VOL. 10, NO. 21, NOVEMBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences
©2006-2015 Asian Research Publishing Network (ARPN). All rights reserved.
www.arpnjournals.com

A RANDOM SYNCHRONOUS-ASYNCHRONOUS PARTICLE SWARM OPTIMIZATION ALGORITHM WITH A NEW ITERATION STRATEGY

Nor Azlina Ab Aziz[1,2], Shahdan Sudin[3], Marizan Mubin[1], Sophan Wahyudi Nawawi[3] and Zuwairie Ibrahim[4]
[1]Faculty of Engineering, University of Malaya, Kuala Lumpur, Malaysia
[2]Faculty of Engineering and Technology, Multimedia University, Melaka, Malaysia
[3]Faculty of Electrical Engineering, Universiti Teknologi Malaysia, Johor, Malaysia
[4]Faculty of Electrical and Electronics Engineering, University Malaysia Pahang, Malaysia
E-Mail: azlina.aziz@mmu.edu.my

ABSTRACT

Particle swarm optimisation (PSO) is a population-based stochastic optimisation algorithm. Traditionally the particles update sequence for PSO can be categorized into two groups, synchronous (S-PSO) or asynchronous (A-PSO) update. In S-PSO, the particles' performances are evaluated before their velocity and position are updated, while in A-PSO, each particle's velocity and position is updated immediately after individual performance is evaluated. Recently, a random asynchronous PSO (RA-PSO) has been proposed. In RA-PSO, particles are randomly chosen to be updated asynchronously, the randomness improves swarm's exploration. RA-PSO belongs to the asynchronous group. In this paper, a new category; hybrid update sequence is proposed. The new update sequence exploits the advantages of synchronous, asynchronous, and random update methods. The proposed sequence is termed as, random synchronous-asynchronous PSO (RSA-PSO). RSA-PSO divides the particles into groups. The groups are subjected to random asynchronous update, while the particles within a chosen group are updated synchronously. The performance of RSA-PSO is compared with the existing S-PSO, A-PSO, and RA-PSO using CEC2014's benchmark functions. The results show that RSA-PSO has a superior performance compared to both A-PSO and RA-PSO, and as good as S-PSO.

Keywords: asynchronous, particle swarm optimisation, random, synchronous.

INTRODUCTION

Particle swarm optimisation (PSO) algorithm was introduced by Kennedy and R. Eberhart in 1995 [1]. It looks for optimal solution of an optimisation problem by mimicking the social behaviour seen in nature, such as flock of birds looking for food. In PSO, these organisms are represented by a swarm of agents called particles. The particles move within the search area looking for optimal solution by updating their velocity and position. These values are influenced by the personal experience of the particles and their social interaction.

PSO has gained a lot of interest since its introduction. However, there are a few fundamental aspects of PSO which are not thoroughly explored yet, such as the synchronicity of the particle update sequence, which is also known as iteration strategy [2]. The traditional PSO iteration strategies can be divided into two categories, synchronous and asynchronous, as shown in Figure-1. In synchronous PSO (S-PSO), a particle's information on the neighbourhood's best found solution is updated after the fitness of the whole swarm is evaluated. The synchronous update in S-PSO provides perfect information on the fitness of the whole swarm. Thus, allowing the swarm to exploit the information of the best neighbour. However, this could cause the particles to converge too fast. Many works has reported that synchronous update leads to a strong exploitation by the particles in S-PSO. However, according to [3] if the improvement of the best found solution is marginal, the synchronous update is not only reducing the exploration, but it also hinders the particles from exploiting and benefiting from the information available.

Another variation of PSO, known as asynchronous PSO (A-PSO), has been discussed in [4]. In A-PSO, the particles evaluate their fitness and update their velocity and position on their own without the need to synchronize with the whole swarm. As soon as a particle finished evaluating its fitness, it immediately identifies the best solution available in the swarm and updates its velocity and position. Hence, the particles updated at the beginning of iteration use more information from the previous iteration, while particles at the end of the iteration use more information from the same iteration to determine the best solution of the swarm. This allows various leads from different best solutions, thus providing more exploration by the swarm.

Recently, random A-PSO (RA-PSO) has been introduced [5], [6]. RA-PSO belongs to asynchronous update group. In RA-PSO, particles to be updated are selected randomly. Therefore, in an iteration, some particles might be updated more than once while other particles may not be updated at all. The randomness helps to prevent particles from being trapped in local optima and encourage more exploration. This is due to various degree of information within the swarm, because some particles might not be updated for several iterations thus possessing outdated information while others might be updated more than once in a single iteration.

