Modified Particle Swarm Optimization with Novel Modulated Inertia for Velocity Update

Abdul Hadi Hamdan[#1], Fazida Hanim Hashim[#2], Abdullah Zawawi Mohamed[*3], W. M. Diyana W. Zaki[#4], Aini Hussain[#5]

[#] Engineering Department of Electric, Electronic & System, Universiti Kebangsaan Malaysia, Bangi, Malaysia.
[1] ab_hadi2329@yahoo.com
[2] fazida@ukm.edu.my
[*] Fujitsu Systems Global Solutions, Petaling Jaya, Malaysia.
[3] abdullah.zawawi@fsgs.my.fujitsu.com

***Abstract*— Particle swarm optimization (PSO) is a population-based stochastic search algorithm for searching the optimal regions from multidimensional space, inspired by the social behaviour of some animal species. However, it has its limitations such as being trapped into a local optima and having a slow rate of convergence. In this paper, a new method of creating a combination of a developed Accelerated PSO and a new modulated inertia coefficient for the velocity update has been proposed. Random term based on particle neighbourhood has been added in the position update formula, inspired by the Artificial Bee Colony (ABC) algorithm. To verify the proposed modified PSO, experiments were conducted on several benchmark optimization problems. The results show that the proposed algorithm is superior in comparison with standard PSO and accelerated PSO algorithms.**

Keyword- Velocity Update, Global Best, Modulated Inertia, Particle Swarm Optimization

I. INTRODUCTION

Particle swarm optimization (PSO) is a population-based stochastic search algorithm for searching the optimal regions from multidimensional space. It is an optimization method inspired by social behaviour of fish schooling and birds flocking and was defined by Kennedy and Eberhart in 1995 [1]. PSO is inspired by general artificial life and random search methods applied in evolutionary algorithm [2]. When travelling in a group, individual birds and fishes have the ability to move without colliding with each other. This is achieved by having each member follow its own group and adjust its position and velocity using the group information, thereby reducing the burden of individual's effort in searching the target (food, shelter). Particle swarm optimization is quite similar to genetic algorithm because both are population-based and are equally effective [2]. The advantage of the PSO method lies in its lower complexity while having comparable performance as there are only a few parameters to be adjusted and manipulated. It also has better computational efficiency, need less memory space, and is less dependent on the CPU speed. Another advantage of PSO over derivative-based local search methods is that when solving a complicated optimization problem, the gradient information is not needed to perform the iterative search.

In PSO, a member in the swarm is called a particle, representing a potential solution of a problem. A population of particles starts to move in a search space by following the current optimum particles and changing their positions in order to find out the optima. The position of a particle refers to a possible solution of the function to be optimized. Each particle has a fitness value, determined by evaluating a function using the particle's position, and a velocity. The experiences of the swarm are used as a learning tool in the search for the global optima [3]. While it has been successfully used to solve many optimization tests and real-life optimization problems, the PSO method often suffers from premature convergence and getting trapped in a local optimum region.

In order to achieve better algorithm performance, the original PSO algorithm has been modified by many researchers to be used in various types of applications. Shi et al. proposed an extended PSO based on inertia weight. A large inertia weight facilitates a global search while a small inertia weight facilitates a local search. By changing the inertia weight dynamically, the search capability is dynamically adjusted [4]. Nickabadi et al. proposed a new dynamic inertia weight PSO. While the former uses the fitness or iteration number as the basis

for its inertia weight updating scheme, the latter proposed a method of using the success rate of the swarm to determine the inertia weight [5]. To improve the flexibility of mutation in PSO, Hui Wang presents an adaptive mutation strategy. In the new approach, three different mutation operators: Gaussian, Cauchy, and Lévy, are utilized [6]. Xin-She Yang simplified the standard PSO to become Accelerated PSO by neglecting the individual particle best. The reason for neglecting the individual particle best is that the diversity resulting from it can be simulated by introducing some randomness.

Jian Hu [3] proposed a general fitness evaluation strategy (GFES), in which a particle is evaluated in multiple subspaces and different contexts in order to take diverse paces towards the destination position [3]. In [8], woven fabrics with the desired quality and low manufacturing cost were designed by optimizing the weave parameters using PSO to find the appropriate combination of weave parameters. In [9], the author added differential evolution (DE) mutation operator to the accelerated particle swarm optimization (APSO) algorithm to solve numerical optimization problems so as to speed up convergence. The mutation operator tunes the newly generated solution for each particle, rather than random walks used in APSO. PSO also had been used to identify inelastic material parameter [10]. Each individual particle is associated to hyper-coordinates in the search space, corresponding to a set of material parameters, upon which velocity operators with random components are applied. The effectiveness of PSO was utilised by applying it in finding the best solution for Maximum Covering Location Problem (MCLP) for ambulance in Malaysia. The best ambulance location was determined to ensure the efficiency of emergency medical services delivery [11].

