Scientific Research Publishing

Journal of Computer and Communications, 2017, 5, 13-23
http://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Particle Swarm Optimization Algorithm Based on Chaotic Sequences and Dynamic Self-Adaptive Strategy

Mengshan Li, Liang Liu, Genqin Sun, Keming Su, Huaijin Zhang, Bingsheng Chen, Yan Wu

Gannan Normal University, Ganzhou, China
Email: jcimsli@163.com

How to cite this paper: Li, M.S., Liu, L., Sun, G.Q., Su, K.M., Zhang, H.J., Chen, B.S. and Wu, Y. (2017) Particle Swarm Optimization Algorithm Based on Chaotic Sequences and Dynamic Self-Adaptive Strategy. *Journal of Computer and Communications*, **5**, 13-23.
https://doi.org/10.4236/jcc.2017.512002

Received: September 13, 2017
Accepted: September 27, 2017
Published: September 30, 2017

Abstract

To deal with the problems of premature convergence and tending to jump into the local optimum in the traditional particle swarm optimization, a novel improved particle swarm optimization algorithm was proposed. The self-adaptive inertia weight factor was used to accelerate the converging speed, and chaotic sequences were used to tune the acceleration coefficients for the balance between exploration and exploitation. The performance of the proposed algorithm was tested on four classical multi-objective optimization functions by comparing with the non-dominated sorting genetic algorithm and multi-objective particle swarm optimization algorithm. The results verified the effectiveness of the algorithm, which improved the premature convergence problem with faster convergence rate and strong ability to jump out of local optimum.

Keywords

Particle Swarm, Algorithm, Chaotic Sequences, Self-Adaptive Strategy, Multi-Objective Optimization

1. Introduction

Particle swarm optimization (PSO) algorithm is a swarm intelligence optimization algorithm, which derives agglomeration of organism behavior, such as a simulation of the behavior of a flock of birds or fish. Compared to other intelligent algorithms, PSO algorithm has simple structure, less parameters and is easy to describe and implement. The global search ability is stronger, without gradient information and many other features, which makes it to be widely used in many fields such as function optimization, multi-objective problem solving and

pattern recognition. In particular, it is applicable to solving the problems of nonlinear, multipolar and non-differentiable and complex optimization [1] [2] [3]. However, the standard PSO algorithm also has shortcomings such as premature convergence and bad local searching ability similar to other intelligent algorithms [4] [5] [6]. For example, in the optimization of complex problems in high-dimension, the population may have accumulated to a certain point of stagnation without finding the global optimization point, forming premature convergence. In other words, the premature convergence problem does not guarantee that the algorithm can converge to the global extreme point. Meanwhile, in the PSO algorithm search process, when the particle is approaching or entering the most advantageous region, the convergence speed is obviously slow. That is, in the later period of particle optimization, the search ability is poor. Thus, the application of PSO algorithm is restricted.

For the lack of PSO algorithm, the researchers propose many improvement strategies [7]. Inertia weighting factor [8], contraction factor and adaptive mutation operator are the most representative, such as linear decrease method [9], fuzzy adaptive method [10], distance information method and other inertial coefficients adaptive adjustment methods [11], PSO algorithm with compression factor, the PSO algorithm of adaptive mutation operator, etc. In addition, the PSO algorithm and the hybrid PSO algorithm combined with the PSO algorithm, synergy polices, chaos theory [12] and other algorithms [13] are also attracted by the researchers, such as quantum PSO algorithm with chaotic mutation operator [14]. In addition, there are also many researches on discrete PSO algorithm, multi-objective PSO algorithm [15] [16], etc. At present, the improvement of PSO algorithm mainly focuses on two aspects: adjustment of algorithm parameters and update of particle structure and trajectory. The aim is to make the algorithm solve or improve the local search slow, precocious convergence and so on, and improve the convergence speed and accuracy of the algorithm to improve the performance of the algorithm [17]. Although the proposed particle swarm improvement algorithm improves both performance and efficiency, it is difficult to improve the local search ability of the algorithm while avoiding precocious convergence. To provide better, more efficient, and cheaper particle swarm algorithms, academics and industry researchers have been exploring and experimenting with new approaches [18].

