An Extension of Ant Colony System to Continuous Optimization Problems

Seid H. Pourtakdoust and Hadi Nobahari

Sharif University of Technology, P.O. Box: 11365-8639, Tehran, Iran
pourtak@sharif.edu
nobahari@mehr.sharif.edu

Abstract. A new method for global minimization of continuous functions has been proposed based on Ant Colony Optimization. In contrast with the previous researches on continuous ant-based methods, the proposed scheme is purely pheromone-based. The algorithm has been applied to several standard test functions and the results are compared with those of two other meta-heuristics. The overall results are compatible, in good agreement and in some cases even better than the two other methods. In addition the proposed algorithm is much simpler, which is mainly due to its simpler structure. Also it has fewer control parameters, which makes the parameter settings process easier than many other methods.

1 Introduction

Ant algorithms were first proposed by Marco Dorigo and colleagues [1, 2] as a multi-agent approach to solve difficult combinatorial optimization problems. The first algorithm inspired from the ant colony functioning, is Ant System (AS) [1, 3], which is considered the base of many other approaches such as Max-Min AS (MMAS) [4], Ant Colony System (ACS) [5] and Ant-Q [6].

Ant algorithms have been applied to many different discrete optimization problems such as the Traveling Salesman Problem, Quadratic Assignment Problem, Job-Shop scheduling, Vehicle Routing, Sequential Ordering, Graph Coloring, Routing in Communications Networks and so on [7, 8]. But there are few adaptations of such algorithms to continuous problems, whereas these problems are frequently met, especially in engineering. The first algorithm designed for continuous functions optimization is CACO (Continuous Ant Colony Optimization) [9, 10], which uses ant colony framework to perform local searches where as the global search is handled by a genetic algorithm.

The main idea utilized in all of the above algorithms has been adopted from the ants pheromone trails-laying behavior, which is an indirect form of communication mediated by modifications of the environment. But this behavior is also a part of the recruitment process, which is defined as a form of communication that leads some individuals to gather at the same place in order to perform a particular action. According to this definition, some other optimization methods such as API [11, 12] have been developed in a continuous framework. This

method is inspired by a primitive ant's recruitment behavior. It performs a *tandem-running* which involves two ants and leads them to gather on a same hunting site. The authors use this particular recruitment technique to make the population proceed towards the optimum solution, by selecting the best point among those evaluated by the ants.

A recent research on modeling ants' behavior [13] has shown that it is possible to start a recruitment sequence even without taking pheromone trails into account. In this model, the stigmergic process is deliberately ignored and only inter-individuals relationships are considered. The model tries to reproduce the flow of ants exiting the nest after the entry of a scout who has discovered a new food source. To differentiate this process from the stigmergic recruitment, the authors have called it mobilization. Another utilized approach to solve continuous optimization problems is by converting them to discrete form. Then discrete versions of ant algorithms can be applied [14, 15].

In this paper a pure pheromone based method has been developed to find the global minimum of continuous functions. Our approach is to extend the well-known ACS [5] to continuous optimization problems. So we will call it *Continuous Ant Colony System* (CACS).

2 Continuous Ant Colony System

It is desired to find the absolute minimum of a positive non-zero function $f(x)$, within a given interval $[a, b]$, in which the minimum occurs at a point x_s. In general f can be a multi-variable function, defined on a subset of $\mathbb{R}^n$ delimited by n intervals $[a_i, b_i]$, $i = 1, ..., n$.

2.1 Ant Colony System Basic Features

Ant Colony System uses a graph representation. In addition to the cost measure, each edge has also a desirability measure, called *pheromone intensity.* To solve the problem, each ant generates a complete tour by choosing the nodes according to a so called *pseudo-random-proportional state transition rule,* which has two major features. Ants prefer to move to the nodes, which are connected by the edges with a high amount of pheromone, while in some instances, their selection may be completely random. The first feature is called *exploitation* and the second is a kind of *exploration.*

While constructing a tour, ants also modify the amount of pheromone on the visited edges by applying a *local updating rule.* It concurrently simulates the *evaporation* of the previous pheromone and the *accumulation* of the new pheromone deposited by the ants while they are building their solutions. Once all the ants have completed their tours, the amount of pheromone is modified again, by applying a *global updating rule.* Again a part of pheromone evaporates and all edges that belong to the global best tour, receive additional pheromone conversely proportional to their length.

