(t);
    For each particle
        Update X_i(t) & V_i(t);
    End
End
```

Figure 4. T-PSO Pseudocode

4. EXPERIMENT, RESULTS AND DISCUSSION

The performance of T-PSO is tested using CEC2014's benchmark functions for single objective optimization. The benchmark functions consist of 30 functions, which are three rotated unimodal functions, 13 multimodal problems which are either shifted only or shifted and rotated, six hybrid functions, and eight composition functions. The search space of the functions is bounded within [-100, 100] along every dimension. The functions are listed in Table 1 and the details of the function can be found at http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2014/CEC2014.htm.

T-PSO is compared with S-PSO and A-PSO. The population size of each swarm is 100 and linearly decreasing inertia weight with the range, [0.4, 0.9] is used. Both learning factors of PSO are set to 2 and the velocity is clamped according to the search space, [-100, 100]. These settings for PSO are made according to [6],[17]. The maximum iteration is set to 2000. The problems dimension size is 50. Each of the test is repeated 50 times.

For T-PSO, the S threshold is set to 100. Where, if $G_i$is found to be static for 100 iterations, then the population of T-PSO changes from asynchronous update to synchronous update. The averaged performance of the PSO variations are listed in Table 2. The bolded values are the best achieved among the three iteration strategies tested. It can be seen that T-PSO found the best solution more often than S-PSO and A-PSO. T-PSO found the best solution for 17 functions out of the 30 test functions, whereas S-PSO found the best for the other 13 functions. A-PSO failed to outperform both T-PSO and S-PSO.

Based on the average performance, statistical analysis is perfomed. The Friedman test is conducted and the three variations of PSO are ranked. The average ranks are shown in Table 3. T-PSO is ranked the best. This is closely followed by S-PSO. A-PSO is ranked the lowest among the three variations. According to this rank, the Friedman statistical value is calculated and it shows significant difference exist between the algorithms. The Holm posthoc procedure is chosen to identify the significant difference. The statistical value of Holm posthoc procedure is shown in Table 4. The numbers show that T-PSO and S-PSO are statistically on par with each other. Both T-PSO and S-PSO are significantly better than A-PSO with significance level of $\alpha = 0.05$.

The boxplots in Figure 5 show the distribution of the results obtained by T-PSO, S-PSO and A-PSO. It can be seen that T-PSO and S-PSO have lower and smaller boxplot compared to A-PSO. This indicate better and more stable performance.

Table 1. CEC2014's Benchmark Functions

	Function ID	Function	Ideal Fitness
Unimodal Function	f1	Rotated High Conditioned Elliptic Function	100