In order to improve search precision and convergence speed of the standard PSO algorithm, this paper tries to propose a more efficient and higher convergence speed algorithm by combining chaos theory with dynamic adaptive weight adjustment strategy. A new chaos self-adaptive particle swarm optimization algorithm (CSAPSO) was proposed. The algorithm improves the convergence speed by adaptive adjustment strategy evolution inertia weight. The chaotic sequence of chaos theory is used to optimize the learning factor of the algorithm, so that it can get out of the local optimum when it comes to precocious convergence. Finally, according to the solution experiment of multi-objective optimization problem, by comparing and analyzing the CSAPSO algorithm with the

standard PSO algorithm and the classical multi-objective algorithm, the feasibility and validity of the algorithm are verified and the convergence speed and accuracy are discussed.

2. CSAPSO Algorithm

2.1. Standard PSO

Particle Swarm Optimization algorithm (PSO) is a group evolution algorithm proposed by scholars Eberhart and Kennedy [19] based on the social behaviors of birds in 1995. The PSO algorithm is derived from the behavior characteristics of biological groups and is used to solve the optimization problems. It has the advantages of easy description, easy implementation, little adjustment parameter, fast convergence speed and low calculation cost. And there is no high requirement for memory and CPU speed. In the process of particle optimization, the potential solution of the problem is assumed to be a "particle" in the n-dimensional space, and the particle will fly at a certain speed and direction in the solution space. In the iterative process, all particles use two global variables to represent the best position of the particle itself (pbest) and the best position of all particles (gbest). It is assumed that, in an n-dimensional search space, the particle population $X=\left(x_1,x_2,\cdots,x_n\right)^{\mathrm{T}}$ is composed of m particles. The position of the ith particle is denoted as $x_i=\left(x_{i,1},x_{i,2},\cdots,x_{i,n}\right)^{\mathrm{T}}$ and the velocity is denoted as $v_i=\left(v_{i,1},v_{i,2},\cdots,v_{i,n}\right)^{\mathrm{T}}$. The individual extremum is $p_i=\left(p_{i,1},p_{i,2},\cdots,p_{i,n}\right)^{\mathrm{T}}$, The global extreme of the population of particles is $p_g=\left(p_{g,1},p_{g,2},\cdots,p_{g,n}\right)^{\mathrm{T}}$. During the $k+1$ iteration, the particle updates its speed and position through Formulas (1) and (2).

$$v_{i,d}^{k+1}=\omega v_{i,d}^{k}+c_1\left(p_{i,d}^{k}-x_{i,d}^{k}\right)+c_2\left(p_{g,d}^{k}-x_{i,d}^{k}\right) \tag{1}$$

$$x_{i,d}^{k+1}=x_{i,d}^{k}+v_{i,d}^{k+1} \tag{2}$$

where, $i=1,\cdots,m$; ω is called an inertial weight factor, it makes the particles keep sport inertia and have the ability to expand search space; C_1 and C_2 are the learning factors, which represent the weight of each particle to the statistical acceleration item of the extremum position; rand () is a random number within (0, 1), $v_{i,d}^{k}$ and $x_{i,d}^{k}$ are respectively the velocity and position of particle i in d–dimension kth iteration; $p_{i,d}^{k}$ is the position of the individual extremum of particle i in d-dimension, is the position of the global extremum of the whole population in d-dimension.

2.2. CSAPSO Algorithm

The standard PSO algorithm has its own limitations, such as the implementation process of the algorithm has a great relationship with the value of the parameters. When the algorithm is applied to the complex optimization problem of high dimension, the algorithm tends to converge to some extreme point and stagnates when the global optimum is not found, that is, precocious convergence is easy to occur. These points can be a point in the local extreme point or local

extreme point area. In addition, the convergence rate of the algorithm becomes slow when approaching or entering the optimal solution area. The early convergence rate of PSO algorithm is fast, but in the later stage, when the algorithm converges to local minimum, due to the lack of effective local search mechanism, the local search speed is slow. According to the formula of particle velocity update of PSO algorithm, the change of particle velocity is determined by three factors: 1) The inertial weight factor, which represents the velocity information at the previous moment. It indicates the relationship between current speed and forward speed. 2) The cognitive factor, which is the development capacity coefficient, represents the error of the optimum of the particle itself, and reflects the local mining and development capability of the particle. 3) The exploration factor, which is the social sharing ability coefficient, represents the error of global optimum, the information sharing and cooperation ability between the particles. Under the circumstances, the inertia coefficient determines the search step length. When it is larger, it is good for global search. When it is smaller, it is helpful for local exploration. Cognitive factors and exploration factors are collectively called learning factors, which represent the effect proportion of the optimum of the particle itself and global optimum. By adjusting the learning factors properly, the global and local search of the particles can be weighed. When the algorithm is in precocious convergence, it is possible to change the exploration factor to achieve out of local optimal.

