Ant Colony Optimization for the Traveling Salesman Problem Based on Ants with Memory

Bifan Li[1], Lipo Wang[1,2], and Wu Song[3]

[1] College of Information Engineering, Xiangtan University, Xiangtan, Hunan, China.
Email: bevan_li@yahoo.cn
[2] Scholl of Electrical and Electronic Engineering, Nanyang Technology University,
Block S1,50 Nanyang Avenue, Singapore 639789
Email: Elpwang@ntu.edu.sg
[3] Dept. of Computer Science, QiongZhou Academy, Wuzhishan, Hainan, China.
Email: evol_song@yahoo.com.cn

Abstract

We propose a new model of Ant Colony Optimization (ACO) to solve the traveling salesman problem (TSP) by introducing ants with memory into the Ant Colony System (ACS). In the new ant system, the ants can remember and make use of the best-so-far solution, so that the algorithm is able to converge into at least a near-optimum solution quickly. We have tested the algorithm in 3 representational TSP instances and compared the results with the original ACS algorithm. According to the result we make amelioration to the new ant model and test it again. The simulations show that the amended ants with memory improve the converge speed and can find better solutions compared to the original ants.

1. Introduction

Ant Colony Optimization (ACO) [6], which studies artificial agent systems, takes inspiration from 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. Because it takes less time for ants to find food and carry them back to the nests through the shortest path, after a long enough period of time, more ants will choose this shortest path and deposit pheromone on it. In this way, all ants eventually choose the shortest path. ACO is used to solve discrete optimization problems such as the Traveling Salesman Problem (TSP) and the Quadratic Assignment Problem (QAP) [6]. The first ant algorithm was introduced by Dorigo et al. [1] in 1991 and was called the Ant System (AS) [1] [2]. Dorigo and Gambardella proposed the Ant Colony System (ACS) [4][5] later in 1996, while Stützle and Hoos proposed the MAX-MIN Ant System (MMAS) [3]. ACO has drawn much research attention and various extended versions of the ACO paradigm were proposed, such as the Best-Worst Ant System (BWAS) [8], the Rank based Ant System (RAS) [7] etc..

The main novel idea of ants with memory algorithm, to be discussed in the remainder of the paper, is the synergistic use of the previous best solution constructed by ants.

This paper is organized as follows: In section 2, we introduce the background knowledge of the TSP, AS and ACS. In section 3, we provide definition of the parameters and environment of the experiments. In section 4, we explain what ants with memory are and simulated them in ACS. In section 5 we amend the ants with memory, then we show the results of the experiments which were generated by amended ants and compare the performance of each algorithm. Finally, in section 6, we are dedicated to discuss the main characters of ants with memory and suggesting directions for further research.

2. Background

2.1 Traveling Salesman Problem

978-0-7695-3304-9/08 $25.00 © 2008 IEEE
DOI 10.1109/ICNC.2008.354

IEEE computer society

TSP is one of the most widely known NP-hard problems. In the TSP we are given a set of cities $c_i,c_j,\ldots c_N$ and the distances $d(c_i,c_j)$ for each pair of distinct cities (c_i,c_j). The salesman has to visit every city once and only once, and return to the starting city in the end. Our goal is to find a closed tour with minimal cost. In this paper, we concentrate on the symmetric TSP, where $d(c_i,c_j)= d(c_j,c_i)$ for $1\le i,j\le N$.

D198.tsp, Rat783.tsp and Pr2392.tsp, these TSP models were widely used by Dorigo et al. [2][4][5], are available at TSPLIB benchmark library [11],

2.2 Ant system

In origin AS [1] [2], all the m ants which have constructed a solution in the loop can update the pheromone. The value of pheromone τ_{ij} which contacted with the edge ij between city i and city j updated with the following formula:

$$\tau_{ij} = (1-\rho)\bullet\tau_{ij} + \sum_{k=1}^{m}\Delta\tau_{ij}^{k} \qquad (1)$$

The parameter 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：

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & \text{edge (i,j) in ant k`s tour} \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

where Q is a constant，L_k is the length of the tour which constructed by ant k in the current loop.

