JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 35, 1263-1277 (2019)
DOI: 10.6688/JISE.201911_35(6).0006

An Adaptive Ant Colony Algorithm for Dynamic Traveling Salesman Problem

AN-XIANG MA, XIAO-HONG ZHANG, CHANG-SHENG ZHANG,
BIN ZHANG AND YAN GAO
School of Computer Science and Engineering
Northeastern University
Shenyang, 110819 P.R. China
E-mail: {maanxiang; zhangxiaohong; zhangchangsheng; zhangbin; gaoyan}@mail.neu.edu.cn

Compared with the traditional static traveling salesman problem, it is more practical to study the dynamic traveling salesman problem in dynamic environment. In this paper, the dynamic optimization problem is considered as a combination of a series of static optimization problems, and an adaptive ant colony algorithm is proposed to solve the dynamic traveling salesman problem. When traveling salesman problem changes, we firstly analyze the environmental changing degree of traveling salesman problem, then an adaptive pheromone initialization mechanism adaptable to changing degree is presented, so as to ensure faster convergence speed without affecting the accuracy of problem solving. In addition, to further improve the quality of solution, we propose an optimization guidance based search mechanism and integrate it into the adaptive ant colony algorithm proposed. Finally, the algorithm is analyzed and compared with the related algorithms on several common real data sets. The results show that the adaptive ant colony algorithm proposed in this paper can solve the dynamic traveling salesman problem more effectively.

Keywords: ant colony algorithm, adaptive ant colony algorithm, dynamic traveling salesman problem, optimization guidance

1. INTRODUCTION

The traveling salesman problem (TSP) is a basic route planning problem, which belongs to the NPC problem. Heuristics and metaheuristics algorithms are usually used to solve this problem. Genetic algorithm, particle swarm optimization and ant colony optimization algorithm have good results on solving TSP [1, 2]. However, most of the existing researches focus on the static traveling salesman problem which doesn't vary during the problem solving. In real life, the TSP is usually in a dynamic environment, such as the path consumption cost among cities which changes with time caused by road congestion. This kind of TSP problem is called dynamic traveling salesman problem (DTSP) [3-5]. For dynamic TSP, the search environment may change during the process of problem solving, which requires dynamic TSP solving method to react to environmental changes quickly. Compared with the traditional traveling salesman problem, it is more practical to effectively solve the dynamic traveling salesman problem.

Among metaheuristics, the ant colony optimization (ACO) algorithm has better adaptive ability, because its pheromone information can be dynamically adjusted according to the environment information. It can be easily extended to solve the dynamic optimization problem. Therefore, the ant colony optimization method is usually used to

Received July 27, 2018; revised November 6, 2018; accepted December 3, 2018.
Communicated by Luís Alberto Lucas.

solve the dynamic optimization problem. The dynamic optimization problem can be considered as a set of continuous static optimization problem instances. When the search environment changes, the dynamic optimization problem can be regarded as a new static optimization problem instance, which can be solved by reinitializing pheromone information [6-8]. However, the approach of reinitializing pheromone information is usually inefficient. To improve the running efficiency, some research works add the local searching techniques into ACO [4, 5, 9]. Reference [5] proposes a hybrid optimization method which integrates local search technique called unstringing and stringing (US) [10] into ant colony optimization approach. The local search method US can effectively improve the convergence speed and improve the quality of the solution. The experimental results show that this method, to some extent, makes up for the low efficiency of the reinitializing pheromone information when the environment changes. To deal with the low efficiency of the reinitializing pheromone approach, under the condition of using the previous pheromone information when the environment changes, researchers try to improve the quality of the solution in dynamic environment by increasing the diversity of solutions [11, 12] and the mode based on storage [13, 14].

Taking into accounts of the differences in the way of reaction to dynamic changes, the existing approaches to solve DTSP based on ant colony optimization can be divided into two categories. The first category is to reinitialize pheromone information to react to the environment changes. The other still uses previous pheromone information when environment changes, which react to the changes by maintaining the diversity of the solutions or adding local search. Through the analysis of the dynamic optimization problems, it is not difficult to find, when the environment changes little, previous pheromone information may provide a priori information for the new environment so as to speed up the search process. When the environment changes a lot, the previous pheromone information may lead to deviate from the optimal solution search direction, thereby reducing the search efficiency. In this case, the reinitialization of pheromone information is more helpful to the problem.

