DOI: 10.2298/CSIS120302057G

An Ant System based on Moderate Search for TSP

Ping Guo and Zhujin Liu

College of Computer Science, Chongqing University, Chongqing, China
Chongqing Key Laboratory of Software Theory & Technology, Chongqing 400044, China
guoping@cqu.edu.cn

Abstract. Ant Colony Optimization (ACO) algorithms often suffer from criticism for the local optimum and premature convergence. In order to overcome these inherent shortcomings shared by most ACO algorithms, we divide the ordinary ants into two types: the utilization-oriented ants and the exploration-oriented ants. The utilization-oriented ants focus on constructing solutions based on the learned experience like ants in many other ACO algorithms. On the other hand, inspired by the adaptive behaviors of some real-world Monomorium ant species who tend to select paths with moderate pheromone concentration, a novel search strategy, that is, a completely new transition rule is designed for the exploration-oriented ants to explore more unknown solutions. In addition, a new corresponding update strategy is also employed. Moreover, applying the new search strategy and update strategy, we propose an improved version of ACO algorithm—Moderate Ant System. This improved algorithm is experimentally turned out to be effective and competitive.

Keywords: Ant Colony Optimization; Adaptive behavior; Traveling Salesman Problem; Local optimum; Premature convergence

1. Introduction

Combinational optimization (CO) problems exist widely in the real world and many of them are non-deterministic polynomial time hard (NP-hard). NP problems are decision problems where the solutions can be recognized in polynomial time by a non-deterministic Turing machine. And NP-hard problems are at least as hard as the hardest problems in NP. Such problems need not be in NP; indeed, they may not even be decision problems [1,2]. As a particular kind of optimization problems, CO problems play an important role be it in theory or in practice. Traditional algorithms can obtain the optimal solution to NP-hard CO problems but often at the cost of great computational time, which is infeasible for any practical application [3]. Swarm intelligence, inspired by the social behaviors of real animals, provides a novel way to deal with such complicated optimization problems [4]. Swarm intelligence algorithms, which usually are approximation algorithms, can compute satisfactory solution in markedly less time. Particularly, ant colony optimization (ACO) is a typical sort of SI algorithms. For many problems, ACO algorithms generate competitive results that are very close to the best known solutions while on some problems like quadratic assignment problem (QAP) they are the state-of-the-art [5].

ACO takes inspiration from the foraging behaviors of real ants which can always find the shortest path between their nests and food sources. The famous experiment, known as “shortest bridge” experiment, simulates the behaviors and explains the optimization ability [6,7]. When there are more than one path between a nest and a food source, each ant will choose one path by chance. And the ants will return to their nest as soon as they arrive at the food source. Meanwhile, they lay pheromone on the paths they have passed. It is obvious that it takes less time for the ants that choose the shorter path to get back to the nest. Accordingly, the shorter path gains more pheromone from ants than longer paths because of the high round-trip frequency. After that, ants in the nest tend to choose the path with higher pheromone concentration instead of random selection when departing for the food source. Thus, an increasing number of ants will be attracted to the shorter path and those ants will lay more pheromone on the shorter path over time. Eventually, the path between the nest and the food source selected by almost all of the ants, that is, the shortest path, is found. An individual ant has little intelligence. However, a swarm of ants achieves incredible high intelligence [8]. The term “stigmergy” is used to illustrate the distinguished capability that owned by ants as well as other animals. Specifically, stigmergy, refers to a self-organized mechanism which describes such an activity: a number of individuals interact with each other indirectly via modifying the common environment which serves as a medium; in other words, ants affect the local environment and the environment feeds back to ants, thus, ants exchange information; as a consequence, the interaction contributes to the update of the environment which affords a new platform for new interaction 5. Drawing on this mechanism, a multitude of ant algorithms have been created and ACO meta-heuristic is used to describe these algorithms in a more general way [9], however, ant algorithms are not confined to ACO.

The usual method to examine the effectiveness of a new algorithm is to apply it to some problems and compare its performance with that of already known algorithms. Traveling Salesman Problem (TSP), probably the best known instance of NP-hard problems [100], is firstly used to evaluate ACO’s performance, which indicates its central role in the development of ACO [11]. Subsequently, TSP has served as a benchmark for the sake of comparison with other optimization methods. TSP stems from a real-world problem: given a set of cities and their pairwise distances, the goal is to find the shortest tour that visits each city exactly once. In more formal terms, the task is to find a Hamiltonian circuit of minimal length on a fully connected graph [12]. There are quite a few TSP data sets that are available, and our experiments are based on these data sets as well.

Overall, the existing ACO algorithms share similar search and update strategies. More precisely, the transition rules that they employ render them in favor of components with higher pheromone concentration during the construction of feasible solutions. And update rules are more likely to give the better solutions more pheromone when updating the pheromone trails [13]. It is evident that transition rules and update rules supplement each other. The average quality of solutions benefits from the selection bias and update bias indeed. On the other hand, taking into account that ACO is an iterative approach, the biases may lead to premature convergence and local optimum, that is, limit the discovery of new or better answers, which is the starting point of our research. As a matter of fact, not all species of ants in the real world tend to select paths with more pheromone. According to some new research [14], some kinds of ants will strike a balance between paths with different quantity of pheromone. They prefer paths with moderate pheromone rather than paths with overly high or overly low pheromone concentration. Inspired by this, we present an improved ACO algorithm by bring in an extra transition rule and the counterpart update rule. The improved ACO algorithm is experimentally proven that it can obtain better solutions than other ACO algorithms in most cases.