During constructing process, ants will visit the following city through a stochastic mechanism：While an ant locating in city i has constructed the partial solution, the probability of move to city j is given by (3) ：

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

The parameter $N(s^p)$ here is a set of suitable elements. In another word, it is a set of edges (i,l) here the parameter l means the city not visited by ant k. The parameters α and β contact with the importance between pheromone and the heuristic information which was given by (4)：

$$\eta_{ij} = 1/d_{ij} \qquad (4)$$

Where the parameter d_{ij} shows the distance between city i and j.

2.3 Ant colony system

One major difference between the ACS[4-6] and the AS is that the ACS introduces Local Pheromone Update into the algorithm at the end of each step of the construction, also known as offline pheromone update. And only when every ant explored the last edge of its tour the algorithm updates the pheromone as:

$$\tau_{ij} = (1-\varphi)\cdot\tau_{ij} + \varphi\cdot\tau_0 \qquad (5)$$

Here $\varphi\in(0,1]$ is the parameter about the pheromone decay rate, the τ_0, which is defined as $\tau_0=(L_{nn})^{-1}$, initializes the value of pheromone [4][5].

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 iteration-best or best-so-far. But there still a little different between them as formula (6) shows:

$$\tau_{ij} = \begin{cases} (1-\rho)\cdot\tau_{ij}+\rho\cdot\Delta\tau_{ij} & \text{(i,j) belongs to best tour,} \\ \tau_{ij} & \text{otherwise.} \end{cases} \qquad (6)$$

Here $\Delta\tau_{ij} = 1/L_{best}$, L_{best} could be L_{ib} or L_{bs}.

There is another important difference between the ACS and the AS while using decision rule by ants. In the ACS, ants use so called pseudorandom proportional rule [4][5], the probability of an ant move from the city i to the city j was determined by a parameter q_0 and a random variable q which is uniformly distributed over[0,1]:

$$j = \begin{cases} \arg\max_{c_{il}\in N(s^p)}\{\tau_{il}\bullet\eta_{il}^{\beta}\} & \text{if } q\le q_0 \text{ (Exploitation)} \\ \text{use formular (3)} & \text{otherwise(Exploration)} \end{cases} \qquad (7)$$

3. Experiment parameters

The environment of the experiments in the paper is listed in table 3.1.

Table 3.1 Experiment environment

CPU	RAM	OS
Double Core Intel Pentium D 2.80G	DDR 1G	Red Flag Linux

The abbreviation parameters used in table5.2 until table5.4 come from ACOTSP program which is coded by Stützle[12]. The parameters of ACS series were

given by Dorigo et al. [4][5]: m=10,α = ρ=1,β=2, $\tau_0=(n \cdot L_{nn})^{-1}$, q_0=0.98, here L_{nn} is the tour length produced by the nearest neighbor heuristic, and the local search heuristic algorithm is 3-opt.

4. Ants with memory and experiment

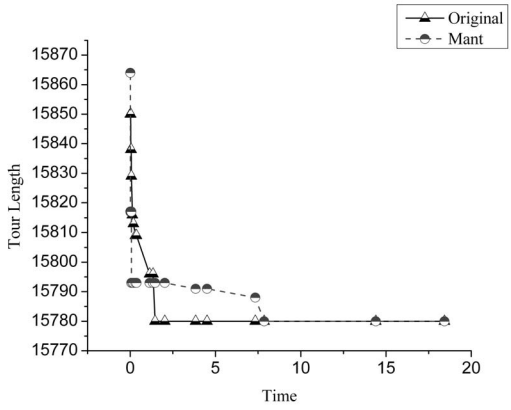
Being different from the preexistent attempts to Ant System, we are trying to introduce a novel character into the agent. We let the new ants memorize the best-so-far solution. This model was called Ants with Memory (Mant).

