# Convergence Analysis of Particle Swarm Optimization via Illustration Styles

Wei-Der Chang

*Abstract*—**Particle swarm optimization (PSO) is the most important and popular algorithm to solving the engineering optimization problem due to its simple updating formulas and excellent searching capacity. This algorithm is one of evolutionary computations and is also a population-based algorithm. Traditionally, to demonstrate the convergence analysis of the PSO algorithm or its related variations, simulation results in a numerical presentation are often given. This way may be unclear or unsuitable for some particular cases. Hence, this paper will adopt the illustration styles instead of numeric simulation results to more clearly clarify the convergence behavior of the algorithm. In addition, it is well known that three parameters used in the algorithm, i.e., the inertia weight *w*, position constants $c_1$ and $c_2$, sufficiently dominate the whole searching performance. The influence of these parameter settings on the algorithm convergence will be considered and examined via a simple two-dimensional function optimization problem. All simulation results are displayed using a series of illustrations with respect to various iteration numbers. Finally, some simple rules on how to suitably assign these parameters are also suggested**

*Index Terms*—**Particle swarm optimization (PSO), convergence analysis, illustration styles.**

## I. Introduction

Particle swarm optimization (PSO) algorithm was originally introduced by Kennedy and Eberhart in 1995 [1]. It belongs to one of evolutionary computations and is also a population-based algorithm. The initial concept of this algorithm is to simulate the social behavior of bird flock and fish school. Over the past two decades, the PSO algorithm has attracted a considerable attention of many researchers and gradually become a popular optimal algorithm because of possessing some good features such as real-valued manipulations, simple updating formulas, easy implementation into computer programs, and quick convergence. Owing to these advantages, a variety of physical engineering optimization problems have been successfully solved via the proposed PSO algorithm, including optimizing reader deployment in RFID systems [2], power and energy optimization [3]-[5], optimal controller design [6], [7], medical diagnosis [8], and other applications [9]-[14]. All of simulation results obtained from them fully revealed that the PSO is a rather powerful and effective algorithm in solving the optimized problems.

Manuscript received June 15, 2021; revised August 12, 2021. This work was supported in part by the Ministry of Science and Technology of Taiwan under Grant MOST 108-2221-E-366-003.

W. D. Chang is with the Department of Computer and Communication, Shu-Te University, Kaohsiung, Taiwan (e-mail: wdchang@stu.edu.tw).

In recent years, in order to further prompt and enhance the searching capacity of the algorithm, some variations of the PSO were successively developed [15]-[22]. It is well known that the PSO algorithm only uses two updating formulas to update the particle positions; that is, the velocity and position updating formulas. In the velocity formula, three system parameters need to be assigned when the algorithm is executed, including the inertia weight $w$ and two positive constants $c_1$ and $c_2$, respectively. These parameters will dominate the final simulation results derived while different value settings are given. As a result, most of the PSO variations mentioned above are to propose a modified version or a new updating scheme for them. In Ref. [15], for example, the authors applied an improved self-adaptive PSO algorithm to the fuzzy clustering in which these three parameters are varying with respect to the number of iteration, not a fixed value, in order to achieve the ideal balance between local and global search. Another updating scheme for $w$, $c_1$, and $c_2$ was introduced in [19]. These three parameters are the linear function of iteration number, and the maximal iteration number is also taken into account. Based on the modified PSO version, the pretension optimization of a double-ring deployable cable net antenna was achieved. In [20], a monotonically decreasing logarithmic function was employed for updating the inertia weight, where the maximum inertia weight $w_{\max}$ and minimal inertia weight $w_{\min}$, and maximum number of iterations were involved. Furthermore, they thought that chaotic sequences embedded in the optimization algorithm are always able to improve the exploitation capacity of the algorithm in the search space. Consequently, in the developed algorithm the inertia weight $w$ decreases linearly with $w_{\max}$ and $w_{\min}$, and the original uniformly random number utilized in the velocity updating formula is replaced by the chaotic random sequences generated by a simple logistic map chaotic system [22].

In the traditional fashion, the authors always provide a series of numerical results and comparisons to show the performance of their proposed PSO algorithms over the general algorithms. For some special cases, it seems to be slightly unclear and unsuitable and there may be no senses to most of the readers. As a result, this paper will illustrate the convergence analysis of the PSO algorithm with a large number of illustrations, instead of numerical results, with respect to the number of iterations in which a simple two-dimensional function optimization problem is considered. Moreover, different sets of parameter settings for three parameters $w$, $c_1$, and $c_2$ utilized in the algorithm are examined to show their influences on finding out the correct system solution. The remainder of this paper is simply outlined as follows. In Section II the general PSO algorithm is introduced in detail including its two updating formulas. In addition, the stability analysis of a simple first-order difference equation is addressed because it is related with the velocity updating formula of the algorithm. Various simulation results by illustration styles are provided in Section III and how to assign these three parameters is concluded as well. Finally, Section IV gives some simple conclusions and future study direction.

## II. The General PSO Algorithm and Its Stability Analysis

### A. PSO Algorithm

Over the past two decades, the PSO algorithm has been proven to be a powerful but simple optimal algorithm by successfully examining a variety of optimization problems. This algorithm is motivated by the social behavior of a flock of birds and a school of fishes, and it is also an iterative and population-based algorithm. In the general PSO, a particle also called an individual is a collection of all the designed variables and can be regarded as a candidate solution of the optimized problem. Usually, the algorithm needs a large number of particles to construct a so-called population. This population will be further evolved by the two simple updating formulas of the PSO including the velocity and position formulas so that all the particles are able to move toward the better system solutions over the search space. During the evolution, two important factors are necessary to be recorded, i.e., the individual best denoted by *pbest* for each particle and the global best by *gbest* for the whole population. These two factors are utilized in the velocity updating formula to significantly guide the moving of all particles. Moreover, the performance of each particle is evaluated by its corresponding cost function which is defined previously. Generally speaking, it is a better particle when its cost function value is smaller.

