Paper 26

by The Jin Ai

Submission date: 19-Jul-2019 03:18PM (UTC+0700)

Submission ID: 1153164281

File name: Paper_26_APIEMS_2008_APSO_for_VRP.pdf (318.34K)

Word count: 4420

Character count: 23262

A Study on Adaptive Particle Swarm Optimization for Solving Vehicle Routing Problems

The Jin Ai †
School of Engineering and Technology
Asian Institute of Technology, Pathumthani 12120, THAILAND
Department of Industrial Engineering
Universitas Atma Jaya Yogyakarta, Yogyakarta 55281, INDONESIA
Email: thejin.ai@ait.ac.th or jinai@mail.uajy.ac.id

Voratas Kachitvichyanukul
School of Engineering and Technology
Asian Institute of Technology, Pathumthani 12120, THAILAND
Email: voratas@ait.ac.th

Abstract. *This paper presents a study on an adaptive version of particle swarm optimization (PSO) algorithm for solving vehicle routing problems (VRPs). Recently, PSO has been showing promising results in solving many optimization problems include VRP. There are some parameters that need to be set in order to obtain a good performance of the PSO algorithm. However, finding the best set of parameters that is good for all problem cases is not an easy task. Many experiments must be performed to set the parameters and yet there is no guarantee that the best obtained parameter set will provide consistently good algorithm performance when it is applied to a new problem cases. Hence, a novel idea to have a self-adaptive PSO, that can adapt its parameters automatically whenever it is applied to solve a problem instance, is an alternative way to overcome this situation. The adaptive version of PSO proposed in this paper has additional capability to self-adapt its inertia weight (w), one of the key PSO parameter, based on the velocity index of the swarm, the searching agents in PSO. The inertia weight is controlled so that the balance between exploration and exploitation phases of the swarm is maintained, since a better balance of these phases is often mentioned as the key to a good performance of PSO. The performance of this adaptive PSO is evaluated for solving VRP instances and is compared with the existing application of PSO for VRP. The computational experiment shows that the adaptive version of PSO is able to provide better solution than the existing non-adaptive PSO with slightly slower computational time.*

Keywords: *Particle Swarm Optimization, Adaptive Parameters, Metaheuristic, Vehicle Routing Problem.*

1. INTRODUCTION

This paper presents a study on an adaptive version of particle swarm optimization (PSO) algorithm which is applied for solving vehicle routing problems (VRPs). Recently, PSO, which is an emerging evolutionary computing method, has been successfully applied for solving some VRP variants, including the capacitated vehicle routing problem (Ai and Kachitvichyanukul; 2007a, 2008a) and the vehicle routing problem with simultaneous pickup and delivery (Ai and Kachitvichyanukul, 2008b).

Similar with other evolutionary computing methods, it is necessary to properly select the PSO parameters in order to yield good performance. The task to find the best set of parameters for all problem cases is not a trivial one. Much experiment needs to be performed to determine proper values of parameters. Moreover, there is no guarantee that the selected parameter set will yield best algorithm performance, especially when the algorithm is applied to solve new problem cases. A novel idea to replace the way to find the best set parameter is through a self-adaptive PSO algorithm that can adapt its parameters automatically whenever it is applied to solve a problem instance. It is noted that in the wider scope of evolutionary algorithm, some approaches for adaptively finding the algorithm's parameter have been proposed, i.e. Annunziato and Pizzuti

---

† : Corresponding Author

(2000) and Back *et al.* (2000).

In the scope of PSO, several researchers have also dealt with adaptive or self-finding parameter. Among PSO parameters, inertia weight has gained enormous attention in the earlier effort to adapt PSO parameters. Since the early development of PSO, the proper setting of inertia weight is believed to have significant effect on the PSO performance. The two most popular setting for the inertia weight are a linear decreasing function that was first proposed by Shi and Eberhart (1998), and a nonlinear decreasing function proposed by Gao and Ren (2007). With these settings, it is expected that the particles are able to explore the problem space more aggressively at the beginning of the iteration steps and to exploit promising solution in the end of iteration steps.

