21

Simultaneous Perturbation Particle Swarm Optimization and Its FPGA Implementation

Yutaka Maeda and Naoto Matsushita
Kansai University
Japan

1. Introduction

The particle swarm optimization technique is one of the promising tools to find a proper optimum for an unknown function optimization. Especially, global search capability of the method is very powerful. The particle swarm optimization utilizes common knowledge of the group and individual experiences effectively. That is, direction for the best estimator that a particle has ever reached, direction for the best one that all particles have ever found and momentum are successfully combined to determine the next direction. At the same time, the method does not utilize gradient of the objective function. Only values of the objective function are used. In many applications, it is difficult or impossible to obtain the gradient of an objective function. Then, the particle swarm optimization can take advantage of the merit.

However, this means that the method does not use local information of the function. Even if a particle is close to a global optimal, the particle moves based on three factors described above. In this case, it seems better to search neighbour area carefully. To do so, local information such as gradient is necessary.

On the other hand, the simultaneous perturbation method is a kind of stochastic gradient method. The scheme can obtain the local information of the gradient without direct calculation of the gradient. The simultaneous perturbation estimates the gradient using a kind of finite difference technique. However, even if dimension of the parameters are large, the simultaneous perturbation requires only two values of the target function. Therefore, we can apply this to high dimensional optimization problems in effect.

As mentioned now, since the simultaneous perturbation is a stochastic gradient method, we cannot expect global search capability. That is, this method cannot give a global optimal but a local one.

Combination of the particle swarm optimization and the simultaneous perturbation optimization will yield interesting algorithms which have advantages of these two approaches. There are some ways to combine the particle swarm optimization and the simultaneous perturbation method. In this paper, we propose four new algorithms based on combinations of the particle swarm optimization and the simultaneous perturbation. Some results for test functions are also shown.

Moreover, hardware implementation of these kinds of algorithms is interesting research target. Especially, the particle swarm optimization has plural search points which are

candidates of optimum. If we can evaluate these search points in parallel processing system, we can realize intriguing optimization scheme as a hardware system. From this point of view, we implemented the particle swarm optimization using the simultaneous perturbation by using field programmable gate array (FPGA). This paper presents detailed description on the implementation of the simultaneous perturbation particle swarm optimization.

2. Particle swarm optimization and simultaneous perturbation

2.1 Particle swarm optimization

The particle swarm optimization is proposed by Eberhart and Kennedy (Kennedy & Eberhart, 1995). This scheme realizes an intelligent interesting computational technique. Intelligence come out swarm behaviour of creatures are successfully modelled as an optimization scheme (Bonabeau et al., 1999)(Engelbrecht, 2006). Many applications of the particle swarm optimization for some fields are reported (Juang, 2004)(Parsopoulos & Vrahatis, 2004)(Bo et al., 2007)(Fernandez et al., 2007)(Nanbo, 2007)(del Valle et al., 2008). Our problem is to find a minimum point of an objective function $f(x) \in \Re^1$ with an adjustable n-dimensional parameter vector $x \in \Re^n$. The algorithm of the particle swarm optimization is described as follows;

$$x_{t+1} = x_t + \Delta x_t \tag{1}$$

$$\Delta x_{t+1} = \chi(\omega \Delta x_t + \phi_1(p_t - x_t) + \phi_2(n_t - x_t)) \tag{2}$$

where, the parameter vector x_t denote an estimator of the minimum point at the t-th iteration. Δx_t is called a velocity vector, that is, a modifying vector for the parameter vector. This term becomes so-called momentum for the next iteration.

p_t is the best estimator that this particle has ever reached, n_t is the best one that all the particles have ever found until the t-th iteration. The coefficients ϕ_1 and ϕ_2 are two positive random numbers in a certain range to decide a balance between the individual best estimator and the swarm best one. Uniform distribution with upper limitation is used in this work. ω denotes a coefficient to adjust the effect of the inertia, χ is a gain coefficient for the update.

As shown in Eq.(2), in the particle swarm optimization algorithm, each individual changes their position based on the balance of three factors; velocity, the individual best estimator and the group best estimator. All the particle change their position using Eq.(2).

2.2 Simultaneous perturbation

The simultaneous perturbation optimization method is very simple stochastic gradient method which does not require the gradient of an objective function but only two values of the function. The simultaneous perturbation was introduced by J.C.Spall in 1987 (Spall, 1987). Convergence of the algorithm was proved in the framework of the stochastic approximation method (Spall, 1992). Y.Maeda also have independently proposed a learning rule of neural networks based on the simultaneous perturbation method and reported a comparison between the simultaneous perturbation type of learning rule of neural networks, the simple finite difference type of learning rule and the ordinary back-

propagation method (Maeda et al.,1995). J.AIespector et al. and G.Cauwenberghs also individually proposed a parallel gradient descent method and stochastic error descent algorithm, respectively, which are identical to the simultaneous perturbation learning rule (Cauwenberghs, 1993) (Alespector et al., 1993). Many applications of the simultaneous perturbation are reported in the fields of neural networks (Maeda, 1997) and their hardware implementation (Maeda, 2003) (Maeda, 2005). The simultaneous perturbation method is described as follows;

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - a\Delta \boldsymbol{g}_t \tag{3}$$

$$\Delta g_t^i = \frac{f(\boldsymbol{x}_t + \boldsymbol{c}_t) - f(\boldsymbol{x}_t)}{c_t^i} \quad (i = 1, \cdots, n) \tag{4}$$

Where, a is a positive constant, c and c_t^i are a perturbation vector and its i-th element which is determined randomly. Δg_t^i represents the i-th element of Δg_t . c_t^i is independent with different element and different iteration. For example, the segmented uniform distribution or the Bernoulli distribution is applicable to generate the perturbation. Δg_t becomes an estimator of the gradient of the function.

