International Journal of Innovative Computing, Information and Control
Volume **3**, Number **1**, February **2007**

ICIC International ©2007 ISSN 1349-4198
pp. 163–173

COMPUTATIONAL INTELLIGENCE BASED ON THE BEHAVIOR OF CATS

Shu-Chuan Chu

Department of Information Management,
Cheng Shiu University
840, Cheng Cing Road, Kaohsiung County 83347, Taiwan
scchu@bit.kuas.edu.tw

Pei-Wei Tsai

Department of Electronic Engineering,
National Kaohsiung University of Applied Sciences
415, Chien Kung Road, Kaohsiung City 80778, Taiwan
pwtsai@bit.kuas.edu.tw

Received July 2006; revised October 2006

Abstract. *Optimization problems are very important in many fields. To the present, many optimization algorithms based on computational intelligence have been proposed, such as the Genetic Algorithm, Ant Colony Optimization (ACO), and Particle Swarm Optimization (PSO). In this paper, a new optimization algorithm, namely, Cat Swarm Optimization (CSO) is proposed. CSO is generated by observing the behavior of cats, and composed of two sub-models by simulating the behavior of cats. According to the experiments, the results reveal that CSO is superior to PSO.*

Keywords: Cat swarm optimization, Swarm intelligence, Soft computing, Evolutionary computing

1. **Introduction.**

Computational intelligence is a hot research topic and many related algorithms have been proposed in recent years, such as the Genetic Algorithm (GA) [1-5], The Ant Colony Optimization (ACO) [6-9], the Particle Swarm Optimization (PSO) [10-14], and the Simulated Annealing (SA) [15,16]. Some of these optimization algorithms were developed based on swarm intelligence by simulating the intelligent behavior of animals. The selection, crossover, mutation and reproduction operations are utilized in GA to solve the optimization problems. ACO solves the optimization problems by moving the tracks of ants with the assistance of the pheromone. PSO finds the solutions by moving the particles in the solution space based on the balance of personal experience and best group experience. SA optimizes the solution based on the control of temperature for the acceptance or rejection of tuning the solution. GA and SA belong to the area of evolutionary algorithms. However, ACO and PSO come under the authority of swarm intelligence.

The idea of computational intelligence may come from observing the behavior of creatures. ACO was presented by studying the behavior of ants, and PSO was presented by examining the movements of flocking gulls. Through inspecting the behavior of the cat, Cat Swarm Optimization (CSO) is proposed in this paper. The artificial structure can be viewed as the model for modeling the common behavior of cats. CSO somehow belongs to the swarm intelligence. PSO with a weighting factor [11] usually finds the optimal solution faster than the others [10], but the proposed CSO may give much better performance than PSO with a weighting factor. This paper considers the extended results of the previous work of the conference paper [17] by adding further descriptions and experiments.

2. **Observing Creatures' Behavior.**

As we mentioned above, the idea of computational intelligence may come from observing the behavior of creatures. How to describe behavior by projecting creatures' motions to formulas, without questions, becomes an extremely important phase.

2.1. **Scrutinizing motions of creatures.**

For musicians, it is effortless to reproduce beautiful melodies in compliance with the music scores. In the same manner, if we can find equations to delineate the exceptional motions of creatures, we can reconstruct the gestures in mathematical form. The mathematical models in this paper are resulted from analyzing the motions of cats, the details of which are described in the following.

2.2. **Inspecting the behaviors of felids.**

There are about thirty different species of feline creatures, e.g., lion, tiger, leopard, cat etc, according to biological classification. Though many have different living environments, felines share similar behavior patterns. The hunting skill of felines is acquired through training although it is not innate for felines. For wild felines, this hunting skill ensures their food supply and survival of their species. Domestic cats also display the same natural hunting skill, and the strong curiosity of moving objects. Although all cats share this strong curiosity, they spend most of their time inactive. If you were to observe cats, you would notice that's cats spend most of their time resting, even when they are awake. Cats have a very high level of alertness. This alertness does not desert them when they are resting. Hence you may find what appears to be a cat lazing around, but a closer examination will show large wide eyes observing their surroundings. Cats appear to be lazy when actually they are very smart and deliberate creatures. It is said that cats have nine lives. This is not meant to be taken literally, but refers to the cat's vigorous vitality. Indoor cat often give off a low frequency sound - purring. Cats will purr when content, in danger or when sick. It is believed that the low frequency of the purr helps with cell repair. This may be the reason for the cats vitality. There are of course many more remarkable things about cats that one may learn from observation, in addition to the above.

3. **Proposed Algorithm.**

Two of the major behavioral traits of cats are modeled for the proposed Cat Swarm Optimization (CSO). These are termed ”seeking mode” and ”tracking mode”. Combination of these two modes allows CSO a better performance. Details of the two modes are given below.