In this study, synchronous, asynchronous, and random updates are merged so that the advantages of each of these methods can be utilized and the weaknesses can be overcome. The proposed random synchronous-asynchronous PSO (RSA-PSO) algorithm divides the particles into smaller groups. The group to be updated are randomly chosen one at a time, asynchronously. The particles within a chosen group are updated synchronously. The search for the optimal solution by the particles in RSA-PSO is led by the best member of the groups and the swarm's best. The RSA-PSO improves the performance of PSO by balancing the exploitation provided by synchronous update, with the exploration by random asynchronous update. RSA-PSO is a method belongs to new category of update strategy; hybrid update strategy, as shown in Figure-1. The CEC2014's benchmark functions for single objective real-parameter numerical optimization are used to evaluate the performance of RSA-PSO and the existing methods, S-PSO, A-PSO and RA-PSO. The results of the existing methods show that stronger exploitation in S-PSO is crucial in ensuring good performance. The RSA-PSO performs as good as S-PSO which is the best update strategy among the traditional methods.

This paper is organized as follows. The two traditional update categories are reviewed in section 2, where S-PSO, A-PSO, and RA-PSO algorithms are discussed. The proposed RSA-PSO algorithm is described in section 3. In section 4, the experiments are discussed, where the CEC2014's benchmark functions are used. The results of the experiments are presented and discussed in section 5. Finally, this work is concluded in section 6.

UPDATE SEQUENCE IN PSO

Traditionally PSO is either implemented as a synchronous update algorithm or asynchronous update. Synchronous update method is the typical method used in PSO while asynchronous update is another available approach. Asynchronous update is a more accurate natural model, it increases the potential of parallelization of an algorithm [7], [8].

In PSO, the search for optimal solution is conducted by a swarm of P particles. At time t, particle ith has a position, xi(t), and velocity, vi(t). The position represents a solution suggested by the particle while velocity is the rate of change to the next position with respect to the current position. At the start of the algorithm, these two values (position and velocity) are randomly initialised [9]. In the subsequent iterations the search process is conducted by updating these values using equation (1) and equation (2) until a position with ideal fitness is attained or maximum number of iteration, T, is reached.

$$v_i(t) = \omega v_i(t-1) + c_1 r_1 (pBest_i(t) - x_i(t-1)) + c_2 r_2 (gBest_i t) - x_i(t-1)) \tag{1}$$

$$x_i(t) = v_i(t) + x_i(t-1) \tag{2}$$

To prevent the particles from venturing too far from the feasible region, the vi(t) value is clamped to ±Vmax. In equation (1), c1 and c2 are the learning factors that control the effect of the cognitive and social influence on a particle. Typically, both c1 and c2 are set to 2 [10]. Two independent random numbers r1 and r2 in the range of [0.0, 1.0] are incorporated in the velocity equation. These random terms provide stochastic behaviour to the particles. Inertia weight, ω, is a term added to control the particles fine tuning [11]. To ensure convergence, a time decreasing inertia weight [12] is more favourable than a fixed inertia weight.

An individual success in PSO is affected by the particle's own effort, experience and also by its surrounding neighbours. As shown in equation (1) the particle's velocity is updated using pBesti(t), which is the best position found so far by particle ith and gBest(t), which is the best position found by the swarm up to tth iteration.

The particle's position, xi(t), is updated using equation (2), in which a particle's next search is launched from its previous position, xi(t-1). Typically, xi(t) is bounded to prevent the particles from searching in an infeasible region. The fitness of xi(t) is evaluated by a problem-dependent fitness function.

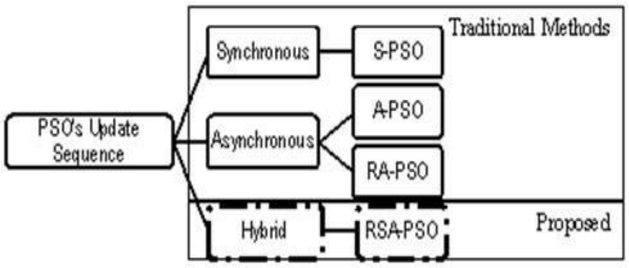

Figure-1. Categories of PSO's update sequence.