2.2 Continuous Pheromone Model

The first step to develop a continuous version of ACS is to define a continuous pheromone model. Although pheromone distribution has been first modeled over discrete sets, like the edges of a traveling salesman problem, in the case of real ants, pheromone deposition occurs over a continuous space. In this regard, consider a food source surrounded by several ants. The ants aggregation around the food source causes the most pheromone intensity to occur at the food source position. Then increasing the distance of a sample point from the food source will decrease its pheromone intensity. To model this variation, a normal probability distribution function (pdf) is proposed as follows:

$$\tau(x) = e^{-\frac{(x - x_{\min})^2}{2\sigma^2}} \tag{1}$$

where $x_{\min}$ is the best point found within the interval $[a, b]$ from the beginning of the trial and σ is an index of the ants aggregation around the current minimum. To initialize the algorithm, $x_{\min}$ is randomly chosen within the interval $[a, b]$, using a uniform pdf and σ is taken at least 3 times greater than the length of the search interval, $(b - a)$, to uniformly locate the ants within it.

2.3 State Transition Rule

In ACS, ants choose their paths by both exploitation of the accumulated knowledge about the problem and exploration of new edges. In CACS, pheromone intensity is modeled using a normal pdf, the center of which is the last best global solution and its variance depends on the aggregation of the promising areas around the best one. So it contains exploitation behavior. In the other hand, a normal pdf permits all points of the search space to be chosen, either close to or far from the current solution. So it also contains exploration behavior. It means that ants can use a random generator with a normal pdf as the state transition rule to choose the next point to move to.

2.4 Pheromone Update

During each iteration, ants choose their destinations through the probabilistic strategy of equation (1). At the first iteration, there isn't any knowledge about the minimum point and the ants choose their destinations only by exploration. During each iteration pheromone distribution over the search space will be updated using the acquired knowledge of the evaluated points by the ants. This process gradually increases the exploitation behavior of the algorithm, while its exploration behavior will decrease. Pheromone updating can be stated as follows: The value of objective function is evaluated for the new selected points by the ants. Then, the best point found from the beginning of the trial is assigned to $x_{\min}$. Also the value of σ is updated based on the evaluated points during the

last iteration and the aggregation of those points around $x_{\min}$. To satisfy simultaneously the fitness and aggregation criteria, a concept of weighted variance is defined as follows:

$$\sigma^2 = \frac{\sum_{j=1}^{k} \frac{1}{f_j - f_{\min}} (x_j - x_{\min})^2}{\sum_{j=1}^{k} \frac{1}{f_j - f_{\min}}} \tag{2}$$

where k is the number of ants. This strategy means that the center of region discovered during the subsequent iteration is the last best point and the narrowness of its width is dependent on the aggregation of the other competitors around the best one. The closer the better solutions get (during the last iteration) to the best one, the smaller σ is assigned to the next iteration.

During each iteration, the height of pheromone distribution function increases with respect to the previous iteration and its narrowness decreases. So this strategy concurrently simulates pheromone accumulation over the promising regions and pheromone evaporation from the others, which are the two major characteristics of ACS pheromone updating rule.

3 Results and Discussion

For a comparative study of CACS with two other meta-heuristic methods, namely API and a Genetic Algorithm [11, 12], similar test functions are chosen as proposed in [16]. These functions are tabulated in table 1.