3.1. **The presentation of solution sets.**

The solution set must be represented for any optimization algorithm. For example, the chromosome is used to represent the solution set for genetic algorithm (GA). Ants are simulated as the agents, and the paths made by the ants depict the solution sets for ant colony optimization (ACO). The positions of particles are used to delineate the solution sets for particle swarm optimization (PSO). In our proposed algorithm, we use cats and the model of cat behavior to solve the optimization problems, i.e. we use cats to portray the solution sets.

In CSO, we first decide how many cats we would like to use in the iteration, then we apply the cats into CSO to solve the problems. Every cat has its' own position composed of M dimensions, velocities for each dimension, a fitness value, which represents the accommodation of the cat to the fitness function, and a flag to identify whether the cat is in seeking mode or tracing mode. The final solution would be the best position of one of the cats. The CSO keeps the best solution until it reaches the end of the iterations.

3.2. **Rest and alert – Seeking mode.** . This sub mode is used to model the cat during a period of resting but being alert - looking around its environment for its next move. Seeking mode has four essential factors, which are designed as follows: seeking memory pool (SMP), seeking range of the selected dimension (SRD), counts of dimension to change (CDC) and self position consideration (SPC). SMP is used to define the size of seeking memory of each cat, indication any points sort by the cat. The cat would pick a point from the memory pool, according to rules described later. SRD declares the mutative ration for the selected dimensions. While in seeking mode, if a dimension is selected for mutation, the difference between the old and new values may not be out of range, the range defines by the SRD. CDC tells how many of the dimensions will be varied. All these factors play important roles in seeking mode. SPC is a Boolean valued variable, and indicates whether the point at which the cat is already standing will be one of the candidate points to move to. SPC cannot influence the value of SMP. Seeking mode is described below.

Step 1: Make j copies of the present position of cat$_k$, where $j = SMP$. If the value of SPC is true, let $j = (SMP - 1)$, then retain the present position as one of the candidates.

Step 2: For each copy, according to CDC, randomly plus or minus SRD percents the present values and replace the old ones.

Step 3: Calculate the fitness values (FS) of all candidate points.

Step 4: If all FS are not exactly equal, calculate the selecting probability of each candidate point by equation (1), otherwise set all the selecting probability of each candidate point be 1.

Step 5: Randomly pick the point to move to from the candidate points, and replace the position of cat$_k$.

$$P_i = \frac{|FS_i - FS_b|}{FS_{max} - FS_{min}}, \qquad where \qquad 0 < i < j \tag{1}$$

If the goal of the fitness function is to find the minimum solution, $FS_b = FS_{max}$, otherwise $FS_b = FS_{min}$.

3.3. **Movement – Tracing mode.** Tracing mode is the sub-model for modeling the case of the cat in tracing targets.

Once a cat goes into tracing mode, it moves according to its' own velocities for each dimension. The action of tracing mode can be described as follows:

Step 1: Update the velocities for every dimension ($v_{k,d}$) according to equation (2).

Step 2: Check if the velocities are in the range of maximum velocity. In case the new velocity is over-range, it is set equal to the limit.

Step 3: Update the position of cat$_k$ according to equation (3).

$$v_{k,d} = v_{k,d} + r_1 \cdot c_1 \cdot (x_{best,d} - x_{k,d}), \qquad d = 1, 2, ..., M \tag{2}$$

where $x_{best,d}$ is the position of the cat, who has the best fitness value; $x_{k,d}$ is the position of cat$_k$, c_1 is a constant and r_1 is a random value in the range of $[0, 1]$.

$$x_{k,d} = x_{k,d} + v_{k,d} \tag{3}$$

3.4. **Core description of cat swarm optimization.** As already discussed, CSO has two sub modes, namely seeking mode and tracing mode. To combine these two modes into the algorithm, we define a mixture ratio (MR) which dictates the joining of seeking mode with tracing mode. It has already been commented on that cats which are awake spend most of their time resting, observing their environment. If they decide to move while resting, the movement is done carefully and slowly. This behavior is represented in seeking mode. Tracing mode models the chasing of a target by the cat. Cats spend very little time chasing things as this leads to over use of energy resources. Hence to guarantee that the cats spend most of their time resting and observing i.e. most of the time is spent in seeking mode, MR is allocated a very small value. The process of CSO is described below.

Step 1: Create N cats in the process.

Step 2: Randomly sprinkle the cats into the M-dimensional solution space and randomly give values, which are in-range of the maximum velocity, to the velocities of every cat. Then haphazardly pick number of cats and set them into tracing mode according to MR, and the others set into seeking mode.

Step 3: Evaluate the fitness value of each cat by applying the positions of cats into the fitness function, which represents the criteria of our goal, and keep the best cat into memory. Note that we only need to remember the position of the best cat (x_{best}) due to it represents the best solution so far.

Step 4: Move the cats according to their flags, if cat$_k$ is in seeking mode, apply the cat to the seeking mode process, otherwise apply it to the tracing mode process. The process steps are presented above.

Step 5: Re-pick number of cats and set them into tracing mode according to MR, then set the other cats into seeking mode.

