Journal of Computer Science 9 (11): 1461-1471, 2013
ISSN: 1549-3636

doi:10.3844/jcssp.2013.1461.1471 Published Online 9 (11) 2013 (http://www.thescipub.com/jcs.toc)

EFFECTIVENESS OF SECOND BEST PARTICLE INFORMATION FOR PARTICLE SWARM OPTIMIZATION

[2]Eisuke Kita and [1]Young-Bin Shin

[1]Department of Complex System Science, Graduate School of Information Science, Nagoya University, Nagoya Japan
[2]Department of Computational Science, Graduate School of System Informatics, Kobe University, Kobe Japan

Received 2013-08-25, Revised 2013-09-02; Accepted 2013-09-23

ABSTRACT

Particle Swarm Optimization (PSO) represents the potential solutions of the optimization problem as the particles and then, the particles move in order to find the better solution. The particle positions are updated from the personal best and the global best particle positions which have been ever found. This research focuses on the use of the second personal best and the second global best particle positions in order to improve the search performance of the original PSO algorithm. In the present algorithm, the second global best or the second personal best particle position is randomly used for updating all particle positions. The algorithms are compared with the original PSO algorithm in five test functions. The results reveal that the use of the second global best and the second personal best particle positions can improve the search performance of the original PSO although the basic idea is simple.

Keywords: Particle Swarm Optimization, Global Best Particle, Personal Best Particle, Second Best Particle

1. INTRODUCTION

The gradient-type algorithms such as the Newton and Steepest gradient methods are very popular algorithms for obtaining the solution of the optimization problem. They sometimes reach not global optimum but local optimum. Therefore, some researchers have been studying other algorithms without the gradient data of the function such as Genetic Algorithm (GA) by Holland (1975) and Goldberg (1989), Simulated Annealing (SA) by Kirkpatrick *et al*. (1983), Particle Swarm Optimization (PSO) by Kennedy and Eberhart (1995); Kennedy (1997) and Shi and Eberhart (1998) and so on. PSO, which has been presented in 1995 by Kennedy and Eberhart (1995), is based on a metaphor of social interaction such as bird flocking and fish schooling. PSO, which is also a population-based optimization algorithm, is available for solving various function optimizations problem and industrial applications (He *et al*., 2009; Lapizco-Encinas *et al*., 2009; Liu *et al*., 2006; Poli, 2008; Qarouni-Fard *et al*., 2007; Zhao *et al*., 2008). Qarouni-Fard *et al*. (2007) presented the timetable design by usign Particle Swarm Optimization. Poli (2008) applied the Particle Swarm Optimization for analysis of publications. He *et al.* (2009); Lapizco-Encinas *et al.* (2009); Liu *et al.* (2006); Qarouni-Fard *et al.* (2007) and Zhao *et al.* (2008) presented the application of the Particel Swarm Optimization for packing problems.

In the original PSO algorithm, the potential solutions of the optimization problem are defined as the particles whose position vector denotes the design vector of the candidate solution. The particle positions are updated from the personal and the global best particle positions. The personal and global best particles denote the best position which each particle has ever found and the best position which all particles have ever found, respectively. One of the basic drawbacks of PSO is the premature convergence problem. The premature convergence means too early convergence of a population of potential solutions, resulting in being not global optimal solution but local (sub-) optimal solution.

This study focuses on the use of second global and second personal best particle positions for improving the search performance of the original PSO algorithm. The

Corresponding Author: Young-Bin Shin, Department of Computational Science, Graduate School of System Informatics, Kobe University, Kobe Japan

PSO algorithms employing second global and personal best particle positions are named as present algorithm 1 and 2, respectively. The present algorithms are compared with the original PSO algorithm in five test functions.

The remaining part of this study is organized as follows. The PSO algorithms and the numerical results are explained in section 2 and 3, respectively. Finally, the conclusions are summarized again in section 4.

2. PSO ALGORITHM

2.1. Optimization Problem

The optimization problem is defined by the objective function and the design variables if the constraint conditions are negligible.

The design variable vector is defined as follows Equation (1):

$$x = \left\{x_1, x_2, ..., x_{N_d}\right\}^T \quad (1)$$

The parameter x_i and D_d denote the design variable and the total number of design variables, respectively.

The objective function to be minimized is defined as the function of the design variables Equation (2):

$$F(x) \to \min \quad (2)$$

In the evolutionary computation, the satisfaction of the particle for the design objective is estimated by the fitness function f(x), which is maximized as follows Equation (3):

$$f(x) \to \max \quad (3)$$

2.2. Original PSO

2.2.1. Search Process

In the PSO algorithm, the particles represent potential solutions of the optimization problem and then, the swarm of the particles moves on the solution space in order to find the better solution. A particle in the swarm has a position vector $x_i(t)$ and a velocity vector $v_i(t)$ in the search space at time. Each particle has memory and hence, can remember the best position which it ever visited in search space. When each particle takes the best fitness function, the position vector is known as the personal best particle position vector and $x_i^p(t)$ the overall best out of all particles in the swarm is as global best particle position vector $x^g(t)$. The particle position vector $x_i(t)$ and the velocity vector $v_i(t)$ are updated by the personal and global best particle position vectors.