Before introducing the updating formulas of the algorithm, let the representation of a particle be $\Theta=[\theta_1,\theta_2,\cdots,\theta_n]$ where $\theta_j$ is the designed variable of the optimized system for $j=1,2,\cdots,n$, and *n* is the total of designed variables. Also, let $\Theta_i=[\theta_{i1},\theta_{i2},\cdots,\theta_{in}]$ be the representation of the *i*th particle for $i=1,2,\cdots,PS$ where $PS$ stands for the population size; that is, the number of particles within the population. Eqs. (1) and (2) are the velocity and position updating formulas, respectively, to adjust the positions of each particle

$$v_{ij}(k+1)=wv_{ij}(k)+c_1r_1(pbest_{ij}(k)-\theta_{ij}(k))$$
$$+c_2r_2(gbest_j(k)-\theta_{ij}(k)), \qquad (1)$$
$$\theta_{ij}(k+1)=\theta_{ij}(k)+v_{ij}(k+1), \qquad (2)$$

where $\theta_{ij}(k)$, $pbest_{ij}(k)$, and $gbest_j(k)$ represent the *j*th position components of the *i*th particle, the *i*th individual best particle, and the global best particle at *k*th iteration, respectively, $v_{ij}(k)$ is the *j*th velocity component of the *i*th particle at *k*th iteration, *w* is the inertia weight that is assigned by the designer to balance the global and local search, $c_1$ and $c_2$ are two assigned positive constants, $r_1$ and $r_2$ are two uniformly random numbers chosen from the interval $[0,1]$. The complete design steps for the general PSO algorithm execution can be listed below and its flow chart corresponding to design steps is shown in Fig. 1.

Step I. Create an initial population consisting of *PS* particles all produced from the search interval $[\theta_{\min},\theta_{\max}]$ randomly.

Step II. If the number of iterations is attained, then the algorithm stops.

Step III. Evaluate the cost function of each particle and record the individual best particle *pbest* and the global best particle *gbest*, respectively.

Step IV. Perform the velocity updating formula of Eq. (1) and position updating formula of Eq. (2) for each particle within the population.

Step V. Check the derived particle position by Eq. (3)

$$\theta_{ij}=\begin{cases}\theta_{\min} & if\ \theta_{ij}<\theta_{\min}\\ \theta_{ij} & if\ \theta_{\min}\le\theta_{ij}\le\theta_{\max}\\ \theta_{\max} & if\ \theta_{ij}>\theta_{\max}\end{cases},\ \text{for } i=1,2,\cdots,PS \text{ and}$$
$$j=1,2,\cdots,n. \qquad (3)$$

Step VI. Go back to Step III.

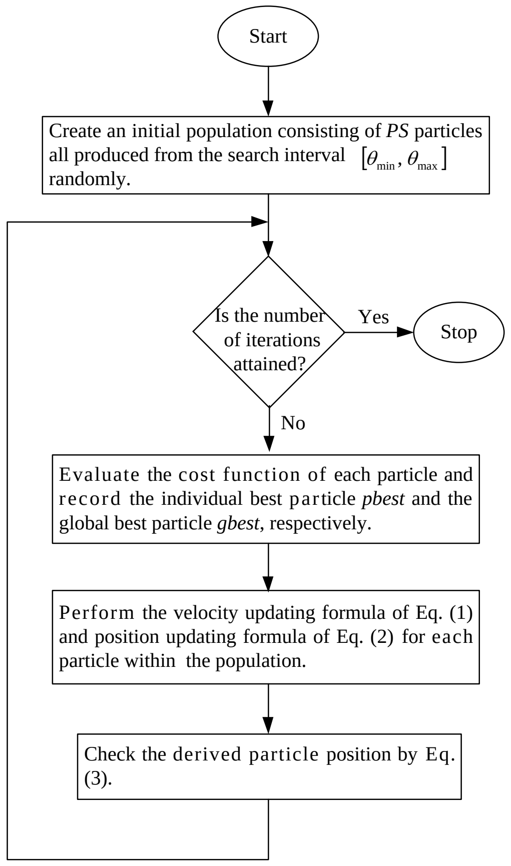
Start
Create an initial population consisting of PS particles all produced from the search interval [θmin, θmax] randomly.
Is the number of iterations attained?
Yes
Stop
No
Evaluate the cost function of each particle and record the individual best particle pbest and the global best particle gbest, respectively.
Perform the velocity updating formula of Eq. (1) and position updating formula of Eq. (2) for each particle within the population.
Check the derived particle position by Eq. (3).

Fig. 1. Design flow chart of the general PSO algorithm execution.

To solve the optimization problem using the PSO algorithm, it is very crucial for properly assigning a set of values for *w*, $c_1$ and $c_2$ because these parameters play the key roles in finding the system solution. Thus, this paper will discuss and analyze the effect of these three assigned parameters *w*, $c_1$ and $c_2$ on convergence behavior in solving the function optimization problem. All the simulation results are displayed by a series of illustrations with respect to different iteration numbers to clearly clarify the searching procedures toward the correct solution. Furthermore, a theoretical analysis for the inertia weight *w* is discussed according to the stability theorem of digital system. This will be introduced in the next subsection and the importance of $c_1$ and $c_2$ to the final results obtained is addressed as well.

### *B. A Simple First-Order Difference Equation*

Let us consider a simple first-order difference equation expressed by Eq. (4)

$$y(k+1) = a\,y(k), \tag{4}$$

where *y* represents the system output and *a* means the system coefficient. According to the stability theorem, the coefficient *a* fully determines whether the discrete system of Eq. (4) is stable or not. In general, there are three different classifications: stable, marginally stable, and unstable cases. The discrete system of Eq. (4) is said to be stable if $|a| < 1$, the system with $|a| = 1$ is marginally stable, and otherwise the system is said to be unstable when $|a| > 1$. These three classifications can be simply verified by giving an initial condition $y(0)$ with different values of *a* to excite the output response of this discrete system. Comparing Eq. (4) with Eq. (1), they are the same when the last two terms on the right side of Eq. (1) are omitted. It means that the inertia weight *w* in the velocity updating formula plays the same role as the coefficient *a* in the first-order discrete system. That is, the new velocity output will grow without bound if $w > 1$ is assigned. After that, the next position output by Eq. (2) is also unbounded. The velocity output is likely to be marginally stable, i.e. neither convergence nor divergence when the inertia weight is given by $w = 1$. It will cause the position of particles failing to sufficiently converge to the system optimum. In the third situation, the velocity output will die away when $w < 1$ is given, and subsequently the position output settles to some steady state. In Section 3, we will provide a large number of simulation illustrations to confirm these.

## III. A Simple Two-Dimensional Function Optimization

In order to show the effect of the algorithm parameters on convergence analysis in the form of illustrations, a simple function optimization problem with two variables *x* and *y* is illustrated. The optimized function can be simply expressed by Eq. (5)

$$f(x, y) = (x+2)^2 + (y+1)^2 . \tag{5}$$

It is very clear that Eq. (5) has only one function minimum point occurred at $(x, y) = (-2, -1)$. In the case, the searching interval is constrained by $-6 \le x \le 6$ and $-6 \le y \le 6$, respectively, i.e., $[\theta_{\min}, \theta_{\max}] = [-6, 6]$ used in Eq. (3). For all the simulations, the software of Borland C++ Builder with version 6 is utilized to implement the above PSO algorithm by means of illustration styles. The number of particles used in the algorithm is fixed and is chosen by $PS = 20$ for any case simulation. Besides, in the figure the red point stands for the correct solution of the function optimization that is the point $(-2, -1)$, and the global best particle *gbest* within the whole population is denoted by the green point. As to the other particles, they are then marked by the black. Furthermore, three different kinds of circumstances are considered including (a) the fixed parameters $c_1$ and $c_2$, and adjustable parameter *w*, (b) the fixed parameters *w* and $c_2$, and adjustable parameter $c_1$, and (c) the fixed parameters *w* and $c_1$, and adjustable parameter $c_2$ to explain the convergence behavior of all the particles on *x*-*y* plane.

