Review on cat swarm optimization algorithms

Conference Paper · November 2013

DOI: 10.1109/CECNet.2013.6703394

2 authors, including:

Pei-Wei Tsai
Swinburne University of Technology
71 PUBLICATIONS **853** CITATIONS

Review on Cat Swarm Optimization Algorithms

Pei-Wei Tsai
Department of Maritime Information and Technology
National Kaohsiung Marine University
Kaohsiung, Taiwan
peri.tsai@gmail.com

Vaci · Istanda
Council of Indigenous Peoples, Executive Yuan
Taipei, Taiwan
biungsu@yahoo.com.tw

***Abstract*—Cat Swarm Optimization (CSO) is an evolutionary method inspired by the creatures in Mother Nature for solving optimization problems. Less than one decade after CSO is proposed, it has been improved and applied in different fields by many researchers in recent years. In this review, we majorly focus on the original CSO algorithm and some improved branches of CSO family algorithms. Some examples of utilizing CSO to solve problems in engineering are also reviewed. In the first section, the concept of the first proposed CSO algorithm is given in brief, and the improved Parallel CSO (PCSO) is described and summarized in a comprehensive way. Finally, the optimization problems in the engineering fields solved by CSO family algorithms are also reviewed. The objective of the review is to provide the overview of CSO algorithms and a brief introduction for new re-searchers to the swarm intelligence research field.**

Keywords—Cat swarm optimization; Swarm intelligence; Bio-inspired algorithm; Evolution-ary computing.

I. INTRODUCTION

Examples of using swarm intelligence methods to solve engineering problem can be found everywhere in different fields. Many algorithms in swarm intelligence are developed by simulating the behaviors of the creatures in Mother Nature. For instance, by taking felid as the model, Chu et al. propose Cat Swarm Optimization (CSO) [5-6] for solving numerical optimization problems; Pan et al. pro-pose Fish Migration Optimization (FMO) [9] in 2010, and Tsai propose Evolved Bat Algorithm (EBA) [15] in 2012.

There are two ways to employ CSO and its family algorithms to solve problems in engineering: the first way is to describe the problem in mathematic formula and exploit the algorithm, directly, to find the optimum solution. [2, 10-13, 16] An-other way is to combine CSO family algorithms with other existing methods or systems such as Artificial Neural Network (ANN) [4, 8, 17] and Simulated Annealing [7]. The rest of the article is organized as follows: we first introduce the original CSO [5-6]. It is followed by the parallel version of CSO algorithm called PCSO [14]. Finally, the brief review on examples of employing CSO to solve problems in the engineering field are summarized at the end.

II. CAT SWARM OPTIMIZATION (CSO)

CSO [5-6] is proposed by Chu et al. by imitating the natural behaviors of cats. It has two modes called the seeking mode and the tracing mode for moving the virtual cats in the solution space. The number of agents participating in seeking mode and tracing mode in one iteration is fixed on a predefined ratio called MR. The process of CSO can be disassembled into 5 steps:

Step 1. Initialization: Create N virtual cats and randomly spread them into the M-dimensional solution space. Generate the velocities for every dimension and select part of the cats pending them to be processed in the tracing mode according to MR, where $MR \in [0,1]$.

Step 2. Evaluation: Calculate the fitness value of each virtual cat with the fitness function and check whether current coordinate present better fitness value then its memorized best solution. The memorized best solution should be updated, accordingly, when the current fitness value is better than the memorized one.

Step 3. Movement: Move the virtual cats in the solution space with the seeking/tracing mode according to the decision made in step 1. The seeking mode and the tracing mode process are described in the next paragraph.

Step 4. Permutation: Repick $(N \times MR)$ virtual cats pending to be processed in the tracing mode in the next iteration.

Step 5. Check whether the process satisfies the termination condition. If the process is terminated, output the coordinate representing the near best solution and stop the program. Otherwise, go back to step 2.