Generally speaking, the Mant can be used in any version of ACO algorithms. In Mant algorithm, we choose random part of the best-so-far solution and make it as the new solution proportion. The Mant based on ACO algorithm is given by table4.1:

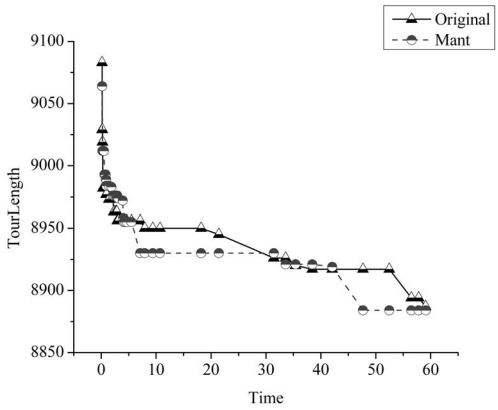
Table 4.1 Mant algorithm

Initialize parameters, pheromone trails,
While (termination condition not met)**Do**
While (not first loop)**Do**
Generate random number i, j(0<i<j<n),
Put visited tag from city i+1 to j-1
Make i as start city and j as end city
Construct solutions as normal ACO
Apply local search(could use 2-opt/3-opt etc.)
Update pheromone trails
End
End

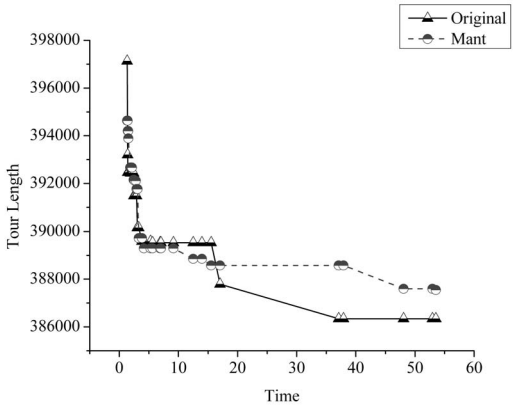
We present the result of the Mant based ACS algorithm versus original ACS algorithm simulated in three tsp models as follows:



(a)D198.tsp



(b)Rat783.tsp



(c)Pr2392.tsp

Fig4.1 Mant Based on ACS
Dashed line with dot is Mant ,
real line with triangle is original ant

We could read from the Fig.4.1 and Fig.4.2 that the Mant ACO algorithm could match the act of original ACO in D198.tsp and Rat783.tsp, but it fall behind in Pr2392.tsp while comparing with original ant ACO. We found that the Mant algorithm has strong tendence which will converge the result into the local optimum, this property can help algorithm to find optimum value faster in small or middle sized problem, however, when the scope of the problem getting larger, this tendence will also astrict the global search ability of the algorithm and fall into the local optimization trap especially.

We have to make some improvement on Mant model in next section.

5. Amelioration on Mant and experiment

5.1 Probabilistic Mant

Aimed at weaken the memory property of Mant, We introduce the probabilistic strategy into Mant. That is, we control the Mant use their memory property by a

probability as used by Dorigo in ACS[4][5](it is called pseudorandom proportional rule). We call the new amended model of Mant with probability p_1 as MantP1, for better illustration, we also provide Mant with probability p_2 as MantP2, here $p_2=1-p_1$. the detailed algorithm shows in table 5.1.

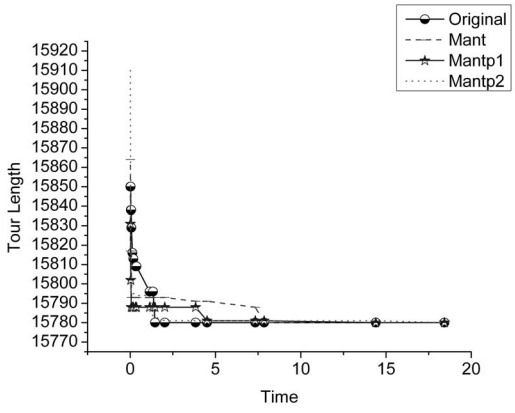
Table 5.1 MantP1 algorithm

Initialize parameters, pheromone trails,
While (termination condition not met)**Do** **While** (not first loop)**Do** Generate random number q, **If** ($q< q_0$), construct solutions as Mant **Else**, construct solutions as normal ACO Apply local search(could use 2-opt/3-opt etc.) Update pheromone trails **End** **End**

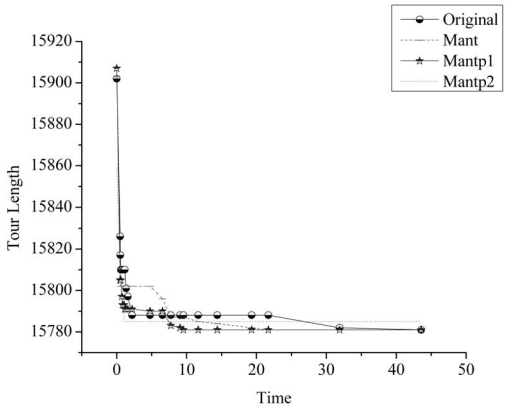
In MantP2 algorithm, we just switch ($q<q_0$) into ($q> q_0$) in MantP1 algorithm.

5.2 Probabilistic Mant in ACS

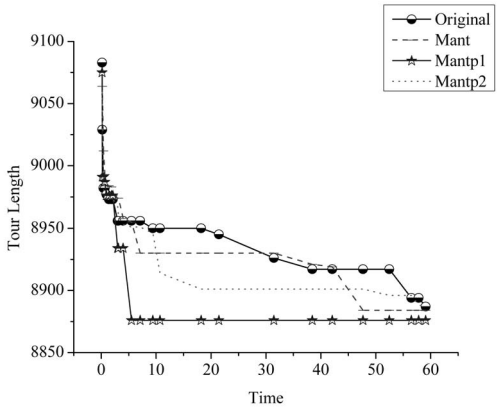
Now we could simulate Mant, MantP1 and MantP2 into ACS algorithm and compare the three Mant models to normal ant in D198.tsp, Rat783.tsp and Pr2392.tsp.



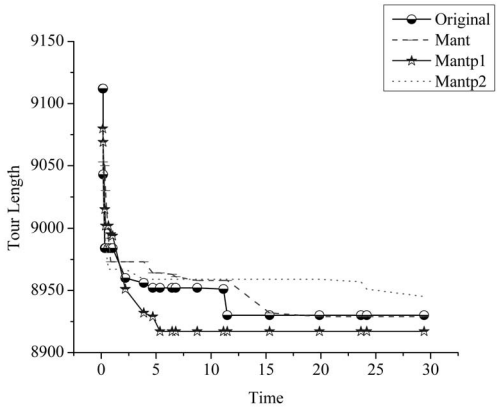
(a)D198.tsp best



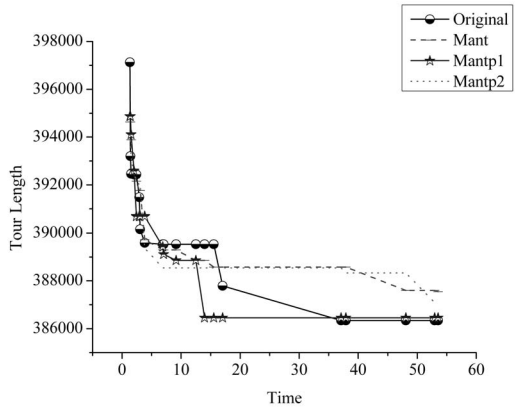
(b)D198.tsp worst



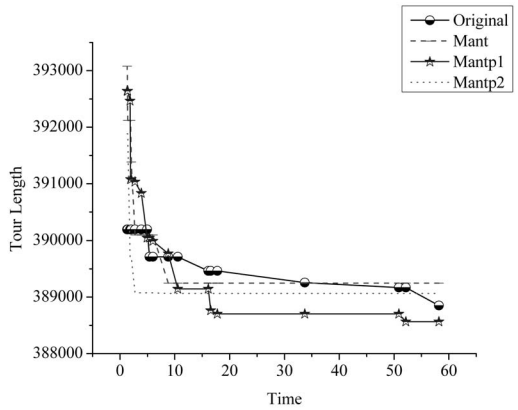
(c) Rat783.tsp best



(d) Rat783.tsp worst



(e) Pr2392.tsp best



(f) Pr2392.tsp worst

Fig.5.1 Mant, MantP1, MantP2 Based on ACS
(Real line with dot is original ant, bold dashed is Mant, real line with pentacle is MantP1 and light dashed is MantP2, best means the condition the best results the algorithms found in all loops while worst means the worst results they found)