Table 1. Utilized test functions (All have zero minimum value)

	Function	Interval of x_i
f_1	$x_1^2 + x_2^2 + x_3^2$	$[-5.12, 5.12]$
f_2	$100(x_1^2 - x_2)^2 + (1 - x_1)^2$	$[-2.05, 2.05]$
f_3	$50 + \sum_{i=1}^{5}(x_i^2 - 10\cos(2\pi x_i))$	$[-5.12, 5.12]$
f_4	$1 + \sum_{i=1}^{2} \frac{x_i^2}{4000} - \prod_{i=1}^{2} \cos(\frac{x_i}{\sqrt{i}})$	$[-5.12, 5.12]$
f_5	$1 + \sum_{i=1}^{5} \frac{x_i^2}{4000} - \prod_{i=1}^{5} \cos(\frac{x_i}{\sqrt{i}})$	$[-5.12, 5.12]$
f_6	$0.5 + (\sin^2(x_1^2 + x_2^2)^{1/2} - 0.5)/(1 + 0.001(x_1^2 + x_2^2))$	$[-100, 100]$
f_7	$(x_1^2 + x_2^2)^{0.25}(1 + \sin^2 50(x_1^2 + x_2^2)^{0.1})$	$[-100, 100]$

3.1 Parameter Study

The proposed CACS has only one parameter to set which is the number of ants, k. In order to determine the optimal value of k, the minimum values obtained with different number of ants have been considered. To make the results comparable with those of the previous researches [11, 12], the maximum number of function evaluations was limited to 10000.

Table 2 gives the average (m), and the standard deviation (ρ) of minima found for each function. All presented results are averaged over 50 different runs. The columns report the average solutions where the numbers in the parentheses are the standard deviations. Since the maximum number of function evaluations is limited to 10000, k can vary from 1 to 10000. The first interesting observation is that the order of minima reached by this algorithm is widely dependent on the complexity of test function, similar to the other known optimization methods. This means that to determine the admissible range of k, each function must be considered separately. The admissible range of k for which the minimum values are relatively accurate with respect to the other results, are shown in bold face. This remark shows that choosing the value of k between 20 and 200 can guarantee a relatively good performance against these test functions.

Table 2. The minimum values obtained with different number of ants (k)