As is shown in Eq.(4), this method requires only two values of the target function despite of dimension of the function. That is, even if the dimension n of the evaluation function is so large, two value of the function gives the partial derivative of the function with respect to all the parameters, although ordinary finite difference requires many values of the function. Combination with the particle swarm optimization is very promising approach to improve performance of the particle swarm optimization.

3. Combination of particle swarm optimization and simultaneous perturbation

We can obtain a global optimal using the particle swarm optimization. However, unfortunately, since the particle swarm optimization itself does not have a capability searching the neighbor of the position, and it may miss the optimal point near the present position. As a result, efficiency of the particle swarm optimization may be limited in some cases.

On the other hand, the simultaneous perturbation estimates gradient of the position. The simultaneous perturbation method searches only local area based on the estimated gradient. If we can add the local search capability of the simultaneous perturbation to global search one of the particle swarm optimization, we will have a useful optimization method with good global search capability and efficient local search ability at the same time. Therefore, combination of the particle swarm optimization and the simultaneous perturbation is promising and interesting approach.

Combined methods of the particle swarm optimization and the simultaneous perturbation is proposed by Maeda (Maeda, 2006). In this work, the update algorithm which is a combination of particle swarm optimization and the simultaneous perturbation is applied for all the particles uniformly. In other words, the same update algorithm is used for all particles.

In population, there are plural particles and we know the best one. The best individual is the best candidate for a global optimal at that iteration. A possibility that the particle is close to

the global optimal is high. We change the movement rule depending on a situation of the particles. Especially, the best particle has a specific meaning;
From this point of view, we propose some schemes which are combinations of the particle swarm optimization and the simultaneous perturbation.

3.1 Scheme 1

We directly combine the idea of the particle swarm optimization and the simultaneous perturbation. In this method, the momentum term of Eq.(2) is replaced by the simultaneous perturbation term. The estimated gradient generated by Eq.(4) is used to change the direction of modification. The main equation is shown as follows;

$$\Delta x_{t+1}=\chi(- a\Delta g_t+ \phi_1(p_t - x_t) + \phi_2(n_t - x_t)) \quad (5)$$

Where the i-th element of Δg_t is defined by Eq.(4). a is a coefficient to adjust the effect of the estimated gradient.
Since the information is estimated by the simultaneous perturbation method, the algorithm does not use the gradient of the function directly but utilizes only two values of the objective function. Therefore, this scheme contains twice observations or calculations for the objective function. However, this number of the observations does not depend on the dimension n of the function. Local information of the gradient of the function is added to the ordinary particle swarm optimization effectively. Fig.l shows elements to generate modifying quantity in the first algorithm.

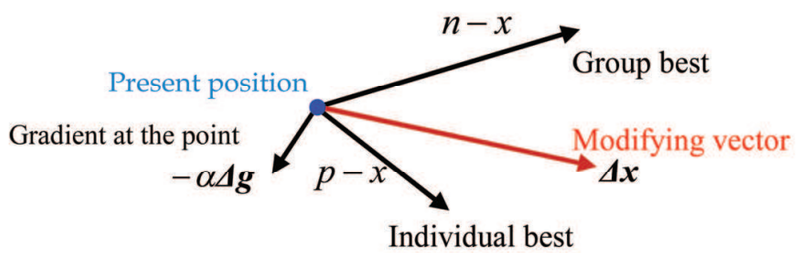


Figure 1. Modifying vector of the algorithm 1

3.2 Scheme 2

In the algorithm 1, all individuals have the same characteristics. That is, Eq.(5) is applied for all particles. However, if the best particle is close to the global minimum, and this is likely, the best particle had better search neighbor of the present point carefully. Then, modification based on the original particle swarm optimization is not suitable for this particle. The gradient type of method is suitable.
Therefore, in this algorithm 2, the simultaneous perturbation method of Eqs.(3) and (4) are applied only to the best particle. All the other individuals are updated by the ordinary particle swarm optimization shown in Eqs.(l) and (2).

3.3 Scheme 3

In this algorithm 3, the particle swarm optimization and the simultaneous perturbation are mixed. That is, in every iteration, half of individuals in the population are updated by the

particle swarm optimization, left half particles are modified only by the simultaneous perturbation.
All the individuals select the particle swarm optimization or the simultaneous perturbation randomly with probability of 0.5 in every iteration.
It is interesting what level of performance does such a simple mixture of the particle swarm optimization and the simultaneous perturbation has. Changing ratio of the particle swarm optimization and the simultaneous perturbation is another option.

3.4 Scheme 4

We have another option to construct new algorithm. Basically, we use the algorithm 3. However, the best individual is updated only by the simultaneous perturbation. The reason is as same as that of the algorithm 2. The best particle has a good chance to be a neighbor of a global minimum. Therefore, we always use the simultaneous perturbation for the best particle.

4. Comparison

In order to evaluate performance of these algorithms, we use the following test functions. These functions have their inherent characteristics about local minimum or slope.

- Rastrigin function
- Rosenbrock function
- 2^n-minima function

Comparisons are carried out for ten-dimensional case, that is, n=10 for all test functions. Average of 50 trials is shown. 30 particles are included in the population. Change of average means that an average of the best particle in 30 particles at the iteration for 50 trials are shown. For the simultaneous perturbation term, the perturbation c is generated by uniform distribution in the interval [0.01 0.5] for the scheme 1 to 4. These setting are common for the following test functions.

1. Rastrigin function

The function is described as follows;

$$f(x) = 10n + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i)) \tag{6}$$

The shape of this function is shown in Fig.2 for two-dimensional case. The value of the global minimum of the function is 0. Searched area is -5 up to +5 for the function. We found the best setting of the particle swarm optimization for the function χ=1.0 and ω=0.9. Upper limitation of ϕ_1 and ϕ_2 are 2.0 and 1.0, respectively. Using the setting (See Table 1), we compare these four methods and the ordinary particle swarm optimization.
As shown in the figure, this function contains many local minimum points. It is generally difficult to find a global minimum using the gradient type of the method. It is difficult also for the particle swarm optimization to cope with the function. The past experiences will not give any clue to find the global minimum. This is one of difficult functions to obtain the global minimum.
Change of the best particle is also depicted in Fig.3. The horizontal axis is number of observations for the function. The ordinary particle swarm optimization requires the same number of observations with the number of particles in the population. Since the scheme 1

contains the simultaneous perturbation procedure, the scheme uses twice number of the observations. However, this does not change, even if the dimension of the parameters increases.