- **Fixed parameters $c_1 = 0.5$ and $c_2 = 0.5$, and adjustable parameter *w***

In this circumstance, six different kinds of cases are further considered including

Case 1: $c_1 = 0.5$, $c_2 = 0.5$, and $w = 2$;
Case 2: $c_1 = 0.5$, $c_2 = 0.5$, and $w = 1$;
Case 3: $c_1 = 0.5$, $c_2 = 0.5$, and $w = 0.8$;
Case 4: $c_1 = 0.5$, $c_2 = 0.5$, and $w = 0.5$;
Case 5: $c_1 = 0.5$, $c_2 = 0.5$, and $w = 0.1$;
Case 6: $c_1 = 0.5$, $c_2 = 0.5$, and $w = 0$.

After executing the PSO algorithm by the Borland C++ Builder 6, simulation results are displayed in Figs. 2-7 for six different cases in illustration styles. In the first case, convergence behaviors of all particles are displayed by iteration=0, iteration=10, and iteration=20. Simulation results are shown in Fig. 2 for Case 1. In Case 1, the parameter $w = 2$ is given. Thus, it can be seen from Fig. 2 that all the particles except for the global best diverge from the correct solution $(-2, -1)$ and are eventually pushed into four corners of the search space as shown in Fig. 2(c). The final optimal solution (the global best particle) marked by the red still locates at $(x, y) = (-2.026924, -1.077583)$, and it is not a correct solution. The PSO algorithm cannot seek for the system optimum for such a simple optimized function under the kind of parameter settings. This fully conforms to the stability analysis of Section II for $w > 1$.

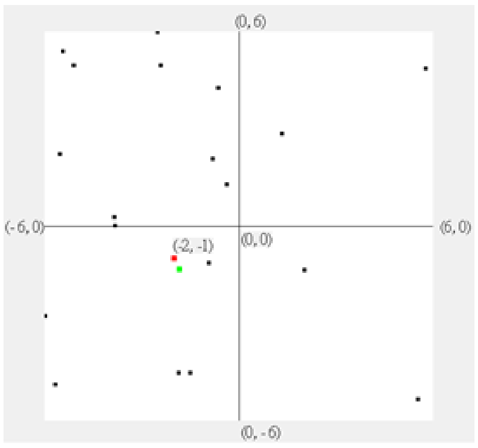

Fig. 2(a). Iteration 0.

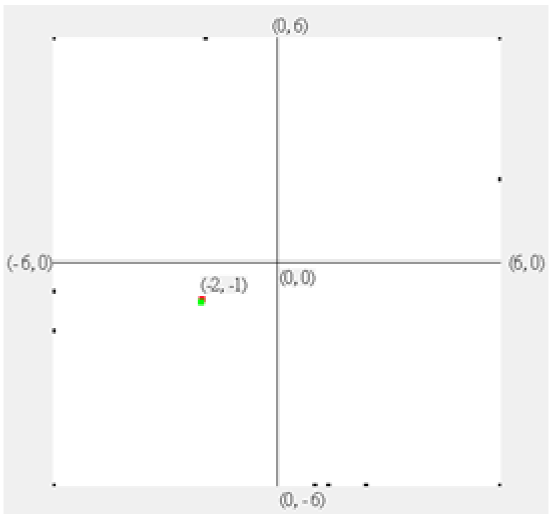

Fig. 2(b). Iteration 10.

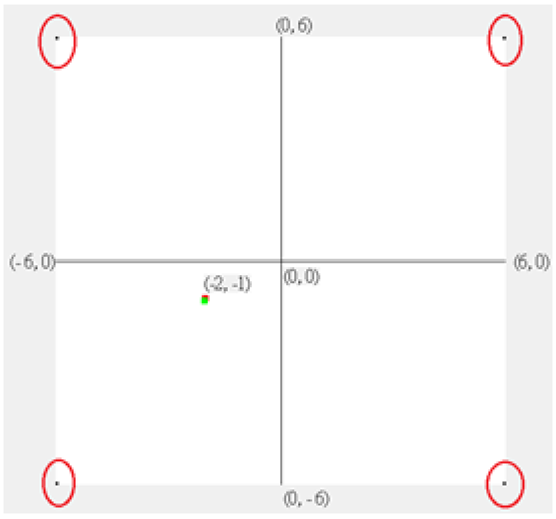

Fig. 2(c). Iteration 20.

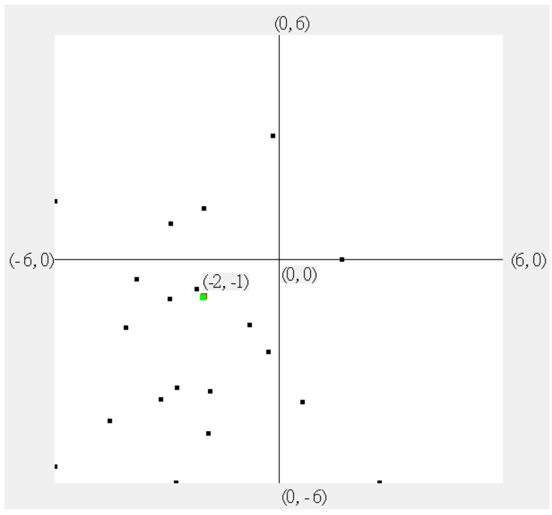

Fig. 3(a). Iteration 10.

Fig. 3 displays the convergence behavior of all particles by various iterations for Case 2 with $w=1$. We can find out from these figures that all particles neither converge nor diverge on the search space due to the parameter $w=1$. It is the case of marginally stable and causes all the particles failing to approximate to the correct solution. In Case 2, the derived optimal solution marked by the green is $x=-2.028547$ and $y=-0.994665$. The PSO algorithm neither catches the correct function solution nor forces all particles toward it for the above two cases.