k	$\min(f_1)$	$\min(f_2)$	$\min(f_3)$	$\min(f_4)$	$\min(f_5)$	$\min(f_6)$	$\min(f_7)$
1	$9.9e+000$	$6.0e+01$	$5.6e+01$	$3.2e-01$	$8.0e-01$	$3.7e-01$	$7.1e+00$
	$(8.0e+000)$	$(1.4e+02)$	$(1.8e+01)$	$(3.0e-01)$	$(2.0e-01)$	$(1.1e-01)$	$(2.6e+00)$
2	$2.0e+000$	$7.9e+00$	$3.2e+01$	$4.9e-02$	$4.7e-01$	$2.7e-01$	$4.5e+00$
	$(3.2e+000)$	$(3.6e+01)$	$(1.5e+01)$	$(1.2e-01)$	$(2.4e-01)$	$(1.5e-01)$	$(2.6e+00)$
5	$\mathbf{2.1e-242}$	$5.6e-05$	$1.5e+01$	$\mathbf{5.2e-03}$	$\mathbf{7.5e-02}$	$4.9e-02$	$9.1e-01$
	$(\mathbf{9.2e-243})$	$(1.5e-04)$	$(7.8e+00)$	$(\mathbf{3.4e-03})$	$(\mathbf{1.2e-01})$	$(7.0e-02)$	$(1.6e+00)$
10	$\mathbf{1.8e-228}$	$\mathbf{6.5e-13}$	$1.1e+01$	$\mathbf{5.8e-03}$	$\mathbf{3.1e-02}$	$1.0e-02$	$\mathbf{2.9e-03}$
	$(\mathbf{2.1e-228})$	$(\mathbf{2.7e-12})$	$(7.1e+00)$	$(\mathbf{3.1e-03})$	$(\mathbf{6.6e-02})$	$(1.4e-02)$	$(\mathbf{8.3e-03})$
20	$\mathbf{2.0e-135}$	$\mathbf{1.5e-19}$	$\mathbf{7.1e+00}$	$\mathbf{5.3e-03}$	$\mathbf{2.8e-02}$	$\mathbf{5.1e-03}$	$\mathbf{7.7e-16}$
	$(\mathbf{1.9e-134})$	$(\mathbf{1.4e-18})$	$(\mathbf{4.5e+00})$	$(\mathbf{3.3e-03})$	$(\mathbf{5.5e-02})$	$(\mathbf{2.1e-03})$	$(\mathbf{1.6e-15})$
50	$\mathbf{1.5e-067}$	$\mathbf{1.2e-31}$	$\mathbf{4.8e+00}$	$\mathbf{5.0e-03}$	$\mathbf{1.1e-02}$	$\mathbf{4.6e-03}$	$\mathbf{4.2e-06}$
	$(\mathbf{9.5e-067})$	$(\mathbf{6.8e-31})$	$(\mathbf{3.9e+00})$	$(\mathbf{3.5e-03})$	$(\mathbf{2.5e-02})$	$(\mathbf{1.1e-03})$	$(\mathbf{7.7e-06})$
100	$\mathbf{3.6e-037}$	$\mathbf{1.6e-33}$	$\mathbf{4.9e+00}$	$\mathbf{4.1e-03}$	$\mathbf{7.8e-03}$	$\mathbf{3.9e-03}$	$\mathbf{2.5e-03}$
	$(\mathbf{2.2e-036})$	$(\mathbf{8.5e-33})$	$(\mathbf{2.8e+00})$	$(\mathbf{3.7e-03})$	$(\mathbf{9.0e-03})$	$(\mathbf{1.8e-03})$	$(\mathbf{1.5e-03})$
200	$\mathbf{1.5e-020}$	$\mathbf{3.2e-22}$	$\mathbf{7.1e+00}$	$\mathbf{2.7e-03}$	$\mathbf{7.7e-03}$	$\mathbf{3.7e-03}$	$\mathbf{5.9e-02}$
	$(\mathbf{3.0e-020})$	$(\mathbf{2.0e-21})$	$(\mathbf{2.4e+00})$	$(\mathbf{3.6e-03})$	$(\mathbf{8.4e-03})$	$(\mathbf{1.8e-03})$	$(\mathbf{1.8e-02})$
500	$3.0e-009$	$\mathbf{1.7e-12}$	$\mathbf{9.4e+00}$	$\mathbf{1.1e-03}$	$\mathbf{1.4e-02}$	$\mathbf{4.2e-03}$	$3.8e-01$
	$(3.2e-009)$	$(\mathbf{4.4e-12})$	$(\mathbf{2.3e+00})$	$(\mathbf{2.6e-03})$	$(\mathbf{1.5e-02})$	$(\mathbf{1.4e-03})$	$(1.2e-01)$
1000	$2.6e-005$	$1.0e-07$	$1.0e+01$	$\mathbf{1.6e-04}$	$\mathbf{5.5e-02}$	$\mathbf{5.2e-03}$	$6.5e-01$
	$(2.4e-005)$	$(2.9e-07)$	$(2.9e+00)$	$(\mathbf{1.1e-03})$	$(\mathbf{2.3e-02})$	$(\mathbf{1.3e-03})$	$(1.8e-01)$
2000	$2.8e-003$	$2.3e-05$	$1.1e+01$	$\mathbf{1.6e-04}$	$\mathbf{7.2e-02}$	$\mathbf{7.1e-03}$	$8.3e-01$
	$(2.0e-003)$	$(4.7e-05)$	$(2.6e+00)$	$(\mathbf{2.3e-04})$	$(\mathbf{3.0e-02})$	$(\mathbf{4.0e-03})$	$(2.4e-01)$
5000	$3.2e-002$	$1.2e-03$	$1.3e+01$	$\mathbf{6.1e-04}$	$\mathbf{9.0e-02}$	$\mathbf{9.5e-03}$	$1.1e+00$
	$(2.2e-002)$	$(1.1e-03)$	$(3.4e+00)$	$(\mathbf{5.6e-04})$	$(\mathbf{3.7e-02})$	$(\mathbf{5.9e-03})$	$(3.3e-01)$
10000	$7.8e-002$	$4.5e-03$	$1.7e+01$	$\mathbf{1.2e-03}$	$1.1e-01$	$1.3e-02$	$1.2e+00$
	$(4.9e-002)$	$(4.6e-03)$	$(3.8e+00)$	$(\mathbf{1.1e-03})$	$(4.0e-02)$	$(8.1e-03)$	$(3.3e-01)$