Other approaches that have been proposed attempts to adjust the inertia weight adaptively based on the particular condition of the swarm. Ueno *et al.* (2005) proposed an adaptive PSO that alternates its inertia weight between a high value and a low value and vice versa in order to control the swarm's velocity. Arumugam and Rao (2008) used the value of local best and global best at a particular iteration as the basis for updating the values of inertia weight. Population diversity of the swarm has also been used as the basis to adaptively adjust the inertia weight, i.e. Dan *et al.* (2006), Jie *et al.* (2006), and Zhang *et al.* (2007).

Borrowing some ideas from those earlier researches in parameter adaptation, especially for adapting the inertia weight, an adaptive PSO algorithm is proposed. The adaptive PSO algorithm proposed in this paper has the capability to self-adapt its inertia weight based on the dynamics of the swarm, the searching agents in PSO. The mechanism of this adaptation is selected so that the existing PSO algorithm for solving VRP is only slightly modified to have the adaptive feature. Furthermore, the selected adaptive mechanism does not significantly increase the computational effort of PSO.

The remainder of this paper is organized as follow: Section 2 briefly reviewed the PSO algorithm for solving VRP. Section 3 presents the adaptive mechanism for setting inertia weight. Section 4 describes the computational results on the benchmark data set. Finally, Section 5 concludes the work presented in this paper and recommends further direction on this work.

2. PSO FOR SOLVING VRP

PSO is a population based search method which imitated the physical movements of the individuals in the swarm as a searching method. In the PSO, the best solution of a specific problem is being searched by a swarm of particles that act as a searching agent. A multi-dimensional particle position is being used to represent problem solution and a velocity vector is being used to represent the searching ability of the particle. Each PSO iteration step consists of the movement of every particle in the swarm from one position to the next based on the velocity. Moving from one position to another, a particle is evaluating different prospective solution of the problem. In imitating swarm's cognitive and social behavior, the PSO mechanism also always keeps the information on the personal best position of each particle, which is defined as the position that gives the best objective function among the positions that have been visited by the particle, and the global best position, which is the best among all personal best. These personal best and global best position are used for updating particle velocity. More information on PSO mechanism, techniques, and applications is provided by Kennedy and Eberhart (2001) and also Clerc (2006).

In the earlier works of Ai and Kachitvichyanukul; (2007a, 2008a, 2008b), a PSO framework for solving VRP had been proposed based on the GLNPSO, a PSO Algorithm with multiple social learning structures (Pongchairerks and Kachitvichyanukul, 2005). This PSO version also incorporates the local best, which is the best position among several adjacent particles, and the near neighbor best, which is social learning behavior concept proposed by Veeramachaneni (2003), besides the global best as components for social learning behavior. The PSO framework is briefly reviewed in Algorithm 1.

Algorithm 1: PSO Framework for VRP

Step 1. *Initialization*
- a. Generate particles as member of the swarm.
- b. Set the initial position and velocity of each particle.

Step 2. *Iteration Process*
- a. Decode each particle position to a set of vehicle routes.
- b. Evaluate the performance of each set of vehicle routes and set the performance value as the fitness value of the corresponding particle.
- c. Update personal best, global best, local best and near neighbor best values.
- d. Update the velocity and position of each particle based on the updated values.

Step 3. *Stopping*
Stop if the stopping criterion is met, otherwise repeat Step 2.