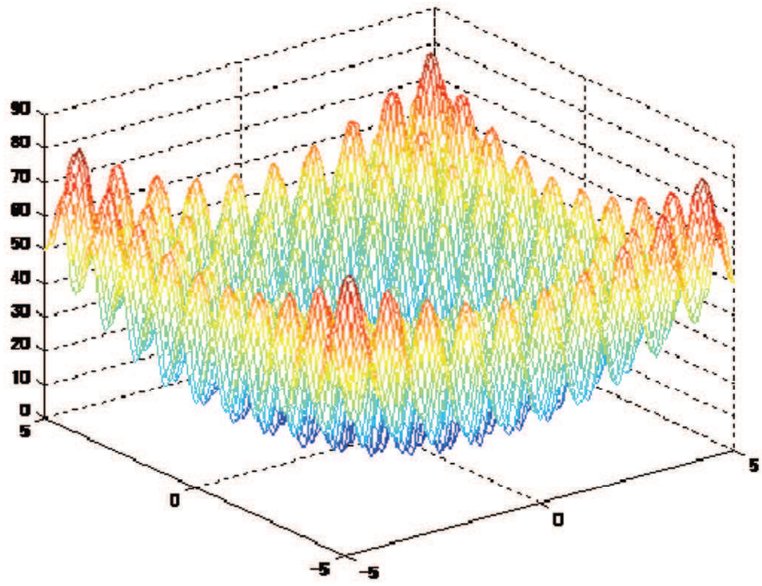


Figure 2. Rastrigin function

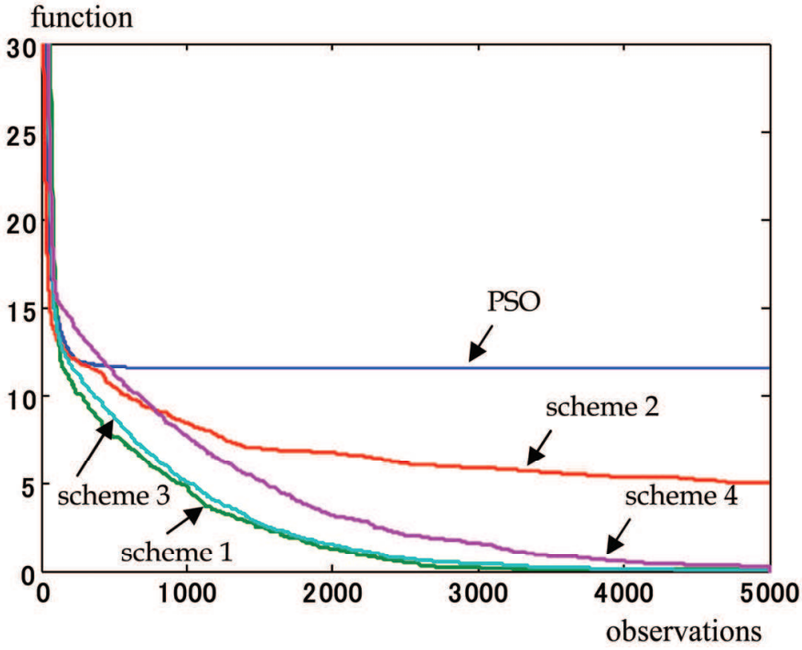


Figure 3. Change of the best particle for Rastrigin function

ϕ_1	ϕ_2	χ	ω	Scheme 1	Scheme 2	Scheme 3	Scheme 4
2.0	1.0	1.0	0.9	$\alpha = 0.000003$	$\alpha = 0.00003$	$\alpha = 0.000006$	$\alpha = 0.000003$

Table 1. Parameters setting for Rastrigin function

The scheme 2 has the number of the observations of the ordinary particle swarm optimization plus one, because only the best particle uses the simultaneous perturbation. The scheme 3 requires 1.5 times of number of the observation of the particle swarm optimization, because half of the particles in the population utilize the simultaneous perturbation. The scheme 4 basically uses the same number of the observations with the scheme 3. In our work, we take these different situations into account. For this function, scheme 1,3 and 4 have relatively good performance.

2. Rosenbrock function

Shape of the function is shown in Fig.4 for two-dimensional case. The value of the global minimum of the function is 0. Searched area is -2 up to +2. Parameters are shown in Table 2. Since the Rosenbrock has gradual descent, the gradient method with suitable gain coefficient will easily find the global minimum. However, we do not know the suitable gain coefficient so that the gradient method will be inefficient in many cases. On the other hand, the particle swarm optimization is beneficial for this kind of shape, because the momentum term accelerates moving speed and plural particles will be able to find the global minimum efficiently.