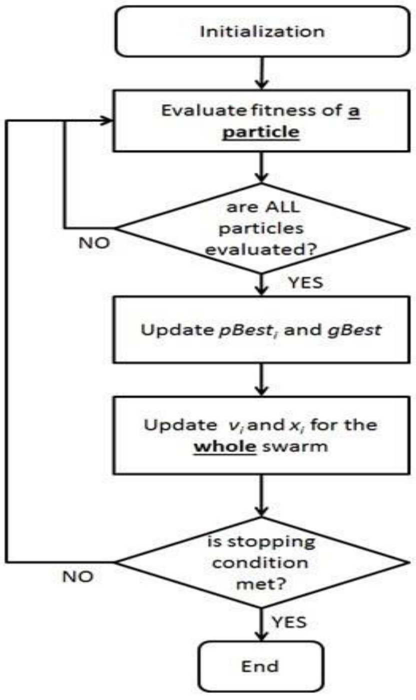
Initialization
Evaluate fitness of a particle
are ALL particles evaluated?
NO
YES
Update pBest_i and gBest
Update v_i and x_i for the whole swarm
is stopping condition met?
NO
YES
End

Figure-2. S-PSO's flowchart.

VOL. 10, NO. 21, NOVEMBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences
©2006-2015 Asian Research Publishing Network (ARPN). All rights reserved.
www.arpnjournals.com

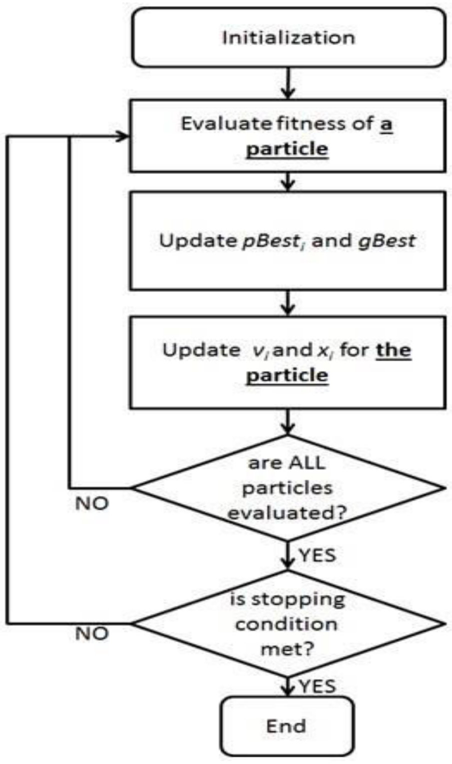
Initialization
Evaluate fitness of a particle
Update $pBest_i$ and $gBest$
Update v_i and x_i for the particle
are ALL particles evaluated?
NO
YES
is stopping condition met?
NO
YES
End

Figure-3. The A-PSO's flowchart.

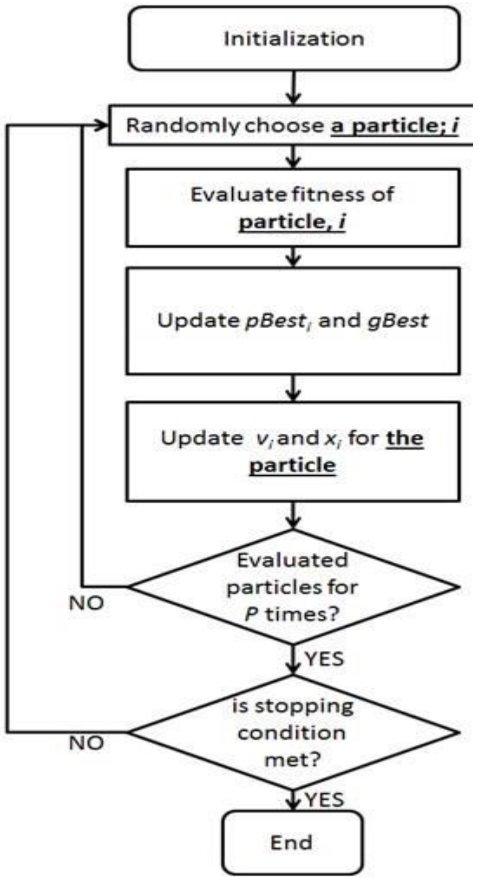
Initialization
Randomly choose a particle; i
Evaluate fitness of particle, i
Update $pBest_i$ and $gBest$
Update v_i and x_i for the particle
Evaluated particles for P times?
NO
YES
is stopping condition met?
NO
YES
End

Figure-4. The RA-PSO's flowchart.