In this framework, *L* particles are initialized in Step 1.a in which each particle dimension is randomly generated between a minimum and a maximum value. The initial velocity vector is zero for all particles. In the iteration process, the following equations are used to update the velocity and position of each position:

$$\omega_{lh}(\tau+1)=w(\tau)\,\omega_{lh}(\tau)+c_p u\left(\psi_{lh}-\theta_{lh}(\tau)\right)+c_g u\left(\psi_{gh}-\theta_{lh}(\tau)\right)+c_l u\left(\psi_{lh}^{L}-\theta_{lh}(\tau)\right)+c_n u\left(\psi_{lh}^{N}-\theta_{lh}(\tau)\right) \quad (1)$$

$$\theta_{lh}(\tau+1)=\theta_{lh}(\tau)+\omega_{lh}(\tau+1) \quad (2)$$

where:

$\omega_{lh}(\tau)$: Velocity of the l^{th} particle at the h^{th} dimension in the τ^{th} iteration

$\theta_{lh}(\tau)$: Position of the l^{th} particle at the h^{th} dimension in the τ^{th} iteration

$w(\tau)$: Inertia weight in the τ^{th} iteration

ψ_{lh} : Personal best position (pbest) of the l^{th} particle at the h^{th} dimension

ψ_{gh} : Global best position (gbest) at the h^{th} dimension

ψ_{lh}^{L} : Local best position (lbest) of the l^{th} particle at the h^{th} dimension

ψ_{lh}^{N} : Near neighbor best position (nbest) of the l^{th} particle at the h^{th} dimension

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

u : Uniform random number in the interval $[0,1]$

In this research, the adaptation is made on the PSO Framework for VRP by using the two solution representations that had been proposed in previous works of Ai and Kachitvichyanukul (2007a, 2008a, 2008b), which are called solution representation SR–1 and SR–2. For representing VRP with n customers and m vehicles, the representation SR–1 is using particle with (n+2m) dimensions and the representation SR–2 is using 3m dimensions. Each representation can be transformed into VRP solution by a specific decoding method. The detail of each decoding method is not presented here, since it has been clearly explained in cited references.

3. PROPOSED ADAPTIVE MECHANISM FOR SETTING INERTIA WEIGHT

The proposed adaptive mechanism for setting inertia weight borrows idea from Ueno *et al.* (2005), in which the algorithm self-adjusts the inertia weight in order to control movement of the swarm which is represented by its velocity index. It is known that different inertia weight value leads to different swarm movement behavior, since high value caused particles in the swarm to maintain its current movement and low value caused the particles to follow the cognitive and social terms. However as compared with Ueno's work, the proposed algorithm uses a different velocity index pattern and a different mechanism for adjusting the inertia weight.

It is noted that the velocity index of the swarm ($\bar{\omega}$) can be calculated using following expression:

$$\bar{\omega}=\frac{\sum_{l=1}^{L}\sum_{h=1}^{H}\left|\omega_{lh}\right|}{L\cdot H} \quad (3)$$

where:

l : Particle index, $l=1\ldots L$

h : Dimension index, $h=1\ldots H$

The velocity index measures how fast the swarm moves in certain iteration and is defined as the average of absolute velocity. This index indicates the moving behavior of the swarm: higher index means the swarm moves more aggressively in the problem space than the swarm with lower index.

Regarding the velocity index pattern that must be followed by the swarm, Ueno *et al.* (2005) used a linear decreasing pattern. However, the study of the dynamic behavior of the swarm in PSO by Ai and Kachitvichyanukul (2007b) implied 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. Hence, the proposed algorithm incorporates the idea of latter work as the velocity index pattern. It is intended that half of iterations are placed as exploration phase and the other half as exploitation phase. Two-step linear decreasing pattern is selected to portray this condition, in which the desired velocity index (ω^*) has following equation:

$$\omega^*=\begin{cases}\left(1-\dfrac{1.8\tau}{T}\right)\omega^{\max}, & 0\le\tau\le T/2\\[2ex] \left(0.2-\dfrac{0.2\tau}{T}\right)\omega^{\max}, & T/2\le\tau\le T\end{cases} \quad (4)$$

where:

τ : Iteration index; $\tau=1\ldots T$

$\omega^{\max}$: Maximum Velocity Index

By using equation 4, the desired 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 at the second half of iterations when the desired velocity index is small enough and slowly reduced from $0.1\omega^{\max}$ to 0. A comparison of the desired velocity index pattern of Ueno's and this proposed mechanism is illustrated in Figure 1.

