ON THE ANT COLONY SYSTEM FOR POSTMAN PROBLEM

Hoang Xuan Huan, Dinh Trung Hoang
Faculty of technology, VNU

Abstract. *The ant colony system (ACS) introduced by Dorigo M. et al (see [7,8,9]) is a distributed algorithm that simulates behavior of real ants of finding the shortest path from a food source to their nest [1] in order to solve the postman problem (or traveling salesman problem). Experimental results have shown that the ACS outperforms other nature-inspired algorithms such as simulated annealing, neural nets, genetic algorithm...This paper first considers the influence of the pheromone updating parameter and the allocation of starting cities for artificial ants in order to make the algorithm more efficient in static problem. Then, we introduce framework for real time problems, using this algorithm.*

I. Introduction

Real ants are capable of finding the shortest path from a food source to their nest [1] without using visual cues by exploiting pheromone information. While walking, ants deposit chemical traces (pheromone) and follow, in probability, pheromone previously deposited by other ants to find a shortest path between two points.

The above behavior of real ants has inspired many ant algorithms (see [2-11];[16]) to efficiently solve different types of combinatorial optimization problems. In particular, ACS algorithm (Dorigo M. et al [7,8,9]) has been shown to be very efficient to solve the symmetric and asymmetric postman problems (PMP). The main idea of ACS is that of having m agents, called ants, search in parallel for good solutions to the PMP and cooperate through pheromone-mediated indirect and global communication by using a common memory that corresponds to the pheromone deposited by real ants. Informally, each ant constructs a PMP solution in an iterative way: it adds news cities to a partial solution by exploiting both informations gained from past experience and a greedy heuristic. Memory takes the form of pheromone deposited on PMP edges, while heuristic information is simply given by the edge's length. This paper first considers the influence of the pheromone updating parameter and the allocation of starting cities for artificial ants to algorithm efficiency in static problem. Experimental results have shown that the efficiency of ACS is improved when we randomly allocate starting cities for artificial ants at each iterative step.

On the other hand, in real time problems, the edge lengths are not previously known and can be stochastic processes determined during run-time. Then, we also propose a framework for this case.

This paper is organized as follows. In section II, we review the postman problem. Section III introduces briefly the ACS for static problem, which has been proposed in [9]

Typeset by $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX and [10]. Section IV is dedicated to consider the pheromone updating parameter and the allocation of starting cities for artificial ants. Section V proposes a framework for real time problems.

II. Postman problem

2.1. Static problem

The static postman problem (PMP) is a relatively old problem, it was documented as early as 1759 by Euler (though not by that mane) whose interest was in solving the knights' tour problem. A correct solution would have a knight visit each of the 64 squares of a chessboard exactly once in its tour.

General PMP can be described as follows. Let $G = (V, E)$ be a graph (simple or directed graph), V be the set of N cities, $E = \{(r, s) : r, s \in V\}$ be the edge set and $l(r, s)$ be a length (or cost) measure associated with edge $(r, s) \in E$. The PMP is the problem of finding a minimal closed tour that visit each city one. If $l(r, s) \neq l(s, r)$ for at least some $(r, s) \in E$ then the PMP is asymmetric.

This problem was proved to be NP-hard (see [12]). It arises in numerous applications and the number of cities might be quite significant as stated in [14].

2.2. Real-time Problem

Real-time problem is an extension of the static model in which the length of edges is not previously known. For every $(r, s) \in E$, its length can be measured during run time as a stochastic process of following form:

$$l(r, s, t) = g(r, s, t) + w(r, s, t), \tag{1}$$

where, $g(r, s, t)$ is trend and $w(r, s, t)$ is white noise. The Real-time problem (RPMP) is the following problem. Basing on trials at a time sequence $\{t_n\}$ before a time T and $\lim_{n\to\infty} t_n = T$, we find a good tour (in average) at the time T.

III. ACS for static problem

In this section we briefly present the ACS for the static problem (see [9],[10] for more detail).

3.1. General description

In this framework, each ant is an agent moving through cities on a PMP graph. Initially, there are m ants placed on cities selected randomly. These artificial ants also have a few capacities that natural ants have not. The ant k can determine how far it is from each city to others, and is endowed with a working memory M_k used to memorize visited cities. At each step, ants move to new cities, modifying the pheromone trail on the edges basing on state transition rule and pheromone updating rules. The process is then iterated R times, where R is selected such that it is large enough.