Based on the above analysis, to solve the dynamic traveling salesman problem, this paper designs an adaptive ant colony optimization method adapted to the environment change degree. An adaptive updating method of pheromone information is presented, so that the adaptive ant colony algorithm can make full use of the accumulated prior information during searching process, and guarantee the quality of solution. Because ACO is easy to fall into local optimal solution in search process, we propose optimization guidance based search mechanism and integrate it into the adaptive dynamic traveling salesman algorithm, to further improve the quality of solution. In the second section, the related works are introduced. In the third section, the adaptive ant colony algorithm is given. The fourth section analyzes the performance of adaptive ant colony optimization algorithm proposed in this paper. Finally, section 5 summarizes the whole paper and plans the future works.

2. RELATED WORKS

2.1 Ant Optimization Algorithm

Ant colony optimization (ACO) [15] is a meta-heuristic algorithm motivated by

ants' behavior, which has some notable features such as distributed computing, information positive feedback and heuristic searching. It has been widely used to solve combinatorial optimization problems in many fields.

ACO is composed of three modules, which are search space representation, probabilistic state transition, and pheromone updating respectively. Main difference among ant colony algorithms is pheromone updating strategies. This paper is based on Max-Min ant system to handle dynamic traveling salesman problem, so take Max-Min ant system as example to introduce three modules of ACO.

(1) Search space representation and pheromone initialization. ACO algorithm uses graph model to represent optimization problem domain, and each path represents a feasible solution of problems. Max-Min ant system sets upper and lower bounds for pheromone value, and uses upper bounds as initial pheromone value.
(2) Probabilistic state transition. Define a problem dependent heuristic value, ants choose next vertex according to pheromone value (τ_{ij}) and heuristic value (η_{ij}). For the ant at vertex i, the probability P_{ij} of going to vertex j is defined by Eq. (1)

$$P_{ij}^{k}=\begin{cases}\dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum\limits_{n\in allowed_k}\tau_{in}^{\alpha}(t)\eta_{in}^{\beta}(t)} & \text{if} \quad j\in allowed_k \\ 0 & \text{otherwise}\end{cases}. \tag{1}$$

Where the α and β are weight parameters that indicate the relative importance of the pheromone and heuristic value. The parameter $allowed_k$ denotes the vertices to be visited by the ant k. The ant k uses parameter $tabu_k$ which represents the vertices visited already.

(3) Pheromone updating. The ant records the quality of paths by depositing pheromone. The path with higher quality is deposited more pheromone. In Max-Min ant system, the pheromone on the best path is reinforced, and the pheromone on other paths is evaporated. Pheromone value cannot exceed the upper and lower bounds.

In ACO algorithm, construction graph representing search space, and the definition of pheromone and heuristic value are all related to the specific problem.

2.2 Traveling Salesman Problem Description

Given a set of cities $V=\{v_1, v_2, \ldots, v_n\}$, the goal of traveling salesman problem is to find a loop that traverses all the cities, and each city can only be visited once. In general, the problem can be modeled by a fully connected weighted graph $G = (V, \Gamma)$. Each edge $e_{ij} = <v_i, v_j>$ has a non negative weight, representing the cost value between city v_i and city v_j.

The objective function of the traveling salesman problem is defined as Eq. (2)

$$f(x)=\min\sum_{i=1}^{n}\sum_{j=1}^{n}d_{ij}\psi_{ij}$$

where

$$\psi_{ij}=\begin{cases}1 & \text{if } \langle v_i,v_j\rangle\in x \\ 0 & \text{otherwise}\end{cases}. \tag{2}$$

Where x represents a feasible solution, and n represents the number of city nodes, d_{ij} is the distance between city v_i and city v_j.

2.3 Dynamic Traveling Salesman Problem Testing Data Generating Tools

In recent years, a number of dynamic benchmark set generating tools have been proposed and applied to dynamic traveling salesman problems, such as dynamic benchmark set generating method based on traffic factors [12] and dynamic benchmark set generating method based on city exchange [16]. These dynamic benchmark set generating methods change the environment state values. When the environment changes dynamically, the optimal solution of the problem will also change. Dynamic benchmark set generating method based on permutation coding [17] simulates dynamic changes by exchanging city positions. This method could remain the optimal solution of the problem constant, and it is easy to compare the performance of the proposed method in the dynamic environment.

This paper uses dynamic benchmark set generating method based on permutation coding. In this method, a random vector V is generated every f iterations, it contains $m \times n$ cities that need periodical changes, where, n represents problem scale, $m \in [0.0, 1.0]$ represents the range of change. Then we get vector U by random reordering of cities in vector V. Comparing U with V, we will get $m \times n$ city exchanging pair. Carrying out the $m \times n$ city exchanging in turn, the cost among the cities varies with the location of the node.