Change of the best particle is depicted in Fig.5. From Fig.5, we can see that the scheme 2 and the ordinary particle swarm optimization have relatively good performance for this function. As we mentioned, the ordinary gradient method has not good performance, the particle swarm optimization is suitable. If we add local search for the best particle, the performance will increase. The results illustrate this. The scheme 1 does not have the momentum term, it is replaced by the estimated gradient term by the simultaneous perturbation. The momentum term accelerates convergence and the gradient term does not work well for flat slope. It seems that this results in this slow convergence.

$$f(x) = \sum_{i=1}^{n-1} \left[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2 \right] \tag{7}$$

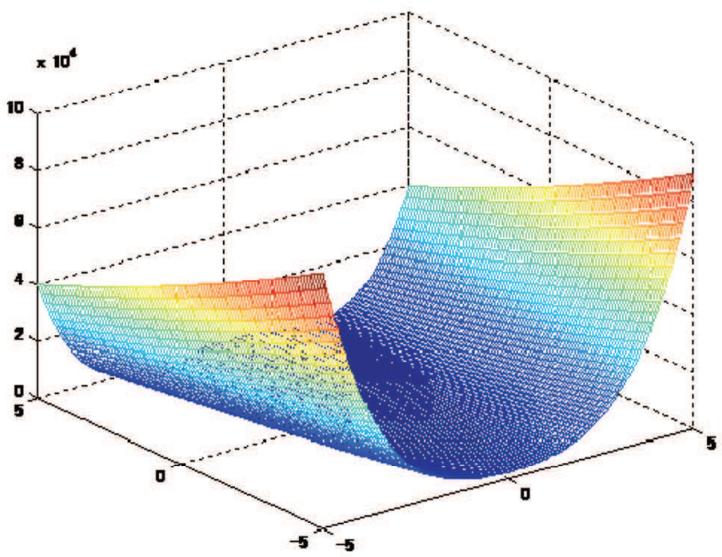


Figure 4. Rosenbrock function

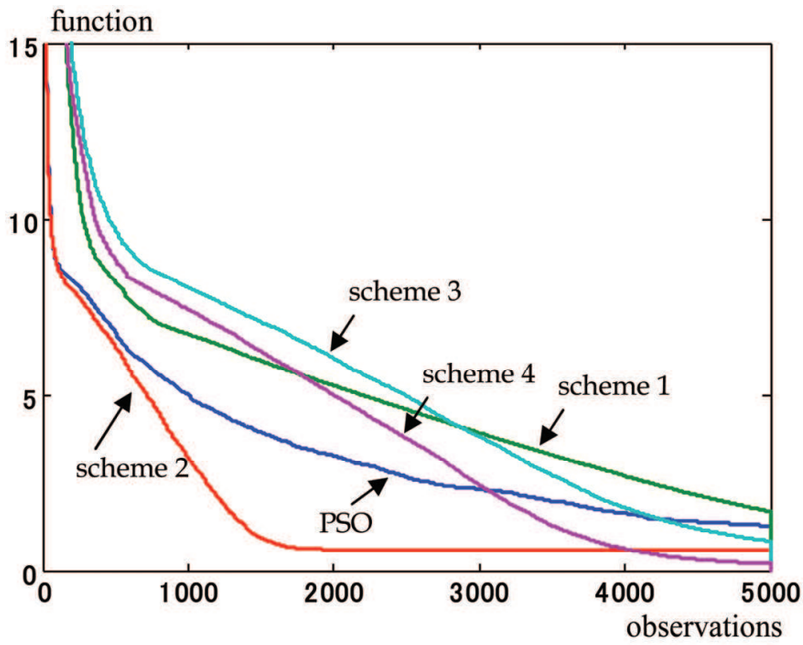


Figure 5. Change of the best particle for Rosenbrock function

ϕ_1	ϕ_2	χ	ω	Scheme 1	Scheme 2	Scheme 3	Scheme 4
1.5	1.0	1.0	0.9	α = 0.0000007	α = 0.00000008	α = 0.000000006	α = 0.00000001

Table 2. Parameters setting for Rosenbrock function

3. 2^n-minima function

The 2^n-minima function is

$$f(x) = \sum_{i=1}^{n} \left[x_i^4 - 16x_i^2 + 5x_i \right] \tag{8}$$

Shape of the function is shown in Fig.6. Searched area is -5 up to +5. Table 3 shows parameters setting. The value of the global minimum of the function is -783.32.

The function has some local minimum points and relatively flat bottom. This deteriorates search capability of the gradient method. Change of the best particle is also depicted in Fig.7. The scheme 4 has relatively good performance for this case. The function has flat bottom including a global minimum. In order to search the global minimum, it seems that the swarm search is useful. Searching the global minimum using many particles is efficient. Simultaneously, local search is necessary to find exact position of the global minimum. It seems that the scheme 4 matched for the case.

As a result, we can say that the gradient search is important and combination with the particle swarm optimization will give us a powerful tool.