Synchronous update

In S-PSO the $pBest_i$ after particle i^{th} is updated and *gBest* is updated after all the particles are evaluated. This is followed by the positions and velocities update of the entire swarm. The S-PSO algorithm is shown in Figure-2.

In a swarm with *P* number of particles, the fitness evaluation is done *P* times per iteration. Thus, the maximum number of fitness evaluation by S-PSO in a run is $(P \times T)$.

Asynchronous update

In synchronous update a particle needs to wait for the whole swarm to be evaluated before it can move to a new position and continue its search. Hence, the first evaluated particle is idle for the longest time, waiting for the whole swarm to be updated. PSO is a nature inspired algorithm. In nature, an individual is typically free to move without the need to synchronize its move with others. This concept is adopted in asynchronous update, where the particles are updated independently without synchronization with the whole swarm.

The flowchart in Figure-3 shows the A-PSO algorithm. A particle evaluates its fitness. After that the particle immediately selects *gBest* and updates its $pBest_i$. The *gBest* is selected depending on the swarm conditions during a particular particle's update process. Using the latest *gBest* and $pBest_i$ a particle updates its velocity and position using the same equations as S-PSO. This process is then continued by the next particles until either ideal solution is found or *T* iterations are reached.

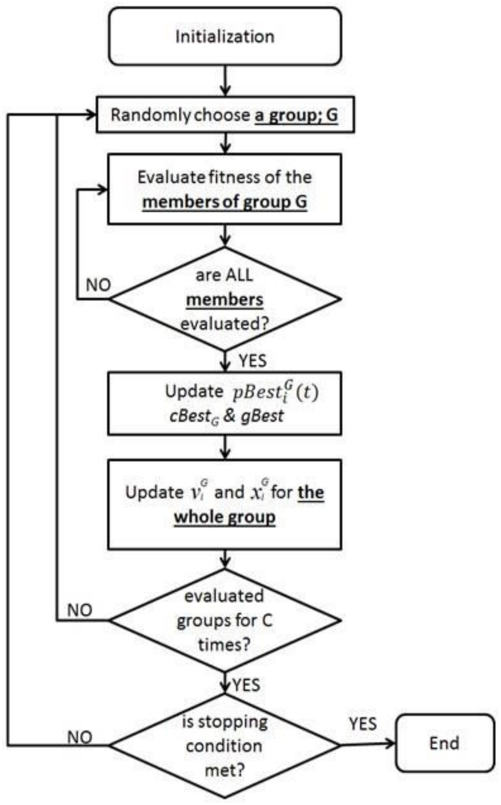
Initialization
Randomly choose a group; G
Evaluate fitness of the members of group G
are ALL members evaluated?
NO
YES
Update $pBest_i^G(t)$ $cBest_G$ & $gBest$
Update v_i^G and x_i^G for the whole group
evaluated groups for C times?
NO
YES
is stopping condition met?
NO
YES
End

Figure-5. The RSA-PSO's flowchart.

VOL. 10, NO. 21, NOVEMBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

Even though, the flow of A-PSO is different than S-PSO, the fitness function is still called for P times per iteration, once for each particle. Therefore, the maximum number of fitness evaluation is (P×T).

Random asynchronous PSO (RA-PSO) is a variation of A-PSO algorithm [5]. The RA-PSO algorithm is presented in Figure-4. The particles to be updated are chosen randomly with repetition allowed. Therefore, a particle can be updated more than once or none at all in a particular iteration. The randomness causes the swarm to have mixture state of particles by the end of each iteration, some particles possessing up to date information while some holding out-dated information. This provides various degrees of information within the swarm. Since the selection of the particles is done randomly, the information flow is different from one iteration to another. This improves the exploration of the particles and prevents them from being trapped in local optima. In RA-PSO, at most (P×T) fitness evaluations are performed. This is similar to S-PSO and A-PSO algorithms.

THE PROPOSED RANDOM SYNCHRONOUS ASYNCHRONOUS PSO

The advantage of synchronous update is strong exploitation which leads to good solution. Meanwhile, the variety of gBest information used in asynchronous update contributes to the strength of A-PSO which is diversity and exploration. RA-PSO enhances the exploration of A-PSO through randomization of the particles' update sequence. In the proposed RSA-PSO, the three update methods are combined to benefit from their advantages and strengths. The RSA-PSO update strategy does not fall within synchronous or asynchronous update, it is a hybrid method. The proposed algorithm is shown in Figure-5.

Table-1. Parameters setting for RSA-PSO, S-PSO, A-PSO and RA-PSO.