3. THE A-MMAS ALGORITHM

MAX-MIN Ant System is an outstanding algorithm among ACO algorithms. To solve DTSP, this paper designs an adaptive MAX-MIN Ant System (A-MMAS) algorithm. In A-MMAS algorithm, to make full use of the accumulated prior information during the search process and guarantee the quality of solution, an adaptive pheromone updating method based on the environmental change degree is presented. To effectively react to the dynamic changes, when the environment changes little, A-MMAS uses previously accumulated pheromone information, however, when the environment changes a lot, it reinitializes pheromone information. To accelerate the speed of convergence and improve the quality of the solution, this paper proposes an optimization guidance based search mechanism, and integrates it into the A-MMAS algorithm.

3.1 Adaptive Ant Colony Optimization Model

In ACO model, construction graph representing search space, and the definition of pheromone and heuristic value are related to the specific problem. Aiming at the characteristics of the DTSP, we define the adaptive ant colony optimization model to solve DTSP.

Each path in construction graph of DTSP represents a feasible solution. Suppose that there are n cities, construction graph is represented as $G = (V, \Gamma)$, where, V represents vertex set $V = \{v_1, v_2, \ldots, v_n\}$, Γ represents edge set $\Gamma = \{e_{ij} = \langle v_i, v_j \rangle \mid i = 1, 2, \ldots, n; j = 1, 2, \ldots, n; i \neq j\}$.

Heuristic information is used to measure the cost value between the vertex to be

visited and the current vertex. Heuristic information is calculated as Eq. (3).

$$\eta_{ij} = 1/d_{ij} \tag{3}$$

Where d_{ij} is the distance between city v_i and city v_j.

In the traditional ant colony optimization algorithm, for a certain path, the amount of pheromones released by an ant is proportional to the quality of the solution corresponding to this path. After each cycle, only the pheromone on the optimal path is added. The pheromone updating rule is shown in Eq. (4),

$$\tau_{ij}(t+1)=\left[\rho\bullet\tau_{ij}(t)+\Delta\tau_{ij}(e_{ij})\right]_{\tau_{\min}}^{\tau_{\max}}$$

where

$$\Delta\tau_{ij}(e_{ij}) = \begin{cases} \dfrac{1}{f(CurBest)} & \text{if } e_{ij} \in CurBest \\ 0 & \text{otherwise} \end{cases}. \tag{4}$$

Where, $\tau_{ij}(t+1)$ represents the amount of pheromones between city v_i and v_j, and the range of $\tau_{ij}(t+1)$ is $[\tau_{\min}, \tau_{\max}]$. Parameter ρ represents pheromone evaporation rate. Parameter $\Delta\tau_{ij}(e_{ij})$ represents pheromone increment. Parameter *CurBest* represents the optimal path in the current iteration. $f(x)$ represents the objective function value of the feasible solution x, which is used to measure the quality of the path and is calculated by Eq. (2).

For DTSP, this paper regards it as a combination of a series of static traveling salesman problem instances. When the environment changes little, A-MMAS uses previously accumulated pheromone information which could accelerate the search process. However, when the environment changes a lot, previous pheromone information may lead to wrong guidance, so it is better to reinitialize pheromone information. Therefore, a parameter m is introduced to record the degree of the environment change. The parameter m is calculated by Eq. (5).

$$m = q/n \tag{5}$$

Where, n represents the number of cities in DTSP, q represents the number of cities whose positions need to be exchanged. Therefore, the range of m is $m \in [0.0, 1.0]$. To assess the degree of environment change, a pheromone adaptive updating threshold ψ is introduced. When the parameters satisfy $m < \psi$, it shows the degree of environment change is small, and A-MMAS will adopt the previous pheromone information in the new environment. When the parameters satisfy $m \geq \psi$, it shows the degree of environment change is large, then A-MMAS will reinitialize pheromone information in the new environment.

Based on the ideas of the A-MMAS algorithm, the improved pheromone updating method is shown in Eq. (6),

$$\tau_{ij}(t+1) = \begin{cases} \left[\rho\bullet\tau_{ij}(t)+\Delta\tau_{ij}(e_{ij})\right]_{\tau_{\min}}^{\tau_{\max}} & \text{if } m < \psi \\ \tau_{\max} & \text{if } m \geq \psi \end{cases}$$

where

$$\Delta\tau_{ij}(e_{ij}) = \begin{cases} \dfrac{1}{f(CurBest)} & \text{if } e_{ij} \in CurBest \\ 0 & \text{otherwise} \end{cases}. \tag{6}$$

Where, ψ represents pheromone adaptive updating threshold, $\tau_{\max}$ represents the upper bound of pheromone value in A-MMAS algorithm.