The shortest tour from the beginning of the trial is the solution of ACS. In general, it is a good enough solution and when R large enough may be an optimal solution. Procedure of ACS is as follows:

```
Initialize
Loop /* at this level each loop is called an iteration */
        Each ant is positioned on a starting node
        Loop  /* at this level each loop is called a step */
                Each ant applies a state transition rule to incrementally
                build a solution and a local pheromone updating rule
        Until all ants have build a complete solution
        A global pheromone updating rule is applied
Until End_Condition
```

3.2. State transition rule

In ACS for static problems (we also denote by ACS), an ant k in city r chooses the city s to move to among those which do not belong to its working memory M_k (it is emptied at the beginning of each new tour and is updated after each time step by adding the new visited city) by applying the following probabilistic formula:

$$s = \begin{cases} \arg\max_{u \in J_k(r)} \left\{ [\tau(r,u)].[\eta(r,u)]^{\beta} \right\} & \text{if } q \leq q_0 \\ S \quad \text{otherwise} \end{cases} \tag{2}$$

where $\tau(r,u)$ is the amount of pheromone trail on edge $(r,u), \eta(r,u) = 1/l(r,u)$ is a heuristic function, $J_k(r)$ is the set of remaining cities to be visited by ant k positioned on city r (to make the feasible solution), β is a parameter which weighs the relative importance of pheromone trail versus length $(\beta > 0), q$ is a value chosen randomly with uniform probability in [0,1], $q_0 \in (0,1)$ is a parameter, and S is a random variable selected according to the following probability distribution, which favors edges which are shorter and have a higher level of pheromone trail:

$$p_k(r,s) = \begin{cases} \frac{[\tau(r,s)].[\eta(r,s)]^{\beta}}{\sum_{u \in J_k(r)} [\tau(r,u)].[\eta(r,u)]^{\beta}} & \text{if } s \in J_k(r) \\ 0 \quad \text{otherwise} \end{cases} \tag{3}$$

The state transitions rule resulting from (3) is called *random proportional rule* and can be performed by using roulette-wheel procedure (see [13,15]).

3.3. Pheromone updating rules

Pheromone trail is changed both locally and globally. Global updating rule is applied only to edges which belong to the best ant tour, and local updating rule is applied to edges while ants construct a solution.

Global updating rule

Global updating is intended to reward edges, which belong to the shortest tour. After all ant have completed their tours, the best ant (1.e. the ant which constructed the shortest tour from the beginning of the trial) deposits pheromone on visited edges which belong to its tour. The pheromone level is update by applying the global updating rule of (4).

$$\tau(r,s) \leftarrow (1-\alpha)\tau(r,s) + \alpha\Delta\tau(r,s) \tag{4}$$

where

$$\Delta\tau(r,s) = \begin{cases} (L_{gb}^{-1}) & \text{if } (r,s) \in \text{global-best-tour} \\ 0 & \text{otherwise} \end{cases}$$

$0 < \alpha < 1$ is the pheromone decay parameter, and L_{gb} is the length of the globally best tour from the beginning of the trial. Expression (4) indicates that only those edges belonging to the globally best tour will receive reinforcement

Local updating rule

While building a solution (i.e., a tour), ants visit edges and change their pheromone level by applying the local updating rule of (5)

$$\delta\tau(r,s) \leftarrow (1-\rho)\tau(r,s) + \rho\delta\tau(r,s) \tag{5}$$

where $0 < \rho < 1$ is a parameter. The term $\delta\tau(r,s)$ can be defined as follows:

(i)

$$\delta\tau(r,s) = \tau_0, \quad \text{where } \tau_0 \text{ initial pheromone level.} \tag{6}$$

(ii)

$$\delta\tau(r,s) = 0. \tag{7}$$

IV. Pheromone updating parameter and starting cities

In [10], Dorigo and Gambardella has taken experiments and found that the experimental optimal values of the parameters were weakly dependent of the problem, except for τ_0. First we study the influence of τ_0 regarding algorithm efficiency.

4.1. Pheromone updating parameter

We denote by BE the optimal tour of PMP and $\gamma = L_{BE}^{-1}$, where L_{BE} is the length of BE.

Proposition 4.1.1. *For every edge $(r,s) \in E$, the following assertions holds*

$$\tau_m := \min\{\gamma, \delta\tau(r,s)\} \le \tau(r,s) \le \max\{\gamma, \tau_0\} := \tau_u. \tag{8}$$

Proof. According to expressions (4), (5) the proof is obvious by induction for iterative steps. This proposition suggests that in order to obtain an optimal solution we have to choose the initial pheromone level $\tau_0 < \gamma$.

Now, we denote by $\tau(r,s,n)$ and $BE(n)$ the pheromone level of (r,s) and the shortest tour from the beginning of the trial when the iterative step n is completed.