Parameter		**Value**
Number of runs for each experiment		50
Number of iterations		2000
Number of particle		100
Velocity clamping, $V_{\max}$		100
Range of inertia weight, ω		0.9-0.5
Learning factors	c_1	2
	c_2	2

The algorithm starts with initialization of particles. The particles in RSA-PSO are divided into *C* groups, which consist of *N* number of particles each. Initially, *C* central particles, one for each group, are randomly placed in the search space. This is followed by random placement of (*N*-1) number of members for each group. The random placements of the members are within the radius of ±Δ from the central particle of their respective group. The Δ is defined as the initial maximum distance of a particle from the central particle of its group. This parameter is only used once throughout the execution of the algorithm, which is during the initialization phase. The membership of the groups' remains fixed throughout the search process. The total number of particles, *P*, for RSA-PSO algorithm is *C*×*N*.

In an iteration, the algorithm randomly chooses the groups to be updated. The selection is done one by one for *C* times. Similar to RA-PSO, repetition is allowed. Therefore, there is a possibility that a group is updated more than once in a single iteration or may not be selected at all. Thus, at the end of an iteration, the swarm may consist of groups having updated velocities and positions, and groups with velocities and positions from previous search.

The particles of a chosen group, *G*, are updated synchronously. The performance of all the members of the group, is evaluated before their $pBest_i^G$ pBest$_i^G$ and the group's cBest$_G$ are identified. The $cBest_G$ cBest$_G$ is the best $pBest_i^G$ of group *G*. If the cBest$_G$is better that *gBest* then *gBest* is updated. The velocity at iteration *t* of particle *i*^th^ that belongs to group $G, v_i^G(t)$, is updated using the following equation:

$$v_i^G(t) = \omega v_i^G(t-1) + c_1 r_1 (cBest_G(t) - x_i^G(t-1)) + c_2 r_2 (gBest\ \ (t) - x_i^G(t-1)) \tag{3}$$

The equation (3) shows that the information used to update the velocity are the current group's best, *cBest*$_G$(*t*) and the global best, *gBest*(*t*). The best *cBest*$_G$(*t*) is the *gBest*(*t*). The position of the particle, $x_i^G(t)$, is updated using equation (4).

$$x_i^G(t) = v_i^G(t) + x_i^G(t-1) \tag{4}$$

In RSA-PSO, when a group is chosen, the particles within the group are evaluated. Hence, fitness function is called for *N* times per group. The groups to be updated are chosen randomly for *C* times per iteration. Therefore, in total, *C*×*N* times of fitness evaluation is conducted every iteration. *C*×*N* is equivalent to total number of particles in the swarm, *P*. Thus, the maximum number of fitness evaluation by RSA-PSO in a run which is limited to *T* iteration is (*P*×*T*). This is similar as S-PSO, A-PSO and RA-PSO.

Experiments

The proposed RSA-PSO and the existing S-PSO, A-PSO, and RA-PSO were implemented using MATLAB R2012a. The parameter settings are summarized in Table-1. Every experiment was subjected to 50 runs. In all algorithms, the particles' velocity and position were initialized as follow. The velocity of every particle was randomly initialized within the velocity clamping range, ±$V_{\max}$, while the position of the particles was randomly initialized within the search space. A linear decreasing inertia weight ranging from 0.9 to 0.5 was employed. The cognitive and social learning factors were set to 2. The search was terminated once the number of iterations reaches 2000 or ideal fitness is attained. The parameters setting for the additional parameters in RSA-PSO are given in Table-2. These values are determined based on separate experiments. Exclusively for RSA-PSO, the members of the groups were initialized randomly around the groups' central particles and based on Δ value.

The CEC2014's benchmark functions for single objective real-parameter numerical optimization are used here to evaluate the performance of RSA-PSO, S-PSO, A-PSO and RA-PSO. The benchmark functions consist of three rotated unimodal functions, thirteen multimodal problems, six hybrid functions and eight composition functions. Due to space limitation, readers are referred to http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2014/CEC2014.htm for further details on the benchmark functions.

Nonparametric statistical analysis is conducted on the results using KEEL software. The test chosen is the Friedman test with Holm post hoc procedure, the significance level used is α=0.05. This test is suitable for comparison of more than two algorithms [14].

Table-2. Setting for the additional parameters in RSA-PSO.

Parameter	Value
Number of groups, C	10
Group size (particles per group)	10
Initial distance to group centre, Δ	50% of the length of the search space

RESULTS AND DISCUSSION

The algorithms' mean for each test functions and their average rank are shown is Table-3. The statistical analysis performed using Friedman statistical test shows that significant differences exist between the algorithms.