The original PSO algorithm is summarized as follows (**Fig. 1**):

- Initialize iteration number: The iteration t number is initialized as t←0
- Initialize particle position and velocity vectors: For i = 1, …, N, the particle position vector $x_i(t)$ and velocity vector $v_i(t)$ are initialized with uniformly distributed random vectors
- Initialize best particle position vectors: The global best particle position vector $x^g(t)$ and the personal best particle position vector $x_i^p(t)$ of the particle are initialized with zero vectors; $x^g(t) = 0$ and $x_i^p(t) = 0$
- Evaluate fitness function: For i = 1,…,N, fitness function $f(x_i(t))$ is evaluated
- Check the convergence criterion: If the criterion is satisfied, the process goes to next step. Otherwise, the process goes to the step 7
- Output results: The results are output and the process is terminated
- Update particle position vectors: The particle velocity vector $v_i(t)$ is updated and then, the particle position vector $x_i(t)$ is updated. (Update algorithm is described in the next section)
- Update iteration number: The iteration number is updated so that and then, the process goes to step 3

2.2.2. Update Algorithm

In the original PSO algorithm, the position and the velocity vectors of the particle i(i = 1,…,N) are updated according to the following rules Equation (4 and 5):

$$v_i(t+1) = \omega v_i(t) + c_1 r_i \times \left(x_i^p(t) - x_i(t)\right) + c_2 r_2 \times \left(x^g(t) - x_i(t)\right) \quad (4)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (5)$$

The parameter w is the inertia weight. The parameter c_1 and c_2 are acceleration coefficient and is the iteration time-step. The variable r_1 and r_2 are random numbers in the range of [0,1]. The parameter N is the swarm size or the total number of particles in the swarm.

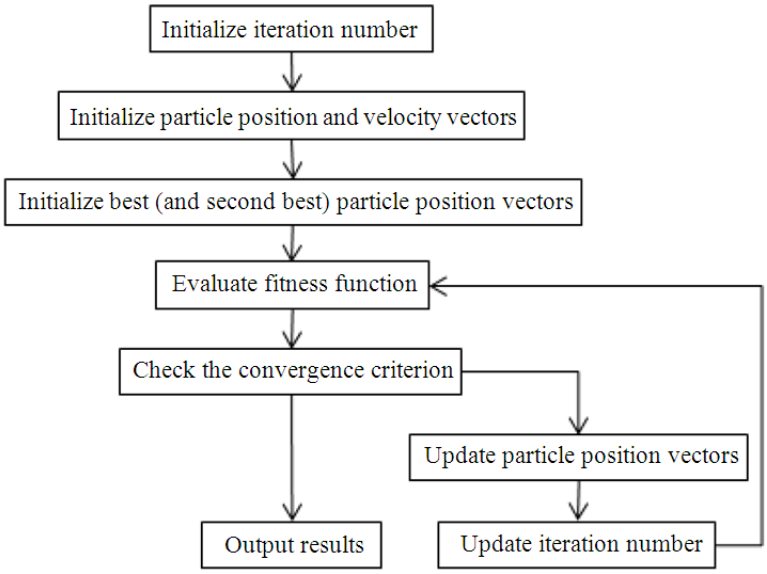


Fig. 1. PSO algorithm

The inertia weight governs how much percentage of the velocity should be retained from the previous time step to the next time step. The inertia weight is updated by the following self-adapting formula Equation (6):

$$w = w_{max} - \left(w_{max} - w_{min}\right) \times \frac{t}{t_{max}} \quad (6)$$

The parameter w_{max} and w_{min} denote the maximum and minimum inertia weights, respectively. The parameter t and t_{max} are the iteration step and the maximum iteration steps in the simulation, respectively.

The parameters c_1 and c_2 determine the relative pull of $x_i^p(t)$ and $x^g(t)$. According to the recent work done by Clerc (1999), the parameters are given as follows:

$$c_1 = c_2 = 1.5 \quad (7)$$

The update algorithm of the particle position is summarized as follows:

- Update the particle velocity vector: The particle velocity vector $v_i(t+1)$ is calculated by Equation (4)
- Update the particle position vector: The particle position vector $x_i(t+1)$ is calculated by Equation (5)
- Update global best particle position vector: The set is defined as follows:

$$S = \{S_i\} = \{x^g(t), x_1^p(t), ..., x_N^p(t)\}$$

The global best particle position vector is updated as follows:

$$x^g(t+1) \leftarrow \arg\max_S f(S_i)$$

- Update personal best particle position vector: For i = 1,…,N, the set is defined as follows:

$$S^p = \{S_i^p\} = \{x_i^p(t), x_i(t)\}$$

The personal best particle position vector is updated as follows:

$$x_i^p(t+1) \leftarrow \arg\max_{S^p} f(S_i^p)$$

2.3. Present Algorithm 1

2.3.1. Search Process

The search process of the present algorithm 1 is similar except for the uses of the second global best particle position vector $x^g(t)$. This algorithm uses the personal best particle position vector $x_i^p(t)$, the first global best particle position vector $x^{g2}(t)$ and the second

global best particle position vector $x^{g2}(t)$ for updating the particle position and velocity vectors.

