Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

A Clustering-based Adaptive Parameter Control Method for Continuous Ant Colony Optimization

Yue-jiao Gong, Rui-tian XU and Jun Zhang
Department of Computer Science
SUN yat-sen University
Guangzhou, P.R.China, 510275
junzhang@ieee.org

Ou Liu
The School of Accounting and Finance
The Hong Kong Polytechnic University
Kowloon, Hong Kong
Ou.Liu@inet.polyu.edu.hk

***Abstract*—Ant colony optimization (ACO) has been widely and successfully applied to NP-hard combinatorial optimization problems for its strong searching ability and robustness. Recently, several extended ACO algorithms have also been proposed to deal with continuous optimization problems. However, the ACO algorithms always have slow convergence speed and encounter premature convergence in engineering applications. This paper proposes a novel adaptive parameter control method for continuous ACO algorithms. Clustering analysis is used to judge the optimization state of the algorithm and the flexible adjustment of the parameters is based on these optimization states during the training process. As an example, the adaptive control method is used to improve the performance of the continuous orthogonal ant colony (COAC). Experimental results demonstrate that the clustering-based adaptive parameters control scheme contributes to both faster convergence speed and higher solution accuracy. The proposed adaptive control method has great practical value and bright prospect.**

Keywords—*ant colony optimization (ACO), continuous orthogonal ant colony (COAC), adaptive parameter, clustering analysis*

I. INTRODUCTION

Ant colony optimization (ACO) was first proposed in 1990s for solving NP-hard combinatorial optimization problems. Because of its strong searching ability and robustness, ACO has been widely and successfully applied in solving problems such as the traveling salesman problem [1][2], scheduling problem [3][4], protein folding problem [5], and other problems [6][7]. Recently several extended ACO algorithms have also been proposed to deal with continuous optimization problems [8][9].

However, the ACO algorithms, especially the continuous ACO algorithms, always have slow convergence speed and encounter premature convergence. The parameter setting strategies for ACO algorithms are deficient. In order to overcome these shortcomings, many researchers have made great efforts to study on the parameter settings, and have proposed some adaptive parameter control schemes[10]-[13]. Dorigo and Gambardella [2] put forward a method for choosing the proper number of ants m based on the value of the pheromone decay parameter ρ and the probability for performing exploitation q_0 in their ant colony system. Zeng and He [10] designed a formula to reduce the q_0 during the training process in order to strengthen the random node selecting probability. Cai and Huang [11] proposed an automatic setting method to adjust the ρ and the relative weight of heuristic value β according to the amount of pheromone and the solution value. In the paper of Qin *et al.* [12] the adjustment of the relative weight of pheromone trail α and β is based on a pheromone distributing weight.

However, most of the previous studies only concentrate on finding the parameter control strategies based on the conventional ACO which deals with the combinatorial optimization problems. They are not applicable for continuous ant colony algorithms. Moreover, almost all the previous schemes use deterministic strategies to adjust the parameters. They are not adaptive to different evolutionary environments, resulting in inefficient optimization when dealing with complex problems. Therefore, research into adaptively controlling the parameters of ACO, especially the continuous ACO, has become significant and promising.

In this paper, a clustering-based adaptive parameter control scheme for continuous ant colony algorithms is proposed. Clustering analysis [14] is used to judge the optimization state of the algorithm. Flexible adjustment of the parameters is based on the optimization states during the training process. The proposed scheme has advantages of robustness, versatility and effectiveness. As an example, the scheme is used to improve the performance of the continuous orthogonal ant colony (COAC) [9], and the experimental results show that it contributes to the COAC with faster convergence speed and higher solution accuracy.

The rest of this paper is organized as follows. The next section presents brief review on the COAC. In Section III, we introduce the clustering-based adaptive parameter control scheme in detail. Then, in Section IV, seventeen benchmark functions are tested, experimental results are reported. At last, conclusions are drawn in Section V.

II. BRIEF REVIEW ON THE COAC ALGORITHM

The COAC was proposed by Hu *et al*. [9] as an extension of the ACO algorithm in order to solve the continuous optimization problems. In COAC, the continuous domain is discretized into μ regions, m ants are distributed to several of them. The principle of the problem is how to direct these ants to find the best region and the optimal point.