Theorem 4.1.2. *The following assertions are valid:*

a) The algorithm mentioned above is always convergent.

b) If there exist a n_0 such that for all $n > n_0$, (r, s) does not receive global updating pheromone then $\tau(r, s, n)$ converges in probability to $\delta\tau(r, s)$.

Proof. Denote by $L(n)$ the length of $BE(n)$. Since sequence $L(n)$ is decrease monotone and is bounded by 0, the assertion a) is obvious.

We will prove b) with local updating rule (6) (the case (7) can be proved analogously). For simplicity, we consider the symmetric graph, the asymmetric case is considered similarly. It follows from $\tau_0 < \gamma$. and (8) that

$$\tau_m = \tau_0 = \delta\tau(r, s) \text{ and } \tau_u = \gamma.$$

In expression (5) , we rewrite:

$$(1 - \rho)\tau(r, s) + \rho\delta\tau(r, s) = \tau_0 + (1 - \rho)[\tau(r, s) - \tau_0].$$

Suppose that from the iterative step n_0 to the one $n = n_0 + p$ the edge (r, s) is updated pheromone h times by local rule then:

$$\tau(r, s, n) = \tau_0 + (1 - \rho)^h[\tau(r, s, n_0) - \tau_0] \leq \tau_0 + (1 - \rho)^h(\gamma - \tau_0). \tag{9}$$

Therefore, for all arbitrary ϵ, there exist H such that $\forall h \geq H$ we have

$$\tau(r, s, n) - \tau_0 < \epsilon. \tag{10}$$

On the other hand, at each iterative step, we have an estimation of probability of event that an ant k locally update the edge (r, s)

$$p_0 = 1 - q_0 \geq P_k(r, s) \geq (1 - q_0)\tau_0\eta^\beta(r, s) / \sum_{(r,s)\in E} \gamma\eta^\beta(r, s) = a > 0, \tag{11}$$

where $a, p0 \in (0, 1)$.

Now, for all $i \leq mp$ we estimate the probability of the event that (r, s) is updated i times from the step n_0 to the one n. In each iterative step, there are m ants, then this problem can be considered as follows: there are mp ants, in any condition each ant can update the edge (r, s) with a probability estimated by (11). We number these ants from 1 to mp and denote by A_j the event that the ant j updates (r, s). from (11) we have:

$$\forall j, P(A_j) \leq p_0 \text{ and } P(\bar{A}_j) \leq 1 - a.$$

Then

$$P(A_1...A_i\bar{A}_{j+1}...\bar{A}_{mp}) = P(A_2...A_i\bar{A}_{j+1}...\bar{A}_{mp})P(A_1/A_2...A_i\bar{A}_{j+1}...\bar{A}_{mp}) \leq$$
$$\leq p_0 P(A_2...A_i\bar{A}_{j+1}...\bar{A}_{mp}).$$

Continuing by reduction we have:

$$P(A_1...A_i\bar{A}_{j+1}...\bar{A}_{mp}) \leq p_0^i(1-a)^{mp-i}.$$

Permuting the order of the ants, we receive: $P((r,s)$ is updated i times$) \leq C_{mp}^i p_0^i(1-a)^{mp-i}$. This implies that :

$$P(|\tau(r,s,n)-\tau_0| > \epsilon) \leq P((r,s) \text{ is updated less than } H \text{ time}) \leq \sum_{i=1}^{H} C_{mp}^i p_0^i(1-a)^{mp-i}$$

Then

$$\lim_{n\to\infty} P(|\tau(r,s,n)-\tau_0| > \epsilon) \leq \lim_{p\to\infty} \sum_{i-=1}^{H} C_{mp}^i p_0^i(1-a)^{mp-i} = 0$$

This completes the proof.

Comment

When we use local updating rule (7) or $\tau_0 \cong 0$, the expression $\tau_0 + (1-\rho)^h(\gamma - \tau_0)$ quickly converges to 0 and the local updating process quickly become invalid. In this case, the algorithm efficiency is worse. This coincides with the experimental results in 9 and [10]. If $\tau_0 \cong \gamma$ then pheromone level change slightly, the algorithm become nearly heuristic.