When a virtual cat is processed by the seeking mode, its movement would be slow and conservative. Four essential factors are defined in the seeking mode:

1. Seeking Memory Pool (SMP): SMP defines the seeking memory size for each cat to consider the candidate coordinates.
2. Seeking Range of the selected Dimension (SRD): SRD is the mutative ratio, which provides the boundary condition of changing, for the selected dimensions.
3. Counts of Dimensions to Change (CDC): CDC indicates the number of dimensions to be varied.
4. Self-Position Considering (SPC): SPC decides whether the current coordinate of the virtual cat will also be considered as the candidate in the seeking memory.

The process of the seeking mode is reviewed as follows:

978-1-4799-2860-6/13/$31.00 ©2013 IEEE

Step 1. Generate *j* copies of cat_k , where *j* is defined by Eq. (1):

$$j = \begin{cases} SMP, & SPC = "true" \\ SMP - 1, & SPC = "false" \end{cases} \quad (1)$$

Step 2. Plus or minus *SRD* percent of current value on the selected dimensions by Eq. (2) to Eq. (4):

$$M = Modify \cup (1 - Modify) \quad (2)$$

$$|Modify| = CDC \times M \quad (3)$$

$$x_{jd} = \begin{cases} x_{jd}, & d \notin Modify \\ (1 + rand \times SRD) \times x_{jd}, & d \in Modify \end{cases}\Bigg|_{d=1,2,.....,M} \forall j \quad (4)$$

where *Modify* denotes the elements of the selected candidate dimensions for the modification, *rand* is a random variable and $rand \in [0,1]$.

Step 3. Calculate the fitness values for all candidates.

Step 4. Calculate the selecting probability P_i for each candidate by Eq. (5):

$$P_i = \begin{cases} 1, & \text{when } FS_{\max} = FS_{\min} \\ \dfrac{|FS_i - FS_b|}{FS_{\max} - FS_{\min}}, & \text{where } 0 < i < j, \text{otherwise} \end{cases} \quad (5)$$

Let $FS_b = FS_{max}$ when the goal of the fitness function is to find the minimum solution. Otherwise, let $FS_b = FS_{min}$, where FS_{max} and FS_{min} indicate the largest and the smallest *FS* presented in the candidates, respectively.

Step 5. Sort the candidates by P_i and select one of them to replace the coordinate of cat_k by roulette wheel selection.

The tracing mode allows the virtual cats imitating the movement of tracing the prey. The process of the tracing mode can be disassembled in 3 steps:

Step 1. Velocity update: Update the velocities on each dimension by Eq. (6):

$$v_{k,d} = v_{k,d} + r \cdot c \cdot (x_{best,d} - x_{k,d}), \ d = 1,2,.....,M \quad (6)$$

where $v_{k,d}$ indicates the velocity of cat_k on the d^{th} dimension, x_{best} is the coordinate with the near best solution found over all virtual cats, $x_{k,d}$ is the coordinate of cat_k, *c* is a constant and *r* is a random variable in the range of [0,1].

Step 2. Check whether the updated velocities exceed the maximum velocity limit. The velocity, which is greater than the maximum velocity should be set to equal to the maximum velocity.

Step 3. Update the coordinate of the virtual cat by Eq. (7):

$$x_{k,d} = x_{k,d} + v_{k,d} \quad (7)$$

III. PARALLEL CAT SWARM OPTIMIZATION (PCSO)

One year after CSO is proposed, Tsai et al. propose PCSO [14] with an information exchanging process. Moreover, Tsai et al. modify the involved element in the tracing mode process in PCSO. The strong points of parallelizing the virtual agents in evolutionary algorithms are pointed out in precedents [1, 3].

In PCSO, the virtual cats share the isolated near best solution between different clusters via the information exchanging process. Since the foundation of PCSO is as the same as CSO, we only review the information exchanging process and the particular parallel tracing mode in this section.