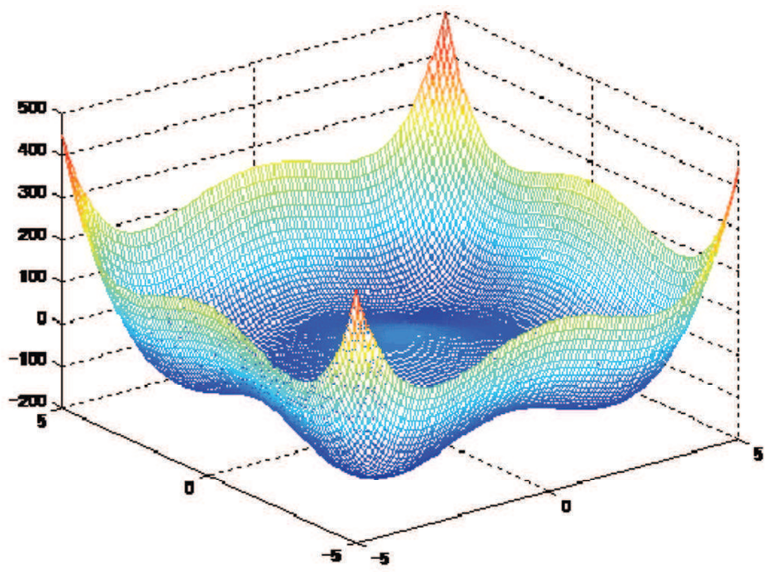


Figure 6. 2^n-minima function

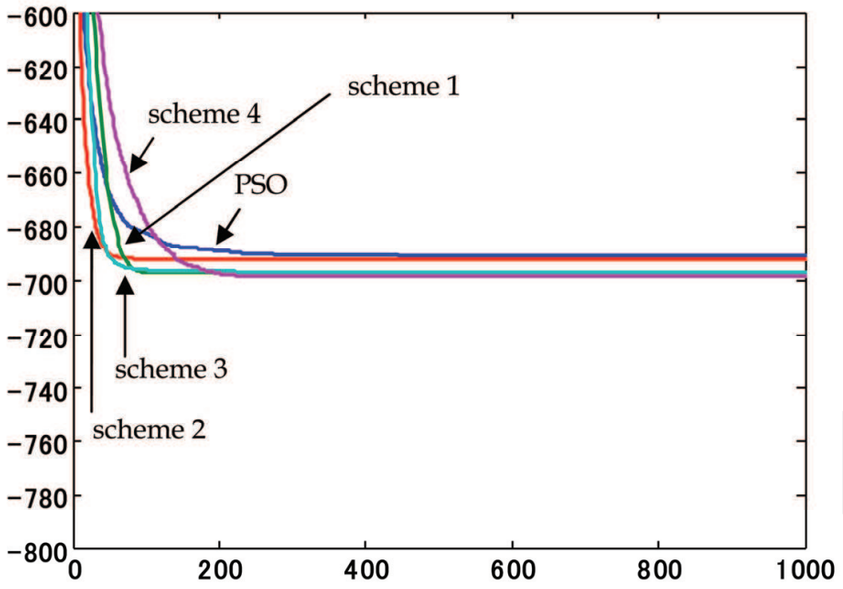


Figure 7. Change of the best particle for 2^n-minima function

ϕ_1	ϕ_2	χ	ω	Scheme 1	Scheme 2	Scheme 3	Scheme 4
2.0	1.5	1.0	0.9	$\alpha = 0.00009$	$\alpha = 0.0008$	$\alpha = 0.0008$	$\alpha = 0.000005$

Table 3. Parameters setting for 2^n-minima function

5. FPGA implementation

Now, we implement the simultaneous perturbation particle swarm optimization using FPGA. Then we can realize one feature of parallel operation of the particle swarm optimization. This results in higher operation speed for optimization problems.
We adopted VHDL (VHSIC Hardware Description Language) in basic circuit design for FPGA. The design result by VHDL is configured on MU200 - SX60 board with EP1S60F1020C6 (Altera) (see Fig.8). This FPGA contains 57,120 LEs with 5,215,104 bit user memory.

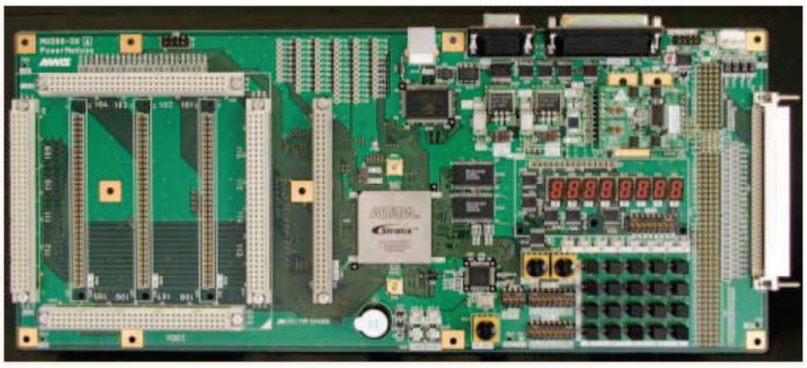


Figure 8. FPGA board MU200-SX60 (MMS)