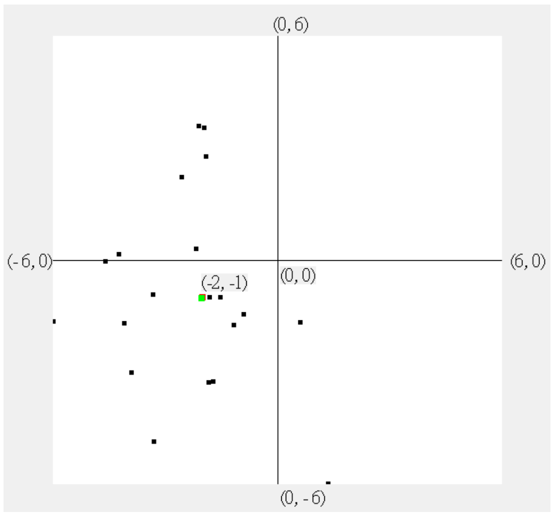

Fig. 3(b). Iteration 20.

In the following, the circumstance of $w<1$ including Cases 3-6 is considered. Simulation results for Case 3 with $w=0.8$ are displayed in Fig. 4. It is clearly seen from Fig. 4 that the moving trend of all the particles is towards the correct function solution $(-2,-1)$. After about executing 50 iterations, the global best particle (green point) can accurately catch the correct solution.

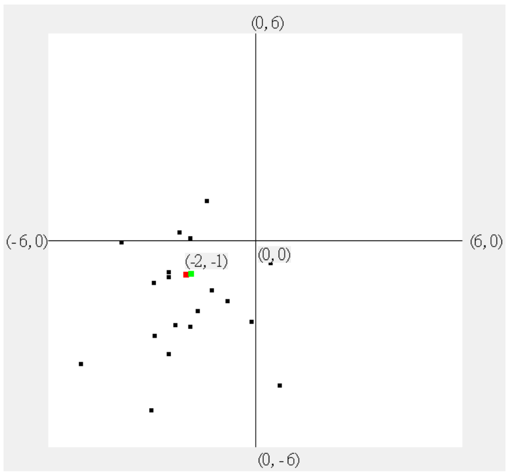

Fig. 4(a). Iteration 10.

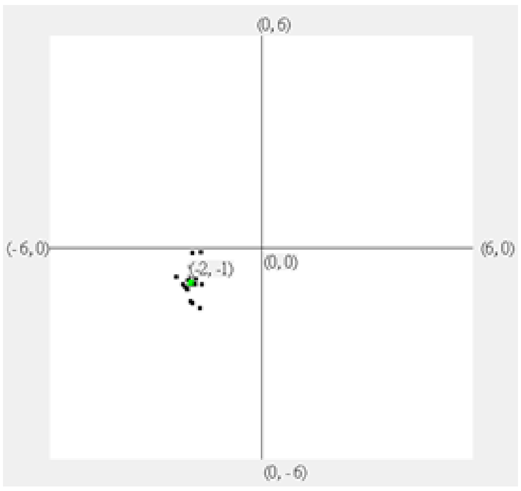

Fig. 4(b). Iteration 20.

In Case 4 with $w = 0.5$, its simulation illustrations are displayed in Fig. 5. About performing 20 iterations, the PSO algorithm successfully solves for the correct function solution. Fig. 5 clearly reveals the evidence that all particles can quickly converge on the system optimum $(-2, -1)$ over than those of Case 3.

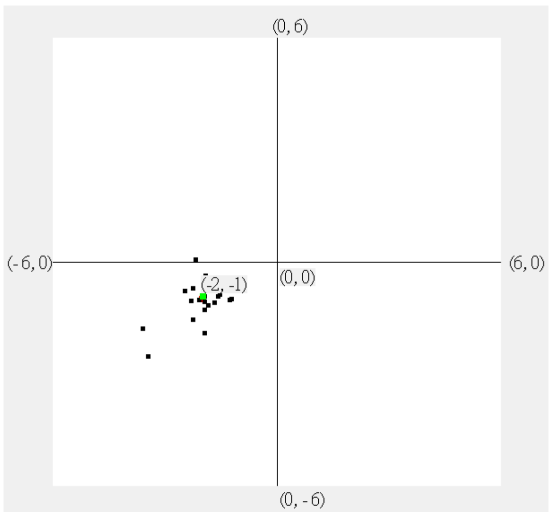

Fig. 5(a). Iteration 10.

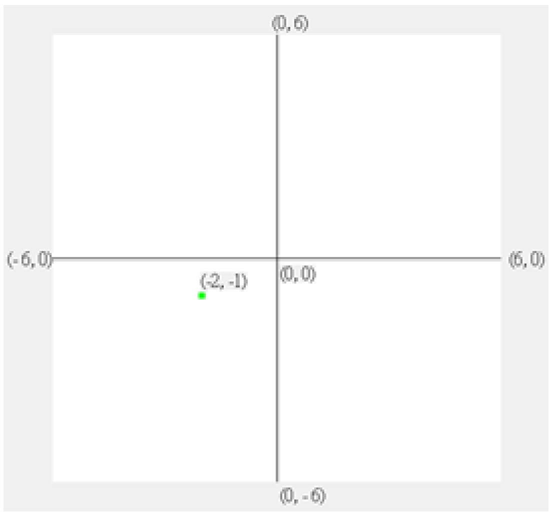

Fig. 5(b). Iteration 20.

Moreover, Figs. 6 and 7 then show the convergence behavior of particles in the population for Case 5 with $w = 0.1$ and Case 6 with $w = 0$, respectively. It is clearly seen from Figs. 6 and 7 that in both cases the PSO algorithm can also seek for the function optimum. It needs about 25 iterations for Case 5 and about 30 iterations for Case 6 to solve for the correct solution. Notice that in Case 6 the correct solution can also be derived by the PSO even though the parameter $w = 0$ is given. It means that the role of the inertia weight $w$ seems not to be important for solving such a function minimization problem. Some simple conclusions based on the above simulation results are addressed in the following.

(a) The PSO algorithm cannot perform well if $w > 1$ such as Cases 1 and 2.

(b) The PSO can accurately find out the correct function solution when $w < 1$ such as Cases 3-6.

(c) The best result is Case 4: $c_1 = 0.5$, $c_2 = 0.5$, and $w = 0.5$.

(d) The PSO algorithm can also derive the correct solution when $w = 0$.

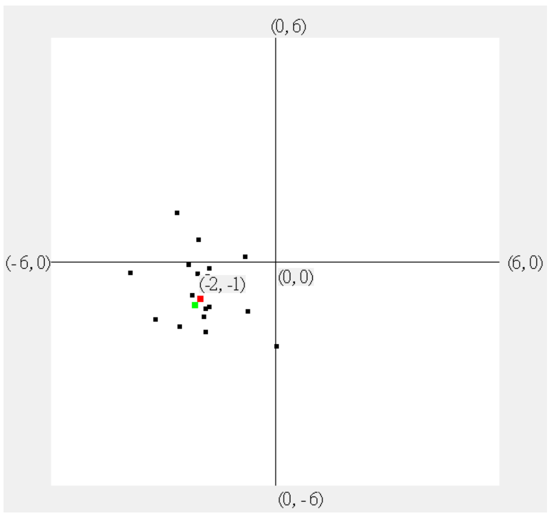

Fig. 6(a). Iteration 10.