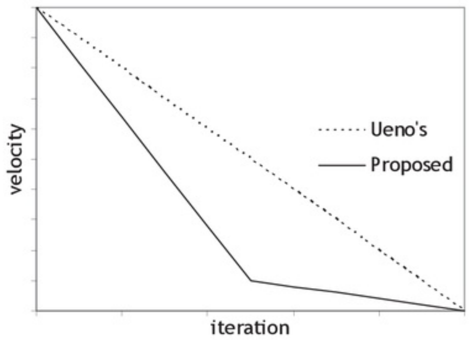

Figure 1: Desired Velocity Index Pattern of the Swarm

In Ueno's work, there are only two preset values of inertia weight, the lower and the higher value, and the inertia weight that is used in certain iteration is selected based on the current swarm velocity index. When the swarm velocity index is greater than the desired velocity index, the inertia weight is set to the lower value in order to reduce swarm velocity index in the subsequent iteration. In the reverse situation, when the swarm velocity index is lower than the desired velocity index, the inertia weight is set to the higher value in order to increase swarm velocity index in the subsequent iteration.

In this proposed mechanism, the inertia weight is set in the range of minimum ($w^{\min}$) and maximum value ($w^{\max}$) instead of using two preset values only. Nevertheless, the updating mechanism principle is similar with Ueno's 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. The amount of increases or decreases of inertia weight depends on the difference between the velocity index of the swarm and the desired velocity index. The following equations are used to update the inertia weight:

$$\Delta w = \frac{(\omega^* - \bar{\omega})}{\omega^{\max}}\left(w^{\max} - w^{\min}\right) \tag{5}$$

$$w = w + \Delta w \tag{6}$$

$$w = w^{\max} \quad \text{if} \quad w > w^{\max} \tag{7}$$

$$w = w^{\min} \quad \text{if} \quad w < w^{\min} \tag{8}$$

Based on the description given above, the proposed mechanism only requires a slight modification of the PSO Framework for VRP (Algorithm 1). Since the inertia weight is only used while updating velocity in the Step 2d, the steps of calculating desired and actual velocity index and updating the inertia weight following equations 5 – 8 must occur before Step 2d. It is expected that these additional steps should have only slight impact on the computational effort.

The complete algorithm of the adaptive PSO algorithm incorporating the proposed mechanism is presented in Algorithm 2, which is called the APSO-1 algorithm. It is noted that the APSO-1 algorithm incorporates the same number of parameters as the non-adaptive PSO algorithm in Algorithm 1, however, its inertia weight is controlled by swarm dynamics instead of strictly followed the pre-defined values.

Algorithm 2: APSO-1 Algorithm for VRP

```
Step 1.  Initialization
         a. Generate particles as member of the swarm.
         b. Set the initial position and velocity of each particle.
Step 2.  Iteration Process
         a. Decode each particle position to a set of vehicle routes.
         b. Evaluate the performance of each set of vehicle routes and set the performance value as the fitness value of the corresponding particle.
         c. Update personal best, global best, local best and near neighbor best values.
         d. Calculate the actual and desired velocity index using Eqs. 3 and 4, and then update the inertia weight using Eqs. 5–8.
         e. Update the velocity and position of each particle based on the updated values.
Step 3.  Stopping
         Stop if the stopping criterion is met, otherwise repeat Step 2.
```

4. COMPUTATIONAL TEST

Computational test is conducted to compare the performance of existing non-adaptive PSO algorithm (Algorithm 1) and the proposed APSO-1 algorithm (Algorithm 2). For this purpose, totally new problem instances of vehicle routing problem are generated which incorporates the features of simultaneously pickup-delivery and time windows of customer. Two classes of 200-customers problem are generated, in which each class consists of four instances. The main difference between the first and the second class is the time windows characteristic, in which the first class (class RL) has wider time windows than the second class (class RT). In both problem classes, the traveled time between two locations is defined to be equal to its Euclidean distance. The detail specification of these two classes of problems is described in Table 1.