4.2. Starting cities

In [9] and [10], authors fixed starting city for each ant. This implies that when an ant arrives final city of its tours, it obligates to return to the starting city without choice although this edge may be long. Basing on this notice, we can select randomly starting city for each ant at each iterative step (motive starting cities) in order to improve the efficiency. We constructed two ACS by using two schemes:

+ Scheme 1 for the case of fixed starting cities

+ Scheme 2 for the case of motive starting cities

The ACS parameters were set $\beta = 2, q_0 = 0.9, \alpha = \rho = 0.1, \tau_0 = (NL)^{-1}$, where L is the tour length produced by the nearest neighbor heuristic and N is the number of cities. We apply these schemes for 50-city problems generated randomly and especially for problems Bayg29 and Bays29 found in TSPLIB:

http://www.iwr.uniheidelberg.de/iwr/comopt/soft/tsplib95/tsplib.html

Experimental observation has shown that scheme 2 is better than the first. The following tables present results applied for problems Bayg29 and Bays29 (with 29 cities). ACS was run for 1000 iterations and the results are averaged over 15 trials with different ant quantity m. The best tour length was obtained out of 15 trials. The best tour length and the best average tour length are in boldface.

Table 1: Applied problem is Bayg29 with

m	scheme 1		scheme 2	
	average	best	average	best
4	1673.67	1652	**1657.33**	**1642**
6	1681.83	1655	**1659.17**	**1634**
8	1658.5	1644	**1645.33**	**1631**
10	1648.83	1634	**1646.67**	**1627**
15	1649.5	1641	**1641.5**	**1624**

Table 2: Applied problem is Bays29

m	scheme 1		scheme 2	
	average	best	average	best
4	2061.67	2045	**2047.33**	**2036**
6	2051.33	2036	**2050**	**2034**
8	2033.67	**2020**	**2033**	**2020**
10	2037.33	2033	**2030.67**	**2020**
15	2034.33	2028	**2024.67**	**2020**

V. A framework for real time problem

5.1. Description

As mention above, in $RPMP$ the length of every edge $(r, s) \in E$ is a stochastic process and not previously known. It has the form (1): $l(r, s, t) = g(r, s, t) + w(r, s, t)$, and can be measured at a time sequence $\{t_n\}(t_n < T)$ and $\lim_{n\to\infty} t_n = T$. Basing on this data set we will find a good tour (in average) at T.

For every edge (r, s), in common memory we use two variables $l(r, s)$ and $T*(r, s)$ in order to store average length of (r, s) and the number of times that (r, s) are visited. The algorithm is composed of two stages: initial stage and ant colony stage.

Initial stage. We measure values $l(r, s, t_0)$ of all edges at time t_0 and set: $l(r, s) = l(r, s, t_0), T*(r, s) = 1$ for every edge (r, s). Then we set the initial pheromone level $\tau_0 = (nL_0)^{-1}$, where L_0 is the tour length produced by the nearest neighbor heuristic for the PMP with edge lengths $l(r, s, t_0)$.

Ant colony stage. We use m artificial ants to measure data. Operation of artificial ants is similar to those in static problem with some modifications. At each time t_n, we also denote by $l_k(r, s, t_n)$ the length value of edge (r, s) measured at this time by an ant k. When visiting edge (r, s) at time t_n an ant k measures value $l_k(r, s, t_n)$, changes variables $l(r, s)$ and $T*(r, s)$ by applying updating variable rules :

$$l(r, s) \leftarrow [l(r, s)T*(r, s) + l_k(r, s, t_n)]/[T*(r, s) + 1], \qquad (12)$$

$$T*(r, s) \leftarrow T*(r, s) + 1. \qquad (13)$$

Then it applies local updating rule by (5). The state transition is not changed.

Global updating rule is modified by *iteration-best* type, instead of global-best type in subsection 3.3. In this type, value L_{gb} in (4) is replaced by L_{ib} (the length of the best tour in current iteration of the trial) and the best ant of this iteration deposits pheromone on its path.

The following is basic for our framework.

Theorem 5.2. *Suppose that in (1) $t = t_n$ and*

$$\lim_{n\to\infty} t_n = T, \quad \lim_{n\to\infty} g(r,s,t_n) = g(r,s,T) \tag{14}$$

then the above variable $l(r,s)$ converges in probability to expectation of $l(r,s,T)$

Proof. Since (12) and (13) , at each iterative step the value $l(r,s)$ is updated by the average of all random values $l_k(r,s,t_h)$ where h is from time t_0 to time t_n. According to (14) and the fact that $W(r,s,t)$ is white noise we easy receive the conclusion of theorem.

By this framework, when n is large enough and t_n near to T we have a good enough solution for $RPMP$.