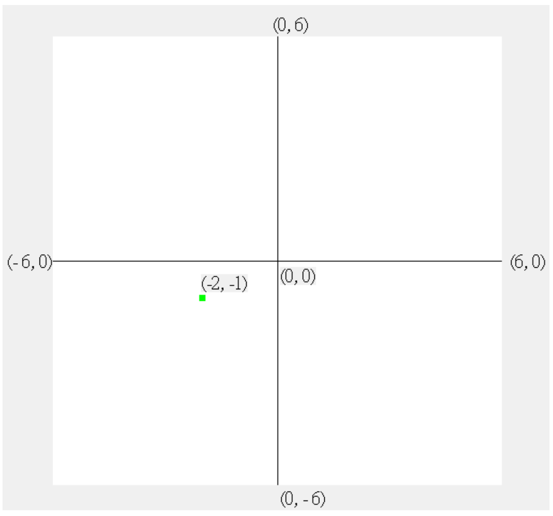

Fig. 6(b). Iteration 20.

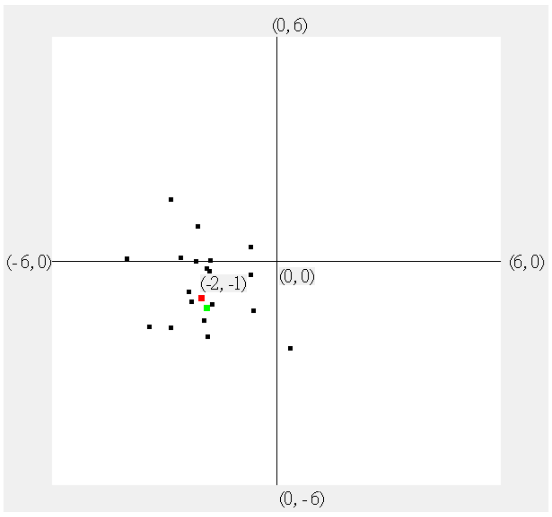

Fig. 7(a). Iteration 10.

- **Fixed** parameters $w = 0.5$ **and** $c_2 = 0.5$ **, and adjustable parameter** $c_1$

In this subsection, the parameters $w = 0.5$ and $c_2 = 0.5$ are fixed and parameter $c_1$ is adjustable. Totally, we have the following four different kinds of cases:

Case 7: $w = 0.5$, $c_2 = 0.5$, and $c_1 = 1$;

Case 8: $w = 0.5$, $c_2 = 0.5$, and $c_1 = 0.6$;

Case 9: $w = 0.5$, $c_2 = 0.5$, and $c_1 = 0.2$;

Case 10: $w = 0.5$, $c_2 = 0.5$, and $c_1 = 0$.

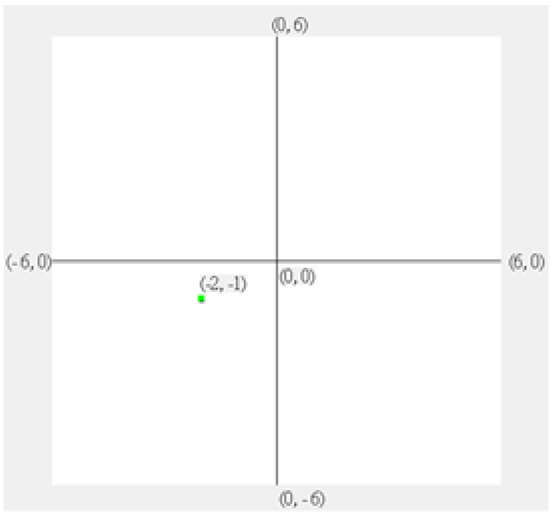

Fig. 7(b). Iteration 20.

After executing the PSO algorithm with the above parameters, simulation results that are in the form of illustrations are displayed in Figs. 8-11 with respect to different iteration numbers. Fig. 8 is shown for Case 7, and the correct solution can be obtained by executing 20 iterations. In this case, all particles also quickly converge to the function minimum point $(-2, -1)$. Similarly, simulation illustrations for Cases 8-10 are displayed in Figs. 9-11, respectively. As can be seen from these figures, the correct function solution can be rapidly caught about performing 20 iterations for any of parameter settings. Based on these simulation results, we can conclude that

(e) The individual best *pbest* that corresponds to the parameter $c_1$ in Eq. (1) is not a key factor of the PSO algorithm.

(f) Because the parameter $c_1 = 0$ is given in Case 10, the PSO algorithm can also perform well for solving this function optimization problem.

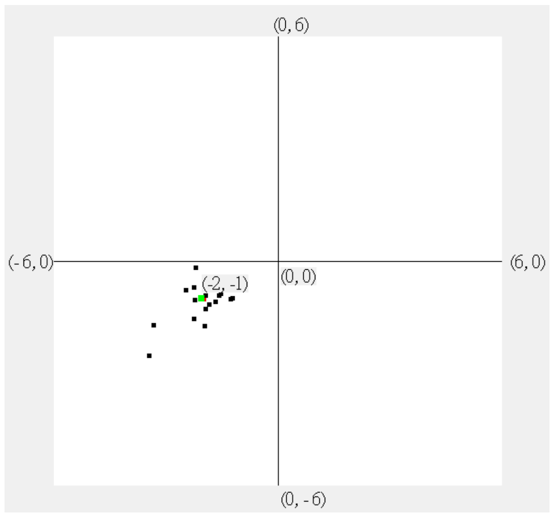

Fig. 8(a). Iteration 10.

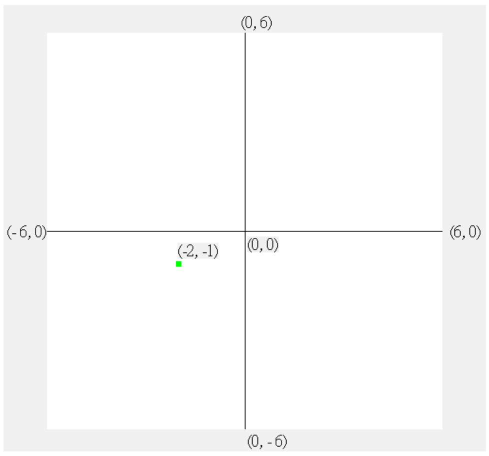

Fig. 8(b). Iteration 20.