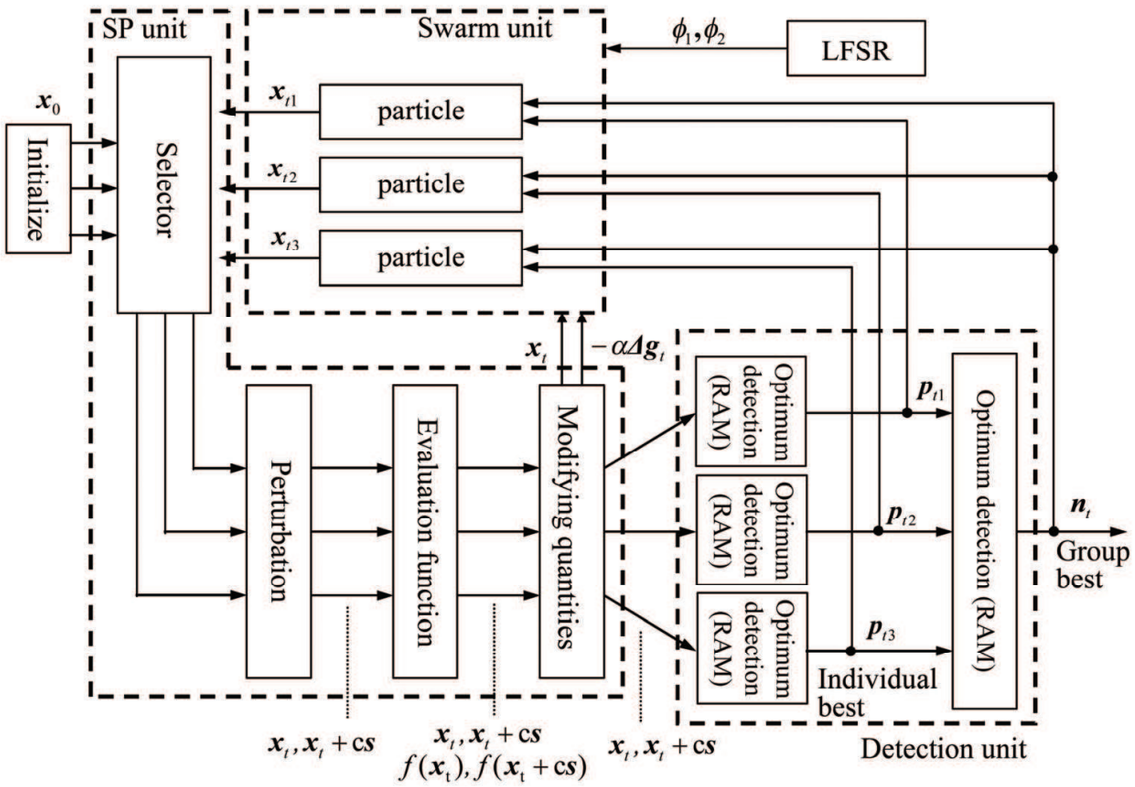


Figure 9. Configuration of the SP particle swarm optimization system

Visual Elite (Summit) is used for the basic deign. Synplify Pro (Synplicity) carried out the logical synthesis for the VHDL. QuartusII (Altera) is used for wiring.Overall system configuration is shown in Fig.9. Basically, the system consists of three units; swarm unit, detection unit and simultaneous perturbation unit.
In this system, we prepared three particles. These particles works parallel to obtain values of a target function, and are updated their positions and velocity. Therefore, even if the system has many particles, this does not effect on the overall operation speed. Number of the particles in this system is restricted by the scale of target FPGA. We should economize the design, if we would like to contain many particles.
The target function with two parameters x_1 and x_2 is defined as follow;

$$f(\boldsymbol{x}) = 16 + \sum_{i=1}^{2}(x_i^2 - 8(1 - 4x_i^2 + \frac{1}{4}x_i^4 - \frac{1}{128}x_i^6 + \frac{1}{8192}x_i^8)) \tag{9}$$

Based on Rastrigin function, we assume this test function with optimal value of 0 and 8th order. We would like to find the optimal point (0.0 0.0) in the range [-5.5 5.5]. Then optimal value of the function is 0. Fig. 10 shows shape of the function.

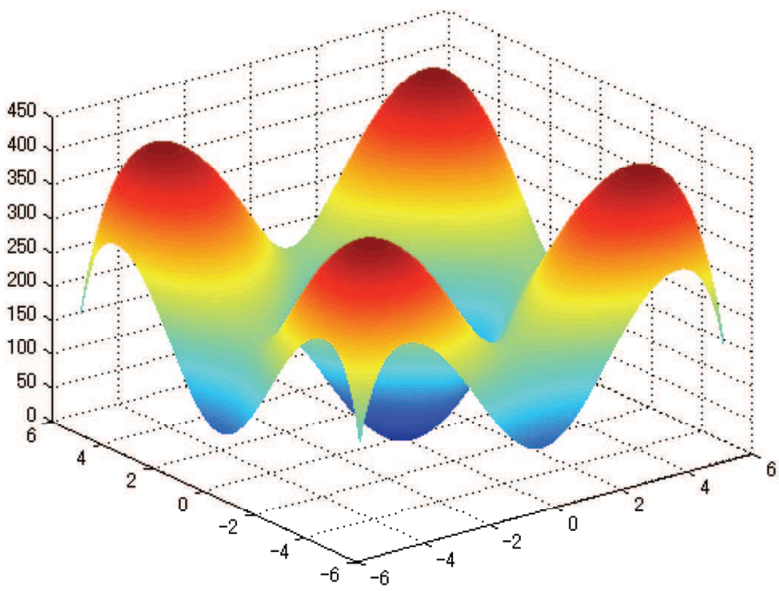


Figure 10. Shape of the target function

5.1 Swarm unit

Swarm unit includes some particles which are candidate of the optimum point. Candidate values are memorized and updated in these particle parts.
Configuration of the particle part is shown in Fig. 11. This part holds its position value and velocity. At the same time, modifying quantities for all particles are sent by the

simultaneous perturbation unit. The particle part updates its position and velocity based on these modifying quantities.

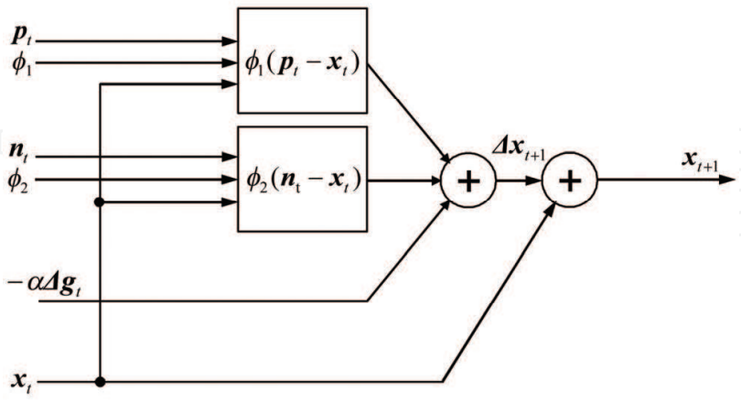


Figure 11. Particle part

5.2 Detection unit

The detection unit finds and holds the best estimated value for each particle. We refer this estimator as individual best. And based on the individual best values of the each particle, the unit searches the best one that all the particles have ever found. We call it the group best. The individual best values and the group best value are stored in RAM. For iteration, new positions for all particles are compared with corresponding values stored in RAM. If new position is better, it is stored, that is, the individual best value is updated. Moreover, these are used to determine the group best value.

These individual best values and the group best value are used in the swarm unit to update the velocity.

5.3 Simultaneous perturbation unit

The simultaneous perturbation unit realizes calculation of evaluation function for each particle, estimation of the gradient of the function based on the simultaneous perturbation. As a result, the unit produces estimated gradient for all the particles. The results are sent to the swarm unit.

5.4 Implementation result

Single precision floating point expression IEEE 574 is adopted to express all values in the system. Ordinary floating point operations are used to realize the simultaneous perturbation particle swarm optimization algorithm.