For the dynamic combinational optimization problem, it is very important to improve the convergence performance to quickly response to the dynamic environment. To improve the convergence speed of A-MMAS, this paper introduces the concept of the optimization guidance information of optimal solution and adds it into the pheromone updating rule. The traditional pheromone information update rules only depend on the optimal path information in the current iteration or so far. It is hard to provide the rich information to guide the search process, so it usually converges slowly. By comparing the optimal solution of current iteration *CurBest* with the optimal solution of last iteration *PreBest*, we can get the optimization guidance information of the optimal solution. Optimization guidance information is a useful heuristic information to guide the search direction in the candidate search space. If adding the optimization guidance information into the existed pheromone information update rules, it will effectively guide the ant to search and improve the convergence speed.

The optimization guidance information of optimal solution could be calculated by analyzing non common edges between *CurBest* and *PreBest*. It is calculated by Eq. (7).

$$OptGuidance = \{e_{ij} =< v_i, v_j > | e_{ij} \in CurBest \wedge e_{ij} \notin PreBest\} \tag{7}$$

Where, *CurBest* represents the optimal solution of current iteration, *PreBest* represents the optimal solution of last iteration.

Based on Eq. (7), we design improved pheromone updating rule which is based on the optimization guidance information and the optimal path information. The formula of improved pheromone updating rule is shown in Eq. (8).

$$\tau_{ij}(t+1) = \begin{cases} \left[\rho \bullet \tau_{ij}(t) + \Delta\tau_{ij}(e_{ij})\right]_{\tau_{\min}}^{\tau_{\max}} & \text{if } m < \psi \\ \tau_{\max} & \text{if } m \geq \psi \end{cases}$$

where

$$\Delta\tau_{ij}(e_{ij}) = \begin{cases} 2 * \dfrac{1}{f(CurBest)} & \text{if } e_{ij} \in OptGuidance \\ \dfrac{1}{f(CurBest)} & \text{if } (e_{ij} \in CurBest \wedge e_{ij} \notin OptGuidance) \\ 0 & \text{otherwise} \end{cases}. \tag{8}$$

3.2 The Description of A-MMAS Algorithm

The core of the A-MMAS algorithm is to determine the degree of environment chang-

es, then adaptively update pheromone information according to Eq. (7). It enables it to make full use of the pheromone information accumulated in the previous environment and to ensure the forward guidance of the search process. In this paper, the dynamic change frequency is set as f, that means a dynamic change is generated each f generation based on benchmark set generating method based on permutation coding. When changes occur, we need to judge the degree of changes, then adaptively update pheromone information according to Eq. (7). The pseudo-code of A-MMAS algorithm is described in Table 1.

Table 1. The pseudo-code of A-MMAS.

Adaptive MAX-MIN Ant System

```
1.  Begin
2.  Start time
3.  Initialize program
4.  Compute_nn_lists //calculate the neighbour node of each city
5.  Malloc the pheromone and total matrix
6.  while (n_try < max_tries)
7.      Initialize n_try
                //initialization for each running
8.      while (!termination_condition)    //judge whether to satisfy the termination condition
9.          If (! (iteration %f)) //change environment each k iterations
10.           Output the best so far ant every k iterations
11.             Dynamic Benchmark Generator
12.             If (m ≥ ψ)
13.                 Initialize the pheromone matrix
14.                 Initialize the best so far ant
15.                 Compute the distances matrix
16.                 Compute nn_lists
17.             End if
18.         End if
19.         Construct the ant solutions
20.         If (ls_flag)
21.             Local search
22.         End if
23.     Update CurBest    //update the optimal solution
24.     Compute OptGuidance    //calculate the optimization guidance information
25.         Pheromone trail update
26.         Search control and CurBest        //update the global statistic data and output
          CurBest
27.           Iteration++
28.     End while
29.     Exit n_try //free memory space for each running
30.     n_try++
31. End while
32. Exit program
33. Free the memory space
34. End
```

4. EXPERIMENTAL EVALUATION

All algorithms are implemented in C language and executed on a Core(i7), 2.93 GHZ, 3GB RAM computer. Four benchmark data sets are used in the experiment, which are eil51.tsp, kroA100.tsp, d198.tsp and lin318.tsp [18].