In order to improve the precocious convergence of the algorithm and improve the convergence speed of the algorithm, this paper uses adaptive weight adjustment strategy to realize the dynamic adjustment of inertia coefficient. The chaotic sequence generated by chaotic mapping is used to optimize the parameters of learning factor C_1 and C_2, and a chaos self-adaptive particle swarm optimization algorithm (CSAPSO) is obtained. The inertial weight factor ω is adjusted by Formula (3).

$$\omega = \omega_{\max} - Pgbest(k)/Plbest_{ave} - (\omega_{\max} - \omega_{\min}) \times k/k_{\max} \tag{3}$$

where, $\omega_{\max}$ and $\omega_{\min}$ respectively represent the maximum and minimum values of inertial weight; $Pgbest\ (k)$ represents the global optimal for the kth iteration; $Plbest_{ave}$ represents the local optimal average of all particles; $k_{\max}$ is maximum number of iterations; k is current iteration times.

Learning factor C_1 and C_2 are adjusted by chaotic sequence generated by chaotic mapping. This paper uses the typical Lorenz's equation to generate chaotic sequences, as shown in Formula (4).

$$\begin{cases} \dfrac{\mathrm{d}x}{\mathrm{d}t} = -(y+z) \\ \dfrac{\mathrm{d}y}{\mathrm{d}t} = x + ay \\ \dfrac{\mathrm{d}z}{\mathrm{d}t} = b + xz - cz \end{cases} \tag{4}$$

where, parameters a, b and c are controlled parameters, which are 0.2, 0.4 and

5.7 respectively. The learning factor (c_1, c_2) is defined as:

$$\begin{cases} c_1 = x(t) \\ c_2 = y(t) \end{cases} \tag{5}$$

Because the change of chaotic variables is random, ergodic and regular, the algorithm can maintain the diversity of population, effectively overcome the problem of precocious convergence, and improve global search performance.

The CSAPSO algorithm performs the following process:

1) To initialize the particle group

The position and velocity of particles in PSO algorithm are initialized. The initial position and velocity of the particles are generated randomly. The current position of each particle is used as the particle individual extremum, and the optimal value of the individual extremum is selected as the global optimal value.

2) To calculate the adaptive value of group particles.

3) The adaptive value of each particle is compared with the adaptive value of the best position it has passed. If it is better, the current position is the best position of the particle.

4) The adaptive value of each particle is compared with the adaptive value of the global best position, and if it is better, the current position is the global best position.

5) The learning factor C_1, C_2 and inertial weight ω were obtained respectively, and the velocity and position of the particles were updated and optimized

6) If the end condition of the algorithm is satisfied, the global best position is the optimal solution, saving the result and ending. Otherwise return to Step (2).

3. Numerical Experiment

3.1. Experiment Function and Evaluation

In order to test the performance of CSAPSO algorithm, this paper selects the multi-objective optimization test functions proposed by Schaffer [20] and Deb [21] as an experimental case. Multi-objective optimization problem is the most typical optimization problems, due to the constant contradiction and constraint among targets, it is difficult to achieve the optimal at the same time, as well as one of the optimization of goals must be at the expense of the other goals. The solutions to such problems are usually not unique, but a series of optimal solutions, also called non-inferior solutions. A collection of non-inferred solutions is often referred to as Pareto optimal solution. Because intelligent algorithm can search multiple solutions of solution space in parallel, multi-objective optimization is more suitable to verify the performance of intelligent algorithm. The multi-objective optimization test functions used in this paper are shown in **Table 1**.

In order to evaluate the merits of non-inferior solutions, this paper adopts the convergence index and the distribution index to evaluate the performance of the algorithm, and the indexes of convergence and distribution uniformity are respectively defined as follows [2] [3]:

Table 1. Test functions used in this paper.