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

In COAC, every region has four properties, the coordinate of its center, the searching radiuses, the amount of pheromone, and the ranks of its desirability [9]. It works as follows: first, μ regions are randomly generated; m ants are randomly positioned on these regions. Then, each ant chooses its next region according to the amount of pheromone, and performs orthogonal exploration in the region. Once all the ants have finished a round of state transition and orthogonal exploration, the pheromone on the visited regions is updated by applying a global modulation rule. An overall flowchart of COAC is illustrated as Fig. 1, where *MAXFES* is predefined to control the number of iterations.

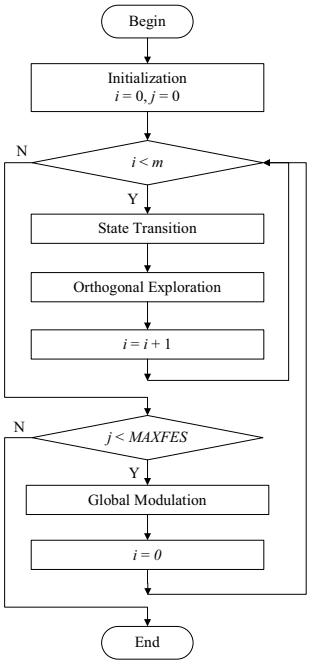

Figure 1. Flowchart of COAC

A. State Transition

Each ant chooses its next region for exploration by applying the state transition rule. That is, the next region j is given by

$$j = \begin{cases} j \,\big|\max_{j\in S_R} \tau_j, & q \le q_0 \quad \text{(exploitation)} \\ RWS, & q > q_0 \quad \text{(biased exploration)} \end{cases}, \qquad (1)$$

where S_R is the set of regions, τ_j is the pheromone value in region j, q is a random number uniformly distributed in [0, 1]. q_0 is a parameter which controls the probability of whether to exploit the current optimal region (exploitation) or to search the whole domain (biased exploration). *RWS* stands for the roulette wheel selection based on the pheromone value. When exploration is used, the probability of choosing the region j is

$$p(j) = \frac{\tau_j}{\sum_{i\in S_R} \tau_i}, \quad j \in S_R . \qquad (2)$$

The state transition rule makes ants tend to search the regions with large amount of pheromone.

B. Orthogonal Exploration

When an ant reaches a region, it performs orthogonal exploration to search better points in the region. The coordinate and radius of the region are updated according to the exploration result.

Use an orthogonal array L (N, k, s), where N is the number of orthogonal combinations which stands for how many points in the region are explored, k is the number of orthogonal factors which must satisfy $k \ge n$ (in order to choose n columns without duplication as a new orthogonal array), and s stands for each factor has s levels. Here n is the dimension of the domain.

Suppose the current center of the region is $\boldsymbol{x}_j = (x_{j1}, x_{j2} \ldots x_{jn})$, and the current radius is $\boldsymbol{r}_j = (r_{j1}, r_{j2} \ldots r_{jn})$. The orthogonal exploration for an ant is defined as follows [9]:

Step 1: Randomly choose n different columns in L (N, k, s) as a new orthogonal array.

Step 2: Generate N neighboring points $\boldsymbol{x}_j^i = (x_{j1}^i, x_{j2}^i \ldots x_{jn}^i)$, i =1,2,…,N by the orthogonal design method where

$$\begin{aligned} & x_{jk}^i = x_{jk} + [\frac{(L_{ik}-1)\cdot 2}{s-1} - 1] \cdot r_{jk} \cdot rand_j , \\ & \text{if} \quad x_{jk}^i < lbound_k , \quad x_{jk}^i = lbound_k , \quad . \\ & \text{if} \quad x_{jk}^i > ubound_k , \quad x_{jk}^i = ubound_k , \\ & \qquad k = 1, 2, ..., n \end{aligned} \qquad (3)$$

L_{ik} is the level of the k-th column in combination i for the chosen orthogonal array, and $rand_j$ is a uniform random number in [0, 1]. Once a better point is found, move the center coordinate of this region to the point, so that more optimal region can be obtained.

Step 3: Adjust the radius by applying the rule given below.

If a better point has been found, expand the radius of the region to search for more diversity [9].

$$r_{jk} = r_{jk} / shrink, \quad k=1,2,...,n . \qquad (4)$$

Otherwise, if no better point is found by the orthogonal exploration, shrink the region to achieve a higher sensitivity [9].

$$r_{jk} = r_{jk} \cdot shrink, \quad k=1,2,...,n . \qquad (5)$$

Here *shrink* is a pre-defined parameter in (0, 1].