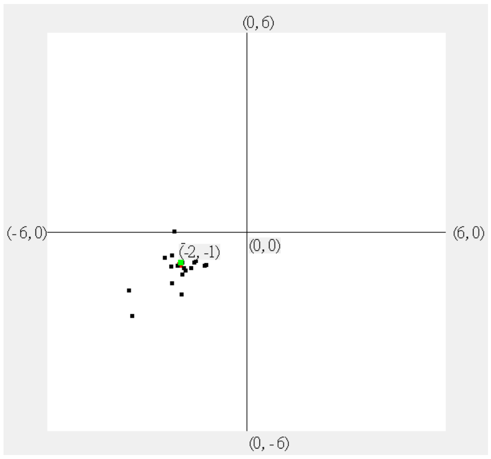

Fig. 9(a). Iteration 10.

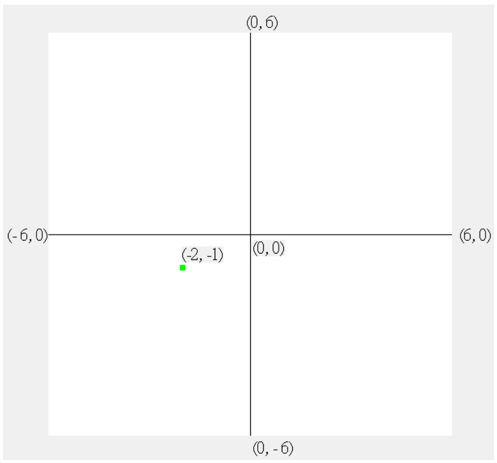

Fig. 9(b). Iteration 20.

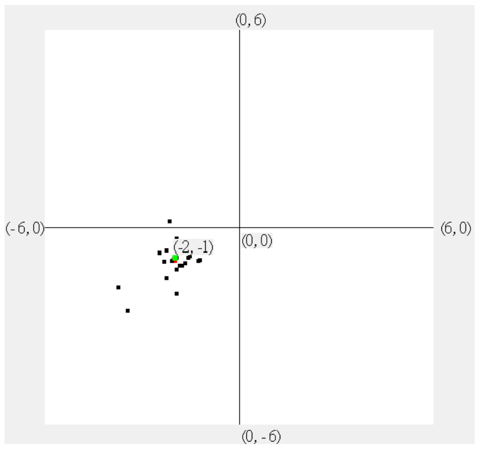

Fig. 10(a). Iteration 10.

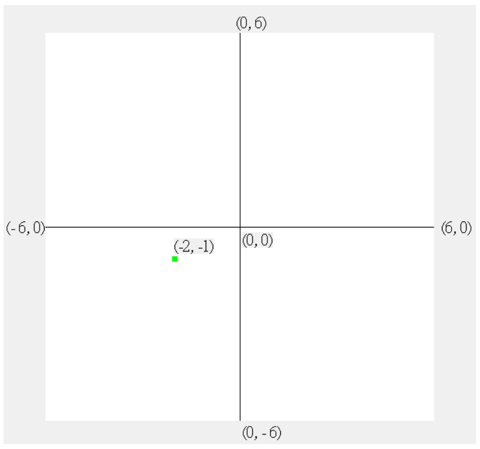

Fig. 10(b). Iteration 20.

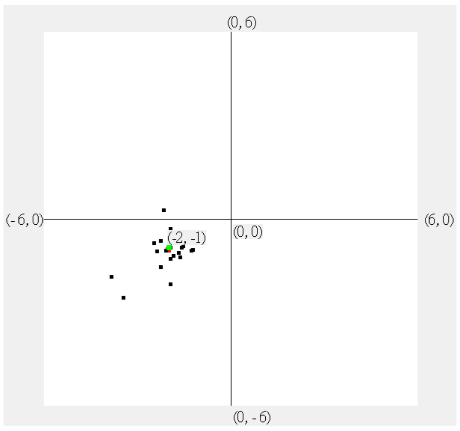

Fig. 11(a). Iteration 10.

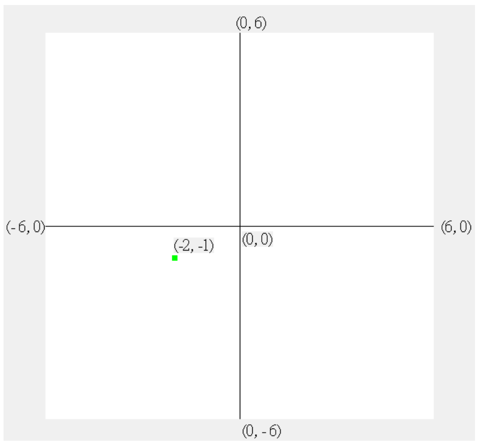

Fig. 11(b). Iteration 20.

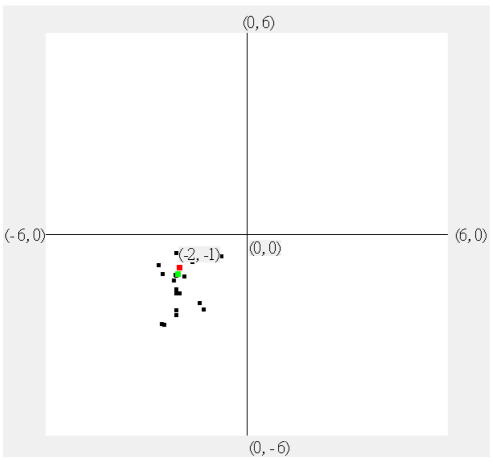

Fig. 12(a). Iteration 10.

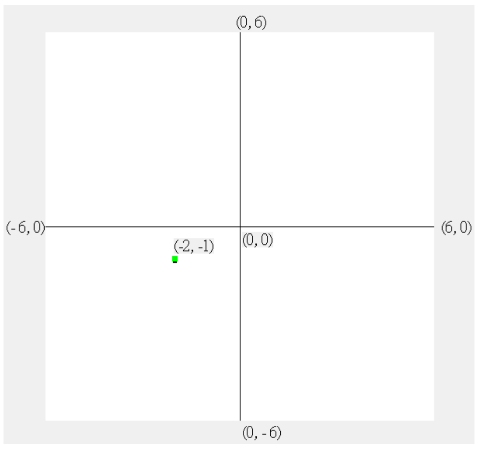

Fig. 12(b). Iteration 20.