The rest of this paper is organized as follows: Section 2 introduces several main ACO algorithms and discusses their design ideas. In Section 3, the original idea of Moderate Ant System (MAS) is explained and we present the search strategy and update strategy of MAS as well as the concrete algorithm. Section 4 gives a performance comparison and analysis between our improved algorithm and main ACO algorithms. Finally, Section 5 concludes the paper.

2. Related Work

A lot of ACO algorithms have been proposed since Dorigo et al. introduced the first ant algorithm in 1996. Here we present typical ones of them: Ant System (AS) [11,15], Max-Min Ant System (MMAS) [16] and Ant Colony System (ACS) [17,18]. In order to be in lines with our experiments, we use TSP as a specific instance to describe the algorithms.

For the convenience, Let G_{best} is the global best solution.

2.1. Ant system

As mentioned above, AS is the first ACO algorithm presented, and meanwhile, it is the original version of various ant algorithms as well. Therefore, we give a relatively complete but brief introduction of AS first, and the same details are omitted when introducing other ACO algorithms. Generally, AS and most of other ACO algorithms share the analogous framework in terms of implementation. The main framework of ACO algorithms including AS is shown in Algorithm 1.

```
Algorithm 1  The framework of ACO algorithms
  Initialize the parameters and pheromone trails;
  while termination conditions not satisfied do
    Step 1: Each ant constructs a solution
    Step 2: Update the pheromone trails
  end while
end
```

It can be seen that the core of the algorithms is a loop body and each loop consists of two crucial steps: the construction of solutions and the update of pheromone trails. The loop body, in fact, is an iterative process through which the accumulation of pheromone can play a role.

Iteration is defined as the interval in (t, t+1) during which each of the m ants constructs a solution. In other words, every ant traverses all of the n cities in each iteration. AS together with other ant algorithms employs the tabu table to record the already visited cities of an iteration in order to obtain feasible answers. In Step 1 of Algorithm 1, during the construction of a solution, ants choose the next city to be visited via certain search strategy which is a stochastic mechanism [19]. When ant k is in city i, the probability of going to city j is given by:

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum\limits_{l \notin tabu_k} \tau_{il}^{\alpha}\eta_{il}^{\beta}} & if\ j \notin tabu_k \\ 0 & otherwise \end{cases} \quad (1)$$

Where, $tabu_k$ is the tabu table of ant k which is comprised of the cities that have been visited in the current iteration. The parameters α and β control the relative importance of the pheromone τ_{ij} versus the visibility η_{ij}. The visibility represents the heuristic information, which is given by:

$$\eta_{ij} = 1/d_{ij} \quad (2)$$

Where, d_{ij} denotes the distance between city i and city j. This transition rule is also shared by other algorithms.

After each iteration, Step 2 of Algorithm 1 is performed. The pheromone τ_{ij} linked to the path connecting city i and j, is updated as Eq. (3).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \quad (3)$$

Where ρ is the evaporation rate, m is the number of ants, and $\Delta\tau_{ij}^{k}$ is the quantity of pheromone laid on path(i,j) by ant k:

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_K & if\ ant\ k\ used\ path(i,j)\ in\ its\ tour \\ 0 & otherwise \end{cases} \quad (4)$$

Where, Q is a constant and L_k is the tour length of the solution obtained by ant k.

2.2. Ant system with elitist strategy

AS_{elite} is the fisrt improved version of the original AS. In order to make the best-so-far solution more attractive to ants in the follwing iterations, the best-so-far solution is given extra pheromone. The best-so-far solution is called global-best solution and the ants who obtain this solution are called elitist ants. The update formula of pheromone is given by:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + n_e \Delta\tau_{ij}^{e} \tag{5}$$

Where e denotes the elitist ants, n_e denotes the number of elitist ants and $\Delta\tau_{ij}^{e}$ is defined as follows:

$$\Delta\tau_{ij}^{e} = \begin{cases} Q / L^{gb} & if\ path(i,j) \in G_{best} \\ 0 & otherwise \end{cases} \tag{6}$$

Where L^{gb} is the tour length of global-best solution.

The elitist strategy enables AS to improve the quality of average solution and to find better solutions at earlier time. However, the strategy narrows the difference between selections of different ants.