C. Global Modulation

After all m ants have finished this round of state transition and orthogonal exploration, a global modulation rule is proposed to update the pheromone value for each region in the domain. The desirability of every region is judged, some undesirable regions are discarded.

The best region in S_R is assigned with rank 1, then, the second-best region is assigned with rank 2, and so on, until $\varphi \times \mu$ regions are marked, where $\varphi \in [0,1]$ is the elitist rate. In each iteration, only $\varphi \times \mu$ regions are reserved, and the pheromone on the selected region j is updated by

$$\tau_j = (1-\alpha) \cdot \tau_j + \alpha \cdot T_0 \cdot (\varphi \cdot \mu + 1 - rank_j + visit_j) , \qquad (6)$$

where $\alpha \in (0,1)$ is the pheromone decay parameter, T_0 is the initial pheromone value and $visit_j$ is the number of times that region j has been visited. The better the region is and the more times it has been visited, the more pheromone is deposited in the region [9].

At last, other unmarked regions are re-initialized.

Although the COAC is useful for discovering the optimal solution for continuous problems, the total iteration number required to find the optimum is unsatisfactory. Therefore, we devise an adaptive parameters control method to improve its performance, which is presented in the next section.

III. Adaptive Parameters Control Method

In conventional ACO, the parameter values such as q_0 and α are fixed. However, the requirement of these values depends on the optimization state during the convergence process, so that the training efficiency by using those fixed parameters may deteriorate. Moreover, for different cases, different parameter settings are required. It brings in a major burden of specifying the value of these parameters. In order to overcome these shortcomings, using adaptive parameter control schemes are promising.

In this paper, a clustering-based adaptive parameter control method is proposed to improve the efficiency of continuous ant colony algorithms, and the COAC is taken for example. The adaptive process is partitioned into two parts: the clustering to judge the optimization state and the adjustment of the parameters. The procedures are described as follows.

A. Judging the Optimization State

In COAC, each region stands for a candidate solution, therefore the training process of the algorithm is the process of all the regions in the domain gathering to the region which represents the optimum solution. Thus, the optimization state can be judged by measuring the current distribution of regions.

By applying a K-means algorithm [14], the regions in the domain is clustered into several groups according to the distribution information in each iteration. The clustering process is defined as below, here K is the number of clusters.

Initialize K clusters $C_1, C_2 \ldots, C_K$.
Each cluster is associated with a cluster center c_j (j=1,2, …, K).
Loop
For each region p_i (1,2, …, μ) in the domain, calculate the distance between p_i and c_j (j=1,2, …, K, $j \neq i$), then assign p_i to its closest cluster.
For each cluster C_j, count the number of regions n_j and compute new cluster center by:

$$c_j = \frac{1}{n_j} \cdot \sum_{p_i \in C_j} p_i, \quad j = 1,2,...,K . \qquad (7)$$

Until End condition.

After the clustering process, the cluster containing the best region is defined as C_{best}, while the cluster containing the worst region is defined as C_{worst}. Based on the size of C_{best} and C_{worst}, the convergence process is divided into three optimization states, as Table I shows. If C_{worst} is the largest cluster, the current process is considered to be in the initial state. Otherwise, if C_{best} is the largest cluster, the current process is considered to be in the matured state. Otherwise, if C_{best} and C_{worst} are both small, the current process is considered to be in the maturing state.

TABLE I. Optimization State

Initial State	Maturing State	Matured State
C_{best} is small C_{worst} is the largest,	C_{best} is small, C_{worst} is small	C_{best} is the largest, C_{worst} is small

B. Adjustment of the Parameters

Adjustment of the parameters is based on the training state. In COAC, there are four parameters including q_0, α, φ and *shrink* can be adjusted. q_0 determines the probability of exploitation towards biased exploration. α is the pheromone decay parameter. φ is the elitist rate, in each iteration only $\varphi \times \mu$ regions are reserved. *shrink* is a parameter that controls the expansion and shrinking rate of the regions. In this paper, the adaptive control strategies for the parameters q_0, φ, and *shrink* are proposed. The value of α is fixed and equal to 0.1, because experimental studies show that, the value of α has little impact on the efficiency in COAC.