The present algorithm 1 is summarized as follows:

- Initialize iteration number: The iteration number t is initialized as t←0
- Initialize particle position and velocity vectors: For i = 1,…,N, the particle position vector $x_i(t)$ and velocity vector $v_i(t)$ are initialized with uniformly distributed random vectors
- Initialize best particle position vectors: The global best particle position vector $x^g(t)$ and the personal best particle position vector $x_i^p(t)$ of the particle are initialized with zero vectors; $x^g(t) = 0$ and $x_i^p(t) = 0$
- Initialize second global best particle position vectors: The global best particle position vector $x^{g2}(t)$ is initialized with zero vectors; $x^{g2}(t) = 0$
- Evaluate fitness function: For i = 1,…,N, fitness function $f(x_i(t))$ is evaluated
- Check the convergence criterion: If the criterion is satisfied, the process goes to next step. Otherwise, the process goes to the step 8
- Output results: The results are output and the process is terminated
- Update particle position vectors: The particle velocity vector $v_i(t)$ is updated and then, the particle position vector $x_i(t)$ is updated. (Update algorithm is described in the next section.)
- Update iteration number: The iteration number is updated so that t←t+1 and then, the process goes to step 3

2.3.2. Update Algorithm with Second Global Best Particle

The original PSO have no handling mechanism for avoiding the local optimization except for the use of $x_i^p(t)$. In the present algorithm 1, each particle can remember the second global best particle position vector $x^{g2}(t)$ in addition to the global best particle position vector $x^g(t)$ and the personal best particle position vector $x_i^p(t)$. The use of $x^{g2}(t)$ can reduce the chance of local optimum convergence of PSO. In this algorithm, the particle velocity vector is updated by the following equation:

$$\begin{aligned} v_i(t+1) &= \omega v_i(t) + c_1 r_1 \times \left(x_i^p(t) - x_i(t)\right) \\ &+ c_2 r_2 \times \left(x^g(t) - x_i(t)\right) \\ &+ c_3 r_3 \times \left(x^{g2}(t) - x_i(t)\right) \end{aligned} \quad (8)$$

The parameter is the inertia weight. The parameter c_1, c_2 and c_3 are the acceleration coefficient and the parameter is the iteration time. Besides, r_1,r_2 and r_3 are random numbers uniformly distributed in the range of The parameter c_1 and c_2 are taken as the same values in the original PSO; $c_1 = c_2 = 1.5$. Effect of the parameter c_3 to the search performance is discussed in the numerical examples.

The update rule (8) has been already presented in the paper. The numerical discussions and the applications were not described in the reference. Therefore, in this study, it is discussed in numerical examples.

The update algorithm of the particle position vector is summarized as follows:

- Generate uniformly distributed random number: The uniformly distributed random number r is generated in the range of [0,1]
- Update particle velocity vector: If r≥0.5, the particle velocity vector $v_i(t+1)$ is calculated by Equation (4). Otherwise, the vector $v_i(t+1)$ is calculated by Equation (8)
- Update the particle position vector: The particle position vector $x_i(t+1)$ is calculated by Equation (5)
- Update global best particle position vector: The set is defined as follows:

$$S^g = \left\{S_i^g\right\} = \left\{x^g(t), x^{g2}(t), x_1^p(t), ..., x_N^p(t)\right\}$$

The global best particle position vector is updated as follows.

$$x^g(t+1) \leftarrow \arg\max_{S^g} f\left(S_i^g\right)$$

- Update second global best particle position vector: The set S^{g2} is defined the set S^g from which $x^g(t+1)$ is excluded as follows:

$$S^{g2} = S^g - x^g(t+1)$$

The second global best particle position vector $x^{g2}(t+1)$ is updated as follows:

$$x^{g2}(t+1) \leftarrow \arg\max_{S^{g2}} \left(S_i^{g2}\right)$$

- Update personal best particle position vector: For i = 1,…N, the set is defined as follows:

$$S^p = \left\{S_i^p\right\} = \left\{x_i^p(t), x_i(t)\right\}$$

The personal best particle position vector is updated as follows:

$$x_i^p(t+1) \leftarrow \arg\max_{S^p} f\left(S_i^p\right)$$

2.4. Present Algorithm 2

2.4.1. Search Process

The search process of the present algorithm 2 is almost same as that of the original PSO algorithm except for the use of the second personal best particle position vector $x_i^{p2}(t)$. This algorithm uses the personal best particle position vector $x_i^p(t)$, the global best particle position vector $x^g(t)$ and the second personal best particle position vector $x_i^{p2}(t)$ for updating the particle position and velocity vectors.