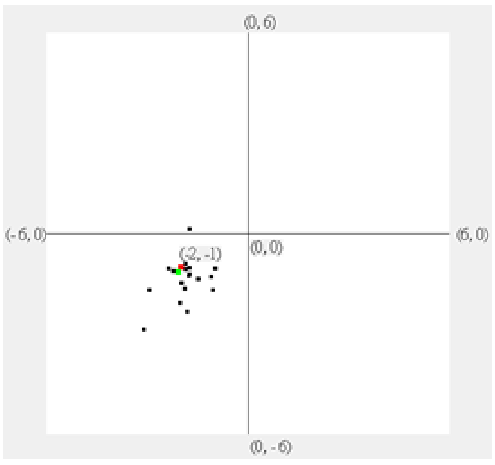

Fig. 13(a). Iteration 10.

- **Fixed** parameters $w = 0.5$ **and** $c_1 = 0.5$ **, and adjustable parameter** $c_2$

The final examination for parameter settings is the fixed parameters $w = 0.5$ and $c_1 = 0.5$, and adjustable parameter $c_2$. Four different sets of parameters are here considered:

Case 11: $w = 0.5$, $c_1 = 0.5$, and $c_2 = 1$;

Case 12: $w = 0.5$, $c_1 = 0.5$, and $c_2 = 0.6$;

Case 13: $w = 0.5$, $c_1 = 0.5$, and $c_2 = 0.2$;

Case 14: $w = 0.5$, $c_1 = 0.5$, and $c_2 = 0$.

Again, after executing the PSO algorithm, simulation results for Cases 11-14 are displayed in Figs. 12-15, respectively. Fig. 12 illustrates the convergence behavior of particles for Case 11 with various iteration numbers, and the correct solution can be accurately solved by executing 25 iterations. As to Cases 12 and 13, they are shown in Figs. 13 and 14, and the correct solution is solved about 20 and 25 iterations, respectively. All of simulation results for Cases 11-13 are almost the same. However, Fig. 15 is the simulation illustrations for Case 14 with $c_2 = 0$. It can be easily seen from Fig. 15 that all the particles on *x*-*y* plane are not any moved and they are identical with initial distribution of particles (iteration 0). Referring to the velocity updating formula of Eq. (1), the next velocity output $v_{ij}(k+1)$ will be zero when $c_2 = 0$ at the beginning of the algorithm with $v_{ij}(0) = 0$, and then it causes all the particle positions no any change by Eq. (2). In accordance with these results, it can be concluded that

(g) the global best particle *gbest* that corresponds to the parameter $c_2$ is the most important factor in the PSO algorithm. The algorithm will fail without it.

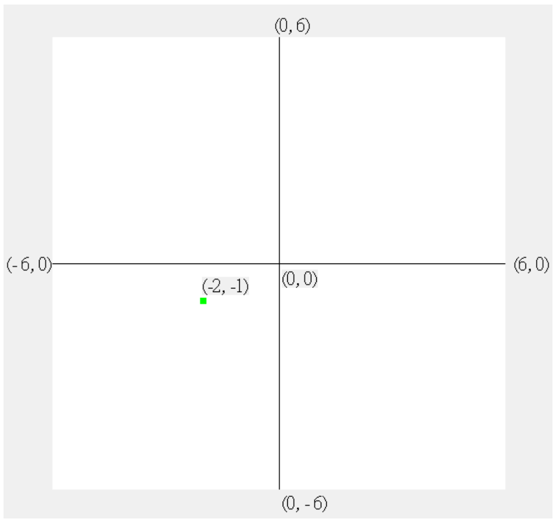

Fig. 13(b). Iteration 20.

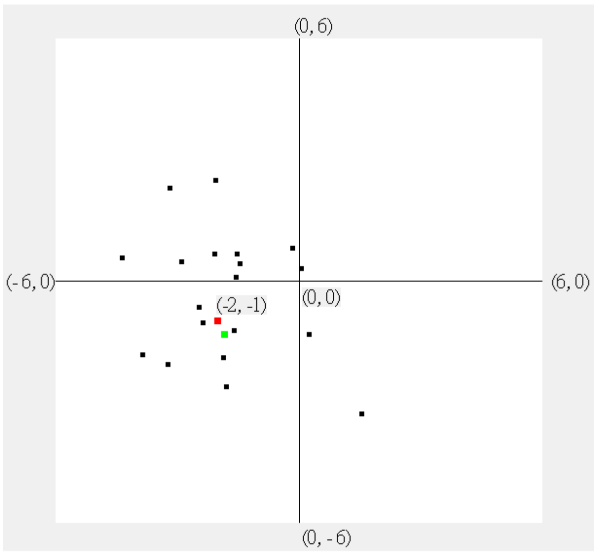

Fig. 14(a). Iteration 10.

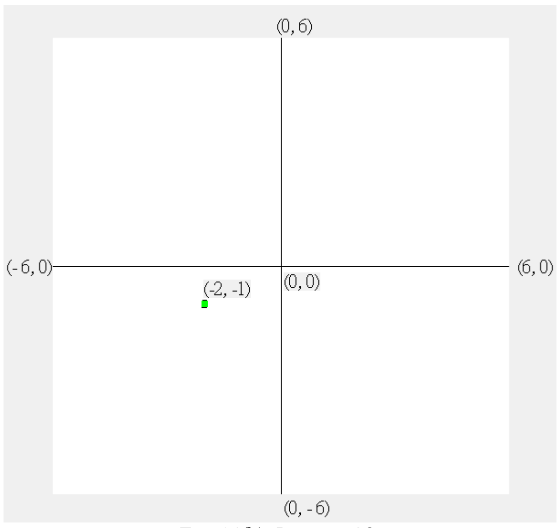

Fig. 14(b). Iteration 20.

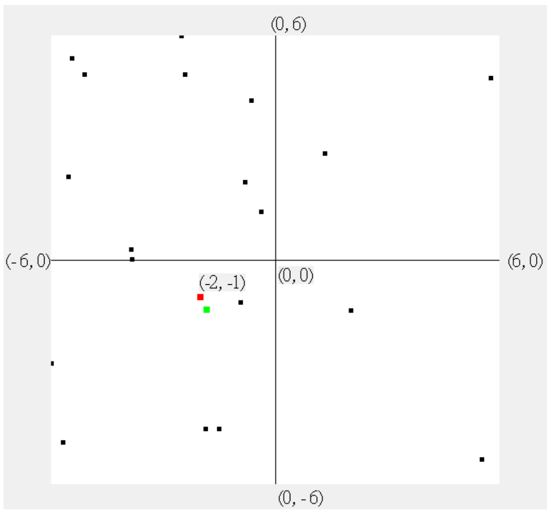

Fig. 15(a). Iteration 10.

