An Ant Colony Optimization Algorithm Based on the Experience Model

Wenjun Pan[1], and Lipo Wang[2]

[1] *College of Information Engineering, Xiangtan University, Xiangtan, Hunan, China.*
Email: WJpan_xtu@yahoo.cn
[2] *Scholl of Electrical and Electronic Engineering, Nanyang Technology University,*
Block S1,50 Nanyang Avenue, Singapore 639789
Email: Elpwang@ntu.edu.sg

Abstract

In this paper, we propose a new model of Ant Colony Optimization (ACO) to solve the traveling salesman problem (TSP). In the new model, we introduce the experience value and a new structure called map, and the experience value is embodied in the reliability of the maps. Each ant takes a map; all ants are guided by the experience gaining from the maps so that the system based on the new model is able to converge into a near-optimum solution quickly. We have proposed an algorithm of MMAS based on the new model, tested the algorithm on several TSP instances and discussed parameter selection. We experimentally show that the algorithm based on the new model improves the converging speed and can find better solutions in comparison with the algorithm which does not use the new model.

1. Introduction

Ant Colony Optimization (ACO) is a metaheuristic for hard discrete optimization problems that was first proposed in the early 1990s [3] – [6]. The inspiring source of ACO is the foraging behavior of real world ants. Biologists noticed that blind ants can find the shortest path between food sources and their nests. Research also found out that ants deposit pheromone on the way while walking. The other ants can follow the previous pheromone by probability while passing by. The more the pheromone, the higher the probability with which the ants will follow. The indirect communication between the ants via the pheromone trails allows them to find shortest paths between their nests and food sources. This functionality of real ant colonies is exploited in artificial ant colonies in order to solve discrete optimization problems.

The first ACO algorithm was introduced by Dorigo et al. [3] in 1991 and was called the Ant System (AS) [3], [4]. Dorigo and Gambardella proposed the Ant Colony System (ACS) [5], [6] later in 1996, while Stützle and Hoos proposed the MAX-MIN Ant System (MMAS) [10]. ACO has drawn much research attention and various extended versions of the ACO paradigm were proposed, such as the Best-Worst Ant System (BWAS) [15], the Rank based Ant System (RAS) [14] etc.. In general, the ACO approach attempts to solve an optimization problem by repeating the following two steps.

1) Candidate solutions are constructed using a pheromone model; that is, a parametrized probability distribution over the solution space.

2) The candidate solutions are used to modify the pheromone values in a way that is deemed to bias future sampling toward high-quality solutions.

In this paper, we introduce a new model for implementing ACO algorithms. We call this model the Experience Model (EM) for ACO. The model is based on changing the probability of transition rule used in ACO algorithms so that it can reduce the blindness of all ants during the search process. In other words, the EM can make the most of the experience which is gained by ants in the former search.

This paper is organized as follows: In section 2, we outline the TSP and some main ACO algorithms. In section 3, we introduce the EM. section 4 shows the running environment of our experiments. In section 5, we present some experimental results in which we show that an algorithm using the proposed model has a rather good performance. Finally, in section 6, we offer a summary and indicate directions for further research.

2. Background

2.1. The TSP

The traveling salesman problem (TSP) is a prominent illustration of NP-hard. In this paper we use the TSP as benchmark [1]. The problem is: given a number of cities and the distance from any city to any other city, the goal is to find the shortest tour that visits each city exactly once and then returns to the starting city. In more formal terms, the goal is to find a Hamiltonian tour of minimal length on a fully connected graph.

2.2. Main ACO Algorithms

Several ACO algorithms have been proposed in this subsection. We use the TSP as a concrete example.

2.2.1. Ant System

AS is the first ACO algorithm proposed in the literature [3], [4]. In the AS, all the *m* ants which have constructed a solution at each iteration can update the pheromone. The pheromone value τ_{ij}, which contacted with the edge between city *i* and city *j*, is updated as follows:

$$\tau_{ij} = (1-\rho)\cdot\tau_{ij} + \sum_{k=1}^{m}\Delta\tau_{ij}^{k} \ , \tag{1}$$

where ρ is the pheromone evaporate rate, *m* is the number of ants, $\Delta\tau_{ij}^{k}$ is the quantity of the pheromone left on edge (*i, j*) by ant *k*:

$$\Delta\tau_{ij}^{k} = \begin{cases} \mathrm{Q}/L_k & \text{if edge } (i,j) \text{ is in the tour of ant } k, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where Q is a constant, L_k is the length of the tour constructed by ant *k*.

During constructing process, ants select the next city through a stochastic mechanism. The probability of moving from city *i* to city *j* is given by:

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\cdot\eta_{ij}^{\beta}}{\sum_{c_{il}\in \mathrm{N}(s^p)}\tau_{il}^{\alpha}\cdot\eta_{il}^{\beta}} & \text{if } c_{ij}\in \mathrm{N}(s^p), \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where $\mathrm{N}(s^p)$ is the set of suitable elements; that is, edge (*i, l*) where the parameter *l* is a city not yet visited by the ant *k*. The parameters α and β contact with the importance between pheromone and the heuristic information which was given by:

$$\eta_{ij} = 1/d_{ij} \ , \tag{4}$$

where the parameter d_{ij} shows the distance between cities *i* and *j*.

2.2.2. Max-Min ant system

The MMAS was introduced by Stützle et al. [10] – [12]. Its characters are that only the best-acted ant can update the value of pheromone trail, and the rank of the pheromone are limited as the following formula：

$$\tau_{ij} = \left[(1-\rho)\cdot\tau_{ij} + \Delta\tau_{ij}^{\mathrm{best}}\right]_{\tau_{\min}}^{\tau_{\max}} , \tag{5}$$

where $\tau_{\max}$ and $\tau_{\min}$ are the upper and lower bounds of the pheromone rank; the operator $[x]_b^a$ is defined as follows:

$$[x]_b^a = \begin{cases} a & \text{if } x > a, \\ b & \text{if } x < b, \\ x & \text{otherwise;} \end{cases} \tag{6}$$

and $\Delta\tau_{ij}^{\mathrm{best}}$ is:

$$\Delta\tau_{ij}^{\mathrm{best}} = \begin{cases} 1/L_{\mathrm{best}} & \text{if } (i,j) \text{ belongs to the best tour,} \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

where L_{best} is the length of the tour constructed by best ant. Alternatively L_{best} can be defined as the length of the best tour in current iteration L_{ib}, we call it : iteration-best，or the length of the best tour that the algorithm find so far from the beginning of the algorithm: L_{bs}. In fact, we use the combination of both models sometime.

With respect to the value of the upper and lower bounds of pheromone, $\tau_{\max}$ and $\tau_{\min}$, we can figure them out on the basis of analytical experiments and fix them to the real condition.

2.2.3. Ant colony system

The ACS [5] – [7] introduces the local pheromone update into the algorithm. Only when each ant explores the last edge of its tour the algorithm updates the pheromone as:

$$\tau_{ij} = (1-\varphi)\cdot\tau_{ij} + \varphi\cdot\tau_0 \ , \tag{8}$$

where $\varphi\in(0,1]$ is the parameter about the pheromone decay rate, and τ_0 is the initial value of pheromone.

The local pheromone update gives the algorithm a character by decreasing the pheromone values on the visited edges, subsequent ants can be encouraged to explore other edges so that new different solutions might be found with greater probabilities.

The offline pheromone update only executed by the last ant at the end of each iteration, called the

iteration-best or best-so-far solution. However, there still a little different as formula (9) shows:

$$\tau_{ij} = \begin{cases} (1-\rho)\cdot\tau_{ij} + \rho\cdot\Delta\tau_{ij} & \text{if } (i,j) \text{ belongs to the best tour,} \\ \tau_{ij} & \text{otherwise.} \end{cases} \quad (9)$$

Here $\Delta\tau_{ij} = 1/L_{\text{best}}$, where L_{best} can be either L_{ib} or L_{bs} .

In the ACS, ants use so-called pseudorandom proportional rule, the probability of an ant move from city i to city j is determined by a parameter q_0 and a random variable q which is uniformly distributed over [0,1]:

$$j = \begin{cases} \text{argmax}_{c_{il}\in \text{N}(s^p)}\{\tau_{il}\cdot\eta_{il}^{\beta}\} & \text{if } q \le q_0, \\ \text{use formula (3)} & \text{otherwise.} \end{cases} \quad (10)$$

3. Experience model (EM)

Much of the outstanding ACO algorithms, such as ACS, MMAS etc., only make the optimization of the pheromone update rules. Ants move from a city to the next one under the influence of pheromone, and most of the ACO algorithms use the origin probability to guide the ants' movement. Relying solely on pheromone, lots of experience gained by ants in the former search will be wasted, so the blindness of ants is inevitable. Here we introduce the experience model (EM). In the EM, we introduce a new ant model: ant with experience memory (Eant), and then discuss the state transition rule as follows:

3.1. Eant

In the Eant model, we introduce a new structure called map. A map consists of two parts: **1.** the best-so-far solution, a Hamiltonian tour of minimal length previously found, showing a ant which city is the best to move to in the previous search. **2.** a new parameter R_{map} , reliability of a map, showing a ant how credible the present best-so-far solution is for the moment. Each ant carries a map in the process of moving. When an ant chooses a city to move to, it will take the map for consideration. At the beginning of the search, the reliability of a map is at a low level, it means the experience value is small; ants are not experienced at the moment. As time passed, the experience value increases gradually, thus the reliability of a map is also on the rise bit by bit. When an ant wants to move from current city to another, it will look at the map and find the neighbor city in the best-so-far solution, then chooses the next city according to the EM state transition rule by the combined action of the map's reliability and the pheromone. We use a new parameter A_{map} ($A_{map} \in [0,1]$) to express the proportion of the reliability of a map to the pheromone. In other words, A_{map} is regarded as the attraction of a map.

Intuitionisticly, the parameter R_{map} – values range from 0 to 1. At the beginning of an algorithm, the value of R_{map} should be small, just closing to 0. In the progress of an algorithm, R_{map} is growing larger. Towards the end of an algorithm, the value of R_{map} is approaching 1. We define R_{map} as a function of time: $R_{map} = F(t)$. Obviously, $F(t)$ is monotone increasing function.

3.2. EM State Transition Rule

In the EM, we make some modifications in the transition probability. When ant k is in city i and chooses city j to move to, there are two cases:

1) If j has been previously visited, obviously the probability of going to city j is 0.

2) If j has not been previously visited, the map will play a active role. In the best-so-far solution, city i has two neighboring cities, the two cities are taken seriously while the ant uses it's map to select the next city. We define $Map(s)$ as the set formed by all edges of the best-so-far solution. The two different cases are the following:

i. If edge $(i,j) \in Map(s)$, the best-so-far solution doesn't make any impact on city j, the ant move from i to j by applying formula (3).

ii. If edge $(i,j) \notin Map(s)$, we give the transition probability a increment in order to make city j more attractive to the ant.

Generally speaking, the EM can be used in any version of ACO algorithms. In algorithms based on EM, we use the following formula instead of formula (3):

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\cdot\eta_{ij}^{\beta}}{\sum_{c_{il}\in N(s^p)}\tau_{il}^{\alpha}\cdot\eta_{il}^{\beta}} & \text{if } c_{ij}\in \text{N}(s^p) \text{ and edge } (i,j)\in Map(s), \\ \dfrac{\tau_{ij}^{\alpha}\cdot\eta_{ij}^{\beta} + R_{map}\cdot A_{map}\cdot\sum_{c_{il}\in \text{N}(s^p)}\tau_{il}^{\alpha}\cdot\eta_{il}^{\beta}}{(1+R_{map}\cdot A_{map})\sum_{c_{il}\in \text{N}(s^p)}\tau_{il}^{\alpha}\cdot\eta_{il}^{\beta}} & \text{if } c_{ij}\in \text{N}(s^p) \text{ and edge } (i,j)\notin Map(s), \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

In order to facilitate the research, we let:

$$R_{map} = F(t) = t/t_{\max}, \quad (12)$$

where $t_{\max}$ is the maximum time for each trial.

For example, in the MMAS based on EM, an ant move from i to j according to formula (11), and in the ACS based on EM the transition probability from i to j is as follows:

$$j = \begin{cases} \text{argmax}_{c_{il} \in \mathrm{N}(s^p)} \{\tau_{il} \cdot \eta_{il}^{\beta}\} & \text{if } q \le q_0, \\ \text{use formula (11)} & \text{otherwise.} \end{cases} \quad (13)$$

4. Experiment Environment

The running environment of the experiments is in Table I. Our programme is coded in Linux C.

Table I
Experiment Environment

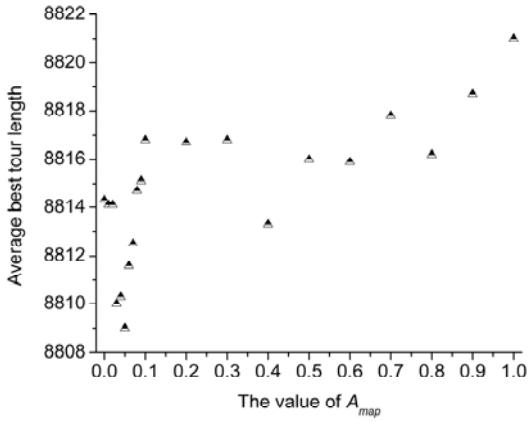
CPU	RAM	OS
Double Core Intel Pentium D 3.00G	DDR 1G	Red Flag Linux

5. Experimental Study

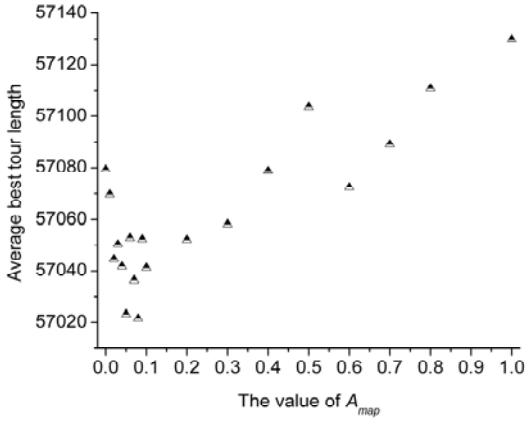
In this section, we choose MMAS for instance. All the tests reported in this section are based, where not otherwise stated, on five TSP instances: the KroA100 problem, the Lin318 problem, the Rat783 problem, the Pcb1173 problem and the Pr2392 problem [1].

5.1. Parameter Setting

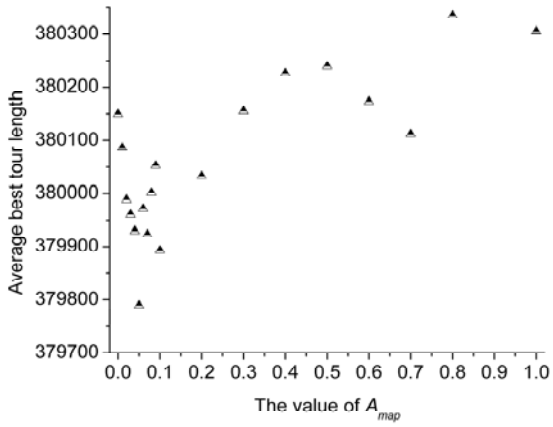
The abbreviation parameters in Table II and Table III come from the ACOTSP program which is coded by Stützle [2]. The following default value of the parameters of MMAS series were given by Stützle et al. [10]: m: the number of ants, m = 25; α: the relative importance of the trail, α = 1; ρ: trail persistence, ρ = 0.8; β: the relative importance of the visibility, β = 2. We implemented the algorithm of MMAS based on EM (EMMAS) and tested 20 values for the new parameter A_{map} while all the other parameters used above values. The values tested were: $A_{map} \in$ {0,0.01,0.02,0.03,0.04,0.05,0.06,0.07,0.08,0.09,0.1,0.2, 0.3,0.4,0.5,0.6,0.7,0.8,0.9, 1}. Intuitively, the value of A_{map} should be small. When $A_{map} = 0$, the algorithm becomes the standard MMAS; and the algorithm is unstable while A_{map} is getting nearer to 1. Fig. 1 shows the average length of the best found tour in different value of the parameter A_{map}. We can read from Fig. 1 that the average length of the best found tour is shorter while A_{map} is in the interval [0, 0.1].



(a) On the Rat783 problem



(b) On the Pcb1173 problem



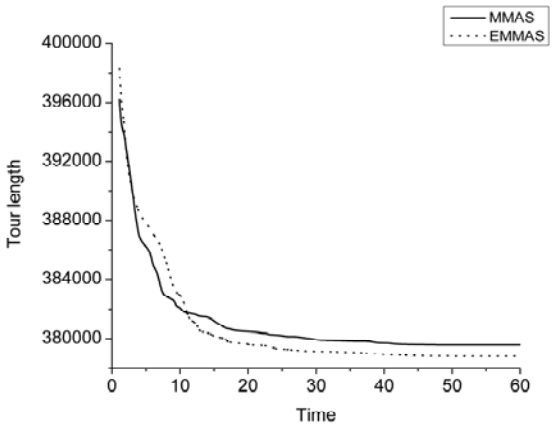
(c) On the Pr2392 problem

Fig. 1. Average length of the best found tour in different A_{map} based on 3 TSP instances.

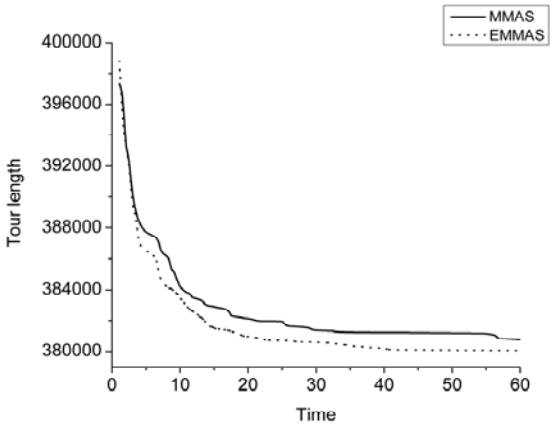
In order to compare the stability of algorithms, we use the standard deviation of the best found tour (Stddev-Best):

$$\text{Stddev-Best}=\sqrt{\frac{\sum_{i=1}^{n}(L_{\text{best-i}}-\overline{L_{\text{best}}})^2}{n}} \quad (14)$$

where $L_{\text{best-i}}$ is the best found tour in a trial, and $\overline{L_{\text{best}}}$ is the average length of the best found tour. The standard deviation is small means that the algorithm is stable. In Fig. 2, we present the Stddev-Best in different value of A_{map} on the Pr2392 problem. It is obvious that the Stddev-Best is becoming larger when A_{map} tend towards 1, that means the algorithm is getting more and more unstable. In Table II, we can see the optimal value of A_{map} is 0.05. We use this value as the default value.



(a) The best on Pr2392



(b) The worst on Pr2392

Fig. 3. EMMAS find better solutions in a shorter time compared with MMAS whether in the best or the worst condition. Test problem: Pr2392.

Table II
Average length of the best found tour based on 3 TSP instances while $A_{map} \in [0, 0.1]$

The value of A_{map}	Average-Best on Rat783	Average-Best on Pcb1173	Average-Best on Pr2392
0.01	8814.1	57069.7	380086.0
0.02	8814.1	57044.8	379988.7
0.03	8810.0	57050.4	379961.1
0.04	8810.3	57041.9	379929.7
0.05	8809.0	57023.3	379788.4
0.06	8811.6	57052.9	379972.1
0.07	8812.5	57036.4	379922.9
0.08	8814.7	57021.6	380001.3
0.09	8815.1	57052.4	380052.4

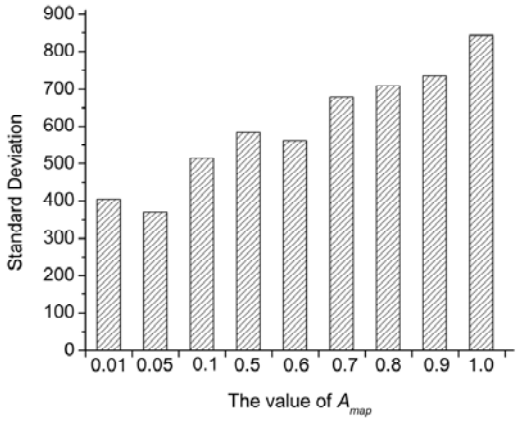


Fig. 2. Standard Deviation in different A_{map}. Test problem: Pr2392.

5.2. EMMAS in Comparison with MMAS

We ran a set of trials in both the EMMAS and the MMAS and then we compared the two algorithms. The default value of the parameters was used for a maximum of 60 seconds per each trial for 10 independent trials. The results are given in Fig. 3 and Table III. In Fig. 3 we compared the best and the worst condition in both the EMMAS and the MMAS based on the Pr2392 problem (the best means the condition the best results the algorithms found in 10 independent trials while the worst means the worst results in 10 independent trials). We can see that at the early stage, the speed of the approximation to the optimum solution of the EMMAS is faster than the speed of the MMAS. It means that the EMMAS converges faster than the MMAS. In the end, the EMMAS can find better solution contrast with the MMAS whether in the best or the worst condition.

Table III shows some statistics, the statistical

Table III
Comparison of EMMAS with MMAS on 5 TSP instances. Average over 10 trials

		On KroA100	On Lin318	On Rat783	On Pcb1173	On Pr2392
EMMAS	Best Known	21282	42029	8806	56892	378032
	Average-Best	21282	42029	8809	57023.3	379788.4
	Average-Iterations	1.7	155.4	856.3	497.4	219.6
	Stddev-Best	0	0	3.13	95.13	372.1
MMAS	Best Known	21282	42029	8806	56892	378032
	Average-Best	21282	42029	8814.3	57079.3	380149.7
	Average-Iterations	2.4	316.3	403.2	365.4	209.3
	Stddev-Best	0	0	5.08	115.96	374.07

comparison has told us that the EMMAS has surpassed the MMAS in the average length of the best found tour based on 5 TSP instances. In Table III, we can easily find out that the Stddev-Best of the EMMAS is smaller than that of the MMAS. In other words, the EMMAS is more stable than the MMAS.

6. Conclusion

This paper analyzes the Basic Ant Colony Algorithm and points out the shortcoming: the blindness of ants. To address the problem, we have proposed the experience model (EM) for ACO, and then compared the result of the MMAS based on EM with the original MMAS and made some analyses. In the experience model, the map can speed up the convergence, but the attraction of maps should be small in order to avoid premature convergence. We experimentally proved this model to be more efficient and stable. It demonstrates that simulating some human action(such as experience) can improve the performance of an ACO algorithm. In the end, how to analyze this model in theory and how to simulate more human actions are the future research content of us.

7. References

[1] TSPLIB: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp

[2] ACO Public Software: http://iridia.ulb.ac.be/~mdorigo/ACO/aco-code/public-software.html

[3] A. Colomi, M. Dorigo, and V. Maniezzo, “Distributed optimization by ant colonies,” Proceedings of the 1st European Conference on Artificial Life, pp.134-142, 1991.

[4] M. Dorigo, V. Maniezzo and A. Colorni, “Ant System: Optimization by a colony of cooperating Agents,” IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol.26, no.2, pp.29–41, 1996.

[5] M. Dorigo and L.M. Gambardella, “Ant Colony System: A cooperative learning approach to the traveling salesman problem,” IEEE Transactions on Evolutionary Computation, vol.1, no.1, pp.53–66, 1997.

[6] M. Dorigo and L.M. Gambardella, “Ant colonies for the traveling salesman problem,” BioSystems, vol.43, no.2, pp.73-81, 1997.

[7] M. Dorigo, G. Di Caro and L.M. Gambardella, “Ant algorithms for discrete optimization,” Artificial Life, vol.5, no.2, pp.137–172, 1999.

[8] M. Dorigo, E. Bonabeau, and G. Theraulaz, “Ant algorithms and stigmergy,” Future Generation Computer Systems, vol.16, no.8, pp.851–871, 2000.

[9] M. Dorigo, M. Birattari, and T. Stützle, “Ant Colony Optimization: Artificial Ants as a Computational Intelligence Technique,” IEEE Computational Intelligence Magazine, vol.1, no.4, pp.28–39, 2006.

[10] T. Stützle and H.H. Hoos, “Improving the Ant System: A detailed report on the MAX–MIN Ant System,” FG Intellektik, FB Informatik, TU Darmstadt, Germany, Tech. Rep. AIDA–96–12, Aug. 1996.

[11] T. Stützle and H. H. Hoos, “The MAX–MIN ant system and local search for the traveling salesman problem,” in Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (ICEC’97), T. Bäck, Z. Michalewicz, and X. Yao, Eds. Piscataway, NJ: IEEE Press, 1997, pp.309–314.

[12] T. Stützle and H. H. Hoos, “The MAX–MIN ant system,” Future Generation Computer Systems, vol.16, no.8, pp.889–914, 2000.

[13] T. Stützle and M. Dorigo, “A Short Convergence Proof for a Class of Ant Colony Optimization Algorithms,” IEEE Transactions on Evolutionary Computation, vol.6, no.4, pp.358–365, 2002.

[14] B. Bullnheimer, R.F. Hartl, and C. Strauss, “A new rank-based version of the Ant System: A computational study,” Central European Journal for Operations Research and Economics, vol.7, no.1, pp.25–38, 1999.

[15] O. Cordón, I.F. de Viana, F. Herrera, and L. Moreno, “A new ACO model integrating evolutionary computation concepts: The best-worst Ant System,” in Proc. ANTS 2000, M. Dorigo et al., Eds., IRIDIA, Université Libre de Bruxelles, Belgium, pp.22–29, 2000.

[16] Bifan Li, Lipo Wang, and Wu Song, “Ant Colony Optimization for the Traveling Salesman Problem Based on Ants with Memory,” in ICNC 2008, Shandong University, China, pp.496–501, 2008.