	f2	Rotated Bent Cigar Function	200
	f3	Rotated Discus Function	300
Simple Multimodal Function	f4	Shifted and Rotated Rosenbrock's Function	400
	f5	Shifted and Rotated Ackley's Function	500
	f6	Shifted and Rotated Weierstrass Function	600
	f7	Shifted and Rotated Griewank's Function	700
	f8	Shifted Rastrigin's Function	800
	f9	Shifted and Rotated Rastrigin's Function	900
	f10	Shifted Schwefel's Function	1000
	f11	Shifted and Rotated Schwefel's Function	1100
	f12	Shifted and Rotated Katsuura Function	1200
	f13	Shifted and Rotated HappyCat Function	1300
	f14	Shifted and Rotated HGBat Function	1400
	f15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500
	f16	Shifted and Rotated Expanded Scaffer's F6 Function	1600
Hybrid Function	f17	Hybrid Function 1 (N=3)	1700
	f18	Hybrid Function 2 (N=3)	1800
	f19	Hybrid Function 3 (N=4)	1900
	f20	Hybrid Function 4 (N=4)	2000
	f21	Hybrid Function 5 (N=5)	2100
	f22	Hybrid Function 6 (N=5)	2200
Composite Function	f23	Composition Function 1 (N=5)	2300
	f24	Composition Function 2 (N=3)	2400
	f25	Composition Function 3 (N=3)	2500
	f26	Composition Function 4 (N=5)	2600
	f27	Composition Function 5 (N=5)	2700
	f28	Composition Function 6 (N=5)	2800
	f29	Composition Function 7 (N=3)	2900
	f30	Composition Function 8 (N=3)	3000
		Search Range: $[-100, 100]^D$	

Table 2. Algorithms' Averaged Performance

Function ID	T-PSO	S-PSO	A-PSO
f1	1.8811E+07	2.3317E+07	9.6780E+10
f2	2.5539E+04	1.6171E+06	7.1563E+11
f3	2.4349E+04	2.0228E+04	2.5847E+10
f4	6.2680E+02	6.4006E+02	8.5606E+05
f5	5.2111E+02	5.2109E+02	5.2197E+02
f6	6.2857E+02	6.2994E+02	7.1490E+02
f7	7.0001E+02	7.0001E+02	7.7211E+03
f8	8.5763E+02	8.6176E+02	2.3308E+03
f9	1.0360E+03	1.0506E+03	3.1912E+03
f10	2.6276E+03	2.5927E+03	2.5277E+04
f11	7.2592E+03	8.0362E+03	2.5106E+04
f12	1.2028E+03	1.2028E+03	1.2255E+03
f13	1.3006E+03	1.3006E+03	1.3245E+03
f14	1.4006E+03	1.4006E+03	3.2076E+03
f15	1.5228E+03	1.5202E+03	1.9224E+10
f16	1.6216E+03	1.6217E+03	1.6260E+03
f17	2.8280E+06	2.7389E+06	4.6500E+10
f18	2.0573E+04	2.6976E+03	1.4964E+11
f19	1.9647E+03	1.9682E+03	1.0694E+05
f20	1.1507E+04	1.2246E+04	5.6039E+10
f21	1.4981E+06	1.8508E+06	3.0662E+10
f22	3.0373E+03	3.1215E+03	2.1121E+09
f23	2.6482E+03	2.6481E+03	2.4277E+04
f24	2.6759E+03	2.6763E+03	1.7339E+04
f25	2.7213E+03	2.7218E+03	1.0138E+04
f26	2.7734E+03	2.7714E+03	1.2577E+04
f27	3.7421E+03	3.7312E+03	9.8385E+04
f28	5.0044E+03	5.3747E+03	8.7626E+04
f29	7.3083E+06	1.1665E+07	6.1162E+10
f30	3.9627E+04	3.9539E+04	1.4758E+09

Table 3. Friedman' Average Rank

Algorithm	Average Rank
T-PSO	1.4
S-PSO	1.6
A-PSO	3

Table 4. Holm Posthoc Procedure

Algorithm	P	Holm
T-PSO vs A-PSO	0	0.016667
S-PSO vs A-PSO	0	0.025
T-PSO vs S-PSO	0.438578	0.05

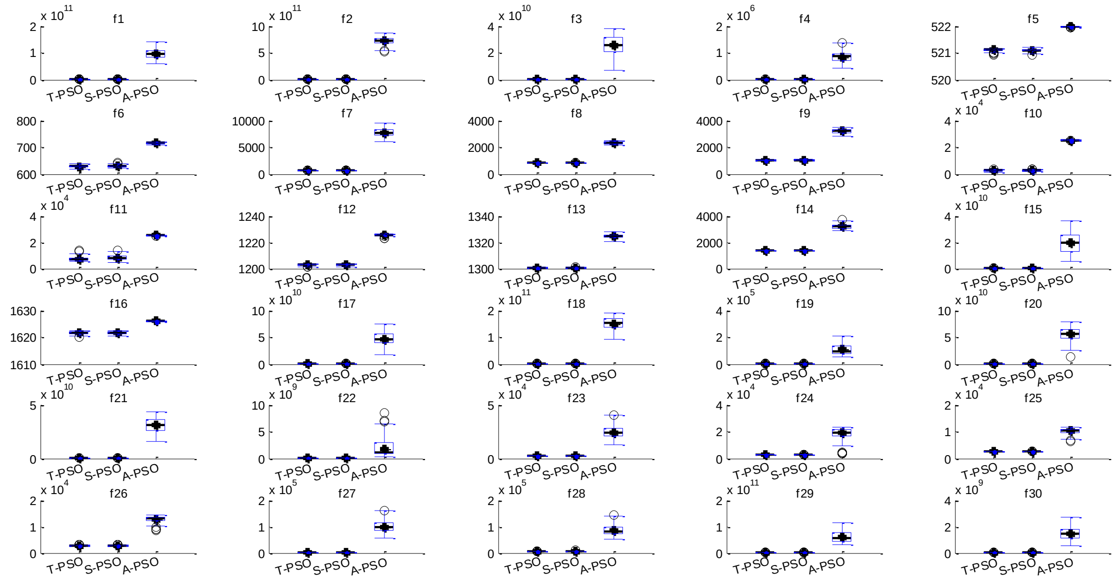

Figure 5. Quality of Results

5. CONCLUSION