The present algorithm 2 is summarized as follows:

- Initialize iteration number: The iteration number t is initialized as t←0
- Initialize particle position and velocity vectors: For i = 1,…N, the particle position vector $x_i(t)$ and velocity vector $v_i(t)$ are initialized with uniformly distributed random vectors
- Initialize best particle position vectors: The global best particle position vector $x^g(t)$ and the personal best particle position vector $x_i^p(t)$ of the particle are initialized with zero vectors; $x^g(t) = 0$ and $x_i^p(t) = 0$
- Initialize second personal best particle position vectors: For i = 1,…,N, the second personal best particle position vector $x_i^p(t)$ is initialized with zero vectors; $x_i^{p2}(t) = 0$
- Evaluate fitness function: For i = 1,…,N, fitness function $f(x_i(t))$ is evaluated
- Check the convergence criterion: If the criterion is satisfied, the process goes to next step. Otherwise, the process goes to the step 8
- Output results: The results are output and the process is terminated
- Update particle position vectors: The particle velocity vector $v_i(t)$ is updated and then, the particle position vector $x_i(t)$ is updated. (Update algorithm is described in the next section)
- Update iteration number: The iteration number is updated so that t←t+1 and then, the process goes to step 3

2.4.2. Update Algorithm with Second Personal Best Particle

The present algorithm 1 uses the second global best particle position vector $x^{g2}(t)$ for avoiding the local optimization. On the other hand, the present algorithm 2 uses the second personal best particle position vector $x_i^{p2}(t)$ instead of the second global best particle position vector $x^{g2}(t)$.

In the present algorithm 1, the second global best particle position vector $x^{g2}(t)$ makes an identical effect on all particles. The second personal best particle position vector $x_i^{p2}(t)$ makes the different effect on each particle. Therefore, particles in the present algorithm 2 tend to search wider region than them in the present algorithm 1.

In the present algorithm 2, each particle can remember the positions of the global best particle position vector $x^g(g)$, the personal best particle position vector $x_i^p(t)$ and the second personal best particle position vector $x_i^{p2}(t)$. In this algorithm, the particle velocity vector is updated by the following equation:

$$\begin{aligned} v_i(t+1) &= \omega v_i(t) + c_1 r_1 \times \left(x_i^p(t) - x_i(t)\right) \\ &+ c_2 r_2 \times \left(x^g(t) - x_i(t)\right) \\ &+ c_4 r_4 \times \left(x^{p2}(t) - x_i(t)\right) \end{aligned} \quad (9)$$

The parameter is the inertia weight. The parameter c_1,c_2 and c_4 are the acceleration coefficient and the parameter t is the iteration time. Besides, r_1, r_2 and r_4 are random numbers uniformly distributed in the range of [0,1]. The parameter c_1 and c_2 are taken as the same values in the original PSO; $c_1 = c_2 = 1.5$. Effect of the parameter c_4 to the search performance is also discussed in the numerical examples.

The present algorithm 2 shares the information of $x_i^p(t)$, $x^g(t)$ and $x_i^{p2}(t)$. Obviously, $x_i^{p2}(t)$ is worse than $x_i^p(t)$. If only Equation (7) is used for updating particle velocity vector, the result must be worse than that of original PSO. Therefore, the update rules (4) and (8) are employed alternately. The update algorithm of the present algorithm 2 is summarized as follows:

- Generate uniformly distributed random number: The uniformly distributed random number r is generated in the range of [0,1]
- Update particle velocity vector: If r≥0.5, the particle velocity vector $v_i(t+1)$ is calculated by Equation (4). Otherwise, the vector $v_i(t+1)$ is calculated by Equation (9)

- Update particle position vector: The particle position vector $x_i(t+1)$ is calculated by Equation (5)
- Update global best particle position vector: The set is defined as follows:

$$S = \{S_i\} = \{x^g(t), x_1^p(t), ..., x_N^p(t)\}$$

The global best particle position vector is updated as follows:

$$x^g(t+1) \leftarrow \arg\max_S f(S_i)$$

- Update personal best particle position vector: For i =1,…,N, the set is defined as follows:

$$S^p = \ll \{S_i^p\} = \{x_i^p(t), x_i^{p2}(t), x_i(t)\}$$

The personal best particle position vector is updated as follows:

$$x_i^p(t+1) \leftarrow \arg\max_{S^p} f(S_i^p)$$

- Update second personal best particle position vector: For i = 1,…,N, the set S^{p2} is defined from the set S^p from which $x_i^p(t+1)$ is excluded as follows:

$$S^{p2} = \{S_i^{p2}\} = S^p - x_i^p(t+1)$$

The second personal best particle position vector is updated as follows:

$$x_i^{p2}(t+1) \leftarrow \arg\max_{S^{p2}} f(S_i^{p2})$$

3. NUMERICAL EXAMPLES

3.1. Test Functions

Sphere, Rosenbrock, Rastrigin, Griewank and Schaffer's f6 functions are considered as test functions.

3.1.1. Sphere function

Sphere function is defined as follows Equation (10):

$$f_1(x) = \sum_{i=1}^{n} x_i^2 \quad (-100 \le x_i \le 100) \tag{10}$$

The vector x is defined as follows Equation (11):

$$x = \{x_1, x_2, ..., x_n\}^T \tag{11}$$

The sphere function of n = 2is shown in **Fig. 2a**.