We searched the area of [-5.5 5.5]. Initial positions of the particles were determined randomly from (2.401, 2.551), (-4.238, 4.026) or (-3.506, 1.753). Initial velocity was all zero. Then we defined value of χ is 1. Coefficients ϕ_1 and ϕ_2 in the algorithm were selected from 2i(=2), 2°(=1), 2-i(=0.5), 2-2(=0.25), 2-3(=0.125), 2-4(=0.0625), 2-5(=0.03125) or 2-6(=0.015625). This simplifies multiplication of these coefficients. The multiplication can be carried out by addition for exponent component.

Design result is depicted in Fig.12. 84% of LE is used for all this system design. We should economize the scale of the design, if we would like to implement much more particles in the system. Or, we can also adopt time-sharing usage of the single particle part. Then total operation speed will deteriorate.

Flow Status	Successful - Tue Apr 24 16:28:30 2007
Quartus II Version	6.0 Build 178 04/27/2006 SJ Full Version
Revision Name	SP_PSO_ver2
Top-level Entity Name	SP_PSO_ver2
Family	Stratix
Device	EP1S60F1020C6
Timing Models	Final
Met timing requirements	No
Total logic elements	47,877 / 57,120 (84 %)
Total pins	13 / 782 (2 %)
Total virtual pins	0
Total memory bits	0 / 5,215,104 (0 %)
DSP block 9-bit elements	40 / 144 (28 %)
Total PLLs	0 / 12 (0 %)
Total DLLs	0 / 2 (0 %)

Figure 12. Design result

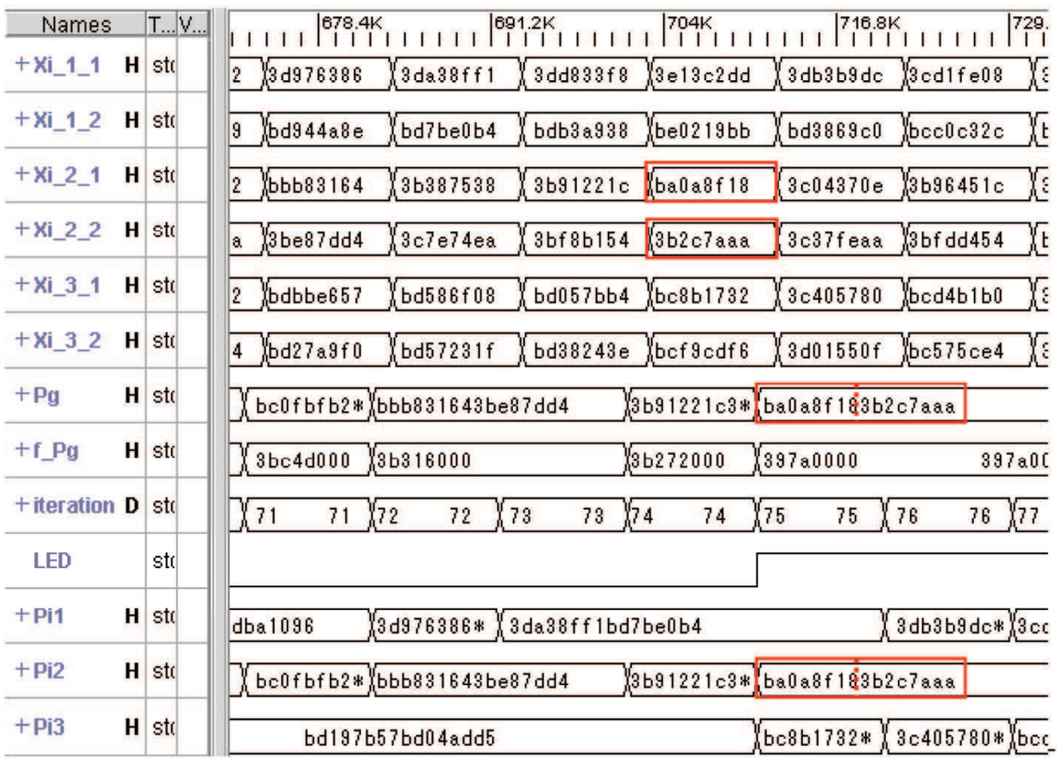


Figure 13. Simulation result

Fig.13 shows a simulation result by Visual Elite. Upper six signals xi_1_1 upto xi_3_2 denote values of parameters x_1 and x_2 of three particles, respectively. Lower three signals HI upto Pi3 are individual best values of three particles at the present iteration. x_1 and x_2 values are consecutively memorized. Between these, the best one becomes group best shown in Pg signal. f_Pg is the corresponding function value. In Fig.13 between three particle, the second particle of Pi2 became the group best value of Pg. We can find END flag of "LED" at 75th iteration.

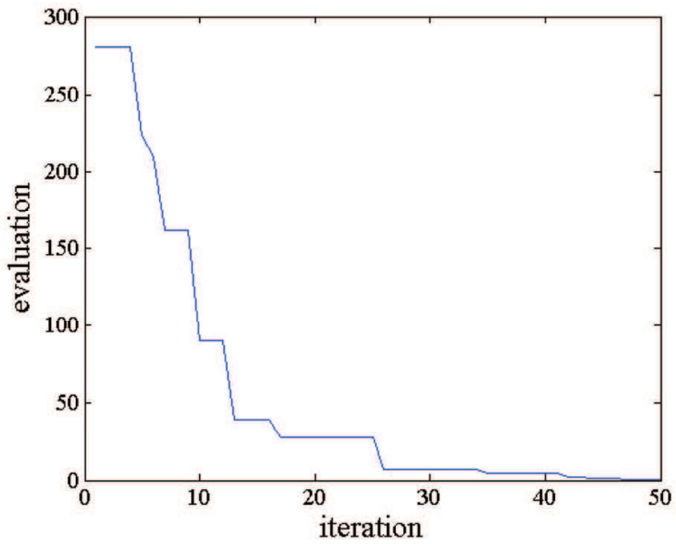


Figure 14. Operation result

Fig.14 shows a change of the evaluation function of the best of the swarm for iteration. The system easily finds the optimum point with three particles. About 50 iteration, the best particle is very close to global optimum. As mentioned before, after 75 iteration, the system stoped with an end condition.

