Paper 25

by The Jin Ai

Adaptive Particle Swarm Optimization Algorithms

The Jin Ai and Voratas Kachitvichyanukul
School of Engineering and Technology, Asian Institute of Technology
P.O. Box 4, Klong Luang, Pathumthani 12120, Thailand
E-mail thejin.ai@ait.ac.th, voratas@ait.ac.th

Abstract
This paper reviews the literature on the mechanisms for adapting parameters of particle swarm optimization (PSO) algorithm. The discussion focused on the mechanisms for adaptively setting such parameters as inertia weight, acceleration constants, number of particles and number of iterations. Two mechanisms are proposed and tested. The velocity index pattern is proposed for adapting the inertia weight while the acceleration constants are adapted via the use of relative gaps between various learning terms and the best objective function values. The mechanisms are demonstrated by modifying GLNPSO for a specific optimization problem, namely, the vehicle routing problem. The preliminary experiment indicates that the addition of the proposed adaptive mechanisms can provide good algorithm performance in terms of solution quality with a slightly slower computational time.

Keywords: Particle swarm optimization, metaheuristic, algorithm's parameter, adaptive PSO, VRP.

1. Introduction

As an emerging evolutionary computing method [1, 2, 3], Particle swarm optimization (PSO) has recently been successfully applied to solve many combinatorial optimization problems including job shop scheduling problem [4] and vehicle routing problem [5, 6]. Similar to other evolutionary computing methods, PSO has several parameters that are required to be properly set in order to yield good performance. Finding the best set of PSO parameters, which include inertia weight, acceleration constants, number of particles and number of iterations, for a specific optimization problem is not an easy task, since the same parameters set may yield different performance on different problem cases. Usually, many experiments are required over many problem cases to determine proper values of parameters. However, there is no guarantee that the selected parameter set will always yield the best algorithm performance, especially when the algorithm is applied to solve a new problem case.

A novel idea to replace the experiments to find the best parameter set is through an adaptive mechanism that can adapt PSO parameters autonomously whenever it is applied to solve a problem instance. It is noted that the concept of adaptive algorithm is also present in the wider scope of evolutionary computing method, i.e. in the genetic algorithm [7, 8]. Also, some earlier works in PSO, which will be further reviewed and discussed in Section 3 of this paper, have dealt with the issue of how to adaptively set its parameters.

The main objective of this paper is to give a perspective on adaptive PSO algorithms. To start with, the GLNPSO, a PSO Algorithm with multiple social learning structures [9], is briefly reviewed in Section 2, altogether with the main role of its parameters. Section 3 reviews the existing adaptive mechanism in the literature and presents other alternative mechanism. Some selected mechanisms are finally embedded into the GLNPSO Algorithm and applied to a specific optimization problem in Section 4. Finally, Section 5 summarizes the material presented in this paper and recommends further works.

2. GLNPSO Algorithm

The GLNPSO Algorithm is a PSO Algorithm with multiple social learning structures. Instead of using only the global best, it also incorporates the local best and near-neighbor best as additional social learning factors. Therefore, in the velocity updating equation, it requires also three different acceleration constants related to each social learning factor. The detail of the GLNPSO Algorithm is presented below.

Notation:

τ	:	Iteration index, $\tau = 1 \ldots T$
l	:	Particle index, $l = 1 \ldots L$
h	:	Dimension index, $h = 1 \ldots H$
u	:	Uniform random number in the interval $[0,1]$
w	:	Inertia weight
$\omega_{lh}(\tau)$	:	Velocity of the particle l at the dimension h in the iteration τ
$\theta_{lh}(\tau)$	:	Position of the particle l at the dimension h