Step 6: Check the termination condition, if satisfied, terminate the program, and otherwise repeat Step3 to Step5.

Here we present the diagram of CSO process in Figure 1.

4. **Experimental Results.** In this section, we evaluate CSO by six test functions and compare it with PSO and PSO with weighting factor (PSO with WF). The details are described in the following.

4.1. **Test functions.** We applied CSO, PSO and PSO with weighting factor into six test functions to compare the performance. These test functions are shown as equation (4) to (9).

$$f_1(x) = \sum_{d=1}^{M} [100(x_d - x_{d-1})^2 + (x_{d-1} - 1)^2] \tag{4}$$

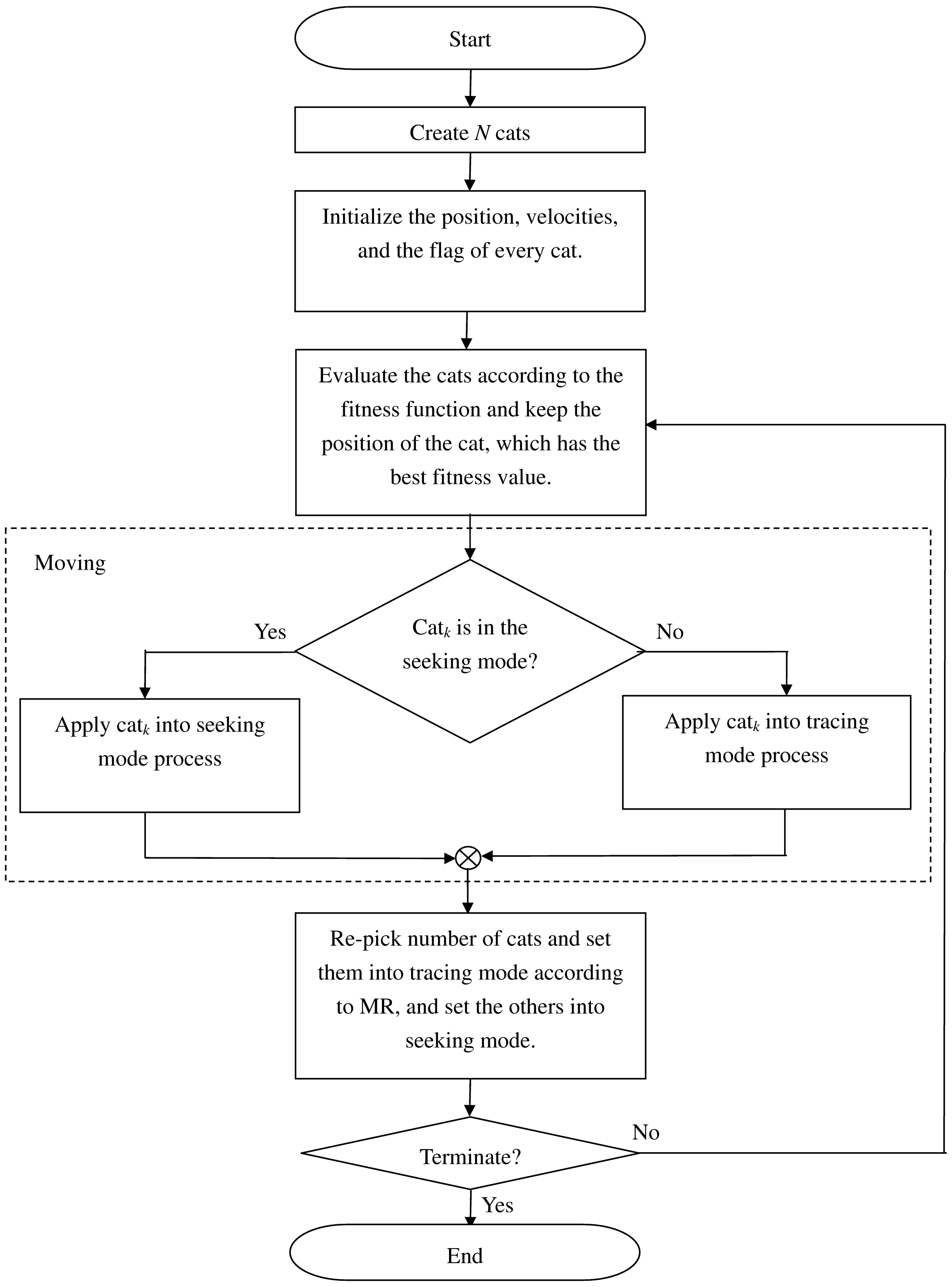
Start
Create N cats
Initialize the position, velocities, and the flag of every cat.
Evaluate the cats according to the fitness function and keep the position of the cat, which has the best fitness value.
Moving
Yes
Cat$_k$ is in the seeking mode?
No
Apply cat$_k$ into seeking mode process
Apply cat$_k$ into tracing mode process
Re-pick number of cats and set them into tracing mode according to MR, and set the others into seeking mode.
No
Terminate?
Yes
End

FIGURE 1. The process diagram of CSO

$$f_2(x) = \sum_{d=1}^{M} [{x_d}^2 - 10 \cdot \cos(2\pi x_d)^2 + 10] \tag{5}$$

$$f_3(x) = \frac{1}{400} \sum_{d=1}^{M} {x_d}^2 - \prod_{d-1}^{M} \cos(\frac{x_d}{sqrtd}) + 1 \tag{6}$$

$$f_4(x) = 20 + \exp^1 - 20 \cdot \exp^{-0.2\sqrt{\frac{\sum_{d=1}^{M} {x_d}^2}{M}}} - \exp^{\sum_{d=1}^{M} \frac{\cos(2\pi x_d)}{M}} \tag{7}$$

$$f_5(x) = \sum_{d=1}^{M} ({x_d}^2 - 10 \cos(2\pi x_d) + 10) \tag{8}$$

$$f_6(x) = \sum_{d=1}^{M} \lfloor x_d + 0.5 \rfloor^2 \tag{9}$$

In the experiments for all test function, we aim at finding the minimum of the fitness value, in other words, the goal of our experiments is to minimize the fitness function.

For each test function, we limit the initial ranges for every dimension according to Table 1. Note that the limits are only suitable for the initialization. When the process moves to the evaluation and movement stage, there is no limit on any dimension for any solution set.

Table 1. The Limitation ranges of dimensions for every test function

Function Name	Limitation Range
Test Function 1	$x_d \in [15, 30]$
Test Function 2	$x_d \in [2.56, 5.12]$
Test Function 3	$x_d \in [300, 600]$
Test Function 4	$x_d \in [-30, 30]$
Test Function 5	$x_d \in [-5.12, 5.12]$
Test Function 6	$x_d \in [-100, 100]$

4.2. **Parameter settings for CSO, PSO and PSO with WF.**

For CSO, we set the parameters, which have been discussed in the above-mentioned according to Table 2. The parameter settings for PSO and PSO with WF are listed in Table 3.

The same conditions for all the three algorithms are initial ranges, which have been shown in Table 1, the dimension of all fitness functions are set to be 30, the count of solution sets are all be set to 160. For each algorithm, we apply 2000 iterations per cycle and 50 cycles per test function, and the final results are composed of the average of the 50 cycles. Besides, the SPC flag for CSO is set to be true, and for PSO and PSO with WF, the maximum velocities for each test function are shown in Table 4. The usage of the parameters for PSO and PSO with WF can be found in [11].

Table 2. Parameter settings for CSO

Parameter	Value or Range
MP	5
SRD	20%
CDC	80%
MR	2%
c_1	2.0
r_1	$[0, 1]$

Table 3. Parameter settings for PSO and PSO with WF

Parameter	Value or Range
Initial Weight	0.9
Final Weight	0.4
c_1	2.0
c_2	2.0
r_1	$[0, 1]$
r_2	$[0, 1]$

Table 4. Maximum velocities for PSO and PSO with WF

Function Name	Maximum Velocity
Test Function 1	100.0
Test Function 2	10.0
Test Function 3	600.0
Test Function 4	100.0
Test Function 5	600.0
Test Function 6	10.0

4.3. **Experimental results.** The results of test function 1 to 6 are shown ordered in Figure 2 to Figure 7. The horizontal axis represents the iterations, and the vertical axis represents the fitness value.

Test function 1 has a name called Rosenbrock function; test function 2 is called Rastrigrin Function, and test function 3 is called Griewank Function.

4.4. **Discussion.** For the first three test functions, the initial ranges of dimensions are not including the point of global best solution; for the last three test functions, the initial ranges for dimensions are including the global best point. From the results, CSO presents a better performance of finding the global best solution, whether or not the initial range includes the global best solution.

Though CSO takes more time to finish the same iteration than PSO-type algorithms, it improves the performance of finding global set solutions. But if considering the same iteration time, CSO still presents a higher performance than PSO-type algorithms.

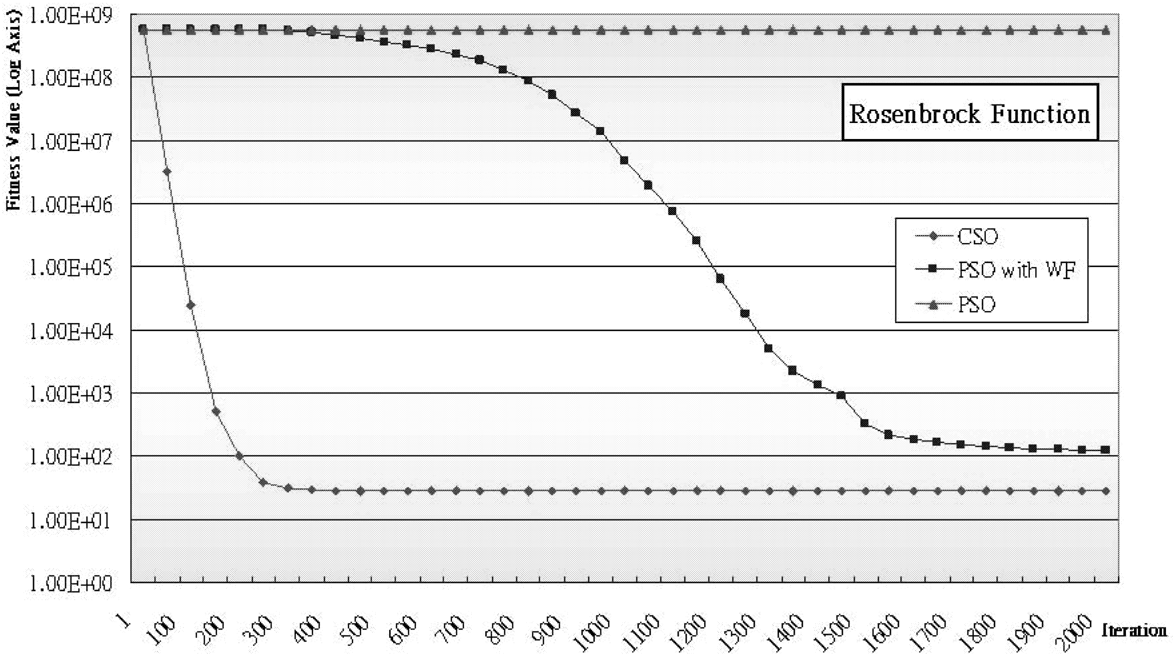

FIGURE 2. The experimental result of test function 1

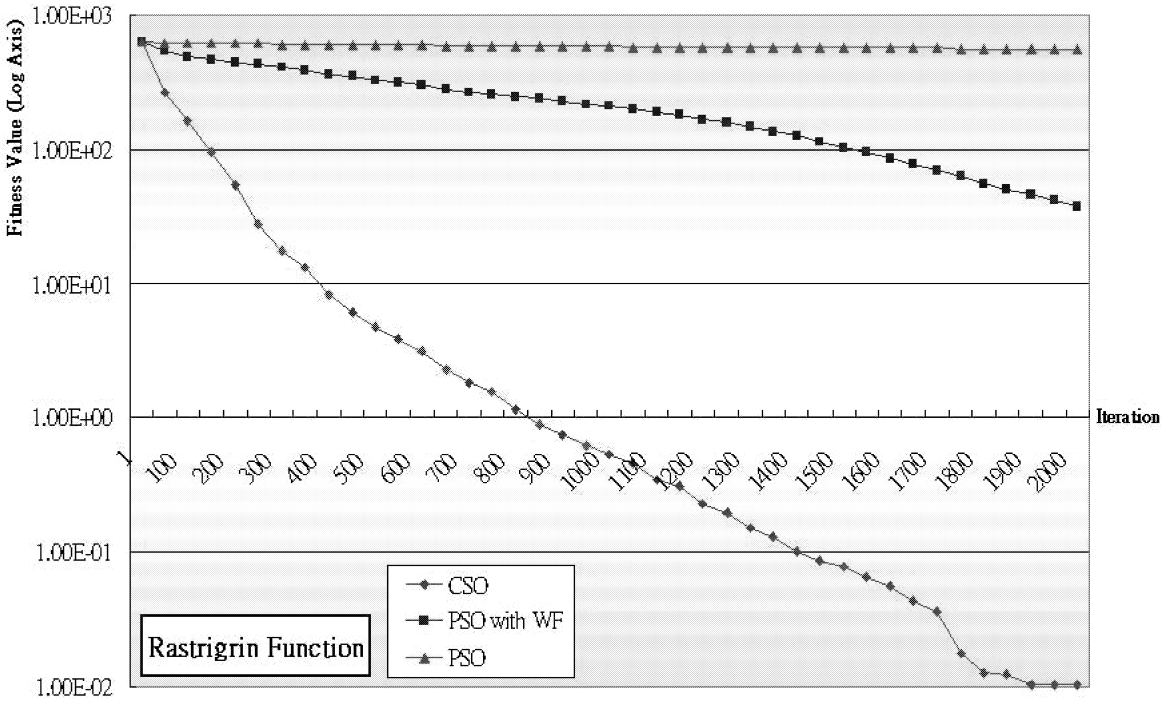

FIGURE 3. The experimental result of test function 2

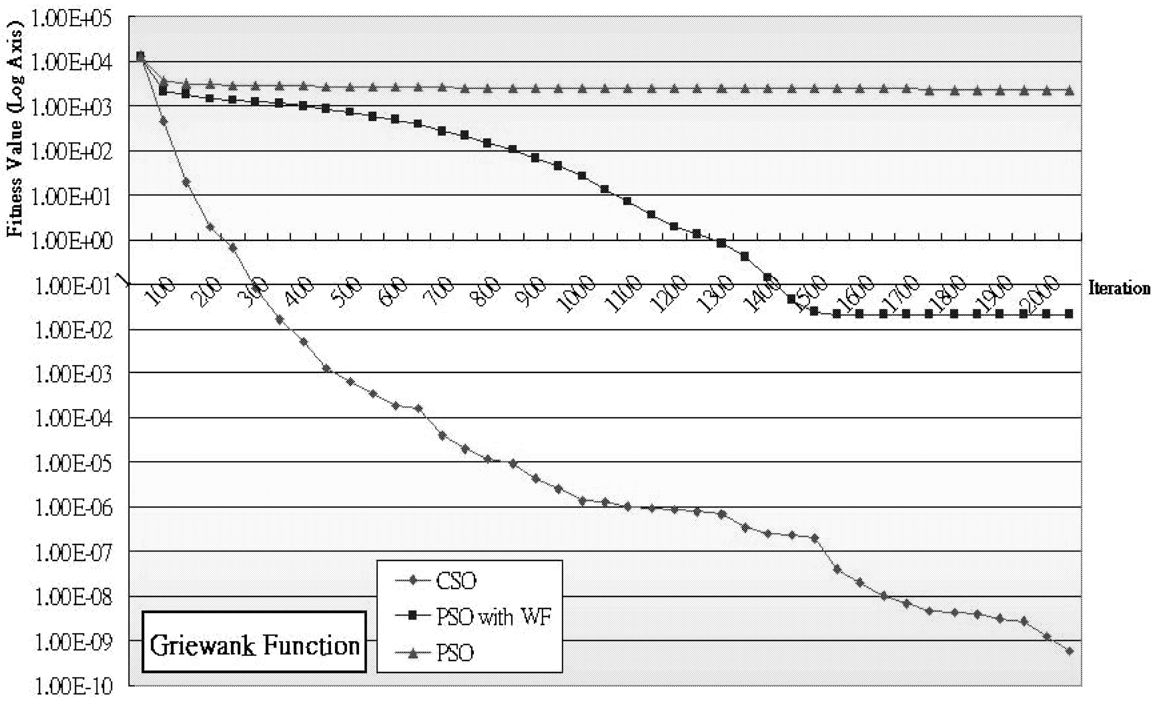

FIGURE 4. The experimental result of test function 1

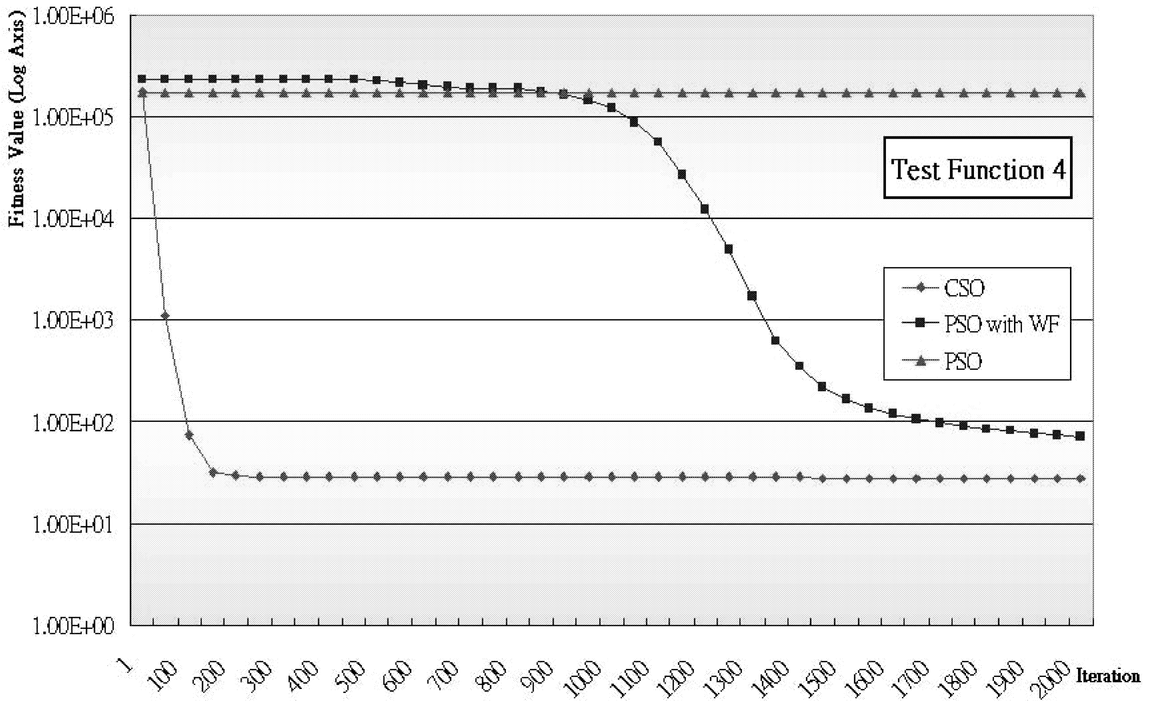

FIGURE 5. The experimental result of test function 1

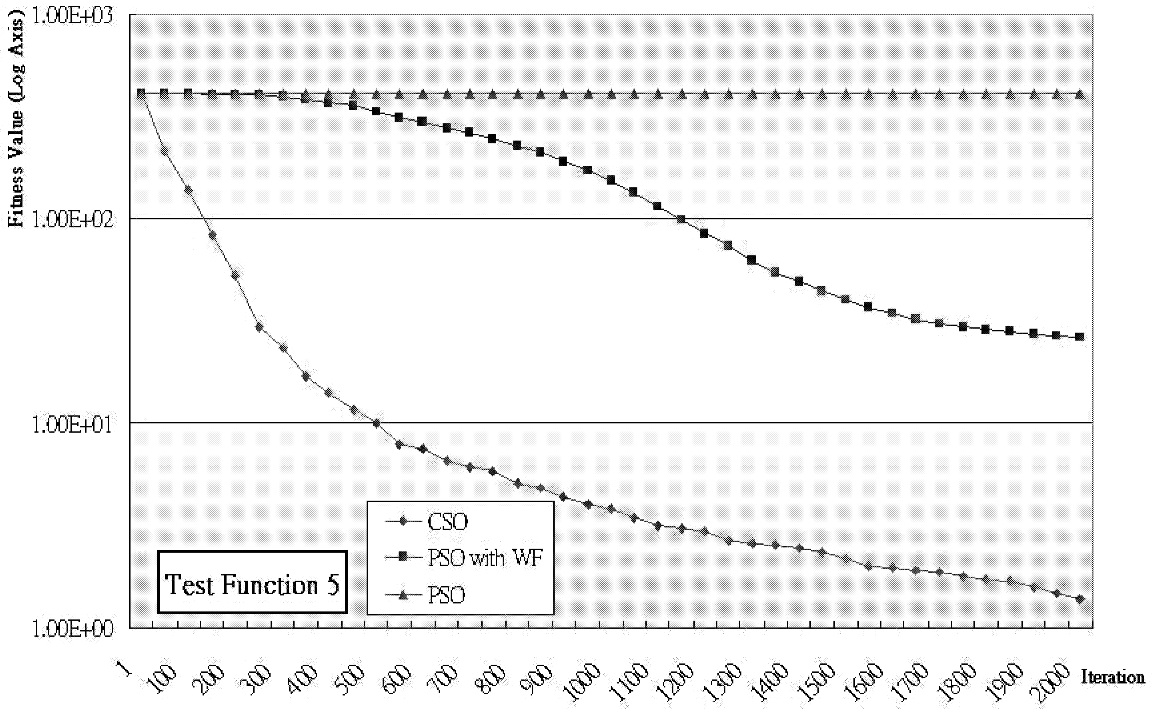

FIGURE 6. The experimental result of test function 1

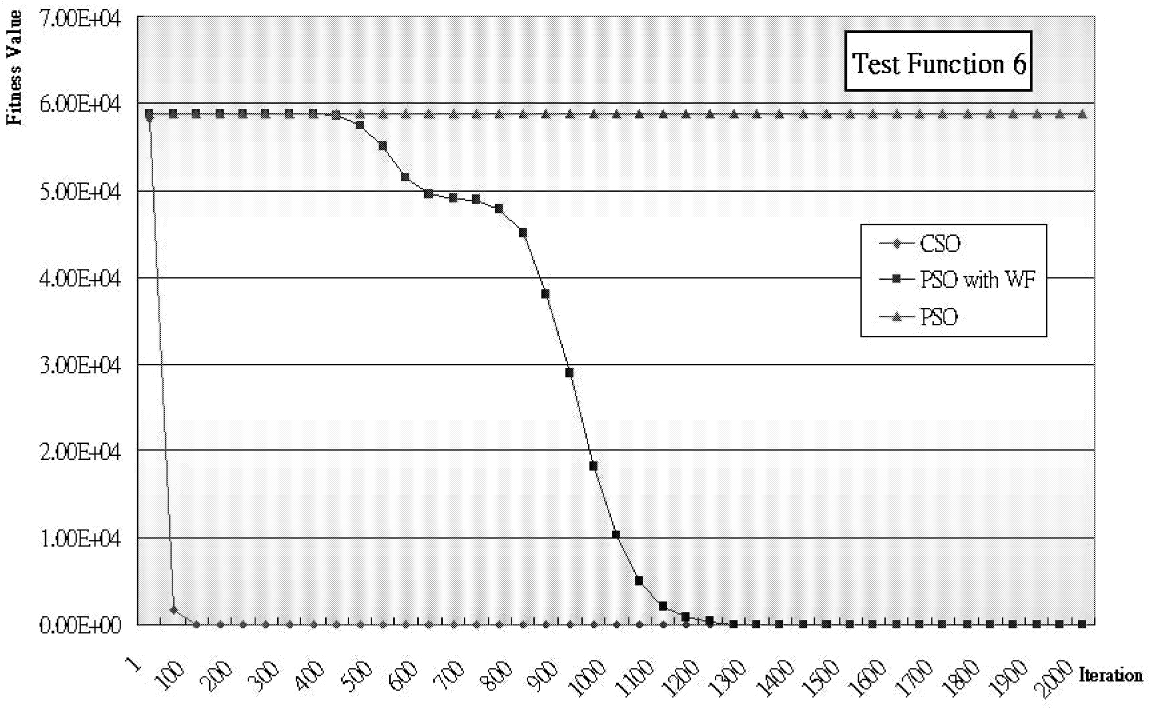

FIGURE 7. The experimental result of test function 1

5. **Conclusion.**
The natural world conceals many characteristics of different creatures, and all of them have some unique behaviors or features to keep them survive. In this paper, we present a new algorithm, Cat Swarm Optimization, through modeling the behaviors of cat to solve the optimization problems.

The experimental results indicate that CSO can better improve the performance on finding the global best solutions. To compare with PSO-type algorithms, CSO avoids the prolix limits, i.e. the maximum velocities, in all iterations. And it can locate the global best solution much faster than PSO-type algorithm.