Function	Definition	Property
SCH1	$\min f(x)=(f_1(x),f_2(x))$ s.t. $x\in[-5,7]$ $f_1(x)=x^2$ $f_2(x)=(x-2)^2$	Convex
SCH2	$\min f(x)=(f_1(x),f_2(x))$ s.t. $x\in[-5,10]$ $f_1(x)=\begin{cases}-x,(x\le 1)\\-2+x,(1<x\le 3)\\4-x,(3<x\le 4)\\-4+x,(x>4)\end{cases}$ $f_2(x)=(x-5)^2$	Discontinuous
ZDT2	$\min f(x)=(f_1(x),f_2(x))$ s.t. $x\in[-5,10]$ $f_1(x)=x_1$ $f_2(x)=g(x)\left[1-(x_1/g(x))^2\right]$ $g(x)=1+9\left(\sum_{i=2}^{n}x_i\right)\Big/(n-1)$	Concave
ZDT3	$\min f(x)=(f_1(x),f_2(x))$ s.t. $x_i\in[0,1]$ $f_1(x)=x_1$ $f_2(x)=g(x)\left[1-\sqrt{x_1/g(x)}-\frac{x_1}{g(x)}\sin(10\pi x_1)\right]$ $g(x)=1+9\left(\sum_{i=2}^{n}x_i\right)\Big/(n-1)$	Discrete

1) Convergence index (GD), GD is used to describe the distance between the ungoverned solution that the algorithm searches for and the optimal front-end of the real Pareto.

$$\mathrm{GD}=\frac{\sqrt{\sum_{i=1}^{N}d_i^2}}{N} \tag{6}$$

where, N represents the number of ungoverned solutions that the algorithm searches for, d_i^2 represents the shortest Euclidean distance between the non-inferior solution i and all solutions in the optimal front-end of the real Pareto.

2) Distribution index (SP), SP is used to evaluate the uniformity of distribution of ungoverned solution.

$$\mathrm{SP}=\frac{\left[\frac{1}{n}\sum_{i=1}^{n}\left(\bar{d}-d_i\right)^2\right]^{1/2}}{\bar{d}} \tag{7}$$

$$\overline{d} = \frac{1}{n}\sum_{i=1}^{n} d_i \tag{8}$$

where, N is the number of ungoverned solutions, and represents the shortest distance between the ith non-inferior solution in the target space and all solutions in the optimal front-end of the real Pareto.

3.2. Experimental Results

The CSAPSO algorithm was used to experiment with SCH1, SCH2, ZDT2 and ZDT3. The algorithm parameter is set to: the particle size is 50; the maximum iteration number is 100; the maximum and minimum values of inertia weight are 0.9 and 0.3 respectively; inertia weight ω and learning factor C_1 and C_2 are obtained according to Formulas (3) and (5) respectively. The Pareto non-inferior solutions of each function are shown in Figures 1-4.

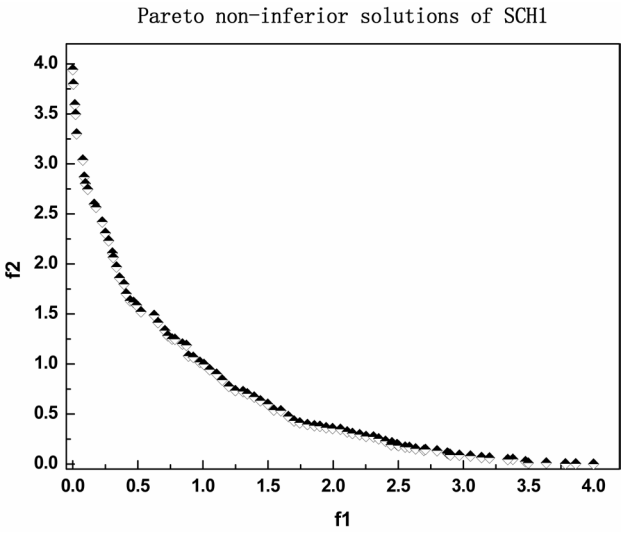

Figure 1. Pareto non-inferior solution of SCH1.