Table II and Fig. 2 depict the adaptive control schemes for different optimization states. Constantly adjust the control parameters accroding to the optimization state makes the parameters meet the need of the evolutionary process better. The searching course of the algorithm can be more flexible and intelligent.

TABLE II. Adaptive Control Scheme

Optimization State	q_0	φ	*shrink*
Initial State	–	–	=0.3
Maturing State	+	–	+
Matured State	+	=0.4	=0.3

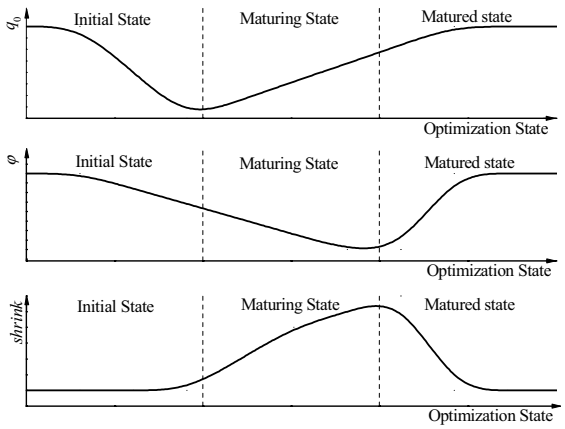

Figure 2. Adaptive Control Scheme

In the initial state, good individuals in the population are very few, the searching direction is undetermined. In order to search in the whole domain and establish an initial searching direction, the value of q_0 and φ are reduced, and the *shrink* is set small to increase the flexibility of the orthogonal exploration.

In the maturing state, the number of good individuals in the population is a little more. So, increase the value of q_0 step by step to guide the search. However, the value of φ should still be reduced in order to prevent the COAC from getting trapped at a local optimum. Because in this optimization state the probabilities of finding better points during the orthogonal exploration is very high, *shrink* should be increased in order to keep the search steady.

In the matured state, most of the regions in the domain have swarmed together near the currently best region, the searching direction is determined. So increase the values of q_0 and φ in order to speed up the convergence. Reduce the *shrink* to make the final orthogonal search more diverse.

These measures can realize the twin goals of maintaining diversity in the domain and enhancing the convergence capability. The comparisons and analysis between the COAC with and without the proposed adaptive method are presented in the next section.

IV. EXPERIMENTAL STUDIES

In this paper, an adaptive parameters control scheme for continuous ant colony algorithms based on clustering analysis is proposed. As an example, it is used to improve the performance of the COAC algorithm [9]. In the following, the clustering-based adaptive COAC is abbreviated to CCOAC. In order to compare the performance of CCOAC with COAC, fourteen benchmark functions [14] are used.

A. Benchmark Functions

The fourteen benchmark functions are given in Table III. f_1-f_6 are unimodal functions. f_7-f_{14} are multimodal functions, among which f_{14} are rotated multimodal functions. The dimensions of these functions are all set as four.

The parameter settings of COAC are the same as [9]. In the CCOAC, the number of regions μ and the number of ants m are set as 30 and 20 for all the fourteen functions, other parameter settings are the same as COAC.

For each function, 100 independent trials are carried out by applying the COAC and CCOAC under the same circumstances.

TABLE III. BENCHMARK FUNCTIONS

Function	Domain	Optimum
$f_1(x)=\sum_{i=1}^{n} x_i^2$	$[-100,100]^n$	0
$f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	$[-10,10]^n$	0
$f_3(x)=\sum_{i=1}^{n}(\sum_{j=1}^{n} x_j)^2$	$[-100,100]^n$	0
$f_4(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	$[-100,100]^n$	0
$f_5(x)=\sum_{i=1}^{n}(\lvert x_i+0.5\rvert)^2$	$[-100,100]^n$	0
$f_6(x)=\sum_{i=1}^{n} ix_i^4$	$[-1.28,1.28]^n$	0
$f_7(x)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2})-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i)+20+e$	$[-32,32]^n$	0
$f_8(x)=\frac{\pi}{n}\{10\sin^2(3\pi y_i)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(3\pi y_{i+1})]+(y_n-1)^2[1+\sin^2(2\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	$[-50,50]^n$	0
$f_9(x)=0.1\{\sin^2(3\pi x_1+\sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+(x_n-1)[1+\sin^2(2\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,10,100,4)$	$[-50,50]^n$	0
$f_{10}(x)=\sum_{i=1}^{11}[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}]^2$	$[-5,5]^n$	3.075×10^{-4}
$f_{11}(x)=-\sum_{i=1}^{5}[(x-a_i)(x-a_i)^T+c_i]^{-1}$	$[0,10]^n$	-10.1532
$f_{12}(x)=-\sum_{i=1}^{7}[(x-a_i)(x-a_i)^T+c_i]^{-1}$	$[0,10]^n$	-10.4029
$f_{13}(x)=-\sum_{i=1}^{10}[(x-a_i)(x-a_i)^T+c_i]^{-1}$	$[0,10]^n$	-10.5664
$f_{14}(x)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}y_i^2})-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi y_i)+20+e,\ y^T=M\cdot x^T$	$[-32,32]^n$	0