T-PSO is proposed in this work. T-PSO attempts to improve PSO via its iteration strategy. The CEC2014 single objective test suite is used here. From the experiment conducted, T-PSO's performance is ranked the best. This proves that iteration strategy does influence the performance of PSO. Manipulation of the iteration strategy is able to improve the performance of PSO. However, more works need to be conducted to fully understand the potential of iteration strategy in improving PSO. Issues such as, which starting strategy is the best and how to select the optimal value of *S* need to be further explored.

ACKNOWLEDGEMENTS

This research is funded by the Multimedia University's Mini Fund (MMUI/150076), and UM-Postgraduate Research Grant (PG097-2013A).

REFERENCES

[1] A. P. Engelbrecht, "Particle Swarm Optimization: Iteration Strategies Revisited," in *BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence*, pp. 119–123, 2013.
[2] J. R. Vilela, *et al.*, "A Performance Study on Synchronicity and Neighborhood Size in Particle Swarm Optimization," *Soft Comput.*, vol. 17, pp. 1019–1030, 2013.
[3] B. I. Koh, *et al.*, "Parallel Asynchronous Particle Swarm Optimization.," *Int. J. Numer. Methods Eng.*, vol/issue: 67(4), pp. 578–595, 2006.
[4] G. Venter and J. S. Sobieski, "A Parallel Particle Swarm Optimization Algorithm Accelerated by Asynchronous Evaluations," in *World Congresses of Structural and Multidisciplinary Optimization*, pp. 1–10, 2005.
[5] S. B. Akat and V. Gazi, "Decentralized Asynchronous Particle Swarm Optimization," in *IEEE Swarm Intelligence*

Symposium, pp. 1–8, 2008.
[6] Y. Shi and R. Eberhart, "A Modified Particle Swarm Optimizer," in *IEEE International Conference on Evolutionary Computation*, pp. 69–73, 1998.
[7] M. Clerc, "Standard Particle Swarm Optimisation," 2012.
[8] M. Clerc and J. Kennedy, "The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space," *IEEE Trans. Evol. Comput.*, vol/issue: 6(1), pp. 58–73, 2002.
[9] K. J. Binkley and M. Hagiwara, "Balancing Exploitation and Exploration in Particle Swarm Optimization: Velocity-based Reinitialization," *Trans. Japanese Soc. Artif. Intell.*, vol/issue: 23(1), pp. 27–35, 2008.
[10] J. Guo and S. Tang, "An Improved Particle Swarm Optimization with Re-initialization Mechanism," in *International Conference on Intelligent Human-Machine Systems and Cybernetics*, pp. 437–441, 2009.
[11] S. Cheng, *et al.*, "Promoting Diversity in Particle Swarm Optimization to Solve Multimodal Problems," in *Neural Information Processing: 18th International Conference, ICONIP 2011, Shanghai, China, November 13-17, 2011, Proceedings, Part II*, vol. 7063 LNCS, no. 60975080, B.-L. Lu, L. Zhang, and J. Kwok, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 228–237, 2011.
[12] K. K. Budhraja, *et al.*, "Exploration Enhanced Particle Swarm Optimization using Guided Re-Initialization," *Adv. Intell. Syst. Comput.*, vol/issue: 2(02), pp. 277–288, 2013.