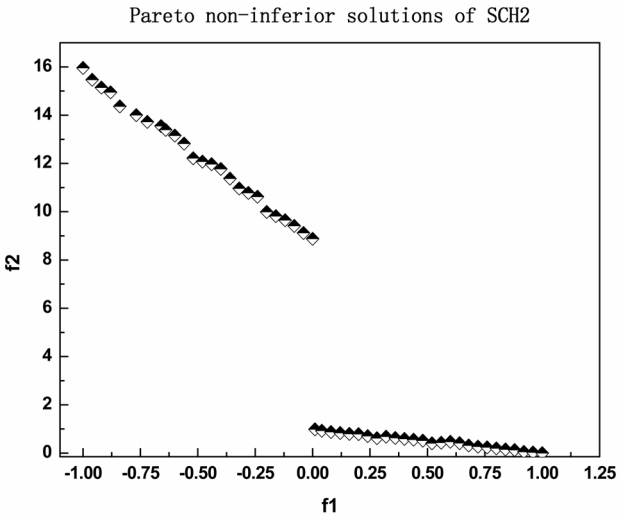

Figure 2. Pareto non-inferior solution of SCH2.

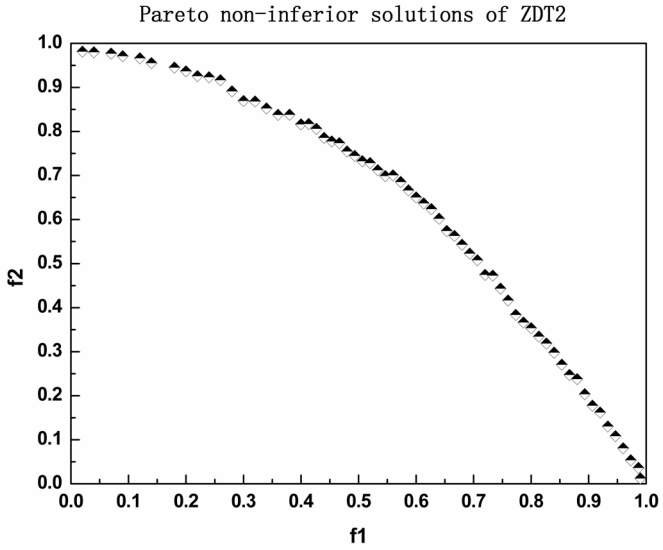

Figure 3. Pareto non-inferior solution of ZDT2.

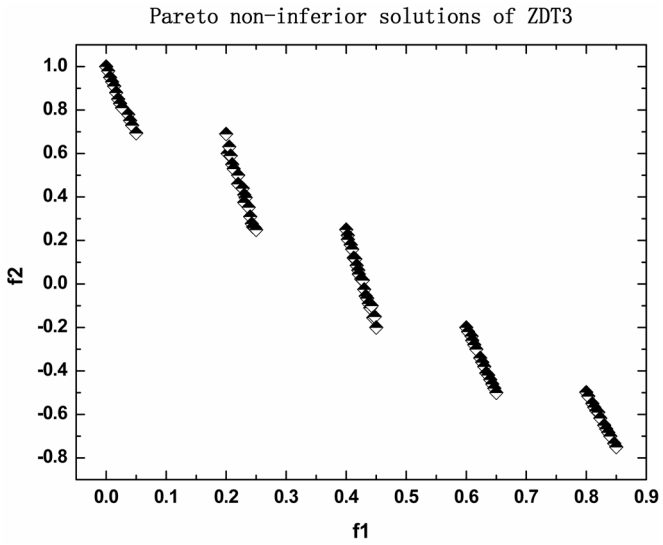

Figure 4. Pareto non-inferior solution of ZDT3.

In the target function space, non-inferior optimal target domain is the boundary of the fitness value region, which is the effective interface. It can be seen from the experimental results that the four test functions accurately give the effective interface, and the complete Pareto non-inferior solution can be obtained. In particular, for the discrete problem of ZDT3, the algorithm also gives a more accurate non-inferior solution. In general, the number of Pareto solutions obtained by the algorithm is more and the distribution is more uniform. The accuracy and reliability of CSAPSO algorithm are verified.

Through CSAPSO algorithm runs 30 times for each test function, the convergence index GD, distribution index SP and the mean value of computed time CT were respectively calculated, and four test function evaluation index are calculated, the results are shown in Table 2.

Table 2. Performance statistics of CASPSO.

Performance Index	SCH1	SCH2	ZDT2	ZDT3	Average
GD[a]	0.000335	0.000334	0.000351	0.000332	0.000338
SP[b]	0.00338	0.00337	0.00333	0.00327	0.00334
CT[c]	6.4	7.8	6.6	9.6	7.6

[a]Convergence index, [b]Distribution index, [c]Calculated time (*s*).