B. Experimental Results and Comparison

In Table IV, the solution accuracy compared between COAC and CCOAC are listed. It can be seen that to most of the fourteen functions, both the COAC and the CCOAC can find the optimum in all trials. In solving f_4, f_7 and f_{10}, the CCOAC can achieve better mean values, and in f_4, f_7, f_{10}, f_{11} and f_{13}, it can achieve better standard deviation. The CCOAC provides higher global optimization ability and stability than the COAC. Especially in solving f_4, the differences are significant.

Fig. 3 illustrates the mean convergence characteristics during the training in solving f_1, f_4, f_7, f_{10}, f_{13} and f_{14}. The vertical axis is the mean function value of 100 trials, and the horizontal axis is the number of function evaluations. Fig. 3 shows that the CCOAC can provide faster convergence speed as well as higher solution accuracy than the COAC.

The comparison results of the convergence speed are listed in Table V. Error is defined to judge whether the optimization meets the precision requirement. During the optimization process, if the absolute value of the current *best* minus *optimum* is smaller than the value of Error (|*best*−*optimum*|<Error), the trial is regarded as successful and the "errFES" of this trial is obtained. The column of "errFESs" lists the average function evaluations in obtaining the Errors in the 100 trials. The column of "stopFESs" lists the average function evaluations in obtaining the last optimal results in the 100 trials. The column of "%ok" lists the successful rates within the Errors.

As shown in Table V, the improvement of the convergence speed is remarkable. The errFESs in CCOAC is smaller than that in COAC when solving all the fourteen functions, the stopFESs in CCOAC is smaller than that in COAC only except for solving f_{11}-f_{13}. The result shows that the CCOAC has extraordinary faster convergence speed for almost all the functions than the COAC. The successful rates in CCOAC are all 100%, which means the encounter premature phenomenon does not occurred when applying the CCOAC to solving the fourteen functions.