References

1. R. Beckers; J.L. Deneubourg; and S. Goss. Trails and U-turns in the selection of the shortest path by the ant Lasius niger, *Journal of Theoretical Biology,* **159**(1992). 397-415
2. H. Bersini; M. Dorigo; S. Langerman; G. Seront and L.M. Gambardella. Result of the first international contest on evolution optimization, *Proc. IEEE. Int.Conf. Evolutionary computation, IEEE-EC,* (1999) 611-661.
3. B. Bullnheimer; R. F. Hart; and C. Strauss. *Applying the ant System to the Vehicle Routing Problem,* in metaheuristics: Advances and Trend in local Search for Optimization, S. Martello, I.H. Osman and C. Roucairol, Kluwer Academic Publishers, Boston (1999) 285-296.
4. G.D. Caro; and M. Dorigo. Ant net: A mobile for Adaptive Routing, *Technical Report 97-12 IRIDIA,* Universite' Libre de Bruxelles, Belgium, 1997.
5. G.D. Caro; and M. Dorigo. Ant net: Distributed stigmergetic control for communications networks, *Technical Report 98-01 IRIDIA,* Universite' Libre de Bruxelles. Belgium, 1998.
6. D. Costa; and A. Hertz. Ants can colour Graphs, *Journal of the Operational Research Society,* **48**(1997) 295-305.
7. M. Dorigo; V. Maniezo and A. Coloni. Positive feedback as a search strategy, *Technical Report 91-019,* Departimento di eletronica e informatica, Poletico di Milano, IT, 1991.
8. M. Dorigo. Optimization learning and natural algorithm, *PhD. dissertation politechnico dimilecno,* Italy,1992
9. M. Dorigo; V. Maniezo and A. Coloni. The Ant System: optimization by a colony of cooperating agents, *IEEE, Trans. Syst., Man, Cybern. B,* vol 26, no. **2** (1996) 29-41
10. M. Dorigo and M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem, *IEEE Trans. on evolutionary computation,* vol, no **1**(april 1997) 53-66.

11. L.M. Gabardeella; E. Taillard and G. Agazzi. MACS-VRPTW: A multiple Ant colony System for Vehicle Routing Problems with time windows. *Technical Report IDSIA 06-99*, Lugano, Switzerland,1999.
12. M. Garey and D. Johnson. *Computers and intractability*. W.H. Freeman, Sanfrancisco, 1990.
13. Hoang Xuan Huan & Nguyen Viet Thang. An evolutionary solution for timetable problem. *Journal of computer and cybernetics, National center for natural sciences and technology of Vietnam*, vol.17,n.**2**(2001), 87-96.
14. D.S. Johnson. Local Optimization and the Traveling Salesman Problem, Proc. of 17 th colloquium on automata, languages and programming, *Lecture notes in computer science*, vol 443, spring verlag (1990) 446-461.
15. Z. Michalewicz. *Genetic algorithm + Data Structure = Evolution Program*. Berlin-Germany, Springer, 1996.
16. R. Schoonderwoerd; O. Holland; J.Bruten and L. Routhkrantz. Ant-based load balancing in telecommunications networks, *Adaptive behavior*, vol 5, no **2**1997.
17. T. Stuzle, M. Dorigo. ASO algorithms for Traveling Salesman Problem, *Evolutionary algorithms in Engineering and Computer Science: Recent Advances in Generic Algoritms, Evolution Strategies, Evolutionary Programming, Genetic Programming and Industrial Applications*, P. Neittaanmaki, J.Periaux, K. Miettinen and M. Makela, (eds), John Wiley & Sons, 1999.

TẠP CHÍ KHOA HỌC ĐHQGHN, KHTN & CN, t.XVIII, n⁰1 - 2002

VỀ HỆ ĐÀN KIẾN CHO BÀI TOÁN NGƯỜI ĐƯA THƯ

Hoàng Xuân Huấn, Đinh Trung Hoàng

Khoa Công nghệ, ĐHQG Hà Nội

Hệ đàn kiến (ACS) là thuật toán phân tán mô phỏng cách tìm đường ngắn nhất từ nguồn thức ăn về tổ của các con kiến thực (xem [7, 8, 9]). Các kết quả thực nghiệm cho thấy nó là thuật toán nổi trội so với các thuật toán nổi trội so với các thuật toán mô phỏng tiến hoá tự nhiên khác như: luyện kim, giải thuật di truyền, mạng nơron... Trong bài này chúng tôi khảo sát theo cách phân tích toán học về ảnh hưởng đối với hiệu quả bài toán của tham số cập nhật mùi và phân bố các điểm xuất phát cho mỗi con kiến để cải tiến thuật toán.

Ngoài ra, các bài toán đang sử dụng hệ đàn kiến thường là bài toán thời gian thực. Để đáp ứng nhu cầu xuất phát từ các bài toán này, chung tôi giới thiệu một lược đồ cho bài toán thời gian thực cho khi độ dài các cạnh là các quá trình ngẫu nhiên và tìm lời giải dựa vào dữ liệu ở các thời điểm trước đó.