3.1.2. Rosenbrock function

Rosenbrock function is defined as follows Equation (12):

$$f_2(x) = \sum_{i=1}^{n-1} \left\{100\left(x_{i+1} - x_i^2\right)^2 (x_i - 1)^2\right\} \quad (-30 \le x_i \le 30) \tag{12}$$

The Rosenbrock function of n = 2 is shown in **Fig. 2b**.

3.1.3. Rastrigin Function

Rastrigin function is a multi-modal function defined as follows Equation (13):

$$f_3(x) = \sum_{i=1}^{n} \left(x_i^2 - 10\cos 2\pi x_i + 10\right) \quad (-5.12 \le x_i \le 5.12) \tag{13}$$

The Rastrigin function of n = 2is shown in **Fig. 2c**. A lot of local optimal solutions exist around a global optimal solution.

3.1.4. Griewank Function

Griewank function is defined as follows Equation (14):

$$f_4(x) = \frac{1}{4000}\sum_{i=1}^{b} x_i^2 - \prod_{i=1}^{n} \cos\frac{x_i}{\sqrt{i}} + 1 \quad (-600 \le x_i \le 6000) \tag{14}$$

The Griewank function of n = 2 is shown in **Fig. 2d**.

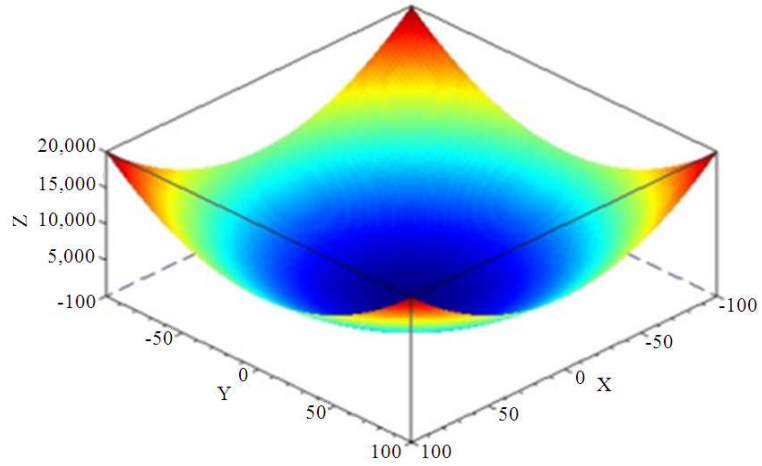
3.1.5. Schaffer's F6 Function

Schaffer's f6 function is defined as follows Equation (15):

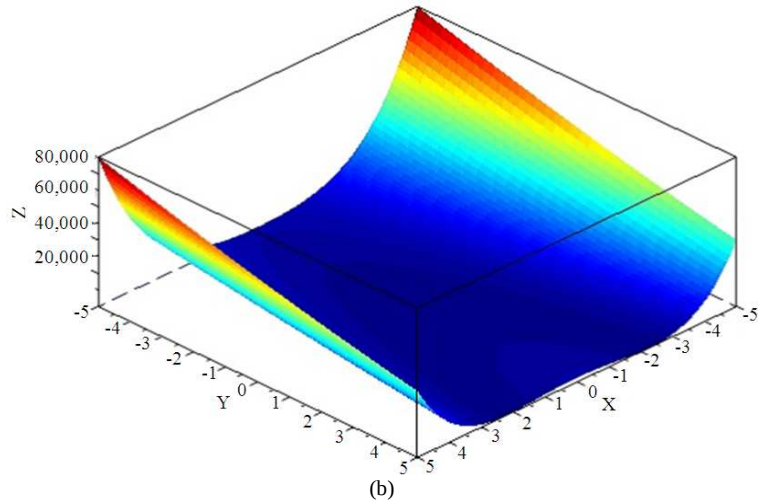
$$f_5(x) = 0.5 + \frac{\sin^2 |x| - 0.5}{\left(1 + 0.001|x|^2\right)^2} \quad (-100 \le x_i \le 100) \tag{15}$$

The function |x| denotes the absolute value of the vector x. The Schaffer's f6 function of is n = 2 shown in **Fig. 2e**.

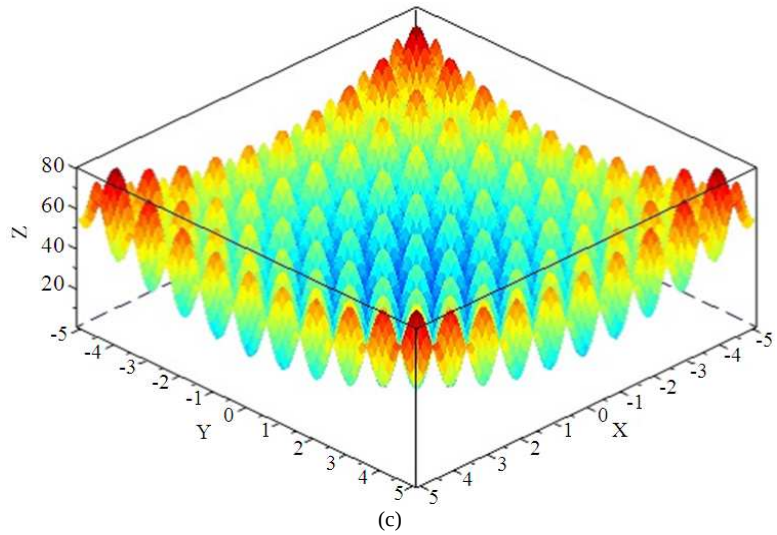
The dimension of functions is n = 2 for Schaffer's f6 function or n = 30 for the other functions. The threshold for function optimization is also shown in the same table. In minimization of the function, it is concluded that the global minimum of the function can be found when the function value is smaller than the threshold value.