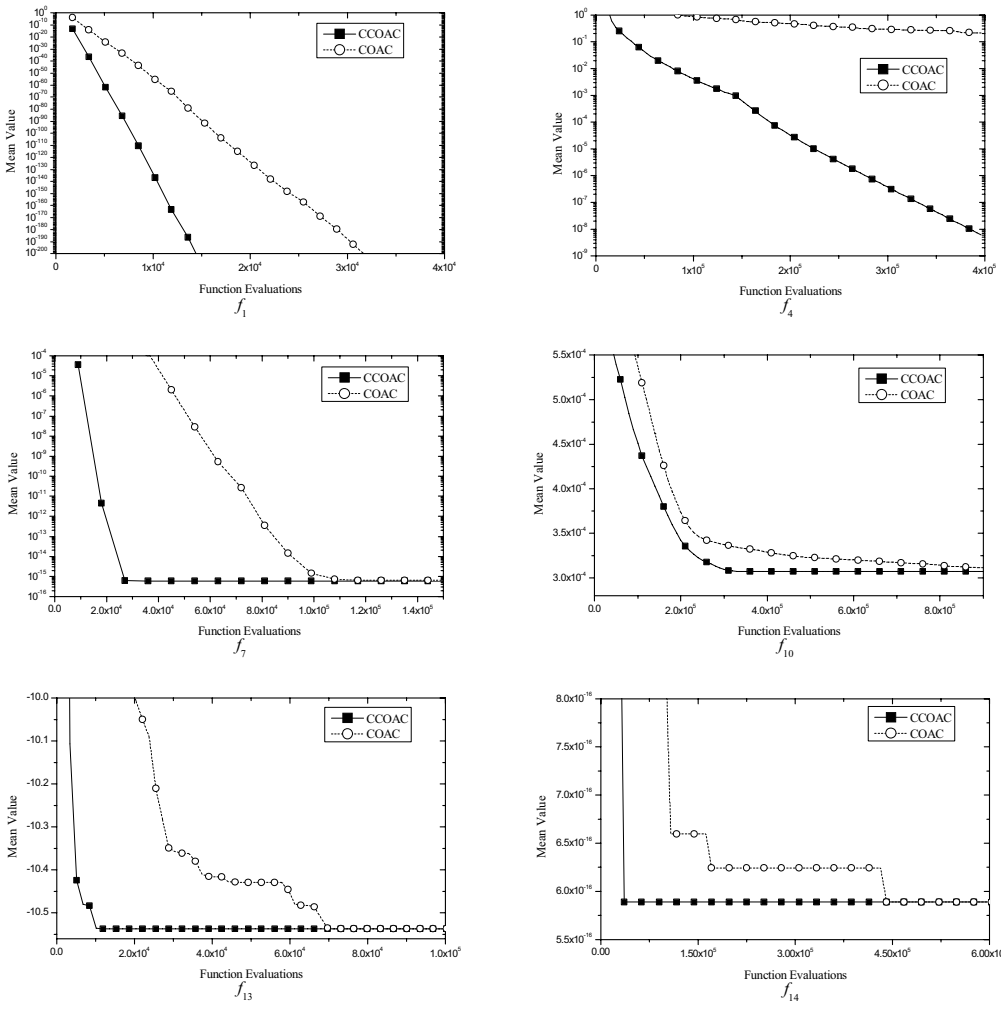

Figure 3. Convergent Curves of COAC and CCOAC

TABLE IV. SOLUTION ACCURACY COMPARISONS

F	Optimum	COAC		CCOAC	
		mean	dev	mean	dev
f_1	0	0	0	0	0
f_2	0	0	0	0	0
f_3	0	0	0	0	0
f_4	0	0.2141	0.6376	**5.03×10^{-9}**	**4.03×10^{-9}**
f_5	0	0	0	0	0
f_6	0	0	0	0	0
f_7	0	6.24×10^{-16}	3.55×10^{-16}	**5.89×10^{-16}**	**0**
f_8	0	1.18×10^{-31}	1.32×10^{-46}	1.18×10^{-31}	1.32×10^{-46}
f_9	0	1.35×10^{-32}	2.48×10^{-47}	1.35×10^{-32}	2.48×10^{-47}
f_{10}	3.075×10^{-4}	3.083×10^{-4}	6.05×10^{-6}	**3.075×10^{-4}**	**1.01×10^{-16}**
f_{11}	-10.1532	-10.1532	1.94×10^{-14}	-10.1532	**1.82×10^{-14}**
f_{12}	-10.4029	-10.4029	1.4310^{-14}	-10.4029	1.56×10^{-14}
f_{13}	-10.5364	-10.5364	1.5110^{-14}	-10.5364	**1.42×10^{-14}**
f_{14}	0	5.89×10^{-16}	0	5.89×10^{-16}	0

TABLE V. SPEED COMPARISONS

F	Error	COAC			CCOAC		
		errFESs	stopFESs	%ok	errFESs	stopFESs	%ok
f_1	0.1	915	46303	100	**579**	**19860**	100
f_2	0.1	965	101963	100	**643**	**60445**	100
f_3	0.1	1043	55570	100	**874**	**41365**	100
f_4	1	63775	407062	97	**9533**	**400065**	**100**
f_5	0.1	799	799	100	**612**	**612**	100
f_6	0.1	130	28334	100	**96**	**15890**	100
f_7	0.1	12795	69827	100	**1559**	**7369**	100
f_8	0.1	3949	31824	100	**1513**	**9099**	100
f_9	0.1	9223	51618	100	**770**	**10897**	100
f_{10}	0.0001	190071	657259	100	**106502**	**653885**	100
f_{11}	0.1	6203	15289	100	**2399**	43162	100
f_{12}	0.1	8351	29677	100	**2303**	43703	100
f_{13}	0.1	9993	25040	100	**2590**	29112	100
f_{14}	0.1	9951	54326	100	**2214**	**9306**	100

The experimental results and comparisons demonstrate that the clustering-based adaptive parameters control scheme contributes to both higher solution accuracy and faster convergence speed to continuous ant colony algorithms.