Holm post hoc procedure is conducted to further analyse this finding. The results of Holm procedure is shown in Table-4. Holm procedure reveals that the performance of RSA-PSO is on par with S-PSO and both RSA-PSO and S-PSO have significant difference in performance with A-PSO and RA-PSO.

Good performance of RSA-PSO is contributed by the fact that the particles are learning and exploiting information from gBest and cBest. The gBest and cBest particles have better information that can be used as reference and guidance by other particles. The random asynchronous update of the groups provides exploration in RSA-PSO thus avoiding premature convergence.

CONCLUSIONS

Random synchronous-asynchronous PSO algorithm (RSA-PSO) a new method belongs to hybrid update strategy is proposed in this paper. The particles in RSA-PSO are updated synchronously in groups. The groups, however, are chosen randomly and asynchronously updated, one group after another. A group's search is led by the group's best performer, cBestG, and the best member of the swarm, gBest. The algorithm benefits from good exploitation and fine tuning provided by synchronous update, while it takes advantage of the exploration in the asynchronous update. The exploration is further enhanced by the random selection of the group to be updated. Statistical analysis performed shows that the performance of RSA-PSO is better than A-PSO and RA-PSO and on par as S-PSO.

ACKNOWLEDGEMENTS

This research is funded by the Department of Higher Education of Malaysia under the Fundamental Research Grant Scheme awarded to UTM (4F374), 2014 MMU's Mini Fund (MMUI/150076), and UM-Postgraduate Research Grant (PG097-2013A). We are grateful to Associate Professor Dr Salvador Garcia, University of Granada for his assistance. The authors also would like to acknowledge the anonymous reviewers for their valuable comments and insights.

REFERENCES

[1] Kennedy J. and Eberhart R. 1995. Particle swarm optimization. In International Conference on Neural Networks, pp. 1942–1948.

[2] Engelbrecht A.P. 2013. Particle swarm optimization: iteration strategies revisited. In BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence, pp. 119–123.

[3] Voglis C.A., Parsopoulos K.E., and Lagaris I.E. 2012. Particle swarm optimization with deliberate loss of information. Soft Computing. Vol. 16, No. 8, pp. 1373–1392.

[4] Carlisle A. and Dozier G. 2001. An Off-The-Shelf PSO. In Proceedings of the Workshop on Particle Swarm Optimization.

[5] Rada-Vilela J., Zhang M. and Seah W. 2011. Random Asynchronous PSO. In The 5th Int. Conf. Autom. Robot. Appl., pp. 220–225.

[6] Rada-Vilela J., Zhang M., and Seah W. 2013. A performance study on synchronicity and neighborhood size in particle swarm optimization. Soft Computing Vol. 17, No. 6, pp. 1019–1030.

[7] Coleman V. 1989. The DEME mode: an asynchronous genetic algorithm. Technical Report University of Massachusetts, Amherst, MA, USA.

[8] Koh B.-I., George A.D., Haftka R.T. and Fregly B.J. 2006. Parallel asynchronous particle swarm

optimization. Int. Journal Numer. Methods Eng. Vol. 67, No. 4, pp. 578–595.

[9] Jakubcov M., Petr M., and Pavel P. 2014. A comparison of selected modifications of the particle swarm optimization algorithm. Journal Appl. Math. 2014(Article ID 293087).

[10] Kennedy J., Eberhart R. and Shi Y. 2001. Swarm Intelligence. US: Morgan Kaufmann.

[11] Shi Y. and Eberhart R. 1998. A modified particle swarm optimizer. In Proceedings of IEEE Int. Conf. Evol. Comput. pp. 69–73.

[12] Shi Y. and Eberhart R. 1998. Parameter selection in particle swarm optimization. In Proceedings of the 7th International Conference on Evolutionary Programming VII (EP '98), V. William Porto, N. Saravanan, Donald E. Waagen, and A. E. Eiben (Eds.). pp. 591-600

[13] Dioşan L. and Oltean M. 2006. Evolving the structure of the particle swarm optimization algorithms. In Proceedings of 6th Eur. Conf. Evol. Comput. Combinatorial Optim. (EvoCOP ’06), Lect. Comput. Sci., Vol. 3906, pp. 25–36.

[14] Derrac J., García S., Molina D., and Herrera F. 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol. Comput., Vol. 1, No. 1, pp. 3–18.