The evaluation index GD, SP and CT confirmed the feasibility, accuracy and efficiency of the CSAPSO algorithm for solving multi-objective optimization problems. GD indicates that the non-inferior solution is very close to the optimal front end of the real Pareto. SP shows that the non-inferior solution has good distribution. CT shows that the time spent running is within acceptable limits.

In order to test the superiority of the algorithm in multi-objective optimization solution, comparing CSAPSO algorithm in this paper with classic non-poor classification multi-objective genetic algorithm(NSGA-II), multi-objective particle swarm optimization, the statistical comparison results are shown in **Table 3**.

According to GD of **Table 3**, the convergence of CSAPSO algorithm is better than the other two algorithms. It is shown that the optimal front distance between the non-inferior solution and real Pareto is smaller. That is, the solution is closer to the real solution. The SP of **Table 3** shows that the distribution of non-inferior solutions obtained by the CSAPSO algorithm is better, that is, the distribution of the non-inferior solution of the algorithm is more uniform than the other two algorithms. For CT, the execution time of CSAPSO algorithm is between two algorithms, lower than NSGA II algorithm but higher than MOPSO. The reason for this is that the standard PSO algorithm is according to the equal step and flying in a single direction search, while CSAPSO algorithm dynamically adjusted with the flight process of particle, the process of dynamic adjustment will consume more time. Although the CSAPSO algorithm spends more computation time, the convergence and distribution of non-inferior solutions are better than the other two algorithms, which can obtain more and more evenly distributed feasible solutions.

In conclusion, through the CSAPSO algorithm for the numerical experiments of four multi-objective optimization problems, compared with the classical multi-objective optimization of NSGA II algorithm and MOPSO algorithm, we can know, CSAPSO algorithm has a better comprehensive performance. Algorithm improves the convergence speed by dynamic adaptive mechanism. Through the chaotic learning mechanism, the precocious convergence problem is improved.

4. Conclusions

This paper presents a chaotic self-adaptive particle swarm optimization algorithm (CSAPSO). The algorithm uses chaos theory and dynamic adaptive adjustment strategy to optimize the parameters in PSO algorithm, improve the

Table 3. Comparison results of three algorithms.

Performance Index	NSGA II	MOPSO	CSAPSO
GD	0.000362	0.000465	0.000338
SP	0.00475	0.00502	0.00334
CT	8.7	6.8	7.6

precocious convergence of PSO algorithm, and improve the convergence speed. By the experiment of four standard test functions, the proposed algorithm can be used to solve the multi-target problem, and the obtained non-inferiority solution can get a good approximation of the optimal solution set of Pareto and distribute evenly. By comparing with other algorithms, CSAPSO algorithm has better property, which can provide practical reference value for many optimization problems in the project. In the future, the convergence strategy and mathematical proof of the PSO algorithm can be discussed in-depth.

Acknowledgements

The authors gratefully acknowledge the support from the National Natural Science Foundation of China (Grant Numbers: 51663001, 51463015, 51377025) and the science and technology research project of the education department of Jiangxi province (Grant Numbers: GJJ151012, GJJ150983).

References

[1] Neri, F., Mininno, E. and Lacca, G. (2013) Compact Particle Swarm Optimization. *Information Sciences*, **239**, 96-121. https://doi.org/10.1016/j.ins.2013.03.026

[2] Tan, K.C., Lee, T.H. and Khor, E.F. (2002) Evolutionary Algorithms for Multi-Objective Optimization: Performance Assessments and Comparisons. *Artificial Intelligence Review*, **17**, 253-290. https://doi.org/10.1023/A:1015516501242

[3] Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M. and da Fonseca, V.G. (2003) Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, **7**, 117-132. https://doi.org/10.1109/TEVC.2003.810758

[4] del Valle, Y., Venayagamoorthy, G.K., Mohagheghi, S., Hernandez, J.C. and Harley, R.G. (2008) Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems. *IEEE Transactions on Evolutionary Computation*, **12**, 171-195. https://doi.org/10.1109/TEVC.2007.896686

[5] Tsekouras, G.E. and Tsimikas, J. (2013) On Training RBF Neural Networks Using Input-Output Fuzzy Clustering and Particle Swarm Optimization. *Fuzzy Sets and Systems*, **221**, 65-89. https://doi.org/10.1016/j.fss.2012.10.004