A. Information Exchanging Process

The near best solutions in different clusters may have chance to be copied into other clusters. A parameter called ECH is defined in order to launch the infor-mation exchanging process. Thus, in PCSO, the information exchanging process is involved every ECH iterations. This process can be described in 3 steps:

Step 1. Sort the virtual cats for every clusters by their fitness values.

Step 2. Randomly pick a near best solution from all clusters and replace the virtual cat, which has the worst fitness value in the cluster. But the near best solution and the virtual cat should not come from the same cluster.

Step 3. Repeat step 2 for all clusters.

B. Parallel Tracing Mode Process

In PCSO, the virtual cats are divided into isolated clusters. It means that cats in different clusters should not be effected by other cats in other clusters. Thus, they can be treated as groups of small-scale CSO clusters. The virtual cats in different clusters should only share their own near best solution. In the parallel tracing mode process, the velocity update step should follow Eq. (8) instead of Eq. (6):

$$v_{k,d} = v_{k,d} + r \cdot c \cdot (x_{l_{best},d} - x_{k,d}), \ d = 1,2,.....,M \quad (8)$$

where $x_{l_{best},d}$ indicates the coordinate of the near best solution in one cluster.

IV. APPLICATIONS

In this section, we give review on two representative works that utilizing CSO or CSO family algorithms in different fields. The first example is using CSO to optimize the hidden points on the carrier media for information hiding proposed by Wang et al. in 2012. [16] The other example is utilizing a newly advanced hybrid CSO method called Enhanced Parallel Cat Swarm Optimization (EPCSO) [13] to solve the problem of aircraft schedule recovery in a limited process time.

A. CSO Supported Information Hiding

In digital watermarking, the secret message is embedded into the carrier media to avoid the detection or be seen by other people. In this case, the carrier media is the digital image, and the secret message is encoded to be a stego-image. When evaluating the quality of an encoded or compressed image, Mean Square Error (MSE) is a general index we want to evaluate with. The chose information hiding method has one characteristic: MSE between the stego-image and the carrier image would be "0" is the encoding parameter perfectly match the encoded result with the carrier; otherwise MSE will be increased when the embedding procedure causes more modification on the carrier image. Moreover, MSE value of the carrier image would decrease when the secret message is embedded into the carrier. The more modification caused by the embedded message, the lower MSE value is presented.

However, by properly selecting the encryption parameters, the MSE value may be able to raise without shrinking the size of the embedded secret message. In addition, when the size of the secret message increases, the solution space of different parameter combination is also increased with a large-scale. Doing full search to find the best solution is not feasible. Thus, CSO is exploited to find the proper location on the carrier image and the optimum parameter combination for embedding the secret message.

The result presented by Wang et al. indicates that with CSO's support, the secret is able to be embedded with an acceptable MSE value in a limited computational time. Their result is adopted in Table I and Table II.

TABLE I. MSE BETWEEN LENA AND STEGO-IMAGE. (ADOPT FROM [16])

	Jet	Tiff
(a) The secret images are 256 × 256		
CSO supported method	2.2394	2.2909
Exhaustive LSB substitution method	2.2394	2.2909
The Optimal substitution	2.2349	2.2909
(b) The secret images are 256 × 512		
CSO supported method	33.6979	34.0179
(Computation time: s)	125	127
Exhaustive LSB substitution method	Not available	Not available
The Optimal substitution	33.3233	33.2183

TABLE II. MSE BETWEEN BABOON AND STEGO-IMAGE. (ADOPT FROM [16])

	Jet	Tiff
(a) The secret images are 256 × 256		
CSO supported method	2.2339	2.2963
Exhaustive LSB substitution method	2.2339	2.2963
The Optimal substitution	2.2339	2.2963
(b) The secret images are 256 × 512		
CSO supported method	33.3411	33.6139
(Computation time: s)	78	126
Exhaustive LSB substitution method	Not available	Not available
The Optimal substitution	33.3242	33.2026