[13] L. Han and X. He, "A Novel Opposition-Based Particle Swarm Optimization for Noisy Problems," in *Third International Conference on Natural Computation*, pp. 624–629, 2007.
[14] J. Kennedy, "Stereotyping: Improving particle swarm performance with cluster analysis," in *Congress on Evolutionary Computation*, pp. 1507–1512, 2000.
[15] K. Premalatha and A. M. Natarajan, "Hybrid PSO and GA for Global Maximization," *Int. J. Open Probl. Compt. Math*, vol/issue: 2(4), pp. 597–608, 2009.
[16] S. Yu, *et al.*, "A Hybrid Particle Swarm Optimization Algorithm based on Space Transformation Search and A Modified Velocity Model," *Int. J. Numer. Anal. Model.*, vol/issue: 9(2), pp. 371–377, 2012.
[17] R. Eberhart and Y. Shi, "Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization," in *Congress on Evolutionary Computation*, pp. 84–88, 2000.

BIOGRAPHIES OF AUTHORS

Ms Nor Azlina Ab Aziz is a lecturer in the Faculty of Engineering and Technology at Multimedia University, Melaka. She is interested in the field of soft computing and its application in engineering problems. More specifically, her focus is in the area of swarm intelligence and nature inspired optimization algorithm.

Associate Professor Dr. Zuwairie Ibrahim obtained a B.Eng (Mechatronics) and M.Eng (Electrical) from Universiti Teknologi Malaysia, and Ph.D in Electrical Engineering from Meiji University, Japan. He is currently Associate Professor of Faculty of Electrical and Electronics Engineering, Universiti Malaysia Pahang. His research interests are on Computational Intelligence, Artificial Intelligence, Molecular and DNA Computing, Machine Vision and Automated Visual Inspection. He has supervised 7 postgraduate students and currently supervising 6 PhD and 2 Master students. He has received various funding from both local and international bodies, as well as holding 3 patents. He has published articles in 2 books, 76 journal articles and many conference proceedings. He is also involved in consultations, the editorial board of local and international journals and has reviewed many technical papers. He received awards of the Best Instrumentation Paper Award in International Conference on Control, Instrumentation, and Mechatronics Engineering 2007, Session Best Presentation Award in Joint 2nd International Conference on Soft Computing and Intelligent Systems and 5th International Symposium on Advanced Intelligent Systems 2004 and Student paper award in the IEEE International Symposium on Industrial Electronics 2004. He has also won several medals in various research exhibitions and has been invited to give lectures and tutorials in his research fields. He is a member of IEEE, BEM and mSET.

Dr Marizan Mubin graduated with an honors degree in telecommunication engineering, from University of Malaya in 2000 and she had continued her post-graduate studies in University of Newcastle Upon Tyne, UK (the MSc in communication and signal processing). In 2003, she was awarded a scholarship by Japan International Cooperation Agency (JICA) to pursue a doctoral degree in Tokai University, Japan. She is currently attached to the Department of Electrical Engineering, University of Malaya as a senior lecturer. Her main research interest is in non-linear control systems.

Dr Sophan Wahyudi Nawawi joined Universiti Teknologi Malaysia since 1999. He received his PhD degree from the Universiti Teknologi Malaysia in 2010. In 2006, he collaborates with control research group at Hong Kong University of Science and Technology as a research scholar and attached to HKUST spin off company Googol Technology (HK) ltd. He was promoted to a senior lecturer in 2010.

Ms Nor Hidayati Abdul Aziz graduated from Multimedia University in 2002 in Electronics Engineering majoring in Computer, and completed her Master's Degree of Engineering at Universiti Teknologi Malaysia in 2005. She worked in Telekom Malaysia Berhad as a field engineer for 4 years immediately after graduation, and then joined Multimedia University as a lecturer after finishing her Master's Degree. She is currently pursuing her PhD part-time at Universiti Malaysia Pahang in Computational Intelligence. At the moment, she is serving Multimedia University at the Faculty of Engineering and Technology, Melaka campus.