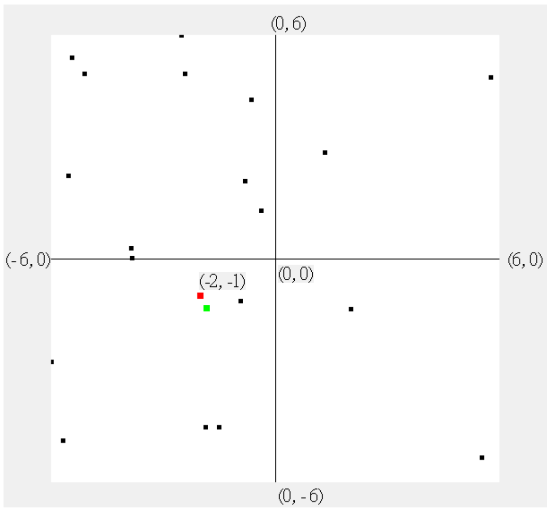

Fig. 15(b). Iteration 20.

## IV. CONCLUSIONS AND FUTURE WORK

In the traditional fashion, simulation results in numerical presentations are always provided to show the searching performance of the PSO algorithm. However, in this paper we have utilized simulation illustrations instead of numerical results to more clearly explain the particle convergence behaviors. The influences of three parameter settings of the PSO on the searching outcomes are also successfully examined by various sets of numerical values. In addition, according to the stability theorem of digital systems the convergence analyses of inertia weight *w* for the PSO algorithm are addressed and also confirmed by some illustration results. In the future work, a simplified version of the PSO algorithm may be presented based on the conclusions that are obtained in this paper because some parameters seem not to be important for solving the system solution.

## REFERENCES

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. the IEEE International Conference on Neural Networks*, vol. IV, Perth, Australia, 1995, pp. 1942-1948.

[2] M. Tao, S. Huang, Y. Li, M. Yan, and Y. Zhou, "SA-PSO based optimizing reader deployment in large-scale RFID systems," *Journal of Network and Computer Applications*, vol. 52, pp. 90-100, 2015.

[3] B. B. Zad, H. Hasanvand, J. Lobry, and F. Vallee, "Optimal reactive power control of DGs for voltage regulation of MV distribution systems using sensitivity analysis method and PSO algorithm,"

*Electrical Power and Energy Systems*, vol. 68, pp. 52-60, 2015.
[4] V. K. Gupta and R. Mahanty, "Optimized switching scheme of cascaded H-bridge multilevel inverter using PSO," *Electrical Power and Energy Systems*, vol. 64, pp. 699-707, 2015.
[5] S. Panda, B. Mohanty, and P. K. Hota, "Hybrid BFOA–PSO algorithm for automatic generation control of linear and nonlinear interconnected power systems," *Applied Soft Computing*, vol. 13, pp. 4718-4730, 2013.
[6] M. Ranjani and P. Murugesan, "Optimal fuzzy controller parameters using PSO for speed control of Quasi-Z Source DC/DC converter fed drive," *Applied Soft Computing*, vol. 27, pp. 332-356, 2015.
[7] A. Lari, A. Khosravi, and F. Rajabi, "Controller design based on analysis and PSO algorithm," *ISA Transactions*, vol. 53, pp. 517-523, 2014.
[8] Y. Z. Hsieh, M. C. Su, and P. C. Wang, "A PSO-based rule extractor for medical diagnosis," *Journal of Biomedical Informatics*, vol. 49, pp. 53-60, 2014.
[9] C. Y. Tsai and C. J. Chen, "A PSO-AB classifier for solving sequence classification problems," *Applied Soft Computing*, vol. 27, pp. 11-27, 2015.
[10] M. Karimi-Nasab, M. Modarres, and S. M. Seyedhoseini, "A self-adaptive PSO for joint lot sizing and job shop scheduling with compressible process times," *Applied Soft Computing*, vol. 27, pp. 137-147, 2015.
[11] Q. Liu, H. Niu, W. Xu, and D. Zhang, "A service-oriented spectrum allocation algorithm using enhanced PSO for cognitive wireless networks," *Computer Networks*, vol. 74, pp. 81-91, 2014.
[12] V. Mangat and R. Vig, "Novel associative classifier based on dynamic adaptive PSO: application to determining candidates for thoracic surgery," *Expert Systems with Applications*, vol. 41, pp. 8234-8244, 2014.
[13] M. Najjari and R. Guilbault, "Formula derived from particle swarm optimization (PSO) for optimum design of cylindrical roller profile under EHL regime," *Mechanism and Machine Theory*, Vol. 90, pp. 162-174, 2015.
[14] H. Garg and M. Rani, "An approach for reliability analysis of industrial systems using PSO and IFS technique," *ISA Transactions*, vol. 52, pp. 701-710, 2013.
[15] T. M. S. Filho, B. A. Pimentel, R. M. C. R. Souza, and A. L. I. Oliveira, "Hybrid methods for fuzzy clustering based on fuzzy c-means and improved particle swarm optimization," *Expert Systems with Applications*, Vol. 42, pp. 6315-6328, 2015.
[16] D. E. G. Trigueros, A. N. Modenes, M. A. S. S. Ravagnani, and F. R. Espinoza-Quinones, "Reuse water network synthesis by modified PSO approach," *Chemical Engineering Journal*, vol. 183, pp. 198-211, 2012.
[17] B. O. Arani, P. Mirzabeygi, and M. S. Panahi, "An improved PSO algorithm with a territorial diversity-preserving scheme and enhanced exploration-exploitation balance," *Swarm and Evolutionary Computation*, vol. 11, pp. 1-15, 2013.
[18] S. Wang and J. Watada, "A hybrid modified PSO approach to VaR-based facility location problems with variable capacity in fuzzy random uncertainty," *Information Sciences*, vol. 192, pp. 3-18, 2012.
[19] F. Guan, L. Dai, and M. Xia, "Pretension optimization and verification test of double-ring deployable cable net antenna based on improved PSO," *Aerospace Science and Technology*, vol. 32, pp. 19-25, 2014.
[20] C. C. Liao, X. L. Zhao, and J. Z. Xu, "Blade layers optimization of wind turbines using FAST and improved PSO algorithm," *Renewable Energy*, vol. 42, pp. 227-233, 2012.
[21] J. Tang, L. He, and S. Fu, "An improved PSO-based charging strategy of electric vehicles in electrical distribution grid," *Applied Energy*, vol. 128, pp. 82-92, 2014.
[22] G. Chen, L. Liu, P. Song, and Y. Du, "Chaotic improved PSO-based multi-objective optimization for minimization of power losses and L index in power systems," *Energy Conversion and Management*, vol. 86, pp. 548-560, 2014.

**W. D. Chang** received his Ph.D. degree in electrical engineering from National Sun Yat-Sen University, Kaohsiung, Taiwan in 2002. He is now a professor at the Department of Computer and Communication, Shu-Te University, Kaohsiung, Taiwan. His research interests include the intelligent signal processing, evolutionary computations, chaotic secure communication, and control engineering.