From the upper fig.5.1(a) to fig.5.1(f), we could easily find out that the MantP1 performs the best both in best and worst condition in all three tsp models. The MantP1 could find the optimum value with less time against the original ant except Pr2392.tsp in best condition fig.5.1(e), however, the MantP1 find the better result witch is 0.5% worse than the original ant in 20 seconds earlier.

Table 5.2 On D198.tsp

Type of Mant	Average Best	Average Iterations	Avg.time best	Stddev Best
ACS	**15780.50**	4760.50	22.83	0.53
Mant	15780.80	3093.50	**14.77**	**0.42**
MantP1	15780.70	**4546.40**	22.13	0.48
MantP2	15781.10	4220.40	20.41	1.45

Table 5.3 On Rat783.tsp

Type of Mant	Average Best	Average Iterations	Avg.time best	Stddev Best
ACS	8911.90	1200.80	29.65	13.48
Mant	8907.30	1769.80	44.22	14.39
MantP1	**8906.20**	**1096.10**	**27.20**	**12.38**
MantP2	8910.70	1384.30	34.77	18.22

Table 5.4 On Pr2392.tsp

Type of Mant	Average Best	Average Iterations	Avg.time best	Stddev Best
ACS	387686.90	442.30	43.06	720.72
Mant	388345.60	360.00	**35.18**	**510.20**
MantP1	**387634.70**	**352.70**	35.40	690.81
MantP2	388270.30	361.30	35.73	570.22

The further information were show in upper three tables, the best values among four algorithms were signed by bold. Except the Average-Best in table 5.2, the statisical comparison told us that the probabilistic Mant has surpassed original ant in ACS at speed and quality while searching for the optimum solution. Even in table 5.2 the MantP1 exceeded other three competitors in all targets.

6. Conclusion

In the end, we have shown through this paper that the Mant is a very simple but interesting novel approach to the ACO system. It has been shown to compare favorably with ACS algorithm, and Its amelioration model probabilistic Mant got inspiring performance in ACS. However, competition on the TSP is very tough, and a utilization of best-so-far tour information which converges these solutions to a near-optimum seems to be a useful strategy. We still long for better performance of probabilistic Mant on the Dynamic TSP[9][10] in further study.

7. References

[1] A. Colomi, M. Dorigo, V. Maniezzo, "Distributed optimization by ant colonies," Proceedings of the 1st European Conference on Artificial Life, pp.134-142, 1991.
[2] 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.
[3] 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.
[4] 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.

[5] M. Dorigo and L.M. Gambardella. “Ant colonies for the traveling salesman problem,” BioSystems,vol.43, o.2,pp.73-81,1997.
[6] M. Dorigo, G. Di Caro and L.M. Gambardella, “Ant algorithms for discrete optimization,” Artificial Life, vol. 5, no. 2, pp. 137–172, 1999.
[7] 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.
[8] 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.
[9] M. Guntsch and M. Middendorf, “Pheromone modification strategies for ant algorithms applied to dynamic TSP,” in pplications of Evolutionary Computing: Proc. EvoWorkshops 2001, ser. LNCS, E. J. W. Boers et al., Eds., Springer Verlag, vol. 2037, pp. 213–222, 2001.
[10] C.J. Eyckelhof and M. Snoek, “Ant systems for a dynamic TSP: Ants caught in a traffic jam,” in Proc. ANTS 2002, ser. LNCS, M. Dorigo et al., Eds., Springer Verlag, vol. 2463, pp. 88–99, 2002.
[11] TSPLIB:
http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp
[12] ACO Public Software:
http://iridia.ulb.ac.be/~mdorigo/ACO/aco-code/public-software.html