B. EPCSO for Aircraft Schedule Recovery

The second example is utilizing Enhanced Parallel Cat Swarm Optimization (EPCSO) [13], which is a hybrid method branch of CSO, to solve the aircraft schedule recovery problem. The weather in Taiwan is known as the insular cli-mate. Thus, the typhoon usually comes and attack Taiwan during summer and fall. If the influence of the typhoon reaches a level that becomes a threat to the safety issue, the airport must close, temporary. In that case, the aircraft schedule must be affected and is necessary to be recovered in a limited time after the airport is back into the service. In addition, some unexpected events or the dysfunctional machines may also cause the closure of the airport. In Tsai et al.'s work, EPCSO is employed to find the optimum solutions in the recovered aircraft schedule. Five conditions including passengers' waiting time, aircraft change cost, etc. are con-sidered in this optimization problem. The experimental result from Tsai et al. is summarised in Table III.

TABLE III. EXPERIMENTAL RESULTS OF EPCSO FOR AIRCRAFT SCHEDULE RECOVERY.

	Value
Total delay time	595 (minutes)
No. of long delay aircraft (delay more than *30* minutes)	5 (flights)
Average iteration no. to find the optimum solution	24 (iterations)
STD of the statistical total delay time for every schedule	1.400×10^{1}
STD of the final fitness value over 25 runs	4.547×10^{-13}

V. DISCUSSION AND CONCLUSION

In this review, two CSO family algorithms are summarized in brief. Moreover, two application examples are also reviewed above. In section IV, Wang et al. utilize CSO to tune the parameter for hiding secret information in to the carrier. The number of possible solutions of the information hiding method they choose is highly related with the size of the secret message.

Hence, the number of possible solution increases to a titanic size, which is too large for full search method to find solutions in a feasible time. According to the results given in Table I and Table II, the Exhaustive LSB substitution method and the CSO supported method can both find the optimal solution when the stego-image is with size 256×256. However, the Exhaustive LSB substitution method fails to produce the solution when the size of the stego-image grows up to 256×512. Even though the size of the stego-image is increased, CSO supported method still produce the acceptable solution (not the optimal solution, but very close to it) in about 2 minutes.

Another example given in section IV is using EPCSO to solve the aircraft schedule recovery problem is reviewed. Tsai et al. improves the PCSO algorithm by adopting the Taguchi method process in the enhanced parallel tracing mode. Table III shows the statistical results of the optimum solutions found by EPCSO. By balancing five considered conditions, the shortest total delay time is 595 minutes over 38 flights. Only 5 flights are delayed relatively longer, i.e., more than 30 minutes. The same experiment is rerun 25 times in order to test the stability of EPCSO's outcome. Although the recovered aircraft schedule is not always the same in every run, the standard deviation (STD) of the fitness value over all runs is quite puny. It implies that the results produced by EPCSO in this application is quite stable.

In general, the evolutionary algorithm such as CSO usually requires recursive computing ability. It implies that the computation process of CSO needs lots of memory space to store the temporary computational result. Thus, swarm intelligence algorithms are usually implemented on personal computers or servers. It's nearly difficult to find swarm intelligence being implemented on the embedding system or some small devices with low computational power. In the future re-search, focusing on reducing the computational complexity of the CSO family algorithms may also be a contributive way for improving the algorithm.

REFERENCES

[1] Abramson, D., and Abela, J., "A Parallel Genetic Algorithm for Solving the School Timetabling Problem," Technical Report, Division of Information Technology, CSIRO, 1991.

[2] Benala, T. R., Satapathy, S. C., Surya Vamsi S. G. S., and Panchumarthy, R., "Cat Swarm Optimization for Optimizing Hybridized Smoothing Filter in Image Edge Enhancement," In Proceedings of International Conference on Systemics, Cybernetics and Informations, 2011, pp. 247-252.