To further study the effect of k on the algorithm performance, two performance indices are defined. Since all test functions have a minimum of zero, a normalized summation of the minima, has been proposed as a good index for performance evaluation [11]. This parameter is defined as follows :

$$S_1 = \sum_{i=1}^{7} \frac{m_i - \min\limits_{j=1}^{7}\{m_j\}}{\max\limits_{j=1}^{7}\{m_j\} - \min\limits_{j=1}^{7}\{m_j\}} \tag{3}$$

where m_i is the average of the minima found for f_i. The second index can also be defined as follows:

$$S_2 = \sum_{i=1}^{7} \frac{\rho_i}{m_i} \tag{4}$$

which is the summation of the normalized standard deviations obtained for the given test functions. Since it is desirable to reach to the smallest possible values of S_1 and S_2, their variations with k are investigated (Figure 1). One interesting observation is that CACS is not very robust or responsive with small number of ants ($k < 10$). In this case the probability of falling in a local minimum increases because it is very probable that during one iteration all ants choose the points near to the current minimum. This in turn decreases the values of the weighted variances magnifying the aggregation of the later iteration points. When the value of the weighted variance becomes several times smaller than the distance to the nearest local minimum, it's no longer possible to jump to a better local minimum and the minimization process stops.

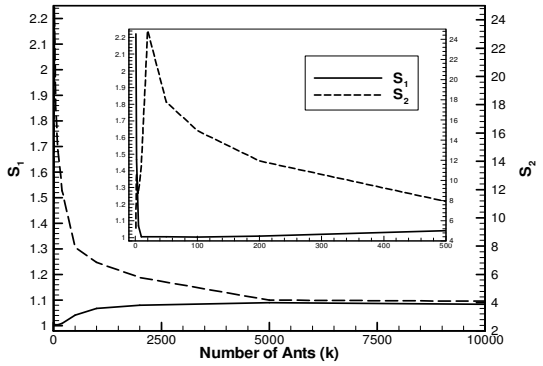

Fig. 1. Variation of S_1 and S_2 with the number of ants

Figure 1 shows that S_1 is relatively constant when the number of ants is between 10 and 200 and it begins to increase for greater values. This is consistent with the previous results about the admissible range of the number of ants. On

the other hand, S_2 first increases with the number of ants up to $k = 20$ and then begins to decrease. However the reduction rate slows down as the number of ants increases. According to figure 1 and the results of Table 2, the range $[50, 100]$ seems to be an admissible range for the number of ants. The convergence rate will decrease for simpler functions like f_1 if the number of ants increases further.

3.2 Comparison with other methods

Table 3 shows the CACS results, under 10000 function evaluations with ($k = 50, 100$), which are compared with those of two other methods. The first one is API algorithm [11, 12], which is another continuous ant-based method and the second one is a generational genetic algorithm (GA) with binary tournament selection, 1 point crossover and with a real-coded representation [17]. Table 3 shows that CACS can find the global minima better than or at least as good as API for all functions. It can also find the global minima better than GA for all functions except for the case of f_3 which is a complex function containing many local minima.

Table 3. Comparision of CACS Scheme with API and GA

Method	$\min(f_1)$	$\min(f_2)$	$\min(f_3)$	$\min(f_4)$	$\min(f_5)$	$\min(f_6)$	$\min(f_7)$
CACS	$1.5e-067$	$1.2e-31$	$4.8e+00$	$5.0e-03$	$1.1e-02$	$4.6e-03$	$4.2e-06$
$N_a = 50$	$(9.5e-067)$	$(6.8e-31)$	$(3.9e+00)$	$(3.5e-03)$	$(2.5e-02)$	$(1.1e-03)$	$(7.7e-06)$
CACS	$3.6e-037$	$1.6e-33$	$4.9e+00$	$4.1e-03$	$7.8e-03$	$3.9e-03$	$2.5e-03$
$N_a = 100$	$(2.2e-036)$	$(8.5e-33)$	$(2.8e+00)$	$(3.7e-03)$	$(9.0e-03)$	$(1.8e-03)$	$(1.5e-03)$
API	0.00000	0.00000	7.47651	0.00413	0.25034	0.00659	0.09307
	(0.00000)	(0.00000)	(2.98922)	(0.00402)	(0.12254)	(0.00443)	(0.01886)
GA	0.00000	0.04029	2.12457	0.03095	0.13955	0.07376	0.13358
	(0.00000)	(0.06515)	(1.30328)	(0.03531)	(0.07620)	(0.06590)	(0.06271)