II. PARTICLE SWARM OPTIMIZATION

PSO generates an exciting, ever-expanding research subject, called swarm intelligence. PSO has been applied to almost every area in optimization, computational intelligence, and designing applications [12, 13]. There are many types of PSO variants developed by researchers. The trend of combining PSO with other existing algorithms is also increasingly popular and generated much interest from researchers.

The movement of a swarming particle consists of two major components: a position component and a velocity component. Each particle is attracted toward the position of the current global best G_{best} and its historical personal best location P_{best}. P_{best} refers to the position that provided the best fitness for the particle. G_{best} refers to the best position obtained by any particle in the swarm in the particular iteration. PSO remembers both the best position found by all particles and the best positions found by each particle in the search process.

The pseudo code of PSO algorithm is as follows:

1 Initialization.
2 Fitness evaluation of each particle.
3 Repeat
4 Compare the fitness of particle with its P_{best}. If current value is better, replaced the previous value.
5 Compare the fitness of particle with its G_{best}. If current value is better, replaced the previous value.
6 Update the velocity and position of the particle.
7 If criterion is met, algorithm is ended. Else, **Repeat**.

Let X_i and V_i be the position vector and velocity for particle i, respectively. The new velocity is determined by the following formula

$$V_i^{new} = V_i^{old} + rand\left(G_{best} - X_i^{old}\right) + rand(P_{best} - X_i^{old}) \quad (1)$$

where *rand* is a random vector, taking the values between 0 and 1. Shi and Eberhart [4] had improved the standard PSO algorithm by adding inertia weight as follows

$$V_i^{new} = inertia * V_i^{old} + rand\left(G_{best} - X_i^{old}\right) + rand(P_{best} - X_i^{old}) \quad (2)$$

where inertia is a value between 0.9 and 0.4, progressively decreasing throughout the process. They claimed that a large inertia weight facilitates a global search while a small inertia weight facilitates a local search. This is so

as to stabilize the motion of the particles, and as a result, the algorithm is expected to converge more quickly. The position update is always

$$X_i^{new} = X_i^{old} + V_i^{new} \tag{3}$$

.

III. MODIFIED PARTICLE SWARM OPTIMIZATION

The standard particle swarm optimization uses both the current global best, G_{best} and the individual best, P_{best}. The reason of using the P_{best} information is to increase the diversity in the quality solutions. However, this diversity can also be generated by introducing some randomness. A simplified version which could accelerate the convergence of the algorithm is to use the G_{best} only, as demonstrated by She Yang as accelerated PSO. This paper proposed a new modulated inertia, p, added in the G_{best} term for velocity update formula, neglecting the P_{best} information without replacing it with some random term.

$$V_i^{new} = V_i^{old} + p * rand\left(G_{best} - X_i^{old}\right) \tag{4}$$

where p is a modulated inertia defined as

$$p = \omega_0(1 - dist_i/\max\_dist) \tag{5}$$

where ω_0 is (0.5,1.0), $dist_i$ is the current Euclidean distance of i_{th} particle from the global best.

$$dist_i = (\sum_{d=1}(\widehat{g_{i_d}} - x_{i,d})^2)^{1/2} \tag{6}$$

where $\widehat{g_{i_d}}$ is the current global best. The maximum distance, $\max\_dist$, of a particle from the global best is calculated using Equation 7.

$$\max\_dist = arg_i \max(dist_i) \tag{7}$$

Inspired by Artificial Bee Colony (ABC) algorithm established by Dervis Karaboga in [14], position update is supplemented by a random term from ABC position update formula, $rand(X_i^{old} - X_i^{neighbour})$. A particle (bee) in ABC selects a food source (solution) and compare it with other food sources within their neighbourhood [15]. The choice is based on the neighbourhood of the previously selected food source. This increases the feasibility of the solution [16]. Hence, equation 3 was manipulated by implementing the random term.

$$X_i^{new} = X_i^{old} + V_i^{new} + rand(X_i^{old} - X_i^{neighbour}) \tag{8}$$

where $X_i^{neighbour}$ denotes another solution selected randomly from the population. X_i^{new} is a new solution that is modified from its previous value, X_i^{old} based on a comparison with the randomly selected position from its neighbouring solution, $X_i^{neighbour}$.

IV. BENCHMARK FUNCTION

In the simulation studies, MPSO was applied for finding the global minimum of three well known benchmark functions. A function of variables is separable if it can be rewritten as a sum of functions of just one variable [17]. A function is multimodal if it has two or more local optima, which makes finding the global optimum more difficult. The most complex case occurs when the local optima are randomly distributed in the search space [17]. Another important factor that affects the complexity of the problem is the dimension of the search space.

Three classical benchmark functions are implemented using PSO, APSO and MPSO. The first function is the Rosenbrock function. The global optimum lies inside a long and narrow valley. Finding the global optimum is difficult, as the variables are strongly independent and the gradients generally do not point toward the optimum. The global minimum is 0. Initialization range for the function is [−15, 15]. The formula is,

$$\sum_{i=1}^{D-1}[100(x_{i+1} - x_i^2) + (x_i - 1)^2\ ] \quad (9)$$

where *D* denotes the number of dimensions.

The second function is Ackley's function. It has a value of 0 at its global minimum (0, 0… 0). Initialization range for the function is [−5, 5]. The formula is,

$$-20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e \quad (10)$$

The third function is the Sphere function whose value is 0 at its global minimum (0, 0). Initialization range for the function is [−100, 100]. It is continuous, convex and unimodal. The formula is,

$$\sum_{i=1}^{D} x_i^2 \quad (11)$$

These functions were used to compare the performance of the improved optimization algorithm against the existing PSO and APSO algorithms.

V. EXPERIMENTAL RESULTS

To make a fair comparison between the three algorithms, all experiments were run for 100 function evaluations. The number of particles in the swarm was set at 36, and the dimension of the search space was 2. A run will be terminated after 100 function evaluations or until the function error dropped below e^{-30} (values less than e^{-30} were reported as 0), whichever comes first. Each of the experiments was repeated 30 times independently and the reported results are the mean of the function values, worst and best values, and the standard deviations of the statistical experimental data. The simulation experiments were implemented on a computer with MATLAB R2010a, Windows 8, and Intel i5 CPU clocked at 2.10 GHz with memory of 8 GB.

The results are presented in Tables 1, 2 and 3. The mean of function value for every algorithm was calculated after 30 experiments. "Best" represent the nearest function value to 0 among 30 attempts. "Worst" represent the most far away function value from 0 among 30 attempts. Standard deviation was calculated to know how far the function values spread out from the average. Hence, precision of an algorithm can be concluded. A standard deviation close to 0 indicates that the data points tend to be very close to the mean (precise), while a high standard deviation indicates that the data points are spread out over a wider range of values (less precise).

Table I showed the results for Rosenbrock function experiment. MPSO mean value was the nearest to 0 compared to others. With the lowest value of standard deviation, 30 function value of MPSO were near to mean value in 0 to 1.25^{e-25} range, hence it can be concluded that MPSO is precise. MPSO was the only one that managed to achieve 0 function value in 30 repeated experiments with 63.33% success while the other two algorithms failed.

TABLE I
Results for the Optimization of Rosenbrock Function Using Three Different Algorithms.

	Function Value			**Probability best zero value (%)**	**Standard Deviation**
	Worst	**Mean**	**Best**		
PSO	4.87E-4	3.88E-05	1.87E-21	0	1.16E-4
APSO	1.19E-21	4.05E-23	5.13E-30	0	2.17E-22
MPSO	1.25E-25	1.76E-26	0	63.33	7.55E-26

Table II showed the results for Sphere function experiment. MPSO mean value was the nearest to 0 compared to others. With the lowest value of standard deviation, 30 function value of MPSO were near to mean

value in 0 to 1.01^{e-28} range, hence it can be concluded that MPSO is precise. MPSO were most accurate among the algorithms, with 50% chance to achieve 0 function value in 30 experiments, while APSO managed to achieve 0 value with 26.67% probability. PSO failed to get any 0 value.

TABLE II
Results for the Optimization of Sphere Function Using Three Different Algorithms.

	Function Value			Probability best zero value (%)	Standard Deviation
	Worst	Mean	Best		
PSO	6.65E-13	2.50E-14	1.53E-24	0	1.21E-13
APSO	1.10E-226	1.05E-27	0	26.67	3.18E-27
MPSO	1.01E-28	1.79E-28	0	50	5.96E-28

Table III showed the results for Ackley's function experiment. MPSO mean value was the nearest to 0 compared to others. With the lowest value of standard deviation, 30 function value of MPSO were near to mean value in 0 to 1.49^{e-13} range, hence it can be concluded that MPSO is precise. MPSO was also the most accurate algorithm among them, with 16.67% probability achieving 0 function value in 30 repeated experiments. APSO came second and PSO failed again.

TABLE III
Results for the Optimization of Ackley's Function Using Three Different Algorithms.

	Function Value			Probability best zero value (%)	Standard Deviation
	Worst	Mean	Best		
PSO	4.97E-05	9.91E-05	2.35E-12	0	5.43E-04
APSO	3.16E-13	1.03E-13	0	6.67	2.21E-13
MPSO	1.49E-13	5.50E-14	0	16.67	1.53E-13

VI. CONCLUSION

This paper proposes the addition of a term in the position update formula, inspired by ABC algorithm technique, and a new modulated inertia, *p,* in velocity update formula to improve the established PSO algorithm.

The benchmark functions which were used for comparison were selected to analyse the performance of the proposed MPSO against the established PSO and APSO algorithms when facing different characteristics like multi-modality, local minima and difficulty of convergence. The results show that the proposed PSO algorithm outperforms the other two for the three selected test functions. Hence, we can conclude that MPSO can be proposed as an optimization algorithm of functions with similar characteristics. This experiment was conducted using 2 dimension problems, thus the results may vary when the dimensionality is increased.

ACKNOWLEDGMENT

This research had been conducted under grant GUP-2015-006.

REFERENCES

[1] J. Kennedy, R. Eberhart, "Particle swarm optimization", in Proc. IEEE ICONN'95,1995, vol. 4, p. 1942
[2] A. Khare, S. Rangnekar, "A review of particle swarm optimization and its applications in solar photovoltaic system", Applied Soft Computing, vol. 13, pp. 2997-3006, 2013
[3] J. Hu, Z. Wang, S. Qiao, J. Gan, "The fitness evaluation strategy in particle swarm optimization", Applied Mathematics and Computation, vol. 217, pp. 8655-8670, 2011
[4] Y. Shi, R. C. Eberhart, "Empirical study of particle swarm optimization", in Proc COEC'99, vol. 3, p. 1945
[5] A. Nickabadi, M. M. Ebadzadeh, R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight", Applied Soft Computing, vol. 11, pp. 3658-3670, 2011.

[6] H. Wang, W. Wang, Z. Wu, "Particle swarm optimization with adaptive mutation for multimodal optimization", Applied Mathematics and Computation, vol. 221, pp. 296-305, 2013
[7] X. S. Yang, S. Deb, S. Fong, "Accelerated particle swarm optimization and support vector machine for business optimization and applications", in NDT'11, Communications in Computer and Information Science, vol. 136, pp. 53-66, 2011.
[8] S. Das, A. Ghosh, D. Banerjee, "Designing of engineered fabrics using particle swarm optimization", International Journal of Clothing Science and Technology, vol. 26, pp. 48 – 57, 2014.
[9] G. G. Wang, A. H. Gandomi, X. S. Yang, A. H. Alavi, "A novel improved accelerated particle swarm optimization algorithm for global numerical optimization", Engineering Computations, vol. 31, pp. 1198 – 1220, 2014.
[10] M. Vaz Jr, E.L. Cardoso. J. Stahlschmidt, "Particle swarm optimization and identification of inelastic material parameters", Engineering Computations, vol. 30, pp. 936 -960, 2013.
[11] W. A Lutfi W. M. Hatta, C. S. Lim, A. Faiz, M. Hafiz, S. S. Teoh, "Solving maximal covering location with particle swarm optimization", International Journal of Engineering and Technology (IJET), vol. 5, pp. 3301-3306, 2013.
[12] D. I. Petropoulos, A. C. Nearchou, "A particle swarm optimization algorithm for balancing assembly lines", Assembly Automation, vol. 31, pp. 118 – 129, 2011.
[13] S. Panda, B. Mohanty, P. K. Hota, "Hybrid BFOA–PSO algorithm for automatic generation control of linear and nonlinear interconnected power systems", Applied Soft Computing, vol. 13, pp. 4718–4730, 2013.
[14] D. Karaboga, "An idea based on honey bee swarm for numerical optimization", Erciyes University, Turkey, Tech. Rep-TR06, 2005.
[15] D. Karaboga, B. Basturk, "On the performance of artificial bee colony (ABC) algorithm", Applied Soft Computing, vol. 8, pp. 687–697, 2008
[16] A. Banharnsakun, T. Achalakul, B. Sirinaovakul, "The best-so-far selection in artificial bee colony algorithm", Applied Soft Computing, vol. 11, pp. 2888-2901, 2011.
[17] V. Nayak, H. Suthar, J. Gadit, "Implementation of artificial aee colony algorithm", IAES International Journal of Artificial Intelligence (IJ-AI)", vol. 1, pp. 112-120,2012.