International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:5, No:11, 2011

Particle Swarm Optimization with Reduction for Global Optimization Problems

Michiharu Maeda and Shinya Tsuda

Abstract—This paper presents an algorithm of particle swarm optimization with reduction for global optimization problems. Particle swarm optimization is an algorithm which refers to the collective motion such as birds or fishes, and a multi-point search algorithm which finds a best solution using multiple particles. Particle swarm optimization is so flexible that it can adapt to a number of optimization problems. When an objective function has a lot of local minimums complicatedly, the particle may fall into a local minimum. For avoiding the local minimum, a number of particles are initially prepared and their positions are updated by particle swarm optimization. Particles sequentially reduce to reach a predetermined number of them grounded in evaluation value and particle swarm optimization continues until the termination condition is met. In order to show the effectiveness of the proposed algorithm, we examine the minimum by using test functions compared to existing algorithms. Furthermore the influence of best value on the initial number of particles for our algorithm is discussed.

Keywords—Particle swarm optimization, Global optimization, Metaheuristics, Reduction.

I. Introduction

FOR optimization problem, it is difficult to obtain an optimal solution and it requires an immense amount of time. Metaheuristics have been a focus of attention for this situation since they are not dependent on a specific problem [1]. Metaheuristics are optimization approaches which make use of the best solution improved iteratively to the next search. Metaheuristics involve, for example, genetic algorithm (GA), differential evolution (DE), and particle swarm optimization (PSO). GA is a search algorithm which carries out the genetic operation of selection, crossover, and mutation [2]. DE adopts mutation as a weighted sum of a base vector and a difference vector. An individual selected from the population becomes the basic vector and the difference between a pair of individuals becomes the difference vector [3]. PSO is a multi-point search algorithm using multiple candidate solutions called particles and performs the solution search by sharing huge amounts of information in each particle [4]. PSO is a popular and utility algorithm without complicated calculations. Although PSO can be applied to a number of optimization problems, the solution may fall into a local minimum and it is difficult to find an optimal solution for a complicated objective function such as a multimodal function with a lot of local minimums [5]–[12]. Various approaches have also been introduced for unit reduction, and many discussions have been made on the multilayer neural networks [13], [14]. For the purpose of vector quantization, self-organizing algorithms are proposed, and significant improvement compared to the conventional techniques has been achieved [15], [16]

M. Maeda and S. Tsuda are with Fukuoka Institute of Technology, Fukuoka, 811-0295 Japan (e-mail: maeda@fit.ac.jp).

In this study, we present an algorithm of particle swarm optimization with reduction for global optimization problems. A number of particles are initially prepared and the positions of these are updated in particle swarm optimization. Particles sequentially reduce to reach a predetermined number of them founded on evaluation value and particle swarm optimization continues until the termination condition is met. In order to show that our algorithm is effective in quality, experimental results are presented in comparison with existing algorithms. Moreover we examine the influence of best value on the initial number of particles for our algorithm.

II. Particle Swarm Optimization

Particle swarm optimization (PSO) is an algorithm referring to the collective motion of flock organisms, such as birds or fishes. PSO is a multi-point search algorithm using multiple candidate solution called particles and performs the solution search by sharing huge amounts of information in each particle. It is a plain and useful algorithm without complicated calculation, and is so versatile that it can adapt to a lot of optimization problems.

The position of the i-th particle in n-dimensional space is defined by $\boldsymbol{x}_i = (x_{i1},\ x_{i2},\ x_{i3},\ \cdots,\ x_{ij},\ \cdots,\ x_{in})^T$. The velocity of the i-th particle is represented by $\boldsymbol{v}_i = (v_{i1},\ v_{i2},\ v_{i3},\ \cdots,\ v_{ij},\ \cdots,\ v_{in})^T$. Each particle stores the best position previously encountered by particle i $\boldsymbol{p}_i = (p_{i1},\ p_{i2},\ p_{i3},\ \cdots,\ p_{ij},\ \cdots,\ p_{in})^T$, and its evaluation value $f(\boldsymbol{p}_i)$, the best position of all particles $\boldsymbol{g} = (g_1,\ g_2,\ g_3,\ \cdots,\ g_j,\ \cdots,\ g_n)^T$, and its evaluation value $f(\boldsymbol{g})$. The current position $\boldsymbol{x}_i^k$ moves to the new position $\boldsymbol{x}_i^{k+1}$ in addition to the velocity $\boldsymbol{v}_i^{k+1}$ (the i-th particle at $k+1$) represented as follows.

$$\boldsymbol{v}_i^{k+1} = w\boldsymbol{v}_i^k + c_1 r(\boldsymbol{p}_i^k - \boldsymbol{x}_i^k) + c_2 r(\boldsymbol{g}^k - \boldsymbol{x}_i^k) \qquad (1)$$

where w is an inertia weight, c_1 and c_2 are cognitive and social parameters, respectively, and r is a random number uniformly distributed in [0, 1].

The new position of particle i is presented as follows.

$$\boldsymbol{x}_i^{k+1} = \boldsymbol{x}_i^k + \boldsymbol{v}_i^{k+1} \qquad (2)$$

Figure 1 shows the movement of PSO. $\boldsymbol{x}_i^k$ is the position of particle, $\boldsymbol{v}_i^k$ is the velocity of particle, $\boldsymbol{g}^k$ is the best position of all particles, $\boldsymbol{p}_i^k$ is the best position previously encountered by particle i, and k is an iteration. These resultant vectors decide the next velocity of each particle and particles move to the new positions according to Eq. (2). Repeating the similar

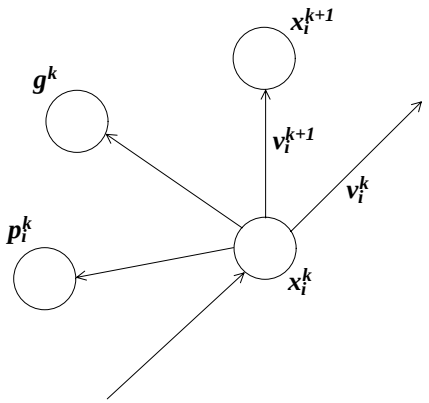


Fig. 1. Particle movement in PSO. $\boldsymbol{x}_i^k$ is the position of particle, $\boldsymbol{v}_i^k$ is the velocity of particle, $\boldsymbol{g}^k$ is the best position of all particles, $\boldsymbol{p}_i^k$ is the best position previously encountered by particle i, and k is an iteration.

procedure, each particle searches for solution space while updating $\boldsymbol{p}_i$ and $\boldsymbol{g}$.

III. PSO with Reduction

In this study, we present reduction of PSO. A number of particles is prepared initially and particles are updated in algorithm of PSO. Particles are sequentially reduced to reach a predetermined number based on evaluation value and PSO continues until the termination condition is met. As a particle with a maximum value is selected and deleted, the worst position of particle reduces. The reduction algorithm of PSO is presented as follows.

[*Reduction algorithm of PSO*]:

1 Initialization:
(1.1) Give parameters w, c_1, c_2, initial number of particles m_0, final number of particles m_f, maximum iteration $T_{\max}$, and partial iteration $u = T_{\max}/(5(m_0 - m_f + 1))$.
(1.2) Yield initial velocity $\boldsymbol{v}_i^0$ and initial position $\boldsymbol{x}_i^0$ for each particle at random.
(1.3) Set $k \leftarrow 0$, $m \leftarrow m_0$, $\boldsymbol{p}_i^0 \leftarrow \boldsymbol{x}_i^0$, and $\boldsymbol{g}^0 \leftarrow \boldsymbol{p}_s^0$, where $s = \arg\min_i f(\boldsymbol{p}_i^0)$.

2 Update of velocity and position:
Adapt velocity $\boldsymbol{v}_i^{k+1}$ and position $\boldsymbol{x}_i^{k+1}$ according to Eqs. (1) and (2).

3 Update $\boldsymbol{p}_i$ and $\boldsymbol{g}$:
If $f(\boldsymbol{x}_i^{k+1}) < f(\boldsymbol{p}_i^k)$, then $\boldsymbol{p}_i^{k+1} \leftarrow \boldsymbol{x}_i^{k+1}$, otherwise $\boldsymbol{p}_i^{k+1} \leftarrow \boldsymbol{p}_i^k$.
Set $\boldsymbol{g}^{k+1} \leftarrow \boldsymbol{p}_s^{k+1}$, where $s = \arg\min_i f(\boldsymbol{p}_i^{k+1})$.

4 Reduction of particle:
If $k = u \times q$ and $m > m_f$, then reduce particle j and set $m \leftarrow m - 1$, where q is a positive integer and $j = \arg\max_i f(\boldsymbol{x}_i^{k+1})$

5 Termination condition:
If $k = T_{\max}$, then terminate, otherwise set $k \leftarrow k + 1$ and go to Step 2.

IV. Numerical Experiments

In numerical experiments, we exhibit the proposed algorithm compared to existing algorithms by using six functions in two-dimensional space. Test functions are expressed in the following descriptions.

F_1 2^n minima function:

$$F_1(\boldsymbol{x}) = \sum_{i=1}^{n} \left[x_i^4 - 16x_i^2 + 5x_i\right] \tag{3}$$

F_2 Rastrigin's function:

$$F_2(\boldsymbol{x}) = 10n + \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i)\right] \tag{4}$$

F_3 Levy's function:

$$\begin{aligned} F_3(\boldsymbol{x}) &= \frac{\pi}{n}\Big\{ \sum_{i=1}^{n-1} \left[(x_i - 1)^2(1 + 10\sin^2(\pi x_{i+1}))\right] \\ &\quad +10\sin^2(\pi x_1) + (x_n - 1)^2\Big\} \end{aligned} \tag{5}$$

F_4 Schwefel's function:

$$F_4(\boldsymbol{x}) = -\sum_{i=1}^{n} (x_i \sin(\sqrt{|x_i|})) \tag{6}$$

F_5 Shubert's function:

$$\begin{aligned} F_5(\boldsymbol{x}) &= \Big\{ \sum_{i=1}^{n-1} i\cos[(i+1)x_1 + i]\Big\} \\ &\quad +\Big\{ \sum_{i=1}^{n-1} i\cos[(i+1)x_2 + i]\Big\} \end{aligned} \tag{7}$$

F_6 Shekel's Foxholes Function:

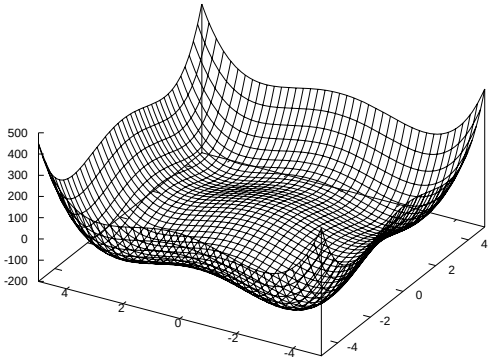
$$F_6(\boldsymbol{x}) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right]^{-1} \tag{8}$$

where

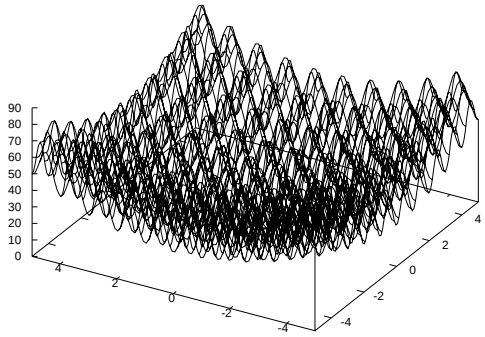
$$\boldsymbol{a}_{ij} = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \cdots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \cdots & 32 & 32 & 32 \end{pmatrix}$$

Tables 1 and 2 show the domain of test functions and the position of minimal solution for each test function, respectively. Figure 2 shows the shape of each function in case of two-dimensional space. Parameters used for experiments are chosen in Table 3 and as follows: $c_1 = 1.4955$, $c_2 = 1.4955$, and $w = 0.729$.

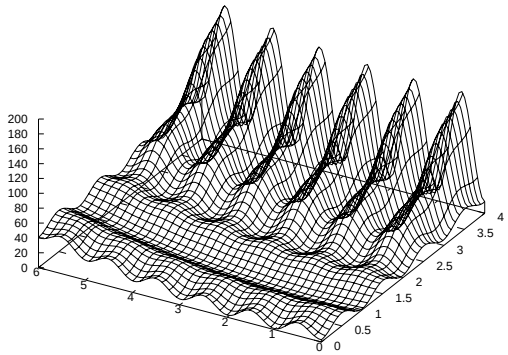
Table 4 shows the minimum in averages of 10000 trials for each of real-coded genetic algorithm (RGA), differential evolution (DE), conventional algorithm (Conv.), and proposed algorithm (Prop.). The number of particle for real-coded



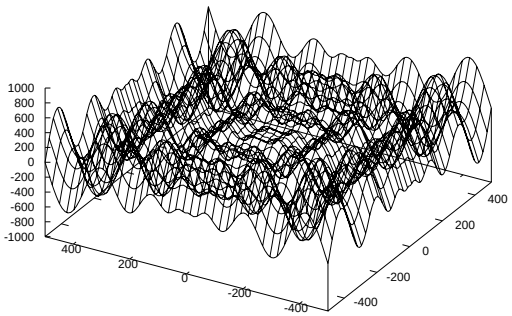
(a) F_1: 2^n minima function



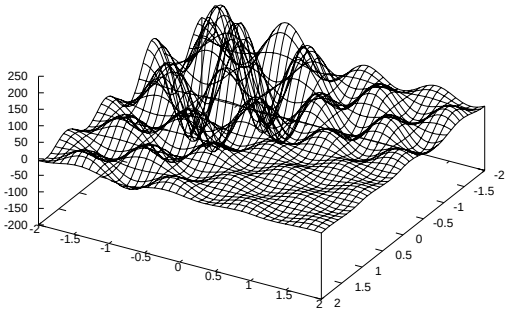
(b) F_2: Rastrigin's function



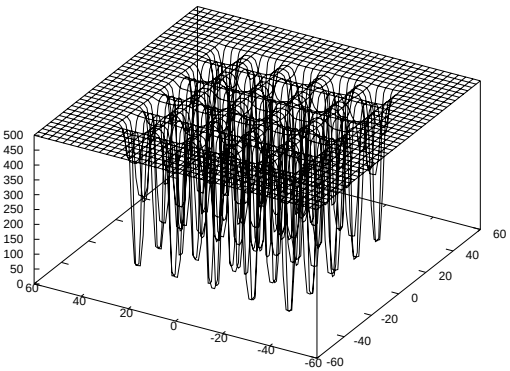
(c) F_3: Levy's function



(d) F_4: Schwefel's function



(e) F_5: Shubert's function



(f) F_6: Shekel's Foxholes function

Fig. 2. Shape of test functions.

TABLE IV
EXPERIMENTAL RESULTS

	RGA	DE	Conv.	Prop.
F_1	-1.55×10^2	-1.56×10^2	-1.55×10^2	-1.57×10^2
F_2	3.32×10^{-2}	6.55×10^{-2}	2.09×10^{-2}	1.99×10^{-3}
F_3	1.35×10^{-1}	3.53×10^{-2}	9.14×10^{-3}	0.00
F_4	-5.95×10^2	-7.80×10^2	-7.83×10^2	-8.04×10^2
F_5	-1.83×10^2	-1.86×10^2	-1.86×10^2	-1.87×10^2
F_6	5.91	2.10	1.39	1.01

TABLE I
DOMAIN OF TEST FUNCTIONS

Function	Domain
F_1	$-5.0 \leq x_i \leq 5.0$
F_2	$-5.0 \leq x_i \leq 5.0$
F_3	$0.0 \leq x_1 \leq 4.0$, $0.0 \leq x_2 \leq 6.0$
F_4	$-500.0 \leq x_i \leq 500.0$
F_5	$-2.0 \leq x_i \leq 2..0$
F_6	$-60.0 \leq x_i \leq 60.0$

TABLE II
POSITION OF MINIMAL SOLUTION FOR EACH TEST FUNCTIONS

Function	Position
F_1	$(-2.90, -2.90)$
F_2	$(0.0, 0.0)$
F_3	$(1.0, 1.0)$
F_4	$(420.9678, 420.9678)$
F_5	$(-1.425, -0.800)$ or $(-0.800, -1.425)$
F_6	$(-32.0, -32.0)$

TABLE III
PARAMETERS

Trials	10000
Maximum iteration	1000
Minimum population	20
Dimensions (n)	2

genetic algorithm, differential evolution, and conventional algorithm is 20, and the number for proposed algorithm reduces 50 to 20. It is proven that the proposed algorithm leads the best position compared to the conventional algorithm. The proposed algorithm is better than existing algorithms.

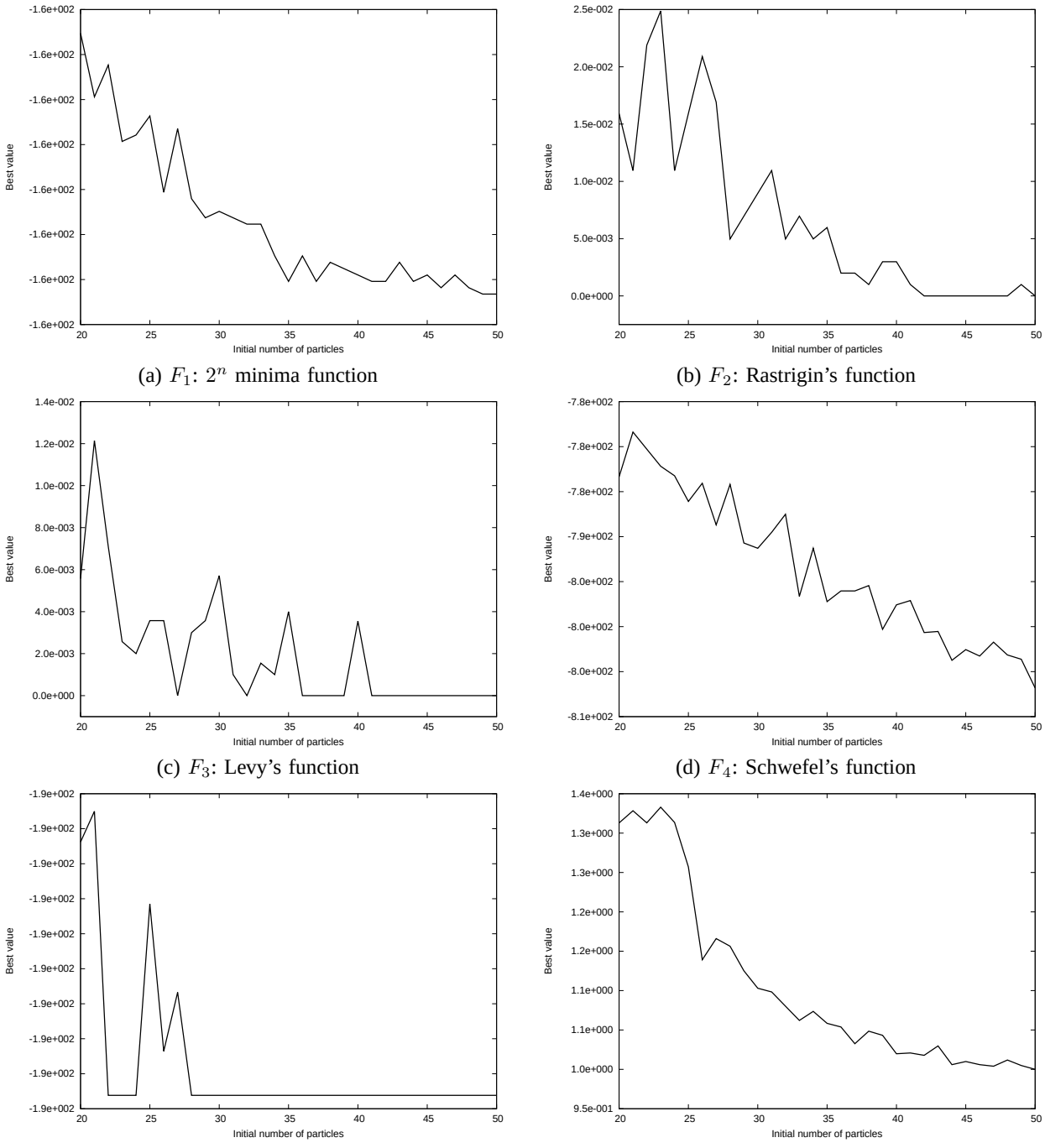
For reduction, Fig. 3 shows the relationship between the best value and the initial number of particles. We studied the effect that the initial number of particles had on accuracy in reduction. When the initial number is 20, the proposed algorithm becomes the conventional algorithm because there are no particles to be deleted. The best value gradually decreases as the initial number of particles increases. It is found that the proposed algorithm is more effective on the complicated functions given in this experiments.

V. CONCLUSIONS

In this paper, we have presented an algorithm of particle swarm optimization with reduction for improving the solution search accuracy. We examined the minimum value by using test functions to show the effectiveness of the proposed algorithm and the influence of best value on the initial number of particles for our algorithm. As a result, our algorithm had a superiority in comparison with existing algorithms for the complicated functions given in this paper. For the future works, we will study more effective techniques of our algorithm.

REFERENCES

[1] E. Aiyoshi and K. Yasuda, *Metaheuristics and Applications*, Ohmsha, 2007.
[2] H. Kitano, *Genetic Algorithm* vol.4, Sangyotosho, 2000.
[3] K. Price, R. Storn, and J. Lampinen, *Differential Evolution*, Springer-Verlag, 2005.
[4] J. Kennedy and R. Eberhart Particle swarm optimization, *IEEE Proc. Int. Conf. Neural Networks*, pp.1942–1948, 1995.
[5] S. Kinoshita, A. Ishigame and K. Yasuda, Particle swarm optimization with hierarchical structure, *The Institute of Electrical Engineers of Japan*, vol.130, pp.100–107, 2009.
[6] W. Langdon and R. Poli, Evolving problems to learn about particle swarm optimization and other search algorithms, *IEEE Trans. Evolutionary Computation*, vol.11, no.5, pp.561–578, 2007.
[7] J. Liang, A. Qin, P. Suganthan, and S. Baskar, Comprehensive learning particle swarm optimization for global optimization of multimodal functions, *IEEE Trans. Evolutionary Computation*, vol.10, no.3, pp.281–295, 2006.
[8] K. Parsopoulos and M. Vrahatis, On the computation of all global minimizers through particle swarm optimization. *IEEE Trans. Evolutionary Computation*, vol.8, no.3, pp.211–224, 2004.
[9] Y. Shi, Particle swarm optimization, *IEEE Connections*, vol.2, no.1, pp.8–13, 2004.
[10] S. Tsuda, T. Liu, and M. Maeda, Solution search algorithm of particle swarm optimization with perturbation term, *ICIC Express Letters*, vol.5, no.5, pp.1515–1521, 2011.
[11] S. Wu and T. Chow, Self-organizing and self-evolving neurons: A new neural network for optimization, *IEEE Trans. Neural Networks*, vol.18, no.2, pp.385–396, 2007.
[12] W. Yeh, Y. Lin, Y. Chung, and M. Chih, A particle swarm optimization approach based on monte carlo simulation for solving the complex network reliability problem, *IEEE Trans. Reliability*, vol.59, no.1, pp.212–221, 2010.
[13] R. Reed, Pruning algorithms—A survey, *IEEE Trans. Neural Networks*, vol.4, pp.740–747, 1993.
[14] M. Ishikawa, Structural learning with forgetting, *Neural Networks*, vol.9, pp.509–521, 1996.
[15] M. Maeda, N. Shigei, and H. Miyajima, Adaptive vector quantization with creation and reduction grounded in the equinumber principle, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol.9, pp.599–606, 2005.
[16] M. Maeda, N. Shigei, H. Miyajima, and K. Suzaki, Reduction models in competitive learning founded on distortion standards, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol.12, pp.314–323, 2008.



(a) F_1: 2^n minima function

(b) F_2: Rastrigin's function

(c) F_3: Levy's function

(d) F_4: Schwefel's function

(e) F_5: Shubert's function

(f) F_6: Shekel's Foxholes function

Fig. 3. Best value and initial number of particles for each function.