6. Conclusion

In this paper, we presented hardware implementation of the particle swarm optimization algorithm which is combination of the ordinary particle swarm optimization and the simultaneous perturbation method. FPGA is used to realize the system. This algorithm utilizes local information of objective function effectively without lack of advantage of the original particle swarm optimization. Moreover, the FPGA implementation gives higher operation speed effectively using parallelism of the particle swarm optimization. We confirmed viability of the system.

7. Acknowledgement

This work is financially supported by Grant-in-Aid for Scientific Research (No.19500198) of the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan, and the Intelligence system technology and kansei information processing research group,

Organization for Research and Development of Innovative Science and Technology, Kansai University, and "Academic Frontier" Project of MEXT of Japan for Kansai University.

8. References

Alespector, J.; Meir, R.; Yuhas, B.; Jayakumar, A. & Lippe,D. (1993). A Parallel Gradient Descent Method for Learning in Analog VLSI Neural Networks, In: *Advances in neural information processing systems,* Hanson, S., Cowan, J., Lee, C. (Eds.), 836-844, Vol.5, Morgan Kaufmann Publisher, Cambridge, MA

Bo Y.; Yunping C. & Zunlian Z.(2007). Survey on Applications of Particle Swarm Optimization in Electric Power Systems, IEEE International Conference on Control and Automation, pp. 481-486

Bonabeau, E.; Dorigo, M. & Theraulaz, G. (1999). Swarm intelligence: from natural to artificial systems, Oxford University Press, 0-19-513159-2, NY

Cauwenberghs, G. (1993). A Fast Stochastic Error-descent Algorithm for Supervised Learning and Optimization,' In: *Advances in neural information processing systems,* Hanson, S., Cowan, J., Lee, C. (Eds.), 244-251, Vol.5, Morgan Kaufmann Publisher, Cambridge, MA

del Valle, Y.; Venayagamoorthy, G.K.; Mohagheghi, S.; Hernandez, J.-C. & Harley, R.G.(2008). Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems, *IEEE transactions on evolutionary computation,* Vol. 12, No. 2,171-195

Engelbrecht, A.P.(2006). Fundamentals of Computational Swarm Intelligence, John Wiley & Sons Ltd., West Sussex, 0-470-09191-6

Fernandez, P. M.; Rubio, B. A.; Garcia, R. P.; Garcia, S.G. & Gomez, M. R.(2007). Particle-Swarm Optimization in Antenna Design: Optimization of Log-Periodic Dipole Arrays, *IEEE Antennas and Propagation Magazine,* Vol. 49, No. 4,34-47

Juang, C. (2004). A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design, *IEEE Transaction on System, Man, and Cybernetics – Part B: Cybernetics,* Vol. 34, No.2,997-1006,

Kennedy, J. & Eberhart, R. (1995). Particle Swarm Optimization, Proceedings of the IEEE Conference on Neural Networks, pp.1942-1948,

Maeda, Y., Hirano, H. & Kanata, Y. (1995). A Learning Rule of Neural Networks Via Simultaneous Perturbation and Its Hardware Implementation, *Neural Networks,* Vol.8, No.2,251-259,

Maeda, Y. & de Figueiredo, R. J. P. (1997). Learning Rules for Neuro-controller Via Simultaneous Perturbation, *IEEE Transaction on Neural Networks,* Vol.8, No.5,1119-1130, Maeda,

Y. & Tada, T. (2003). FPGA Implementation of a Pulse Density Neural Network with Learning Ability Using Simultaneous Perturbation, *IEEE Transaction on Neural Networks,* Vol.14, No.3, 688-695,

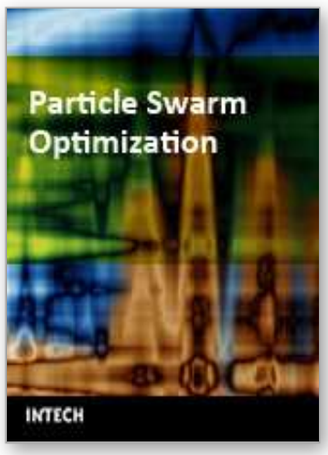
Maeda, Y. & Wakamura, M. (2005). Simultaneous Perturbation Learning Rule for Recurrent Neural Networks and Its FPGA Implementation, *IEEE Transaction on Neural Networks,* Vol. 16, No.6,1664-1672,

Maeda, Y. & Kuratani, T. (2006). Simultaneous Perturbation Particle Swarm Optimization, 2006 IEEE Congress on Evolutionary Computation, pp. 2687-2691,

Nanbo, J. & Rahmat-Samii, Y.(2007). Advances in Particle Swarm Optimization for Antenna Designs: Real-Number, Binary, Single-Objective and Multiobjective Implementations, *IEEE Transactions on Antennas and Propagation,* Vol. 55, No. 3, Part 1,556-567

Parsopoulos, K. E. & Vrahatis, M. N. (2004). On the Computation of All Global Minimizers Through Particle Swarm Optimization, *IEEE Transaction on Evolutionary Computation,* Vol. 8, No.3,211-224,

Spall, J. C. (1987). A Stochastic Approximation Technique for Generating Maximum Likelihood Parameter Estimation, Proceedings of the 1987 American Control Conference, pp.1161-1167, Spall, J. C. (1992). Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation, *IEEE Transaction on Automatic Control,* Vol. 37, No.3,332-341,



Particle Swarm Optimization

Edited by Aleksandar Lazinica

ISBN 978-953-7619-48-0
Hard cover, 476 pages
Publisher InTech
Published online 01, January, 2009
Published in print edition January, 2009

Particle swarm optimization (PSO) is a population based stochastic optimization technique influenced by the social behavior of bird flocking or fish schooling.PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. This book represents the contributions of the top researchers in this field and will serve as a valuable tool for professionals in this interdisciplinary field.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen