Average-Inertia Weighted Cat Swarm Optimization

Maysam Orouskhani[1,*], Mohammad Mansouri[2] and Mohammad Teshnehlab[3]

[1] Msc Student, Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran.
orouskhani@ce.sharif.edu
[2] Intelligent System Laboratory (ISLAB), faculty of Electrical Engineering, Control department, K.N. Toosi University of Technology, Tehran, Iran.
[3] Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

Abstract. For improving the convergence of Cat Swarm Optimization (CSO), we propose a new algorithm of CSO namely, Average-Inertia Weighted CSO (AICSO). For achieving this, we added a new parameter to the position update equation as an inertia weight and used a new form of the velocity update equation in the tracing mode of algorithm. Experimental results using Griewank, Rastrigin and Ackley functions demonstrate that the proposed algorithm has much better convergence than pure CSO.

Keywords: Cat Swarm Optimization, Average-Inertia Weighted Cat Swarm Optimization, Swarm Intelligence

1 Introduction

Optimization and functions minimization are very important problems. So there are many algorithms to solve these problems. Some of these optimization algorithms were developed based on swarm intelligence by simulating the intelligent behavior of animals, like Ant Colony Optimization (ACO) [1-6] which imitates the behavior of ants, Particle Swarm Optimization (PSO) [2] which imitates the behavior of birds, Bee Colony Optimization (BCO) [3] which imitates the behavior of bees and the recent finding, Cat Swarm Optimization (CSO) [4] which imitates the behavior of cats.
In order to solve the optimization problems, CSO models the behavior of cats into two sub-models namely seeking mode and tracing mode.
In the cases of functions optimization, CSO is one of the best algorithms to find the best global solution. In comparison with other heuristic algorithms such as PSO and PSO with weighting factor [7], CSO usually achieves better result. But sometimes in some cases pure CSO takes a long time to find an acceptable solution. So it affects on performance and convergence of the algorithm. Therefore high speed processor is needed for getting reasonable result.
In this article, our aim is to introduce a new version of CSO in order to improve the performance and achieve better convergence in less iteration. First we add a new parameter to the position equation as an inertia weight that will be chosen randomly

* Corresponding author

(ICSO). Then by making a new form of the velocity equation composing two terms, we introduce Average-Inertia Weighted CSO (AICSO).
Experimental results indicate that the proposed algorithm rather than pure CSO can improve performance on finding the best global solution and achieves better accuracy level of convergence.

2 Cat Swarm Optimization

Chu et al. [4] divided CSO algorithm into two sub-models based on two of the major behavioral traits of cats. These are termed ”seeking mode” and ”tracing mode”. 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 D 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 [5].

A. Seeking Mode

This sub model 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).
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 percent 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 $_k$.

$$P_i = \frac{|SSE_i - SSE_{max}|}{SSE_{max} - SSE_{min}}, \qquad 0 < i < j \tag{1}$$

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

B. Tracing Mode

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

$$v_{k,d} = v_{k,d} + r_1 c_1 \left(x_{best,d} - x_{k,d}\right) \quad (2)$$

Step 3: Update the position of cat_k according to (3)

$$x_{k,d} = x_{k,d} + v_{k,d} \quad (3)$$

C. Core Description of CSO

In order to combine these two modes into the algorithm, we define a mixture ratio (MR) which dictates the joining of seeking mode with tracing mode. The MR decides how many cats will be moved by seeking mode process. For example, if the population size is 50 and the MR is equal to 0.7, there should be 50*0.7=35 cats move to seeking mode and 15 cats move to tracing mode in this iteration. So the process of CSO is summarized below:
First we create N cats and initialize the position, velocities and the flag of every cat. (*) According to the fitness function, evaluate the fitness value of the each cat and keep the best cat into memory. In next step, according to cat's flag, apply cat to the seeking mode or tracing mode process. After finishing the related process, re-pick the number of cats and set them into seeking mode or tracing mode according to MR. In the final step check the termination condition, if satisfied, terminate the program, otherwise go to (*).

3 AICSO Algorithm

In the pure CSO, we should have a condition on the velocity equation in order to control the velocities of the cats for every dimension and check whether the velocities are in the range of maximum or not.
For modifying this part, we use a parameter as an inertia weight to handle this problem. Here we choose the value of inertia weight (w) randomly and experimental

results indicate that it is better to choose w in the range of [0.4, 0.9]. So we select the largest value for w in the first iteration (w = 0.9) and then it will be reduced to 0.4 in the next iterations. CSO with inertia weight can converge under certain conditions even without using v_{max} . For w>1, velocities increase over time, causing cats to diverge eventually beyond the boundaries of the search space. For w<1, velocities decrease over time, eventually reaching 0, resulting in convergence behavior. So the new position update equation can be written as

$$v_{k,d} = wv_{k,d} + r_1 c_1 \left(x_{best,d} - x_{k,d}\right) \tag{4}$$

Where c_1 is acceleration coefficient and usually is equal to 2.05 and r_1 is a random value uniformly generated in the range of [0,1] and w is inertia weight (ICSO).
Next step, we use a new form of the position update equation composing two terms. In the first term, the average information of current and previous position and in the second, the average of current and previous velocity information will be used (AICSO). So new position equation is described below

$$x_{i+1} = \frac{(x_{i+1} + x_i)}{2} + \frac{(v_{i+1} + v_i)}{2} \tag{5}$$

4 Experimental Results

In this paper, simulation experiments are conducted in GCC compiler on an Intel Core 2 Duo at 2.53GHz with 4G real memory. Our test is about finding the global minima of three test functions: Rastrigrin, Griewank and Ackley. In table1 we determine the parameter values of the algorithm and the limitations range of three test functions are mentioned in table2.

Table 1.parameters settings (1)

Parameter	Value or range
SMP	5
SRD	20%
CDC	80%
MR	2%
c_1	2.05
r_1	[0,1]
w	[0.4,0.9]

Table 2.parameters settings (2)

Function	Limitation Range	Object
Rastrigrin	[2.56,5.12]	Minimize
Griewank	[300,600]	Minimize
Ackley	[-32.768,32.768]	Minimize

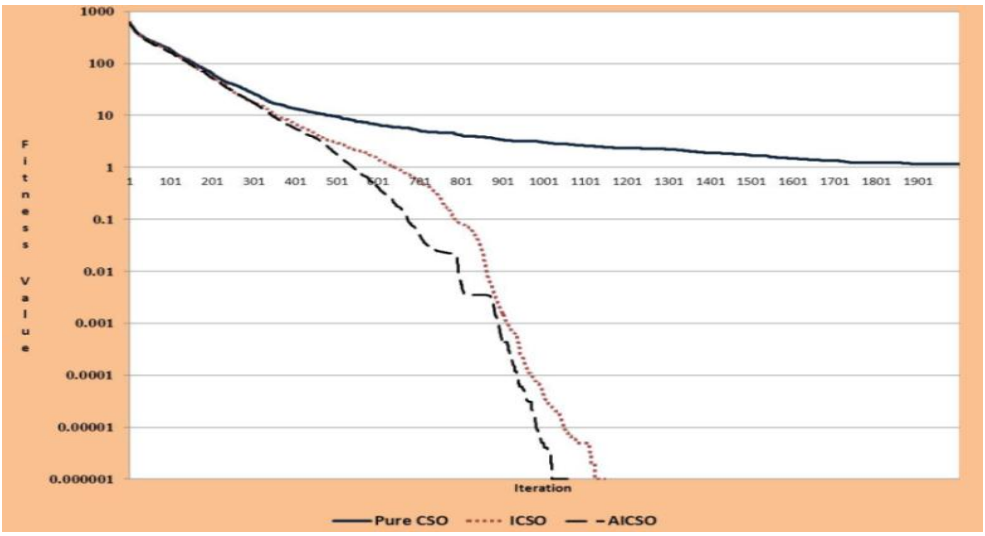

Fig.1.The experimental result of Rastrigin function

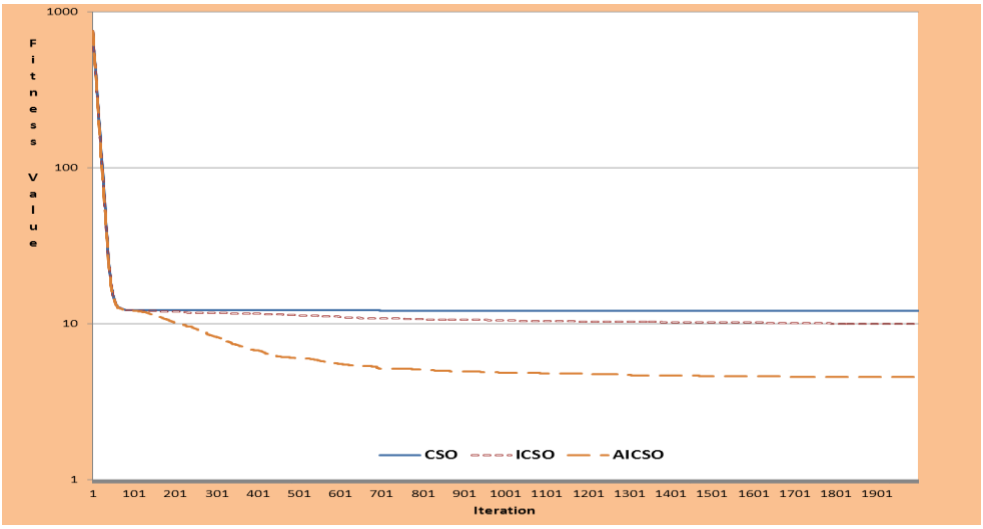

Fig.2. The experimental result of Griewank function

Rastrigin's function is based on the function of DeJong with the addition of cosine modulation in order to produce frequent local minima. Thus, the test function is highly multimodal. Function has the following definition

$$f(x) = \sum_{d=1}^{M} [{x_d}^2 - 10\,cos(2\pi x_d)^2 + 10] \qquad (6)$$

Where -5.12<x<5.12, its global minimum is equal to 0.

Fig.3.The experimental result of Ackley function

As shown in Fig.1, AICSO in the 900th iteration achieves the best solution and its fitness value is so close to the answer of Rastrigin's function. So in comparison with pure CSO and ICSO, AICSO has so much better solution and takes less iteration to converge.

Griewangk's function has many widespread local minima regularly distributed. Where -600<x<600, its global minimum is equal to 0. Function has the following definition

$$f(x) = \frac{1}{4000}\sum_{d=1}^{M} {x_d}^2 - \prod_{d=1}^{M} cos\left(\frac{x_d}{\sqrt{d}}\right) + 1 \qquad (7)$$

As demonstrated in Fig.2, Pure CSO and ICSO have the same fitness value approximately but ICSO can find the better solution and its final solution is 4.57.

Ackley's function is a widely used multimodal test function and Where 32.68<x<32.68, its global minimum is equal to 0. It has the following definition

$$f(x) = -20.\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} {x_i}^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) - 20 + \exp(1) \qquad (8)$$

As shown in Fig.3 AICSO rather than Pure CSO can find the better solution and the best fitness value is 0.000192 in the last iteration.
With comparisons of the average fitness value, numbers of iteration and the best global solution for CSO, ICSO and AICSO which are mentioned in Table3, table4 and

table5, results indicate that AICSO has better performance and usually takes less iteration to converge rather than pure CSO and ICSO. For example, global results of Griewank function are shown in table4. The best fitness values of CSO, ICSO and AICSO are 12.08, 10.01, 4.57 respectively. So it is clear that AICSO has much better result and its solution is more acceptable than CSO and ICSO.

Table 3. Global Results of Rastrigin Function

Algorithm	Average Solution	Iteration	Best Solution
CSO	*26.64208*	*2000*	*1.161127*
ICSO	*22.45198*	*1100*	*0.000000*
AICSO	*21.718749*	*950*	*0.000000*

Table 4. Global Results of Griewank Function

Algorithm	Average Solution	Iteration	Best Solution
CSO	*16.34888*	*2000*	*12.081966*
ICSO	*1485353*	*2000*	*10.017866*
AICSO	*9.937433*	*2000*	*4.572061*

Table 5. Global Results of Ackley Function

Algorithm	Average Solution	Iteration	Best Solution
CSO	*0.38096*	*3500*	*0.004589*
AICSO	*0.370672*	*3500*	*0.000192*

Conclusions

Function minimization is a very important problem in the optimization theory. Cat Swarm Optimization is one of the useful algorithms to solve these problems. But pure CSO in some cases takes a long time to converge and in some problems cannot find the best global solution correctly. So we modified the tracing mode of the algorithm. To do this, we added an inertia weight to the velocity update equation (ICSO) and then we changed the position update equation to a new form using average of current and previous position / velocity information (AICSO). Experimental results for three benchmarks indicated that the proposed algorithm in comparison with pure CSO and ICSO improves the performance on finding the best global solution and achieves the better accuracy level of convergence in the less iteration.

References

1. Dorigo, M., Gambardella, L. M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. on Evolutionary Computation. 26 (1) (1997) 53-66.
2. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. Sixth International Symposium on Micro Machine and Human Science (1995) 39-43.
3. D. Karaboga : An Idea Based On Honey Bee Swarm for Numerical Optimization, TechnicalReport-TR06,ErciyesUniversity, Engineering Faculty, Computer Engineering Department 2005.
4. S. C. Chu, P. W. Tsai, and J. S. Pan, "Cat Swarm Optimization," LNAI 4099, 3 (1), Berlin Heidelberg: Springer-Verlag, 2006, pp. 854– 858.
5. Santosa, B., Ningrum, M.:Cat Swarm Optimization for Clustering. International Conference of Soft Computing and Pattern Recognition.(2009) 54-59.
6. Chu, S. C., Roddick, J. F., Pan, J. S.: Ant colony system with communication strategies. Information Sciences 167 (2004) 63-76.
7. Shi, Y., Eberhart, R.: Empirical study of particle swarm optimization. Congress on Evolutionary Computation. (1999) 1945-1950.
8. Molga, M., Smutnicki, C.: Test functions for optimization needs, 3 kwietnia 2005.