[6] Nickabadi, A., Ebadzadeh, M.M. and Safabakhsh, R. (2011) A Novel Particle Swarm Optimization Algorithm with Adaptive Inertia Weight. *Applied Soft Computing*, **11**, 3658-3670. https://doi.org/10.1016/j.asoc.2011.01.037

[7] Akay, B. (2013) A Study on Particle Swarm Optimization and Artificial Bee Colony Algorithms for Multilevel Thresholding. *Applied Soft Computing*, **13**, 3066-3091. https://doi.org/10.1016/j.asoc.2012.03.072

[8] Zhan, Z.H., Zhang, J., Li, Y. and Chung, H.S.H. (2009) Adaptive Particle Swarm

Optimization. *IEEE Transactions on Systems Man and Cybernetics Part B: Cybernetics*, **39**, 1362-1381. https://doi.org/10.1109/TSMCB.2009.2015956

[9] Khare, A. and Rangnekar, S. (2013) A Review of Particle Swarm Optimization and Its Applications in Solar Photovoltaic System. *Applied Soft Computing*, **13**, 2997-3006. https://doi.org/10.1016/j.asoc.2012.11.033

[10] Valdez, F., Melin, P. and Castillo, O. (2011) An Improved Evolutionary Method with Fuzzy Logic for Combining Particle Swarm Optimization and Genetic Algorithms. *Applied Soft Computing*, **11**, 2625-2632. https://doi.org/10.1016/j.asoc.2010.10.010

[11] Marinakis, Y. and Marinaki, M. (2013) Particle Swarm Optimization with Expanding Neighborhood Topology for the Permutation Flowshop Scheduling Problem. *Soft Computing*, **17**, 1159-1173. https://doi.org/10.1007/s00500-013-0992-z

[12] Cheng, M.Y., Huang, K.Y. and Chen, H.M. (2012) K-Means Particle Swarm Optimization with Embedded Chaotic Search for Solving Multidimensional Problems. *Applied Mathematics and Computation*, **219**, 3091-3099.

[13] Li, M.W., Kang, H.G., Zhou, P.F. and Hong, W.C. (2013) Hybrid Optimization Algorithm Based on Chaos, Cloud and Particle Swarm Optimization Algorithm. *Journal of Systems Engineering and Electronics*, **24**, 324-334. https://doi.org/10.1109/JSEE.2013.00041

[14] Coelho, L.D. (2008) A Quantum Particle Swarm Optimizer with Chaotic Mutation Operator. *Chaos Solitons & Fractals*, **37**, 1409-1418.

[15] Qasem, S.N., Shamsuddin, S.M., Hashim, S.Z.M., Darus, M. and Al-Shammari, E. (2013) Memetic Multiobjective Particle Swarm Optimization-Based Radial Basis Function Network for Classification Problems. *Information Sciences*, **239**, 165-190.

[16] Zheng, Y.J. and Chen, S.Y. (2013) Cooperative Particle Swarm Optimization for Multiobjective Transportation Planning. *Applied Intelligence*, **39**, 202-216. https://doi.org/10.1007/s10489-012-0405-5

[17] Gandomi, A.H., Yun, G.J., Yang, X.S. and Talatahari, S. (2013) Chaos-Enhanced Accelerated Particle Swarm Optimization. *Communications in Nonlinear Science and Numerical Simulation*, **18**, 327-340.

[18] Tsekouras, G.E. (2013) A Simple and Effective Algorithm for Implementing Particle Swarm Optimization in RBF Network's Design using Input-Output Fuzzy Clustering. *Neurocomputing*, **108**, 36-44.

[19] Kennedy, J. and Eberhart, R. (1995) Particle Swarm Optimization. In: 1995 *IEEE International Conference on Neural Networks Proceedings*, *Proceedings of International Conference on Neural Networks*, IEEE Australia Council, Perth, 1942-1948. https://doi.org/10.1109/ICNN.1995.488968

[20] Schaffer, J.D. (1985) Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In: *International Conference on Genetic Algorithms*, Lawrence Erlbaum Associates, 93-100.

[21] Deb, K. (1999) Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, **7**, 205-230. https://doi.org/10.1162/evco.1999.7.3.205