2.3. Rank-based Ant System

AS_{rank} draws on the concept of ranking and adopts it into the pheromone update procedure. The n ants are ranked according to the quality of their solutions in decresing order (i.e., $L_1 \le L_2 \le \ldots \le L_m$) . σ is a parameter of AS_{rank}. Only the paths included in the solutions of σ-1 best ants acquire extra pheromone and the amout depends directly on the ant's rank μ and the quality of its solution. Moreover, the paths belong to global-best solution L^{gb} receive an additional amount of pheromone which depends on the length of L^{gb}, weighted by parameter σ. The pheromone update function is given by:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sigma\Delta\tau_{ij}^{e} + \sum_{\mu=1}^{\sigma-1} (\sigma-\mu)\Delta\tau_{ij}^{\mu} \tag{7}$$

where $\Delta\tau_{ij}^{e}$ as Eq.(6) and $\Delta\tau_{ij}^{\mu}$ as Eq.(8).

$$\Delta\tau_{ij}^{\mu} = \begin{cases} \dfrac{Q}{L_{\mu}} & \text{if the } \mu th \text{ best ant travels } path(i, j) \\ 0 & otherwise \end{cases} \tag{8}$$

2.4. Max-Min Ant System

MMAS differs from the AS in two main aspects: only the best ant is allowed to update the pheromone trails, and the value of pheromone on the paths is bound. The pheromone update function is implemented as follows:

$$\tau_{ij}(t+1) = [(1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}^{best}]_{\tau_{\min}}^{\tau_{\max}} \tag{9}$$

Where τ_{max} and τ_{min} are the upper bound and lower bound of the amount of pheromone on the paths, respectively. And $[x]_b^a$ as Eq.(10) and $\Delta\tau_{ij}^{best}$ as Eq.(11).

$$[x]_b^a = \begin{cases} a & if\ x > a \\ b & if\ x < b \\ x & otherwise \end{cases} \tag{10}$$

$$\Delta\tau_{ij}^{best} = \begin{cases} \dfrac{1}{L_{best}} & path(i, j) \in G_{best} \\ 0 & otherwise \end{cases} \tag{11}$$

L_{best} is the length of the tour obtained by the best ant. It can be the best solution found in the current iteration—iteration-best, L_{ib} or the best solution found since the beginning of the algorithm—global-best, L_{gb} , or a combination of both.

With respect to the upper and lower bounds on the pheromone values, τ_{max} and τ_{min} tend to be set empirically and are determined on a case-by-case basis. Nevertheless, there are still some guide lines that have been proposed for defining τ_{max} and τ_{min} on the basis of analytical considerations.

2.5. Ant Colony System

The most significant contribution of ACS is the introduction of a local pheromone update in addition to the pheromone update applied at the end of the construction process, which is more in line with the natural behavior of real ants.

The local pheromone update is executed by all the ants after each construction step of the *n* steps of an iteration. Moreover, each ant applies it only the last path which has just been traversed:

$$\tau_{ij} = (1-\varphi)\tau_{ij} + \varphi\tau_0 \tag{12}$$

Where $\varphi \in (0, 1]$ is the pheromone decay coefficient, and τ_0 is the initial value of the pheromone trail on the path.

Similar to the MMAS algorithm, the pheromone is also updated at the end of each iteration by the iteration-best or the global-best ant only. However, the update function is slightly different:

$$\tau_{ij}(t+1) = \begin{cases} (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij} & path(i, j) \in G_{best} \\ \Delta\tau_{ij} & otherwise \end{cases} \tag{13}$$

where

$$\Delta\tau_{ij} = 1/L_{best} \tag{14}$$

As in MMAS, L_{best} can be either L_{ib} or L_{gb}.

ACS also uses a different transisiton rule, called the pseudorandom proportional rule. If let *k* be an ant located on city *i*, $q_0 \in [0, 1]$ be a parameter and *q* be a random value in [0, 1], then the next city *j* is selected according to Eq.(15) with $q \le q_0$, else Eq.(1).

$$p_{ij}^k = \begin{cases} 1 & if\ j = \arg\max\{\tau_{ij}\eta_{ij}^{\beta} \mid j \notin tabu_k\} \\ 0 & otherwise \end{cases} \tag{15}$$

Additionally, ACS employs the candidate lists to make the selections prefer some nearer cities during the construction process. In the TSP instance, each of the *n* has a candidate list. The city *i*'s candidate list consists of a certain number of cities which are the closest ones to city *i*. Obviously, the candidate lists of the cities can be built prior to the loop body and they remain stable when performing the algorithm. When an ant is at city *i*, it will choose the next city among those of city *i*'s candidate list which are not visited yet in the current iteration. Only if all of the cities in the candidate list are included in tabu table, would the other cities be selected according the transition rule.

3. Moderate Ant System

3.1. Design Ideas of MAS

Typically, the designs of existing ant algorithms manifest two common apparent principles: ants prefer paths with higher pheromone concentration when constructing a feasible solution at an iteration; the paths belong to the good solutions tends to gain greater amount of pheromone when updating the pheromone trails. Clearly, the two principles strengthen each other. These two principles are able to improve the quality of the solutions and speed up the convergence to some extent.

However, if ants select the paths at a probability in proportion to the pheromone concentration, the chances of finding new solutions would decrease over time. As can be seen from Figure 1, suppose that there is only one food source in the upper path at the beginning, definitely, all the ants will converge to the upper path, and new ants will make the same choice as well because of the great amount of pheromone laid on the upper path, even when a new food source appears in the lower path. According to Figure 2, there are two food sources in the two paths. The upper food source is nearer to the nest than the lower food source. Accordingly, ants go there and back between the upper food source and the nest more frequent and deposit more pheromone on the upper path. Thus, ants will converge to the upper path eventually. Under both the two circumstances, the ability to find new food sources (solutions) are limited by selection bias and update bias of traditional principles which are illustrated above when solving a concrete problem such as TSP.

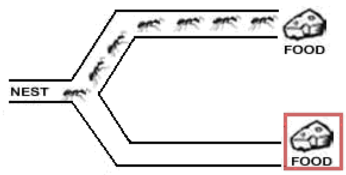

Fig. 1. A new food source appears after convergence

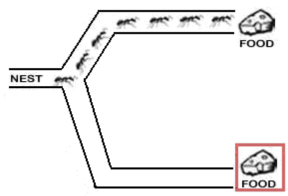

Fig. 2. Two food sources in different length of paths

Actually, the advantages that acquired by applying the principles are accompanied by premature convergence and local optimum which are highly criticized. In consideration of this, we intend to explore other principles to overcome these drawbacks. In particular, some research in animal behavior field offers us an alternative way to design new ACO algorithms.

In the natural world, there are some Monomorium ant species (e.g., M.niloticum, M.najrane, M.mayri) that dislike overly high pheromone concentration. Instead, moderate pheromone concentration renders them more active and paths with too high or too low pheromone concentration induce either no response or repellency [14]. This is the adaptive behavior of the three Monomorium ant species. They regard paths with moderate amount of pheromone as promising sources and tend to choose them, which seems to run contrary to the traditional principles used in ant algorithms.

In the view of traditional design ideas, ants have a better chance to choose paths with higher pheromone concentration. There are two main reasons that cause the high pheromone. Firstly, if there are numerous ants traveling the same path, it indicates that ants for the food source on the path are enough. Thus, a multitude of ants travel the path and lays a great amount of pheromone on the path. Secondly, if ants can frequently travel the same path, it indicates that the food source on the path is close to the nest and do not require too much ants. In this case, a few ants travel the path frequently and also deposit a great amount of pheromone. Two different behaviors end with the same result. At a deeper level, it involves an important problem to ants, that is, distributing the workforce. Be it sufficient ants selecting a path or ants traveling a path frequently enough, it is unnecessary to allocate more ants to the path whose pheromone concentration is already high enough. Otherwise, it would be a waste of workforce. Therefore, the Monomorium ant species' adaptive behavior (tend to choose the paths with moderate pheromone concentration) is understandable and reasonable. It equips the ants with the ability to distribute their workforce more probably and efficiently.

Inspired by the adaptive behavior of the Monomorium ant species, we develop a novel ACO algorithm -- Moderate Ant System (MAS). In MAS, a new transition rule which can enhance the capability to find new solutions is designed. Furthermore, in order to match the new transition rule and improve the performance of the algorithms, new update rule is designed as well.

3.2. Concrete Algorithm

MAS is an improved version based on AS, differs from AS in search and update strategy (i.e., transition rule and update rule) corresponding to the two main steps of Algorithm 1.

Search strategy of MAS. Selection bias contributes to the average quality of solutions while it brings the premature convergence and local optimum at the same time. In order to conquer the disadvantages, more explorative operations are needed. Meanwhile, as a sort of iterative algorithms, the solutions are improved gradually, so the average quality of solutions is supposed to be guaranteed as well. Therefore, a balance should be struck between exploring new knowledge and exploiting the already-known knowledge. In the ACO algorithms introduced in Section II, all the ants fulfill the same obligation. To be more exact, all of the ants construct the solutions according to the same transition rule. For that reason, in MAS, the ants are divided into two categories: the utilization-oriented ants and the exploration-oriented ants. The two sorts of ants are entrusted with different tasks, respectively.

First, actually, the utilization-oriented ants share the same business with that of the ants in most ACO algorithms. This kind of ants mainly concentrates on utilizing the search experience accumulated by the iterative process in previous iterations. They choose the paths with higher pheromone concentration at a higher probability. During the construction of solutions, the utilization-oriented ants make their choices according to the probability distribution given by Equation 1 which is same to the ordinary ants.

Second, the design of exploration-oriented ants is inspired by the Monomorium ant species and is the core of MAS. The utilization-oriented ants perform a biased search and they can choose poor paths (with lower pheromone concentration) but at a low probability. Unlike the utilization-oriented ants, in order to explore more possibilities, the exploration-oriented ants employ a more aggressive transition rule which is discussed in detail.

The pheromone concentration determines the choices of ants in real world, while in ACO algorithms, the combination of pheromone and visibility determines the choices of ants for the sake of better performance. Here, $info_j$ is used to represent the integrated factors on the path from city *i* to city *j*, and $info_{ij}$ is given by:

$$info_{ij} = \tau_{ij}^{\alpha}\eta_{ij}^{\beta} \tag{16}$$

Drawing on the search strategy of Monomorium ant species, the new transition rule should make the ants tend to select paths with moderate *information* instead of too much or too little *information*.

Suppose that ant *k* is at city *i* at step *t* of an iteration. When selecting the next city as part of the solution, ant *k* has *n-t* adjacent cities available. According to Figure 3, from city *i*'s point of view, the *information* values on the *n-t* adjacent paths can be regarded as a stochastic variable follow a normal distribution $N(\mu,\sigma^2)$ since *n* in TSP is usually a big enough number.

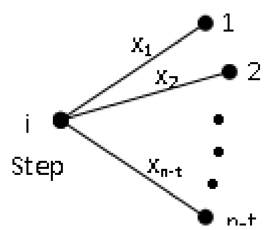

Fig. 3. n-t choices of an ant at city i at step t

The parameters of N(μ,σ2) are estimated as follow:

$$\mu = \frac{1}{n-t}\sum_{j \notin tabu_k} info_{ij} \tag{17}$$

$$\sigma^2 = \frac{1}{n-t} \sum_{j \notin tabu_k} (info_{ij} - \mu)^2 \tag{18}$$

In addition, $attraction_{ij}$ is defined as follows:

$$attraction_{ij} = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(info_{ij}-\mu)^2}{2\sigma^2}} & if\ j \notin tabb_k \\ 0 & otherwise \end{cases} \tag{19}$$

Which denotes the attraction of path from city *i* to city *j* for the exploration-oriented ants in MAS. If let *k* be an ant located on city *i*, $q_0 \in [0, 1]$ be a parameter, and *q* be a random variable uniformly distributed in [0, 1], then the exploration-oriented ants will choose the next city *j* according to Eq.(20) with $q \leq q_0$, else Eq.(19) is used.

$$p_{ij}^k = \begin{cases} 1\ if\ \ j = \arg\max_{j \notin tabu_k} arrtration_{ij}, \\ 0\ \ otherwise, \end{cases} \tag{20}$$

With probability q_0, the exploration-oriented ants select the most moderate path in terms of *attraction*, while with probability (1- q_0), the exploration-oriented ants incline to select the more moderate paths but have a chance to select other paths.

The number of the utilization-oriented ants and the number of the exploration-oriented ants are important parameters of MAS. Tuning their numbers can affect the degree of utilization and exploration (i.e. decide whether to attach more importance to utilizing the learned search experience or exploring new solutions). If there are far more exploration-oriented ants than utilization-oriented ants, the algorithm is about to lose the benefit brought by the iterative process on the basis of relatively good solutions and achieve poor performance.

Update strategy of MAS. In order to achieve the greatest effect of our new transition rule, the counterpart update strategy is designed and employed in MAS. The update strategy in MAS consists of two parts: the evaporation of pheromone trails and the accumulation of pheromone trails. The evaporation lessens the pheromone value on paths while the accumulation adds the pheromone value on paths.

In MAS, similar to the phenomenon in the real world, the pheromone on all of the paths evaporates over the time, which is implemented by:

$$\tau_{ij}'(t) = (1-\rho)\tau_{ij}(t) + \rho\tau_0 \tag{21}$$

Where $\tau_{ij}'(t)$ represents the pheromone concentration on the path between city *i* and city *j* after evaporation but before given extra pheromone at the *t*th iteration. Additionally, τ_0 is initialized by a very small value. It is evident that the lower bound of the pheromone value on a path is τ_0. The evaporation can avoid unlimited accumulation of the pheromone trails. Moreover, if a path does not gain extra pheromone in several consecutive iterations, its corresponding pheromone concentration decreases rapidly until the pheromone trail value is equal to τ_0.

After evaporation, the accumulation of pheromone trails is performed. Not all of the paths gain extra pheromone. As in AS_{rank}, only paths belonging to the best n_b-1 solutions in an iteration are given extra pheromone in MAS and the amount of extra pheromone that paths gain depends on the rank of solutions. The better the rank is, the more pheromone paths of the solutions gain. These paths are provided with strong additional reinforcement and in the following iterations will attract more utilization-oriented ants. However, the update rule is quite different from that of AS_{rank}, which is given by:

$$\tau_{ij}(t+1) = \tau_{ij}'(t) + \sum_{\mu=1}^{n_b-1} \frac{n_b - \mu}{n_b} (\Delta\tau_{ij} - \tau_{ij}'(t)) \quad (22)$$

It means that n_b-1 ants (be it utilization-oriented ants or exploration-oriented ants) will deposit extra pheromone to the paths they visited at the current iteration. It is notable that these n_b-1 ants accumulate the pheromone trails one by one and $\tau_{ij}'(t)$ is updated at the same time, so $\tau_{ij}'(t)$ may differ from ant to ant. Hence, it is easy to prove that the upper bound of the pheromone value on a path is $\Delta\tau_{ij}$. According to our research, using L_{gb} to calculate $\Delta\tau_{ij}$ in MAS achieves better performance than using L_{ib}, especially when MAS is applied to TSP data sets with more than 200 cities.

MMAS employs pheromone limits to prevent the pheromone value on the paths from overly low or overly high, thus, MMAS is able to avoid algorithm stagnation. In particular, MAS implements the similar limits in an implicit manner. The pheromone trails in MAS are guaranteed that $\forall(i,j)$: $\tau_0 \le \tau_{ij} \le \Delta\tau_{ij}$, which will contribute to the performance of MAS as well.

Pseudo-code of MAS. On the basis of the designed search strategy and update strategy, as can be seen in Algorithm 2, the pseudo-code of MAS is given.

```
Algorithm 2  Moderate Ant System
  Initialize the parameters and pheromone trails;
  while termination conditions not satisfied do
    begin
      Step 1:
        For each utilization-oriented ant
          Construct a solution according to Eq. 1.
        End For
        For each exploration-oriented ant
          Construct a solution according to Eq. 19, 20.
        End For
      Step 2:
        For all the paths
          Update the pheromone trail as Eq. 21.
```

```
        End For
        For the paths belonging to the nb best solutions
          Update the pheromone trail use Eq. 22.
        End For
    end while;
end.
```

4. Experimentation

To show the performance of MAS and compare it with other ACO algorithms (i.e., AS, AS_{elite}, AS_{rank}, MMAS, and ACS), these experimentations are completed that based on the TSPLIB's data sets (http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/).

4.1. Data Sets Used

In the following experimentations, 10 test data sets are employed. Table 1 illustrates the data sets and their size used in the experiments.

Table 1. TSP test data sets

Data set	Number of cities
att48.tsp	48
eil76.tsp	76
u159.tsp	159
ts225.tsp	225
pr439.tsp	439
rat575.tsp	575
p654.tsp	654
d1291.tsp	1291
vm1748.tsp	1748
u2152.tsp	2152

4.2. Parameters Setup

Our experimentation is based on ACOTSP which offers a high-quality implementation of various ACO algorithms for TSP [20]. With the common code, MAS is developed and compared in the same framework. And the parameters adopted in the experimentation are the default parameters set in ACOTSP. The main parameters used in the experiments are set as shown in Table 2.

As the solutions that ACO algorithms obtain may vary from try to try due to the use of probability. To be fair and get accurate results, we run each algorithm 10 times, respectively. And in each try, the maximum runtime is set to 10 seconds. Because ACO algorithms are iterative algorithms, the runtime should not be too short. 10 seconds is an acceptable time for most application and the obtained results are relatively stable. And we also do the experiments with runtime set to 15s and 20s, however, the difference of solutions' quality over the three runtimes is quite small, not greater than 0.5%. In addition, in order to examine the usefulness of our new search strategy and update strategy, local search is not employed in all tested algorithms.

Table 2. Main parameters

Parameters	Value	Memo
α	1.0	For all
β	2.0	For all
ρ	0.5	For all
m	25	For all
tries	10	For all
runtime	10s	For all
n_e	100	For AS_{elit}
σ	6	For AS_{rank}
q	0.5	For ACS
φ	0.1	For ACS
size of candidate list	20	For ACS
number of utilization-oriented ants	13	For MAS
number of exploration-oriented ants	12	For MAS
q_0	0.8	For MAS
n_b	6	For MAS

4.3. Experimental Results

The results obtained are presented in Table 3 and Table 4. Table 3 illustrates the shortest tour lengths (best solutions in 10 tries) obtained by the six tested ant algorithms on the 10 test data sets, while Table 4 illustrates the average tour lengths (average solutions of 10 tries) obtained by the six tested ant algorithms on the test data sets. Table 5 and Table 6 give the results of the best performer for each algorithm regarding the shortest tour and the average tour, respectively.

In order to show the performance of different ACO algorithms more clearly, Relative Solution Quality (RSQ) of algorithm *A*, achieved on data set *D*, is defined as follows:

$$RSQ_{D,A} = (1 - \frac{solution_A - Best}{Worst - Best}) \times 100\% \qquad (23)$$

Where $solution_A$ denotes the solution (the shortest tour length or average tour length) obtained by algorithm A. *Best* denotes the best solution (the shortest tour length or average tour length) obtained by the six tested ant algorithms, and *Worst* denotes the worst solution (the shortest tour length or average tour length) obtained by the six tested ant algorithms. Obviously, the greater the value of RSQ is, the better the solution.

Given the definition of RSQ, two line graphs are created. According to Figure 4, the RSQ of MAS stabilizes at 100% or very close to 100%. To be more exact, MAS obtains the best solution on all data sets except att48.tsp (RSQ= 96.55%), rat575.tsp (RSQ= 95.39%) and vm1748.tsp (RSQ= 89.99%). MMAS and ACS are commonly regarded as very effective ACO algorithms. Both of them achieve good performance but second to MAS in terms of finding the shortest tour. As shown in Figure 5, in terms of average solution quality, MMAS performs best. MAS also obtains decent average solutions. Specifically, MAS get best average solutions in four TSP data sets. And it performs better than ACS and only second to MMAS. By comparison, AS's performance is the poorest regarding both the best solutions and the average solutions.

Table 3. The shortest tour lengths obtained

Data set	AS	AS_{elite}	AS_{rank}	MMAS	ACS	MAS
att48.tsp	11438	11572	11300	11021	11021	11040
eil76.tsp	557	561	564	554	554	551
u159.tsp	47850	47301	47671	45319	45423	43270
ts225.tsp	133006	131776	131386	130814	130711	129167
pr439.tsp	123827	122977	121706	118915	120870	118274
rat575.tsp	7889	7877	7822	7668	7607	7620
p654.tsp	41935	41585	41408	40618	41234	39975
d1291.tsp	58890	57140	57357	55728	56244	55698
vm1748.tsp	405534	398635	396208	396144	397473	397084
u2152.tsp	75140	75173	75002	73809	74058	73773

MAS's quality of average solutions is not as good as that of best solutions. It is understandable because the new search strategy allocate part of the ants (the exploration-oriented ants) to focus on founding new tours and diversify MAS's choices rather than finding solutions around best-so-far tours. By and large, in lines with the design intent, MAS is able to find better solutions than other ant algorithms in most cases. To a large extent, MAS addresses the inherent problems (i.e., premature convergence and local optimum) existing widely in ACO algorithms by applying new search strategy as well as the counterpart update strategy. Although the average solutions of MAS are not the best among all of the tested algorithms, MAS obtains satisfactory average solutions as well due to the reservation of the ordinary ants (the utilization-oriented ants in MAS). Considering that TSP aims at finding the shortest tour, all other solutions become worthless as soon as a shorter tour is found. Therefore, in conclusion, MAS is very competitive and significant.

Table 4. The average tour lengths obtained

Data set	AS	AS_{elite}	AS_{rank}	MMAS	ACS	MAS
att48.tsp	11889.3	11682.3	11581.7	11071.7	11067.8	11584.8
eil76.tsp	567.9	586.8	581.5	559.9	561.2	572.2
u159.tsp	49168.4	48348.6	48396.2	46208.5	46378.9	43340.6
ts225.tsp	135751.7	134408.1	132394.3	132575.9	132289.4	132198.4
pr439.tsp	126033.8	125511.9	124594.4	123619.4	124463.8	124065.4
rat575.tsp	7950	7921.8	7902.6	7830.9	7834.7	7800.5
p654.tsp	42682.7	42818.6	42700.6	41873.2	42233.9	41661.8
d1291.tsp	59487.7	58324.2	58073.3	57397.4	57725.6	57605.1
vm1748.tsp	409406.6	404314.6	401487.7	401273.0	403220.4	402692.3
u2152.tsp	76209.4	75608.9	75621.3	74486.6	74733.5	74759.0

Table 5. Results of best performers in terms of the shortest tour

Data set	Best Performer	Tour Length
att48.tsp	MMAS, ACS	11021
eil76.tsp	MAS	551
u159.tsp	MAS	43270
ts225.tsp	MAS	129167
pr439.tsp	MAS	118274
rat575.tsp	ACS	7607
p654.tsp	MAS	39975
d1291.tsp	MAS	55698
vm1748.tsp	MMAS	396144
u2152.tsp	MAS	73773

Table 6. Results of best performers in terms of the average tour

Data set	Best Performer	Tour Length
att48.tsp	ACS	11067.8
eil76.tsp	MMAS	559.9
u159.tsp	MAS	43340.6
ts225.tsp	MAS	132198.4
pr439.tsp	MMAS	123619.4
rat575.tsp	MAS	7800.5
p654.tsp	MAS	41661.8
d1291.tsp	MMAS	57397.4
vm1748.tsp	MMAS	401273.0
u2152.tsp	MMAS	74486.6

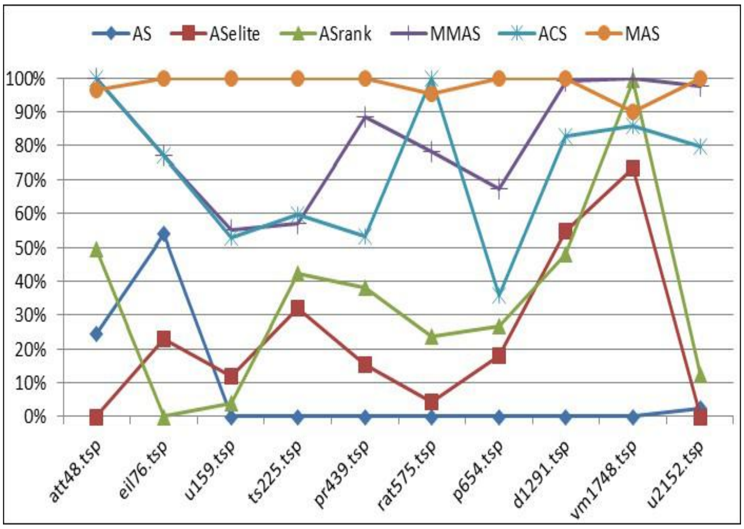

Fig. 4. Best solutions obtained by the tested algorithms. The data sets are sorted by the size in ascending order.

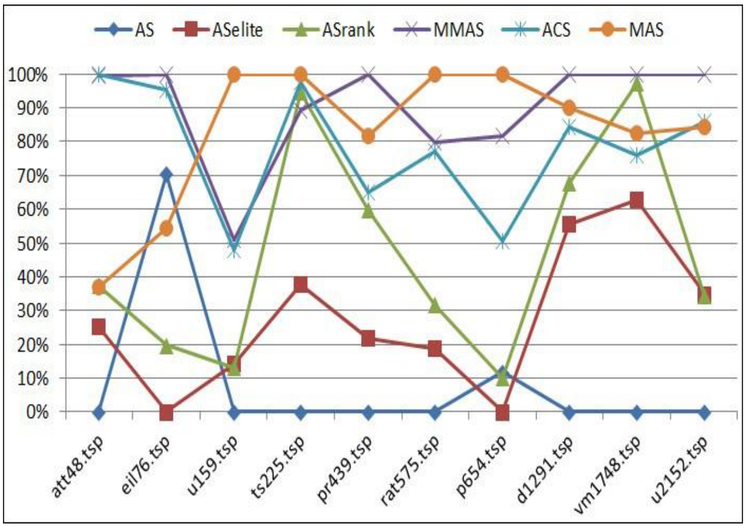

Fig. 5. Average solutions obtained by the tested algorithms. The data sets are sorted by the size in ascending order.

5. Conclusion

This paper discussed the shortcomings of several main ACO algorithms which often get in trouble in local optimum and premature convergence when searching the optimum solution owing to the selection bias and update bias. To overcome these obvious defects, we propose a new ACO algorithm named Moderate Ant System. In MAS, the ordinary ants are divided into two groups, the utilization-oriented ants and the exploration-oriented ants. As for the exploration-oriented ants, taking inspiration from the adaptive behavior of the Monomorium ant species, a novel transition rule is designed and adopted by the exploration-oriented ants. The new transition rule requires a little more time and space, but the extra overhead is linear complexity. Moreover, we also design new update rules to match MAS's search strategy. Taking TSP as an example, ten standard TSP test data sets are employed to examine the performance of MAS algorithm in the experiments. As a result, MAS does achieve our design intent and conquer the known defects to some extent. Overall, the best solution of MAS is better than that of other ant algorithms at the expense of slightly losing the average solution quality.

In the future work, more thorough experimentation and further investigation into MAS algorithm are required in order to explore more characteristics of MAS and perfect the performance of MAS. Moreover, we are planning to apply MAS to other classical problems and give a further study about the performance of MAS.

References

1. R. Demontis, "A simple NP-hard problem", ACM SIGACT News, vol. 40, no. 2, pp. 45-48, 2009.
2. D. S. Hochba, "Approximation algorithms for NP-hard problems" ACM SIGACT News, vol. 28, no. 2 , pp. 40-52, 1997.
3. K. Socha, and M. Dorigo, "Ant colony optimization for continuous domains," European Journal of Operational Research, vol. 185, no. 3, pp. 1155-1173, 2008.
4. Z. Wu, N. Zhao, G. Ren, T. Quan, "Population declining ant colony optimization algorithm and its application", Expert Systems with Applications, vol. 36, no. 3, pp. 6276-6281, 2009.
5. M. Dorigo, M. Birattari, and T. Stützle, "Ant colony optimization," Computational Intelligence Magazine, IEEE, vol. 1, no. 4, pp. 28-39, 2006.
6. O. Cordon, F. Herrera, and T. Stützle, "A review on the ant colony optimization metaheuristic: basis, models and new trends," Mathware & Soft Computing, vol. 9, pp. 141-175, 2002.
7. B. Chandra Mohan, R. Baskaran, "A survey: Ant Colony Optimization based recent research and implementation on several engineering domain", Expert Systems with Applications, vol. 39, no. 4, pp. 4618-4627, 2012.
8. J. Yang, Y. Zhuang, "An improved ant colony optimization algorithm for solving a complex combinatorial optimization problem", Applied Soft Computing, vol. 10, no. 2, pp. 653-660, 2010.

9. M. Dorigo, and G. Di Caro, “Ant colony optimization: a new meta-heuristic,” Proceedings of the 1999 Congress on Evolutionary Computation, Washington, DC, July, 1999.
10. Uğur, D. Aydin, “An interactive simulation and analysis software for solving TSP using Ant Colony Optimization algorithms”, Advances in Engineering Software, vol. 40, no. 5, pp. 341-349, 2009.
11. 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: Cybernetics, vol. 26, no. 1, pp. 29-41, 1996.
12. S. M. Chen, C. Y. Chien, “Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques”, Expert Systems with Applications, vol. 39, no. 12, pp. 14439-14450, 2011.
13. R. J. Mullen, D. Monekosso, S. Barman, and P. Remagnino, “A review of ant algorithms,” Expert Systems with Applications, vol. 36, no. 6, pp. 9608-9617, 2009.
14. M. A. Mashaly, “Monomorium ant's trail pheromones: Glandular source, optimal concentration, longevity and specificity,” Journal of Asia-Pacific Entomology, vol. 13, no. 1, pp. 23-26, 2010.
15. 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.
16. T. Stützle, and H. H. Hoos, “MAX-MIN ant system,” Future Generation Computer Systems, vol. 16, no. 8, pp. 889-914, 2000.
17. 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.
18. M. Dorigo, and T. Stützle, Ant Colony Optimization. Cambridge: The MIT Press, 2004, pp. 76-80.
19. O. Baskan, S. haldenbilen, H. Ceylan, H. Ceylan, “A new solution algorithm for improving performance of ant colony optimization”, Applied mathematics and computation, vol. 211, no. 1, pp. 75-84, 2009.
20. T. Stützle, ACOTSP, Version 1.0. Available from http://www.aco-metaheuristic.org/aco-code, 2004.

Ping Guo (b. September 7, 1963) received his M.Sc. (1989) and PhD (2004) in Computer software and theory from Chongqing University. Now he is full professor of Computer Science at School of Computer Science, Chongqing University, China. His current research interests include different aspects of Artificial Intelligence and Biological computing model. He has (co-) authored 1 books and more than 130 papers.

Zhujin Liu (b. November 30, 1986) received her M.Sc. (2012) in Computer science and technology from Chongqing University of Chongqing. Now he is an engineer of access network in Huawei Company. His current research interests include multicast, gpon and Qos. He has (co-) authored more than 5 papers.

Received: March 02, 2012; Accepted: December 03, 2012.