V. CONCLUSIONS

In this paper, an adaptive parameters control method for continuous ant colony algorithms has been proposed. Clustering analysis is used to judge the optimization state and the adjustment of the parameters is based on the optimization states during the training process.

As an example, the clustering-based adaptive parameter control method is used to improve the performance of the continuous orthogonal ant colony (COAC). From the experimental results, it can clearly be seen that the method not only improves the accuracy and stability of the algorithm, but also improve the convergence speed.

The proposed clustering-based adaptive parameter control method has practical value and bright prospect. Future work is applying the method to conventional ACO algorithms for solving combinatorial optimization problems.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation of China under Project 60573066, in part by the NSF of Guangdong under Project 5003346, in part by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, China, in part by the National Natural Science Foundation of China Joint Fund with Guangdong under Key Project U0835002, in part by the National High-Technology Research and Development Program of China, No. 2009AA01Z208. Jun Zhang is the corresponding author, e-mail: junzhang@ieee.org.

REFERENCES

[1] M. Dorigo, V. Maniezzo and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Trans on Systems, Man, and Cybernetics – part B Cybernetics*, vol. 26, pp. 19-41, February 1996.

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

[3] W.-N. Chen, and J. Zhang, "An ant colony optimization approach to a Grid workflow scheduling problem with various QoS requirements," *IEEE Trans on System, Man, and Cybernetics, Part C*, vol. 39, no. 1, pp. 29-43, 2009.

[4] J. Zhang, X.-M. Hu, X. Tan, et al, "Implementation of an ant colony optimization technique for job scheduling problem," *Transactions of the Institute of Measurement and Control*, vol. 28, no. 1, pp. 93-108, 2006.

[5] X.-M. Hu, J. Zhang, J. Xiao, et al, "Protein folding in hydrophobic-polar lattice model: a flexible ant-colony optimization approach," *Protein & Peptide Letters*, vol. 15, no. 5, pp. 469-477, 2008.

[6] J. Zhang, H. Chung, W. L. Lo, et al, "Extended Ant Colony Optimization Algorithm for Power Electronic Circuit Design," *IEEE Trans on Power Electronic*. vol.24,no.1, pp.147-162, 2009.

[7] X.-M. Hu, J. Zhang, and H. Chung, "An Intelligent Testing System Embedded with an Ant Colony Optimization Based Test Composition Method," *IEEE Trans on Systems, Man, and Cybernetics--Part C: Applications and and Reviews*, (in press).

[8] J. Zhang, W.-N. Chen, J.-H. Zhong, et al, "Continuous function optimization using hybrid ant colony approach with orthogonal design scheme," *LNCS4247*, *SEAL'06*, pp. 126 – 133, October 2006.

[9] X.-M. Hu, J. Zhang and Y. Li, "Orthogonal methods based ant colony search for solving continuous optimization problems," *Journal of Computer Science and Technology*, vol. 23, pp. 2-18, January 2008.

[10] Y. Zeng and Y. He, "Ant routing algorithm for mobile ad-hoc networks based on adaptive improvement," *Wireless Communications, Networking and Mobile Computing, Proceedings 2005*, vol. 2, pp. 678-681, September 2005.

[11] Z. Cai and H. Huang, "Ant colony optimization algorithm based on adaptive weight and volatility parameters," *Intelligent Information Technology Application 2008 (IITA'08)*, vol. 2, pp. 75-79, December 2008.

[12] L. Qin, J. Luo, Z. Chen, et al, "Phelogenetic tree construction using self adaptive ant colony algorithm," *Computer and Computational Sciences 2006 (IMSCCS'06)*, vol. 1, pp. 179-187, June 2006

[13] Y. Lin and J. Zhang, J. Xiao, "A pseudo parallel ant algorithm with an adaptive migration controller," *Applied Mathematics and Computation*, vol.205, pp.677-687, 2008.

[14] J. Zhang, H. Chung and W. L. Lo, "Clustering-Based Adaptive Crossover and Mutation Probabilities for Genetic Algorithms", *IEEE Trans on Evolutionary Computation,* vol.11, no.3, pp. 326-335, 2007

[15] X. Yao, Y. Liu, and G.-M. Lin, "Evolutionary programming made faster," *IEEE Trans on Evolutionary Computation*, vol. 3, no. 2, pp. 82-102, 1999.