		the iteration τ
ψ_{lh}	:	Personal best position (pbest) of the particle l at the dimension h
ψ_{gh}	:	Global best position (gbest) at the dimension h
ψ_{lh}^{L}	:	Local best position (lbest) of the particle l at the dimension h
ψ_{lh}^{N}	:	Near neighbor best position (nbest) of the particle l at the dimension h
c_p	:	Personal best position acceleration constant
c_g	:	Global best position acceleration constant
c_l	:	Local best position acceleration constant
c_n	:	Near neighbor best position acceleration constant
$\theta^{\max}$	:	Maximum position value
$\theta^{\min}$	:	Minimum position value
Θ_l	:	Vector of position of the particle l, $[\theta_{l1} \quad \theta_{l2} \quad \cdots \quad \theta_{lH}]$
Ω_l	:	Vector of velocity of the particle l, $[\omega_{l1} \quad \omega_{l2} \quad \cdots \quad \omega_{lH}]$
Ψ_l	:	Vector of personal best position of particle l, $[\psi_{l1} \quad \psi_{l2} \quad \cdots \quad \psi_{lH}]$
Ψ_g	:	Vector of global best position, $[\psi_{g1} \quad \psi_{g2} \quad \cdots \quad \psi_{gH}]$
Ψ_l^{L}	:	Vector of local best position of particle l, $[\psi_{l1}^{L} \quad \psi_{l2}^{L} \quad \cdots \quad \psi_{lD}^{L}]$
R_l	:	Problem specific solution corresponding to the particle l
$Z(\Theta_l)$	:	Fitness value of Θ_l
FDR	:	Fitness-distance-ratio

GLNPSO Algorithm (For minimization):

1. Initialize a swarm with L particles; generate the particle l with random position Θ_l in the range $\left[\theta^{\min}, \theta^{\max}\right]$, velocity $\Omega_l = 0$ and personal best $\Psi_l = \Theta_l$ for $l = 1 \ldots L$. Set iteration $\tau = 1$.
2. For $l = 1 \ldots L$, decode $\Theta_l(\tau)$ to a problem specific solution R_l.
3. For $l = 1 \ldots L$, compute the performance measurement of R_l, and set this as the fitness value of Θ_l, represented by $Z(\Theta_l)$.
4. Update pbest: For $l = 1 \ldots L$, update $\Psi_l = \Theta_l$, if $Z(\Theta_l) < Z(\Psi_l)$.
5. Update gbest: For $l = 1 \ldots L$, update $\Psi_g = \Psi_l$, if $Z(\Psi_l) < Z(\Psi_g)$.
6. Update lbest: For $l = 1 \ldots L$, among all pbest from K neighbors of the particle l, set the personal best which obtains the least fitness value to be Ψ_l^{L}.
7. Generate nbest: For $l = 1 \ldots L$, and $h = 1 \ldots H$, set $\psi_{lh}^{N} = \psi_{oh}$ that maximizing fitness-distance-ratio (FDR) for $o = 1 \ldots H$. Where FDR is defined as

$$FDR = \frac{Z(\Theta_l) - Z(\Psi_o)}{\left|\theta_{lh} - \psi_{oh}\right|} \quad \text{which } l \neq o \qquad (1)$$

8. Update the velocity and the position of each particle l:

$$\begin{aligned} \omega_{lh}(\tau+1) = {} & c_p u\left(\psi_{lh} - \theta_{lh}(\tau)\right) + c_g u\left(\psi_{gh} - \theta_{lh}(\tau)\right) \\ & + c_l u\left(\psi_{lh}^{L} - \theta_{lh}(\tau)\right) + c_n u\left(\psi_{lh}^{N} - \theta_{lh}(\tau)\right) \\ & + w\omega_{lh}(\tau) \end{aligned} \qquad (2)$$

$$\theta_{lh}(\tau+1) = \theta_{lh}(\tau) + \omega_{lh}(\tau+1) \quad (3)$$

If $\theta_{lh}(\tau+1) > \theta^{\max}$, then

$$\theta_{lh}(\tau+1) = \theta^{\max} \quad (4)$$

$$\omega_{lh}(\tau+1) = 0 \quad (5)$$

If $\theta_{lh}(\tau+1) < \theta^{\min}$, then

$$\theta_{lh}(\tau+1) = \theta^{\min} \quad (6)$$

$$\omega_{lh}(\tau+1) = 0 \quad (7)$$

9. If the stopping criterion is met, i.e. $\tau = T$, stop. Otherwise, $\tau = \tau + 1$ and return to step 2.

It can be seen from the algorithm above that there are some parameters that are required by GLNPSO, including inertia weight (w), acceleration constants (c_p, c_g, c_l, c_n), number of particles (L) and number of iterations (T). The inertia weight and acceleration constants play very important role in the velocity updating equation (Eq. 2). Since the velocity drives the movement of particles from one position to the next (Eq. 3), it implies that the movement of the swarm of particles as a searching agent in PSO is affected by these parameters. Movement of the swarm is closely linked to the algorithm performance, since each distinct position may correspond to different solution and the final solution obtained by PSO must be one of the positions that have been visited by the swarm. Therefore, the number of particles and the number of iterations are also related to the algorithm performance, since these parameters partially determine the number of positions visited by the swarm. However, simply increase the number of particles and number of iterations does not always improve the algorithm performance, since the velocity updating mechanism also depends on the cognitive learning (pbest) and social learning (gbest, lbest, and nbest). In addition, the number of particles also has influence on the social information values and theirs updating behavior.

3. Parameters Adaptation

3.1 Inertia Weight

Among PSO parameters, inertia weight has gained enormous attention since the early development of PSO. The proper setting of inertia weight is believed to have significant effect on the performance of PSO algorithm. Instead of using constant inertia weight, A linear decreasing function has been proposed for setting the inertia weight [10]. For the case of GLNPSO algorithm described above, this concept is implemented by using following expression as the inertia weight (w) in Eq. 2:

$$w(\tau) = w(T) + \frac{\tau - T}{1 - T}\left[w(1) - w(T)\right] \quad (8)$$

where

$w(\tau)$: Inertia weight in iteration τ

Similar with this approach, a nonlinear decreasing function was proposed for setting inertia weight [11]. With these decreasing inertia weight settings, it is expected that the particles are able to explore the problem space more widely at the beginning of iteration steps and to exploit promising solution in the end of iteration steps. As seen in Eq. 2, the inertia weight is the multiplication factor of the previous velocity. Therefore, applying large inertia weight at the beginning causes the particles to maintain their previous velocity and makes the particles move more freely. When this inertia weight is step by step reduced at the latter iteration steps, the particles are influent less by previous velocity and their movements are influent more by theirs cognitive and social learning information.

Other approaches that have been proposed attempts to adjust the inertia weight adaptively based on the particular condition of the swarm. An adaptive PSO was proposed [12] that alternating its inertia weight between a high value and a low value and vice versa in order to control the swarm's velocity. For this purpose, the velocity index of the swarm ($\bar{\omega}$) is defined by the expression given in Eq. 9. The index can be continuously observed from iteration to iteration:

$$\bar{\omega} = \frac{\sum_{l=1}^{L}\sum_{h=1}^{H}\left|\omega_{lh}\right|}{L \cdot H} \quad (9)$$

Then, the swarm velocity index is compared with the target velocity (ω^*), which is a linear decreasing function:

$$\omega^* = \left(1 - \frac{\tau}{T}\right)\omega^{\max} \quad (10)$$

Whenever the velocity index is bigger than the target velocity, the low value of inertia weight is selected. Reversely, the inertia weight is set at the high value when the velocity index is smaller than the target velocity.

Another study of the dynamic behavior of the swarm in PSO was carried out to determine which velocity index pattern should be followed by the swarm [13]. The key finding in [13] stated that different pattern should be used in order to achieve balance between exploration and exploitation process. It is noted that a better balance between these phases is often mentioned as the key to a good performance of PSO. This idea can be implemented based on the velocity index pattern, so that half of iterations placed as exploration phase and the other half placed as exploitation phase. For example, two-step linear decreasing pattern can be selected to portray this condition, in which the target velocity follows this expression:

$$\omega^* = \begin{cases} \left(1 - \dfrac{1.8\tau}{T}\right)\omega^{\max}, & 0 \le \tau \le T/2 \\ \left(0.2 - \dfrac{0.2\tau}{T}\right)\omega^{\max}, & T/2 \le \tau \le T \end{cases} \qquad (11)$$

By using Eq. 11, the target velocity index is gradually decreased from $\omega^{\max}$ at the first iteration to $0.1\omega^{\max}$ at the first half of iterations. It is expected that the problem space is well explored by the swarm in this phase, so that the swarm is able to exploit the existing solutions during the second half of iterations when the desired velocity index is small enough and slowly reduced from $0.1\omega^{\max}$ to 0.

Extending the idea of using two preset values of inertia weight, it is also possible to set the inertia weight in the range of minimum ($w^{\min}$) and maximum value ($w^{\max}$). The updating mechanism principle is similar with the existing work: whenever the swarm velocity index is lower than the desired velocity index, the inertia weight is increased, and reversely when the swarm velocity index is greater than the desired velocity index, the inertia weight is decreased. It can be defined that the amount of increases or decreases of inertia weight depends on the difference between the velocity index of the swarm and the target velocity index. An example of equations that are used to update inertia weight is as follow:

$$\Delta w = \frac{(\omega^* - \bar{\omega})}{\omega^{\max}}\left(w^{\max} - w^{\min}\right) \qquad (12)$$

$$w = w + \Delta w \qquad (13)$$

$$w = w^{\max} \quad \text{if} \quad w > w^{\max} \qquad (14)$$

$$w = w^{\min} \quad \text{if} \quad w < w^{\min} \qquad (15)$$

Other proposed mechanisms to adaptively adjust the inertia weight are based on the value of local best and global best at a particular iteration [14] or the population diversity of the swarm [15, 16, 17]. In addition, fuzzy logic rules based on the swarm fitness values also had been proposed to adaptively adjust the inertia weight [18, 19].
Instead of using single value of inertia weight for the whole swarm, another approach tried to adaptively set the inertia weight for each individual particle in the swarm. Feng *et al.* [20] used the velocity and the acceleration component of each particle to set the individual inertia weight. Panigrahi *et al.* [21] proposed a method to spread the inertia weight between the range of minimum and maximum value, in which particle with the best performance is given the smallest weight so that it moves the slowest and particle with the worst performance is given the biggest weight so that it moves the fastest. It is noted that setting inertia weight for each individual particle in the swarm required more computational effort than setting single weight for whole swarm. Therefore, the effectiveness of this mechanism should be carefully studied in order to evaluate whether the additional computational effort can significantly improve the performance of the algorithm.

3.2 Acceleration Constants

In terms of setting acceleration constants of PSO adaptively, the values of local best and global best at a particular iteration are proposed as the basis for updating the acceleration constants [14]. Alternatively, a time-varying acceleration coefficient (TVAC) proposed to replace the same constant during the whole iteration process [22]. In TVAC, the cognitive acceleration constant is linearly reduced and the social acceleration constant is linearly increased through the iterations.

One mechanism for adaptively setting the important weight of each acceleration term is presented here. As illustration, there are four cognitive/social terms that are taken into consideration in the GLNPSO presented above: personal best, global best, local best, and near-neighbor best. The acceleration constant gives relative importance of respective term when the velocity is updated. A heavier weight for a specific term means that term is more dominant than the others and the particles tend to move in the direction of this term. The proposed adaptive mechanism reverses this property by first determining a relative importance of the cognitive/social term from the current swarm characteristics before setting the acceleration constants.

The importance measurement that is employed here is the difference between the corresponding objective function of particle's position and the objective function of respective term. For a minimization problem, a bigger difference on a particular term represents a higher degree of importance on this term. It is implied that particles have opportunity to gain more improvement in its objective function if moving towards this term. However, negative difference is avoided since it will lead to worsening objective function.

As shown in Fig 1, for a single particle which is located at position Θ and surrounded in its corresponding cognitive/social terms (personal best Ψ, global best Ψ_g, local best Ψ^L, and near neighbor best Ψ^N), the degree of importance of each term can be defined as $\max\{Z(\Theta)-Z(\Psi),0\}$, $\max\{Z(\Theta)-Z(\Psi_g),0\}$, $\max\{Z(\Theta)-Z(\Psi^L),0\}$, $\max\{Z(\Theta)-Z(\Psi^N),0\}$, respectively for personal best, global best, local best, and near neighbor best.

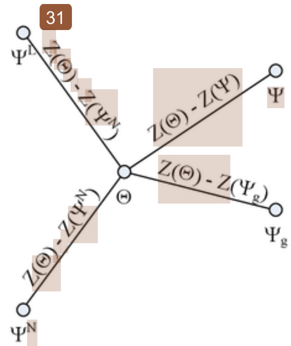

Fig. 1 Particle position and its corresponding social terms.

Then, the acceleration constants can be determined as the proportion of respective degree of importance to the constant c^*, which is defined as the sum of the acceleration constants. The expression for the degree of importance of a single particle can be expanded for the whole swarm which consists of L particles by combining all particles properties, as follow:

$$\Delta Z_P = \sum_{l=1}^{L} \max\{Z(\Theta_l)-Z(\Psi_l),0\} \qquad (16)$$

$$\Delta Z_G = \sum_{l=1}^{L} \max\{Z(\Theta_l)-Z(\Psi_g),0\} \qquad (17)$$

$$\Delta Z_L = \sum_{l=1}^{L} \max\{Z(\Theta_l)-Z(\Psi_l^L),0\} \qquad (18)$$

$$\Delta Z_N = \sum_{l=1}^{L} \max\{Z(\Theta_l)-Z(\Psi_l^N),0\} \qquad (19)$$

where:

ΔZ_P	:	Degree of importance for personal best
ΔZ_G	:	Degree of importance for global best
ΔZ_L	:	Degree of importance for local best
ΔZ_N	:	Degree of importance for near neighbor best

Finally, the acceleration constants can be determined as the proportion of degree of importance. Also, in order to avoid rapid changing of parameters, the acceleration constant is updated using exponential weighted moving average technique:

$$\Delta Z = \Delta Z_P + \Delta Z_G + \Delta Z_L + \Delta Z_N \qquad (20)$$

$$c_p = \alpha c_p + (1-\alpha)\frac{\Delta Z_P}{\Delta Z} c^* \qquad (21)$$

$$c_g = \alpha c_g + (1-\alpha)\frac{\Delta Z_G}{\Delta Z} c^* \qquad (22)$$

$$c_l = \alpha c_l + (1-\alpha)\frac{\Delta Z_L}{\Delta Z} c^* \qquad (23)$$

$$c_n = \alpha c_n + (1-\alpha)\frac{\Delta Z_N}{\Delta Z} c^* \qquad (24)$$

3.3 Number of Particles

Recently PSO with adaptive population size has been proposed [23]. The total iteration steps are divided into some ladders with same number of iterations. At the end of each ladder, the diversity of swarm is measured, and then the population size is adjusted based on the measured diversity. If the swarm diversity is lower than a threshold value, the population size is increased. Otherwise, if the swarm diversity is higher than the threshold, the population size is decreased.

3.4 Other Parameters

Although existing adaptive mechanism for some parameters is not yet available in the literature, such as number of Iterations and number of neighbor, there are also possibilities to set these parameters adaptively.

4. Example Application

An example of adaptive PSO algorithm is presented in this section. In this example some adaptive features are added to the GLNPSO algorithm in which the algorithm is only slightly modified and the computational effort is not significantly increased. To be more specific, the example algorithm can adaptively set the inertia weight and acceleration constants. Therefore, the only change to the GLNPSO is in the Step 8, in which it is updated to:

a. Update inertia weight following Eq. 9, 11–15.
b. Update accelerations constant following Eq. 16–24.
c. Update the velocity and the position of each particle following Eq. 2–7.

In order to save some computational effort, the adaptive mechanism of inertia weight (step a) and acceleration constants (step b) is not performed in every iteration, but only performed every fixed number of iterations, for example 10 iterations.

To make the adaptive feature works, the following initialization is required. For the inertia weight, the $\omega^{\max}$ is taken from the velocity index at the first iteration. Also, the $w^{\max}$ and $w^{\min}$ are being set as 0.9 and 0.4, respectively. For the acceleration constants, the value of c^* is 4 and α is 0.8. Initially, equal acceleration constant is employed, i.e. $c_p = c_g = c_l = c_n = 1$.

For a test case, this adaptive PSO algorithm is applied to solve vehicle routing problem (VRP), in which the solution representation of this problem for PSO and the corresponding decoding method have been proposed before using GLNPSO [5]. It is noted that the adaptive PSO algorithm can be applied to any optimization problem that have been solved by respective non-adaptive PSO algorithm, since the adaptive PSO algorithm only changes its parameters, not other algorithm mechanism. A problem instance, which consists of 200 customers and 16 vehicles, is generated for computational experiment.

In a typical run with 1000 iterations, the velocity index pattern of both GLNPSO algorithm (without adaptive feature) and the adaptive PSO algorithm are displayed in Fig. 1. It can be seen that the velocity index pattern of GLNPSO is steadily decreasing, so that in the first half of the run (approximately from the first iteration to the 500th iteration) velocity index of adaptive PSO algorithm is bigger than velocity index of GLNPSO. Also, in the second half of the run velocity index of adaptive PSO algorithm is smaller than velocity index of GLNPSO. This pattern implies that the adaptive PSO algorithm is better at exploring the solution space in the first half of the run than GLNPSO. Also, the adaptive PSO algorithm is better than the GLNPSO algorithm to exploit the solution space in the second half of the run.

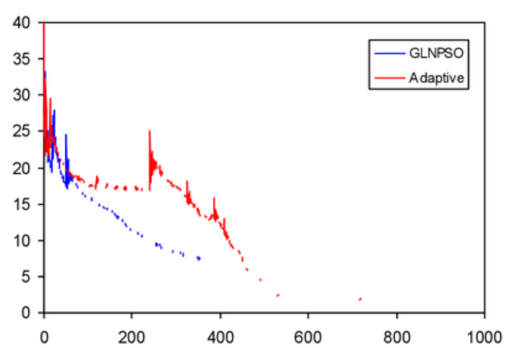

Fig. 1 Comparison of typical velocity index patterns of GLNPSO and Adaptive PSO.

In the same algorithm run, the dynamic behavior of the best objective function values (the objective function of gbest) is presented in Fig 2. Following the pattern of velocity index, the best objective function of GLNPSO is improving steadily. However, the objective function values from GLNPSO are better than those from the adaptive PSO only in the early part of run.

After about the 600th iteration, the adaptive PSO provides better objective function. Finally, the best objective function values found are 2720.86 and 2671.59 for the original version of GLNPSO and from the adaptive version of GLNPSO, respectively.

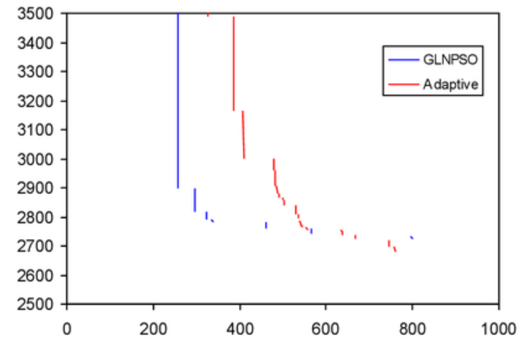

Fig. 2 Typical convergent behavior of objective function values from GLNPSO and Adaptive PSO.

The computational time of the adaptive PSO is not significantly bigger than the GLNPSO. It is empirically shown for the typical run tested above, the computational time of both algorithms are 08:12 and 08:21 minutes, respectively for GLNPSO and adaptive PSO.

5. Conclusion and Further Works

Some possibilities to enable particle swarm optimization algorithm to self-adapt its parameter are discussed in this paper based on some ideas from literature and new proposed mechanism. For illustrative purpose, a demonstrated example of adaptive PSO algorithm is proposed to adaptively set the inertia weight and acceleration constants.

The computational experiment on a typical vehicle routing instance implies that the adaptive PSO algorithm can perform better than GLNPSO algorithm in terms of solution quality but with slightly slower computational time. However, more computational experiment is required in order to make generalization of the result. Also, further works is still needed to explore more mechanisms for adapting other parameters of PSO algorithms, such as: number of particles, number of neighbors, and number of iterations.

Acknowledgments

The research is a part of the research program of the High Performance Computing Group at Asian Institute of Technology (AITHPC) which received support from the Thai GRID project. The authors wish to thank the AITHPC and the Thai GRID Center for the access of computing facility and the technical support.

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE International Conference on Neural Networks*, pp. 1942–1948, 1995.

[2] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*, San Francisco: Morgan Kaufmann Publishers, 2001.

[3] M. Clerc, *Particle Swarm Optimization*, London: ISTE, 2006.

[4] P. Pongchairerks and V. Kachitvichyanukul, "A Two-level Particle Swarm Optimization Algorithm on Job Shop Scheduling Problem," *International Journal of Operational Research*, (in press)

[5] T. J. Ai and V. Kachitvichyanukul, "A particle swarm optimization for the capacitated vehicle routing problem," *International Journal of Logistics and SCM Systems*, vol. 2(1), pp. 50–55, 2007.

[6] T. J. Ai and V. Kachitvichyanukul, "A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery," *Computers and Operations Research*, doi: 10.1016/j.cor.2008.04.003, 2008.

[7] M. Annunziato and S. Pizzuti, "Adaptive parameterization of evolutionary algorithms driven by reproduction and competition," in *Proc. European Symposium on Intelligent Techniques 2000*, pp. 246–256, 2000.

[8] T. Back, A. E. Eiben, N. A. L. Van Der Vaart, "An empirical study on GAs without parameters," in *Lecture Notes in Computer Science Vol. 1917: Parallel Problem Solving from Nature PPSN VI*, pp. 315–324, 2000.

[9] P. Pongchairerks and V. Kachitvichyanukul, "A non-homogenous particle swarm optimization with multiple social structures," in *Proc. International Conference on Simulation and Modeling*, paper A5-02, 2005.

[10] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE International Conference on Evolutionary Computation 1998*, pp. 69–73, 1998.

[11] Y. Gao and Z. Ren, "Adaptive particle swarm optimization algorithm with genetic mutation operation," in *Proc. Third International Conference on Natural Computation*, pp. 211–215, 2007.

[12] G. Ueno, K. Yasuda, N. Iwasaki, "Robust adaptive particle swarm optimization," in *Proc. IEEE International Conference on Systems, Man and Cybernetics 2005*, pp. 3915–3920, 2005.

[13] T. J. Ai and V. Kachitvichyanukul, "Dispersion and velocity indices for observing dynamic behavior of particle swarm optimization," in *Proc. IEEE Congress on Evolutionary Computation 2007*, pp. 3264–3271, 2007.

[14] M. S. Arumugam and M. V. C. Rao, "On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems," *Applied Soft Computing Journal*, vol. 8(1), pp. 324–336, 2008.

[15] L. Dan, G. Liqun, Z. Junzheng and L. Yang, "Power system reactive power optimization based on adaptive particle swarm optimization algorithm," in *Proc. World Congress on Intelligent Control and Automation*, pp. 7572–7576, 2006.

[16] J. Jie, J. Zeng and C. Han, "Adaptive particle swarm optimization with feedback control of diversity," in *Lecture Notes in Computer Science Vol. 4115 LNBI-III*, pp. 81–92, 2006.

[17] D. Zhang, Z. Guan and X. Liu, "An adaptive particle swarm optimization algorithm and simulation," in *Proc. IEEE International Conference on Automation and Logistics 2007*, pp. 2399–2402, 2007.

[18] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proc. IEEE Congress on Evolutionary Computation 2001*, pp. 101–106, 2001.

[19] P. Bajpai and S. N. Singh, "Fuzzy adaptive particle swarm optimization for bidding strategy in uniform price spot market," *IEEE Transactions on Power Systems*, vol. 22(4), pp. 2152–2160, 2007.

[20] C. S. Feng, S. Cong, and X. Y. Feng, "A new adaptive inertia weight strategy in particle swarm optimization," in *Proc. IEEE Congress on Evolutionary Computation 2007*, pp. 4186–4190, 2007.

[21] B. K. 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(6), pp. 1407–1415, 2008.

[22] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8(3), pp. 240–255, 2004.

[23] D. B. Chen and C. X. Zhao, "Particle swarm optimization with adaptive population size and its application," *Applied Soft Computing Journal*, doi: 10.1016/j.asoc.2008.03.001, 2008

Paper 25

ORIGINALITY REPORT

33%	21%	28%	14%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	The Jin Ai, Voratas Kachitvichyanukul. "A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery", Computers & Operations Research, 2009 Publication	5%
2	Kasemset, Chompoonoot. "Application of Adaptive Particle Swarm Optimization to Bi-level Job-Shop Scheduling Problem", Industrial Engineering and Management Systems, 2014. Publication	4%
3	Ai, T.J.. "A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery", Computers and Operations Research, 200905 Publication	2%
4	www.reportworld.co.kr Internet Source	2%
5	Voratas Kachitvichyanukul, Siriwan Sitthitham. "A two-stage genetic algorithm for multi-	1%

objective job shop scheduling problems", Journal of Intelligent Manufacturing, 2009
Publication

6	www.cs.uoi.gr Internet Source	1%
7	Submitted to University of Macau Student Paper	1%
8	"Advances in Swarm Intelligence", Springer Science and Business Media LLC, 2017 Publication	1%
9	mini-chip.eu Internet Source	1%
10	www.vpaa.ait.ac.th Internet Source	1%
11	Submitted to Nazarbayev University Student Paper	1%
12	Submitted to Asian Institute of Technology Student Paper	1%
13	link.springer.com Internet Source	1%
14	Jingneng Ni, Jiuping Xu, Mengxiang Zhang. "Constructed wetland planning-based bi-level optimization to balance the watershed ecosystem and economic development: A case study at the Chaohu Lake watershed, China",	1%

Ecological Engineering, 2016
Publication

15 PISUT PONGCHAIRERKS, VORATAS KACHITVICHYANUKUL. "A PARTICLE SWARM OPTIMIZATION ALGORITHM ON JOB-SHOP SCHEDULING PROBLEMS WITH MULTI-PURPOSE MACHINES", Asia-Pacific Journal of Operational Research, 2011 1%
Publication

16 Jiuping Xu, Liming Yao. "Random-Like Multiple Objective Decision Making", Springer Nature, 2011 1%
Publication

17 DeBao Chen, ChunXia Zhao. "Particle swarm optimization with adaptive population size and its application", Applied Soft Computing, 2009 1%
Publication

18 Bharti, Kusum Kumari, and Pramod Kumar Singh. "Opposition chaotic fitness mutation based adaptive inertia weight BPSO for feature selection in text clustering", Applied Soft Computing, 2016. <1%
Publication

19 Zhang, Limin, Yinggan Tang, Changchun Hua, and Xinping Guan. "A new particle swarm optimization algorithm with adaptive inertia weight based on Bayesian techniques", Applied <1%

Soft Computing, 2015.
Publication

20	Jiuping Xu. "Bi-Random Multiple Objective Decision Making", Lecture Notes in Economics and Mathematical Systems, 2011 Publication	<1%
21	"Toward Sustainable Operations of Supply Chain and Logistics Systems", Springer Science and Business Media LLC, 2015 Publication	<1%
22	J. C. Bansal, P. K. Singh, Mukesh Saraswat, Abhishek Verma, Shimpi Singh Jadon, Ajith Abraham. "Inertia Weight strategies in Particle Swarm Optimization", 2011 Third World Congress on Nature and Biologically Inspired Computing, 2011 Publication	<1%
23	www.iaeng.org Internet Source	<1%
24	Qigao Feng. "A Branch and Bound Algorithm for Generalized Polynomial Programming Problems", 2009 International Conference on Information Engineering and Computer Science, 12/2009 Publication	<1%
25	Gang, Jun, Yan Tu, Benjamin Lev, Jiuping Xu, Wenjing Shen, and Liming Yao. "A multi-	<1%

objective bi-level location planning problem for stone industrial parks", Computers & Operations Research, 2015.
Publication

26	www.ijens.org Internet Source	<1%
27	Beckmann, A.. "Ordinal notations and well-orderings in bounded arithmetic", Annals of Pure and Applied Logic, 20030415 Publication	<1%
28	www.thaiscience.info Internet Source	<1%
29	www.wseas.us Internet Source	<1%
30	Studies in Computational Intelligence, 2014. Publication	<1%
31	tel.archives-ouvertes.fr Internet Source	<1%
32	documents.mx Internet Source	<1%
33	Lu Gan, Jiuping Xu. "Retrofitting Transportation Network Using a Fuzzy Random Multiobjective Bilevel Model to Hedge against Seismic Risk", Abstract and Applied Analysis, 2014 Publication	<1%

34	www.scielo.br Internet Source	<1%
35	oplab.im.ntu.edu.tw Internet Source	<1%
36	users.monash.edu.au Internet Source	<1%
37	Ouyang, Hai-bin, Li-qun Gao, Xiang-yong Kong, Steven Li, and De-xuan Zou. "Hybrid harmony search particle swarm optimization with global dimension selection", Information Sciences, 2016. Publication	<1%
38	"Computational Intelligence and Bioinformatics", Springer Science and Business Media LLC, 2006 Publication	<1%
39	csd.ijs.si Internet Source	<1%
40	hal.inria.fr Internet Source	<1%
41	ijarcce.com Internet Source	<1%
42	slideheaven.com Internet Source	<1%

43	Bing Wang, Zhen Yang. "A Particle Swarm Optimization Algorithm for Robust Flow-shop Scheduling with Fuzzy Processing Times", 2007 IEEE International Conference on Automation and Logistics, 2007 Publication	<1%
44	Xunlin Jiang, Haifeng Ling, Jun Yan, Bo Li, Zhao Li. "Forecasting Electrical Energy Consumption of Equipment Maintenance Using Neural Network and Particle Swarm Optimization", Mathematical Problems in Engineering, 2013 Publication	<1%
45	Submitted to University of Hong Kong Student Paper	<1%
46	Niknam, T.. "A novel hybrid particle swarm optimization for economic dispatch with valve-point loading effects", Energy Conversion and Management, 201104 Publication	<1%
47	Submitted to Savitribai Phule Pune University Student Paper	<1%
48	A. Kaveh. "Advances in Metaheuristic Algorithms for Optimal Design of Structures", Springer Nature, 2017 Publication	<1%
49	Vincent F. Yu, Parida Jewpanya, Ching-Jung	

Ting, A.A.N. Perwira Redi. "Two-level particle swarm optimization for the multi-modal team orienteering problem with time windows", Applied Soft Computing, 2017
Publication

<1%

---

50 Xiaoling Huang, Yating Chen. "An optimization model for multimodal transportation decisions based on congestion information", Proceeding of the 11th World Congress on Intelligent Control and Automation, 2014
Publication

<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		