20,000
15,000
10,000
5,000
Z
-100
-50
0
50
100
Y
-100
-50
0
50
100
X

(a)


80,000
60,000
40,000
20,000
Z
-5
-4
-3
-2
-1
0
1
2
3
4
5
Y
-5
-4
-3
-2
-1
0
1
2
3
4
5
X

(b)


80
60
40
20
Z
-5
-4
-3
-2
-1
0
1
2
3
4
5
Y
-5
-4
-3
-2
-1
0
1
2
3
4
5
X

(c)

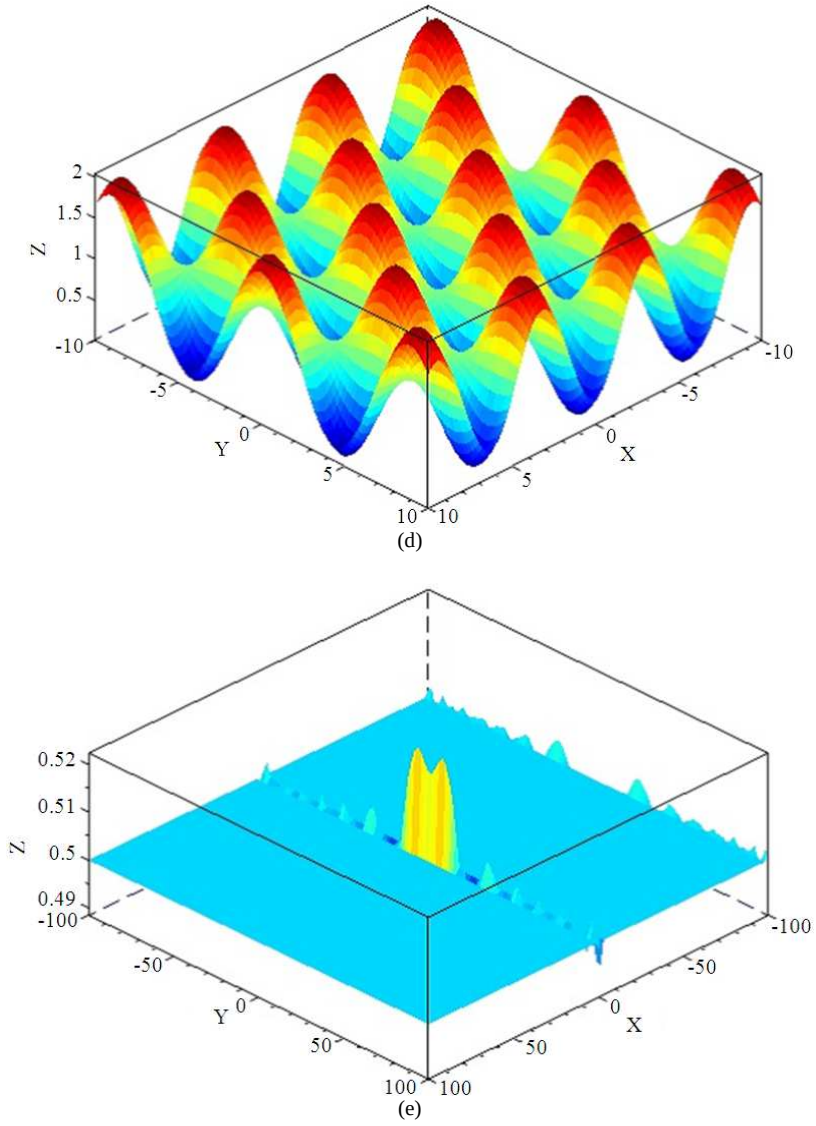


Fig. 2. Test Functions (a) Sphere function (b) Rosenbrock function (c) Rastrigin function (d) Griewank function (e) Schaffer's f6 function

Swarm size and maximum iteration number are shown in **Table 1**. According to the work done by Clerc (1999), the parameters c_1 and c_2 are specified as $c_1 = c_2 = 1.5$.

The results are compared in the estimation value, which is defined as the quotient of the average search time and the success rate as follows Equation (16):

$$\text{Estimation} = \frac{\text{Average search time}}{\text{Success rate}} \quad (16)$$

The average search time denotes the average iteration number at which the global optimum could be found. The success rate denotes, in total number of simulations, the percentage of the number of simulations at which the minimum solution can be found. The threshold for finding the optimal solution is shown in **Table 2**. When a smaller solution than the threshold can be found, it is concluded that the simulation is terminated successfully.

Table 1. Simulation parameters

Swarm size	033
Maximum iteration	10000

Table 2. Threshold for test functions

Function	Threshold
Sphere	0.01
Rosenbrock	100.00
Rastrigin	100.00
Griewank	0.10
Schaffer f6	10^{-5}

3.2. Effect of c_3 on Present Algorithm 1

Simulations are performed 20 times from different initial populations by the present algorithm 1. The results are shown in **Table 3**. The results show that the best value of the parameter c_3 depends on the function to be solved. Comparison of the estimation values shows that the best values of the parameter c_3 are $c_3 = 5$ for Sphere, Rosenbrock, Griewank and Schaffer's f6 functions and $c_3 = 2.5$ or 5 for Rastrigin function. It is concluded that $c_3 = 5$ is good for all functions.

3.3. Effect of c_4 on Present Algorithm 2

Simulations are performed 20 times from different initial populations by the present algorithm 2. The results are shown in **Table 4**. The results show that the best parameter c_3 depends on the function. The best values of the parameter are for Sphere, Rosenbrock, Rastrigin and Griewank functions c_4 and $c_4 = 5$ for Schaffer's f6 functions. It is concluded that is $c_4 = 5.5$ good for all functions.

3.4. Comparison with Other Studies

Swarm size and maximum iteration number are 30 and 10000, respectively. According to the work done by Clerc (1999), the parameters c_1 and c_2 are specified as $c_1 = c_2 = 1.5$. The best results by present algorithms are compared with the results in the study by Eberhart and Shi (2000) and Trelea (2003). The results are shown in **Table 5**. The results by the present algorithms are better than them in the references. Comparison of the present algorithm 1 and 2 shows that the present algorithm 1 is better for Rosenbrock and Griewank functions and the present algorithm 2 is for other functions.

Table 3. Results by present algorithm 1

c_3	1	1.5	2	2.5	3
Sphere function					
Estimation	345.00	614.0	538.0	659.0	429.0
c_3	3.50	4.0	4.5	5.0	5.5
Estimation	299.00	351.0	308.0	292.0	416.0
Rosenbrock function					
c_3	1.00	1.5	2.0	2.5	3.0
Estimation	1431.00	936.0	543.0	477.0	444.0
c_3	3.50	4.0	4.5	5.0	5.5
Estimation	287.00	282.0	352.0	240.0	244.0
Rastrigin function					
c_3	1.00	1.5	2.0	2.5	3.0
Estimation	1.65	266.0	419.0	1.9	238.0
c_3	3.50	4.0	4.5	5.0	5.5
Estimation	155.00	119.0	202.0	112.0	780.0
Griewank function					
c_3	1.00	1.5	2.0	2.5	3.0
Estimation	728.00	313.0	561.0	486.0	373.0
c_3	3.50	4.0	4.5	5.0	5.5
Estimation	355.00	270.0	328.0	223.0	209.0
Schaffer function					
c_3	1.00	1.5	2.0	2.5	3.0
Estimation	11.80	8.9	12.1	11.4	8.6
c_3	3.50	4.0	4.5	5.0	5.5
Estimation	9.00	10.6	11.0	6.8	12.0

Table 4. Results by present algorithm 2

c_4	1	1.5	2	2.5	3
Sphere function					
Estimation	1024.0	1205.0	1178.0	922.0	489.0
c_4	3.5	4.0	4.5	5.0	5.5
Estimation	421.0	685.0	851.0	387.0	291.0
Rosenbrock function					
c_4	1.0	1.5	2.0	2.5	3.0
Estimation	618.0	894.0	1415.0	1035.0	645.0
c_4	3.5	4.0	4.5	5.0	5.5
Estimation	729.0	370.0	586.0	398.0	273.0
Rastrigin function					
c_4	1.0	1.5	2.0	2.5	3.0
Estimation	131.0	148.0	98.0	115.0	81.0
c_4	3.5	4.0	4.5	5.0	5.5
Estimation	62.0	84.0	86.0	68.0	58.0
Griewank function					
c_4	1.0	1.5	2.0	2.5	3.0
Estimation	1235.0	563.0	1136.0	590.0	675.0
c_4	3.5	4.0	4.5	5.0	5.5
Estimation	512.0	315.0	352.0	402.0	309.0
Schaffer f6 function					
c_4	1.0	1.5	2.0	2.5	3.0
Estimation	14.9	9.5	5.6	10.2	5.4
c_4	3.5	4.0	4.5	5.0	5.5
Estimation	3.8	6.0	10.3	2.8	3.1

Table 5. Comparsion with other studies

Algorithm	Eberhart and Shi (2000)	Trelea (2003)	Present 1	Present 2
Sphere function				
Estimation	530	344	292	291
Rosenbrock function				
Algorithm	Eberhart and Shi (2000)	Trelea (2003)	Present 1	Present 2
Estimation	669	614	240	273
Rastrigin function				
Algorithm	Eberhart and Shi (2000)	Trelea (2003)	Present 1	Present 2
Estimation	213	156	112	58
Griewank function				
Algorithm	Eberhart and Shi (2000)	Trelea (2003)	Present 1	Present 2
Estimation	323	348	223	309
Schaffer f6 function				
Algorithm	Eberhart and Shi (2000)	Trelea (2003)	Present 1	Present 2
Estimation	532	215	6.8	2.8

4. CONCLUSION

This study describes the use of the second best particle position for improving the original PSO. In the original PSO, the particle position vectors are updated from the personal best and the global best position vectors which particles have ever found. This research focuses on the use of the second global best and the second personal best particle positions in order to improve the search performance of the original PSO. In the present algorithms, the second global best and the second personal best particle positions are randomly used for updating the particle position vectors.

Present algorithms are compared with the original PSO algorithm in five test functions. The results revealed that the use of the second best positions can improve the search performance of the original PSO. In all cases, the success rate is bigger than or equal to 0.9 and the estimation, which is defined as the quotient of the average search time and the success rate, is also better than the previous studies. The present results were compared with the previous study. The results by the present algorithms were better than them in the references.

In the near future, we would like to discuss the applicability of the present algorithms to actual engineering applications.

5. REFERENCES

Clerc, M., 1999. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. Proceedings of the Congress on Evolutionary Computation, Jul. 6-9, IEEE Xplore Press, Washington, DC., pp: 1951-1957. DOI: 10.1109/CEC.1999.785513

Eberhart, R.C. and Y. Shi, 2000. Comparing inertia weights and constriction factors in particle swarm optimization. Proceedings of the Congress on Evolutionary Computation, Jul. 16-19, IEEE Xploe Press, La Jolla, CA., pp: 84-88. DOI: 10.1109/CEC.2000.870279

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. 1st Edn., Addison-Wesley, Reading, ISBN-10: 0201157675, pp: 412.

He, C., Y.B. Zhang, J.W. Wu and C. Chang, 2009. Research of three-dimensional container-packing problems based on Discrete Particle Swarm Optimization algorithm. Proceedings of the International Conference on Test and Measurement, Dec. 5-6, IEEE Xplore Press, Hong Kong, pp: 425-428. DOI: 10.1109/ICTM.2009.5413015

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. 1st Edn., University of Michigan Press, Ann Arbor, ISBN-10: 0472084607, pp: 183.

Kennedy, J. and R. Eberhart, 1995. Particle swarm optimization. Proceedings of the IEEE the International Conference on Neural Networks, Nov. 27-Dec. 01, IEEE Xplore Press, Perth, WA., pp: 1942-1948. DOI: 10.1109/ICNN.1995.488968

Kennedy, J., 1997. The particle swarm: Social adaptation of knowledge. Proceedings of the IEEE International Conference on Evolutionary Computation, Apr. 13-16, IEEE Xplore Press, Indianapolis, IN., pp: 303-308. DOI: 10.1109/ICEC.1997.592326

Kirkpatrick, S., C.D. Gelatt Jr. and M.P. Vecchi, 1983. Optimization by simulated annealing. Science, 220: 671-680. DOI: 10.1126/science.220.4598.671

Lapizco-Encinas, G., C. Kingsford and J. Reggia, 2009. A cooperative combinatorial particle swarm OPTIMIZATION algorithm for side-chain packing. Proceedings of the IEEE Swarm Intelligence Symposium, Mar. 30-Apr. 2, IEEE Xplore Press, Nashville, TN., pp: 22-29. DOI: 10.1109/SIS.2009.4937840

Liu, D.S., K.C. Tan, C.K. Goh and W.K. Ho, 2006. On Solving multiobjective bin packing problems using particle swarm optimization. Proceedings of the IEEE Congress on Evolutionary Computation, Jul. 16-21, IEEE Xplore Press, Vancouver, BC., pp: 2095-2102, DOI: 10.1109/CEC.2006.1688565.

Poli, R., 2008. Analysis of the publications on the applications of particle swarm optimization. J. Artif. Evolut. Applic., 3: 1-3. DOI: 10.1155/2008/685175

Qarouni-Fard, D., A. Najafi-Ardabili and M.H. Moeinzadeh, 2007. Finding Feasible Timetables with Particle Swarm Optimization. Proceedings of the 4th International Conference on Innovations in Information Technology, Nov. 18-20, IEEE Xplore Press, Dubai, pp: 387-391. DOI: 10.1109/IIT.2007.4430422

Shi, Y. and R. Eberhart, 1998. A modified particle swarm optimizer. Proceedings of the IEEE International Conference on Evolutionary Computation Proceedings, May 4-9, IEEE Xplore Press, Anchorage, AK., pp: 69-73. DOI: 10.1109/ICEC.1998.699146

Trelea, I.C., 2003. The particle swarm optimization algorithm: Convergence analysis and parameter selection. Inform. Process. Lett., 85: 317-325. DOI: 10.1016/S0020-0190(02)00447-7

Zhao C., L. Lin, C. Hao and X. Liu, 2008. Solving the rectangular packing problem of the discrete particle swarm algorithm. Proceedings of the International Seminar on Business and Information Management, Dec. 19-19, IEEE Xplore Press, Wuhan, pp: 26-29. DOI: 10.1109/ISBIM.2008.114