4 Conclusion

In this study a new continuous optimization method was proposed based on Ant Colony Optimization. The proposed scheme was tested over several standard test functions in order to set its control parameters and to compare its results with two other methods, namely API and a Genetic Algorithm. The overall results of CACS are compatible and in some cases even better than those of the two other methods. Also CACS possesses higher convergence rate for some functions due to its adaptive pheromone updating rule strategy. One of the best advantages of CACS is its simplicity, mainly due to its simpler structure and having only one control parameter. Obviously, this feature, makes the parameter setting process easier than many other methods. In addition CACS does not seem to be sensitive

with respect to its control parameter. Numerical results show that choosing the number of ants within the range $[20, 200]$ provides relatively good accuracy for the considered test functions.

References

[1] Dorigo, M.: Optimization, learning and natural algorithms. PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, IT (1992)

[2] Colorni A., Dorigo, M. and Maniezzo, V.: Distributed optimization by ant colonies. Proceedings of the First European Conference on Artificial Life, Elsevier Science Publisher (1992) 134-142

[3] Dorigo, M., Maniezzo, V. and Colorni, A.: The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man and Cybernetics-Part B **26** (1) (1996) 29-41

[4] Stutzle, T. and Hoos, H.: MAX-MIN Ant System. Future Generation System **16** (8) (2000) 889-914

[5] Dorigo, M. and Gambardella, L. M.: Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation **1** (1) (1997) 53-66

[6] Gambardella, L. M. and Dorigo, M.: Ant-Q: A reinforcement learning approach to the traveling salesman problem. Proceedings of the 12th International Conference on Machine Learning, ML-95, Palo Alto (1995) 252-260

[7] Dorigo, M., Caro, G. D. and Gambardella, L. M.: Ant algorithms for discrete optimization. Artificial Life **5** (3) (1999) 137-172

[8] Dorigo, M., Bonabeau, E. and Theraulaz, G.: Ant algorithms and stigmergy. Future Generation Computer Systems **16** (2000) 851-871

[9] Bilchev, G. and Parmee, I.C.: The ant colony metaphor for searching continuous design spaces. Lecture Notes in Computer Science **993** (1995) 25-39

[10] Wodrich, M. and Bilchev, G.: Cooperative distributed search: the ant's way. Control and Cybernetics **26** (1997) 413-445

[11] Monmarche, N., Venturini, G. and Slimane, M.: On how the ants Pachycondyla apicalis suggesting a new search algorithm. Internal Report No. 214, E3i, Downloadable from website http://www.antsearch.univ-tours.fr/webrtic (1999)

[12] Monmarche, N., Venturini, G. and Slimane, M.: On how Pachycondyla apicalis ants suggest a new search algorithm. Future Generation Computer Systems **16** (2000) 937-946

[13] Dreo, J. and Siarry, P.: A new ant colony algorithm using the heterarchical concept aimed at optimization of multi-minima continuous functions. Lecture Notes in Computer Science **2463** (2002) 216-221

[14] Ling, C., Jie, S., Ling, O. and Hongjian, C.: A method for solving optimization problems in continuous space using ant colony algorithm. Lecture Notes in Computer Science **2463** (2002) 288-289

[15] Jun L. Y. and Jun, W. T.: An adaptive ant colony system algorithm for continuous-space optimization problems. Journal of Zhejiang University Science **4** (1) (2003) 40-46

[16] Whitley, D., Mathias, K., Rana S. and Dzubera, J. Building better test functions. Proceedings of the 6th International Conference on Genetic Algorithms, Morgan Kaufmann Publishers (1995) 239-246

[17] Michalewicz, Z. Genetic Algorithms+Data Structures=Evolution Programs. Springer, Berlin, 3rd Edition (1996)