[3] Chang, J.-F., Chu, S.-C., Roddick, J. F., and Pan, J.-S., "A Parallel Particle Swarm Optimization Algorithm with Communication Strategies," Journal of Information Science and Engineering, vol.21, no.4, 2005, pp.809-818.

[4] Chittineni, S., Abhilash, K., Mounica, V., Sharada, N., and Satapathy, S. "Cat Swarm Optimization Based Neural Network and Particle Swarm Optimization Based Neural Network in Stock Rates Prediction," In Proceedings of 2011 the 3rd International Conference on Machine Learning and Computing, 2011, pp. V4-292-V4-296.

[5] Chu, S.-C. and Tsai, P.-W., "Computational Intelligence Based on the Behavior of Cats," International Journal of Innovative Computing, Information and Control, vol. 3, no. 1, 2007, pp. 163-173.

[6] Chu, S.-C., Tsai, P.-W., and Pan, J.-S., "Cat Swarm Optimization," In Proceedings of the 9th Pacific Rim International Conference on Artificial Intelligence, LNAI 4099, 2006, pp. 854-858.

[7] Liu, Y., Wu, X., and Shen, Y., "Cat Swarm Optimization Clustering (KSACSOC): A Cat Swarm Optimization Clustering Algorithm," Scientific Reserach and Essays, vol. 7(49), 2012, pp. 4176-4185.

[8] Orouskhani, M., Mansouri, M., and Teshnehlab, M., "Average-Inertia Weighted Cat Swarm Optimization, In: Advances in Swarm Intelligence," Lecture Notes in Computer Science (LNCS 6782), 2011, pp. 321-328.

[9] Pan, J.-S., Tsai, P.-W., and Liao, B.-Y., "Fish Migration Optimization Based on the Fishy Biology," In Proceedings of 2010 4th International Conference on Genetic and Evolutionary Computing, 2010, pp. 783-786.

[10] Panda, G., Pradhan, P. M., and Majhi, B., "IIR System Identification Using Cat Swarm Optimization," Expert Systems with Applications, vol. 38, 2011, pp. 12671-12683.

[11] Pardhan, P. M., and Panda, G., "Solving Multiobjective Problems Using Cat Swarm Optimization," Expert Systems with Applications, vol. 39, 2012, pp. 2956-2964.

[12] Saha, S. K., Ghoshal, S. P., Kar, R., and Mandal, D., "Cat Swarm Optimization Algorithm for Optimal Linear Phase FIR Filter Design," ISA Transactions, *in Press*, 2013.

[13] Tsai, P.-W., Pan, J.-S., Chen, S.-M., and Liao, B.-Y., "Enhanced Parallel Cat Swarm Optimization Based on the Taguchi Method," Expert Systems with Applications, vol. 39, 2012, pp. 6309-6319.

[14] Tsai, P.-W., Pan, J.-S., Chen, S.-M., Liao, B.-Y., and Hao, S.-P., "Parallel Cat Swarm Optimization," In Proceedings of the 7th International Conference on Machine Learning and Cybernetics, 2008, pp. 3328-3333.

[15] Tsai, P.-W., Pan, J.-S., Liao, B.-Y., Tsai, M.-J., and Istanda, V., "Bat Algorithm Inspired Algorithm for Solving Numerical Optimization Problems," Applied Mechanics and Materials, vol. 148-149, 2012, pp. 134-137.

[16] Wang, Z.-H., Chang, C.-C., and Li, M.-C., "Optimizing Least-significant-bit Substitution Using Cat Swarm Optimization Strategy," Information Science, vol. 192, 2012, pp. 98-108.

[17] Yusiong, J. P. Y., "Optimizing Artificial Neural Networks Using Cat Swarm Optimization Algorithm," International Journal of Intelligent Systems and Applications, vol. 01, 2013, pp. 69-80.