REFERENCES

[1] Goldberg, D. E., *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company, 1989.

[2] Davis, L., *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991.

[3] Pan, J.-S., F. R. McInnes and M. A. Jack, Application of parallel genetic algorithm and property of multiple global optima to VQ codevector index assignment, *Electronics Letters*, vol.32, no.4, pp.296-297, 1996.

[4] Kim, K.-W., M. Gen and M.-H. Kim, Adaptive genetic algorithms for multi-resource constrained project scheduling problem with multiple modes, *International Journal of Innovative Computing, Information & Control*, vol.2, no.1, pp.41-49, 2006.

[5] Maeda, Y. and Q. Li, Parallel genetic algorithm with adaptive genetic parameters tuned by fuzzy reasoning, *International Journal of Innovative Computing, Information & Control*, vol.1, no.1, pp.95-107, 2005.

[6] Dorigo, M., V. Maniezzo and A. Colorni, The ant system: Optimization by a colony of cooperating agents, *IEEE Trans. on Systems, Man, and Cybernetics -Part B*, vol.149, no.5, pp.379-386, 1996.

[7] Dorigo, M. and L. M. Gambardella, Ant colony system: A cooperative learning approach to the traveling salesman problem, *IEEE Trans. on Evolutionary Computation*, vol.26, no.1, pp.53-66, 1997.

[8] Chu, S.-C., J. F. Roddick and J. S. Pan, Ant colony system with communication strategies, *Information Sciences*, vol.167, pp.63-76, 2004.

[9] Chu, S.-C., J. F. Roddick, C.-J. Su and J.-S. Pan, Constrained ant colony optimization for data clustering, *Proc. of the 8th Pacific Rim International Conference on Artificial Intelligence*, LNAI 3157, pp.534-543, 2004.

[10] Angeline, P. J., Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences, in *Evolutionary Programming VII*, V. W. Porto, N. Saravanan, D. Waagen and A. E. Eiben (eds.), *Lecture Notes in Computer Science*, vol.1447, pp.601-610, 1998.

[11] Shi, Y. and R. Eberhart, Empirical study of particle swarm optimization, *Proc. of the Congress on Evolutionary Computation*, pp.1945-1950, 1999.

[12] Kennedy, J. and R. Eberhart, Particle swarm optimization, *Proc. of the IEEE International Conference on Neural Networks*, pp.601-610, 1995.

[13] Chang, J.-F., S.-C. Chu, J. F. Roddick and J.-S. Pan, A parallel particle swarm optimization algorithm with communication strategies, *Journal of Information Science and Engineering*, vol.21, no.4, pp.809-818, 2005.

[14] Iwasaki, N. and K. Yasuda, Adaptive particle swarm optimization using velocity feedback, *International Journal of Innovative Computing, Information & Control*, vol.1, no.3, 2005.

[15] Kirkpatrick, S., Jr. C. D. Gelatt and M. P. Vecchi, Optimization by simulated annealing, *Science*, pp.671-680, 1983.

[16] Huang, H. C., J. S. Pan, Z. M. Lu, S. H. Sun and H. M. Hang, Vector quantization based on generic simulated annealing, *Signal Processing*, vol.81, no.7, pp.1513-1523, 2001.

[17] Chu, S.-C., P.-W. Tsai and J.-S. Pan, Cat swarm optimization, *Proc. of the 9th Pacific Rim International Conference on Artificial Intelligence*, LNAI 4099, pp.854-858, 2006.

View publication stats