Table 1: Specification of Generated VRP Benchmark Data

Characteristic	**Class RL**	**Class RT**
Depot		
Location	(50, 50)	(50, 50)
Customer		
Location	U[(0, 0); (100, 100)]	U[(0, 0); (100, 100)]
Pickup Quantity	U[0, 30]	U[0, 30]
Delivery Quantity	U[0, 30]	U[0, 30]
Service Time	10	10
Earliest Time for Starting Service (ET)	U[0, 100]	U[0, 400]
Latest Time for Starting Service (LT)	ET + U[0, 400]	ET + U[0, 100]
Vehicle		
Fixed Cost	0	0
Variable Cost	1	1
Capacity	300	300
Duration Limit	500	500

For computational test purpose, both algorithms are written in C# language using Microsoft Visual Studio.NET 1.1 and run on a PC with Intel P4 3.4 GHz processor and 1 GB RAM. The test is conducted using only the solution representations SR–2 for representing VRP solution in both non-adaptive PSO and APSO-1 algorithm. It is noted that the non-adaptive PSO is using following parameters setting: $L=50$, $T=1000$, $K=5$, $w(1)=0.9$, $w(T)=0.4$, $c_p=1$, $c_g=1$, $c_l=1$, and $c_n=1$. In addition, the APSO-1 is using the first iteration velocity index as the maximum velocity index ($\omega^{\max}$), $w^{\max}=0.9$, and $w^{\min}=0.1$. For the remaining fixed parameters, the APSO-1 incorporated the same parameters as non-adaptive PSO, in which $L=50$, $T=1000$, $K=5$, $c_p=1$, $c_g=1$, $c_l=1$, and $c_n=1$. For each instance, five replications of algorithm runs are performed. The computational results comprising the average objective function found and computational result for each instance are presented in Table 2.

As seen in Table 2, the APSO-1 result is relatively better than the non-adaptive PSO result. In the objective function column of this table, the bold typeface indicates smaller result between two algorithms results. It is found that the APSO-1 algorithm provides smaller average objective function value than the non-adaptive PSO algorithm in five out of eight instances.

Table 2: Computational Results

Instance	**Objective Function**		**Comp. Time***	
	PSO	**APSO-1**	**PSO**	**APSO-1**
RL1	2159.78	***2150.28***	15:39.6	16:29.0
RL2	***2060.00***	2080.51	16:23.4	16:52.5
RL3	2106.58	***2072.19***	15:03.6	16:04.1
RL4	1988.08	***1961.28***	16:48.3	17:58.5
RT1	2653.43	***2645.32***	18:22.4	19:24.3
RT2	***2622.28***	2626.85	18:35.9	20:30.1
RT3	***2674.92***	2686.89	20:07.0	20:36.9
RT4	2548.23	***2535.68***	20:09.7	21:19.8

* in minutes:seconds

It is also empirically shown from Table 2 that the adaptive versions of PSO algorithm, the APSO-1 algorithm, require slightly more computational time than the non-adaptive one. This additional time is a consequence of additional effort to adjust the inertia weight in the APSO-1 algorithm.

Observation on the details of each instance run may give better understanding of the behavior of both algorithms. In figure 2, velocity index over iteration of some algorithm runs is drawn. It is clearly seen that the velocity index of non-adaptive PSO algorithm is continuously decreasing and very fast approaching the exploitation phase. On the other hand, the velocity index of APSO-1 algorithm is decreased with the lower rate than the non-adaptive PSO algorithm. As a result, it is always higher than the corresponding value of the non-adaptive PSO algorithm in whole iteration steps. Hence, it is implied that the APSO-1 algorithm has more potential to explore the search space than the non-adaptive PSO algorithm in the exploration phase. This higher level of exploration is desirable, since it may avoid the searching process being trapped into local optima and also lead to better final solution.

However, the velocity index of the APSO-1 algorithm is also still slightly higher than the index of non-adaptive PSO algorithm in the exploitation phase. It is implied that the APSO-1 does not have the same level of exploitation as the non-adaptive PSO algorithm. This difference level of exploitation might be suspected as the source of inconsistency in the APSO-1 algorithm results, in which some problem instances of APSO-1 result are worse than the non-adaptive PSO result. If this hypothesis was true, the desired velocity index (ω^*) pattern could be slightly change, i.e. decreased from $\omega^{\max}$ at the first iteration to $0.05\omega^{\max}$ at the first half of iterations and slowly reduced from $0.05\omega^{\max}$ to 0 at the second half of iterations, to improve the result. Though, more experiments should be conducted for this purpose.

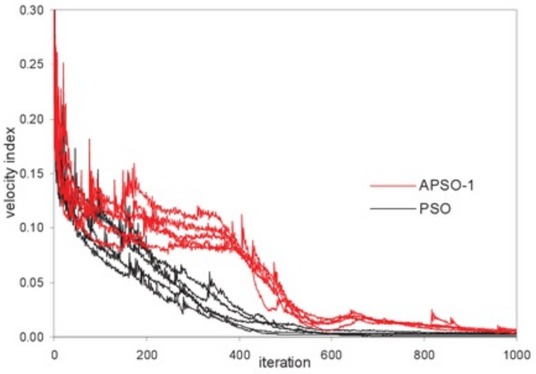

Figure 2: Velocity Index Pattern of Typical Runs on Non-Adaptive PSO and APSO-1 Algorithms

5. CONCLUSION AND FURTHER WORKS

A possibility to enable particle swarm optimization algorithm to self-adapt its parameter is presented in this paper in which an adaptive version of PSO is proposed with capability to self-adapt its inertia weight, one of the key PSO parameter. The computational experiment on some vehicle routing problem instance shows that the proposed adaptive PSO algorithm is able to provide better solution than the existing non-adaptive PSO with slightly slower computational time.

Further works is still required to explore more mechanisms for adapting other parameters of PSO algorithms, such as: acceleration constants, number of particles, number of neighbors, and number of iterations.

REFERENCES

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

Ai, T.J. and Kachitvichyanukul, V. (2007b) Dispersion and velocity indices for observing dynamic behavior of particle swarm optimization. *Proceedings of the 2007 IEEE Congress on Evolutionary Computation, CEC 2007*, 3264–3271.

Ai, T.J. and Kachitvichyanukul, V. (2008a) Particle swarm optimization and two solution representations for solving the capacitated vehicle routing problem. *Computers and Industrial Engineering*, doi:10.1016/j.cie.2008.06.012

Ai, T.J. and Kachitvichyanukul, V. (2008b) 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

Annunziato, M. and Pizzuti, S. (2000) Adaptive parameterization of evolutionary algorithms driven by reproduction and competition. *Proceedings of European Symposium on Intelligent Techniques 2000*, 246–256.

Arumugam, M.S. and Rao, M.V.C. (2008) 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*, **8**, 324–336.

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

Dan, L., Liqun, G., Junzheng, Z. and Yang, L. (2006) Power system reactive power optimization based on adaptive particle swarm optimization algorithm. *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, 7572–7576.

Gao, Y. and Ren, Z. (2007) Adaptive particle swarm optimization algorithm with genetic mutation operation. *Proceedings of the Third International Conference on Natural Computation, ICNC 2007*, 211–215.

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

Pongchairerks, P. and Kachitvichyanukul, V. (2005) A non-homogenous particle swarm optimization with multiple social structures. *Proceedings of International Conference on Simulation and Modeling*, Paper A5-02.

Shi, Y. and Eberhart, R. (1998) A modified particle swarm optimizer. *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation*, 69–73.

Ueno, G., Yasuda, K., Iwasaki, N. (2005) Robust adaptive particle swarm optimization. *Proceedings of the 2005 IEEE International Conference on Systems, Man and Cybernetics*, 3915–3920.

Veeramachaneni K, Peram T, Mohan C and Osadciw L.A. (2003) Optimization using particle swarms with near neighbor interaction. *Proceedings of Genetic and Evolutionary Computation Conference 2003*, 110–121.

Zhang, D., Guan, Z. and Liu, X. (2007) An adaptive particle swarm optimization algorithm and simulation. *Proceedings of the IEEE International Conference on Automation and Logistics, ICAL 2007*, 2399–2402.

AUTHOR BIOGRAPHIES

The Jin Ai is a Doctoral Candidate in Industrial Engineering & Management, School of Engineering and Technology, Asian Institute of Technology, Thailand. He is on study leave from Universitas Atma Jaya Yogyakarta, Indonesia. His research interests include evolutionary computation and its application on industrial systems. Currently, he is working on the application of particle swarm optimization on vehicle routing problems. His email address is <thejin.ai@ait.ac.th> or <jinai@mail.uajy.ac.id>

Voratas Kachitvichyanukul is an Associate Professor in Industrial Engineering & Management, School of Engineering and Technology, Asian Institute of Technology, Thailand. He received a Ph. D. from the School of Industrial Engineering at Purdue University in 1982. He has extensive experiences in modeling of manufacturing systems. He had worked for FORTUNE 500 Companies such as Compaq Computer Corporation and Motorola Incorporated. He had also worked for SEMATECH as technical coordinator of the future factory program. His teaching and research interests include evolutionary algorithms for combinatorial optimization, planning and scheduling, high performance computing and applied operations research with special emphasis on industrial systems. His email address is <voratas@ait.ac.th>

Paper 26

ORIGINALITY REPORT

36%	27%	31%	16%
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	4%
2	www.citeulike.org Internet Source	4%
3	Kasemset, Chompoonoot. "Application of Adaptive Particle Swarm Optimization to Bi-level Job-Shop Scheduling Problem", Industrial Engineering and Management Systems, 2014. Publication	3%
4	210.72.131.170 Internet Source	3%
5	The Jin Ai, Voratas Kachitvichyanukul. "Particle swarm optimization and two solution representations for solving the capacitated vehicle routing problem", Computers & Industrial Engineering, 2009	2%

	Publication	
6	www.vpaa.ait.ac.th Internet Source	2%
7	docplayer.net Internet Source	2%
8	eprints.uns.ac.id Internet Source	1%
9	archive.org Internet Source	1%
10	Kanchana Sethanan, Woraya Neungmatcha. "Multi-objective particle swarm optimization for mechanical harvester route planning of sugarcane field operations", European Journal of Operational Research, 2016 Publication	1%
11	Jiuping Xu. "Bi-Random Multiple Objective Decision Making", Lecture Notes in Economics and Mathematical Systems, 2011 Publication	1%
12	pt.scribd.com Internet Source	1%
13	Jun Gang, Yan Tu, Benjamin Lev, Jiuping Xu, Wenjing Shen, Liming Yao. "A multi-objective bi-level location planning problem for stone industrial parks", Computers & Operations	1%

	Research, 2015	Publication	
14	www.knu.edu.tw	Internet Source	1 %
15	Submitted to Erasmus University of Rotterdam	Student Paper	1 %
16	Pisut Pongchairerks. "Particle Swarm Optimization algorithm with multiple social learning structures", International Journal of Operational Research, 2009	Publication	1 %
17	www.cs.uoi.gr	Internet Source	1 %
18	Submitted to University of South Australia	Student Paper	<1 %
19	"Electrical, Information Engineering and Mechatronics 2011", Springer Nature, 2012	Publication	<1 %
20	www.inderscience.com	Internet Source	<1 %
21	www.slideshare.net	Internet Source	<1 %
22	"Simulated Evolution and Learning", Springer Science and Business Media LLC, 2010	Publication	<1 %

23	www.deepdyve.com Internet Source	<1%
24	Lecture Notes in Computer Science, 2006. Publication	<1%
25	Studies in Computational Intelligence, 2014. Publication	<1%
26	S. Nguyen, V. Kachitvichyanukul. "Movement Strategies for Multi-Objective Particle Swarm Optimization", International Journal of Applied Metaheuristic Computing, 2010 Publication	<1%
27	epdf.tips Internet Source	<1%
28	Seyedeh Atefeh Mohammadi, Morteza Rahmani, Majid Azadi. "Meta-heuristic CRPS minimization for the calibration of short-range probabilistic forecasts", Meteorology and Atmospheric Physics, 2016 Publication	<1%
29	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 Publication	<1%

30	Ahmad Nickabadi, Reza Safabakhsh, Mohammad Mehdi Ebadzadeh. "Inertia weight control strategies for PSO algorithms", Institution of Engineering and Technology (IET), 2018 Publication	<1%
31	zimmer.csufresno.edu Internet Source	<1%
32	Sun, Shufang, Jiahai Zhang, Jianhui Wang, and Lin Xu. "The Application of New Adaptive PSO in AGC and AFC Combination Control System", Procedia Engineering, 2011. Publication	<1%
33	Zhang, Limin, Yinggan Tang, Changchun Hua, and Xinping Guan. "A new particle swarm optimization algorithm with adaptive inertia weight based on Bayesian techniques", Applied Soft Computing, 2015. Publication	<1%
34	repository.ntu.edu.sg Internet Source	<1%
35	Submitted to University of the Western Cape Student Paper	<1%
36	Suraya Masrom, Irene Moser, James Montgomery, Siti Zaleha Zainal Abidin, Nasiroh Omar. "Hybridization of Particle Swarm	<1%

	Optimization with adaptive genetic algorithm operators", 2013 13th International Conference on Intellient Systems Design and Applications, 2013 Publication	
37	Submitted to University of Witwatersrand Student Paper	<1%
38	Terraneo, Federico, Luigi Rinaldi, Martina Maggio, Alessandro Vittorio Papadopoulos, and Alberto Leva. "FLOPSYNC-2: Efficient Monotonic Clock Synchronisation", 2014 IEEE Real-Time Systems Symposium, 2014. Publication	<1%
39	mini-chip.eu Internet Source	<1%
40	ojs.academypublisher.com Internet Source	<1%
41	I. Montalvo, J. Izquierdo, R. Pérez, P. L. Iglesias. "A diversity-enriched variant of discrete PSO applied to the design of water distribution networks", Engineering Optimization, 2008 Publication	<1%
42	Byung-In Kim, So-Jung Son. "A probability matrix based particle swarm optimization for the capacitated vehicle routing problem", Journal of Intelligent Manufacturing, 2010 Publication	<1%

---

43 Djoko Budiyanto Setyohadi, Sri Kusrohmaniah, Sebastian Bagya Gunawan, Pranowo Pranowo, Anton Satria Prabuwono. "Galvanic Skin Response Data Classification for Emotion Detection", International Journal of Electrical and Computer Engineering (IJECE), 2018

Publication

<1%

---

44 The Jin Ai. "A Particle Swarm Optimisation for Vehicle Routing Problem with Time Windows", International Journal of Operational Research, 2009

Publication

<1%

---

45 ALHASAN, Hussein ALRUIM and GÜNEŞ, Mahit. "Yeni Bir Adaptif Parçacık Sürü Optimizasyon Algoritması Kullanarak DC Motor ", Çukurova Üniversitesi, 2017.

Publication

<1%

---

46 Mojtaba Taherkhani, Reza Safabakhsh. "A novel stability-based adaptive inertia weight for particle swarm optimization", Applied Soft Computing, 2016

Publication

<1%

---

47 Li-Jun Yun, Bao-Cang Ding, He-Xu Sun, Yong-Xiao Niu. "Stability of State Feedback Two-Step MPC for Input Saturated Uncertain Hammerstein Model", 2005 International Conference on Machine Learning and

<1%

Cybernetics, 2005

Publication

48	Proceedings of the Institute of Industrial Engineers Asian Conference 2013, 2013. Publication	<1%
49	Lecture Notes in Computer Science, 2010. Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		