4.1 Parameters Turning

In A-MMAS algorithm, there are some parameters needing to be discussed which are the parameters α, β and ρ [19] related to ACO, and the parameter ψ related to adaptive pheromone updating rule. Initial pheromone value is used to make the algorithm to find solutions without preference in the beginning. As the running of the algorithm, the qualities of solutions are different, and the pheromone increments of corresponding edges are different. The ants always search towards the direction with high pheromone value. So the performance of the algorithm is related to the pheromone increment value not initial pheromone value (or the upper and lower bound of pheromone value). So the parameters τ_{max} and τ_{min} are not discussed in this paper.

Except for τ_{max} and τ_{min}, other parameters are more complex and sensitive in this algorithm. Their ranges are shown in Table 2. To set appropriate values for these parameters, we turn them in the sequential order α, β, ρ and ψ. For the parameter α, we vary its value one at a time, while setting the values of the other parameters to their default values. For the next unturned parameter β, we vary its value one at a time while setting the values of turned parameters to the obtained most appropriate ones and the values of the other unturned parameters to their default values. Then the other two parameters are turned in the same way as the parameter β. For A-MMAS, the ants colony size n is set as 25, environment change frequency is set as f = 100. The termination condition of this algorithm is that the number of iterations is 1000 generations or the running time exceeds the 1000ms. The A-MMAS algorithm is run ten times on the selected test instances, and the average of running results is the final result.

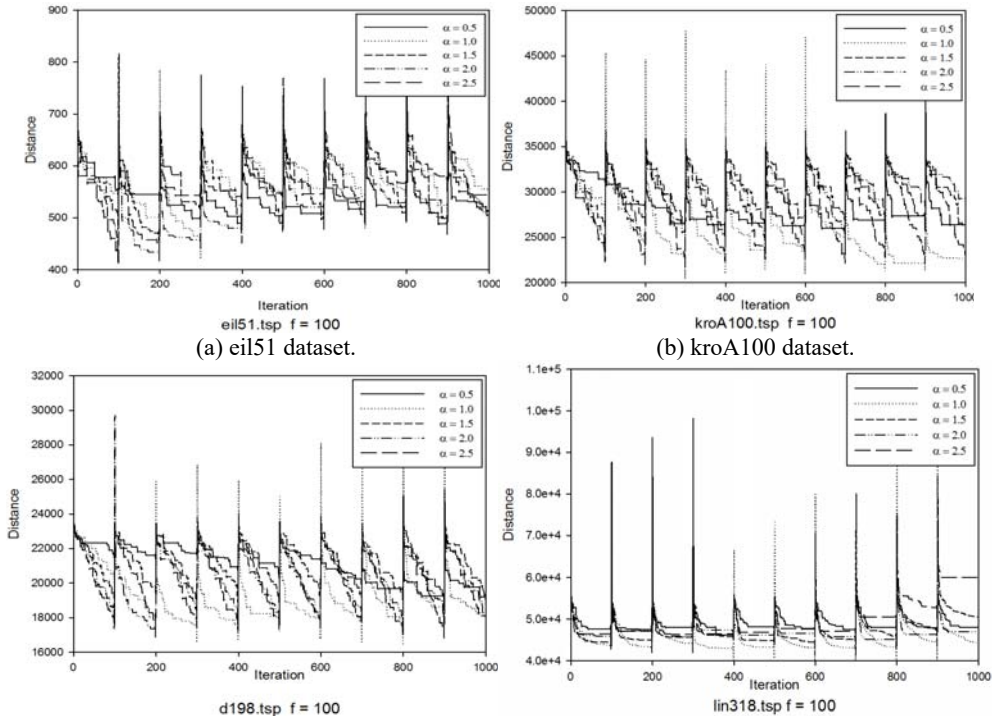
Table 2. The turned parameters.

Para	Meaning	Default	Range
α	relative importance weight of pheromone	1.5	0.5, 1.0, 1.5, 2.0, 2.5
β	relative importance weight of heuristic information	3	1, 2, 3, 4, 5
ρ	pheromone evaporation rate	0.35	0.25, 0.30, 0.35, 0.40, 0.45
ψ	pheromone adaptive updating threshold	0.50	0.35, 0.40, 0.45, 0.50, 0.55, 0.60, 0.65

To assess the quality of solution, we introduce the concept *Distance* which represents the traveling route length. It can be calculated by Eq. (9).

$$Distance = \sum_{i=1}^{n-1} d_{tour[i]tour[i+1]} \tag{9}$$

Where, $tour[i]$ represents the ith city in the solution, $d_{tour[i]tour[i+1]}$ represents the distance between the ith city and the $(i+1)$th city in the solution.



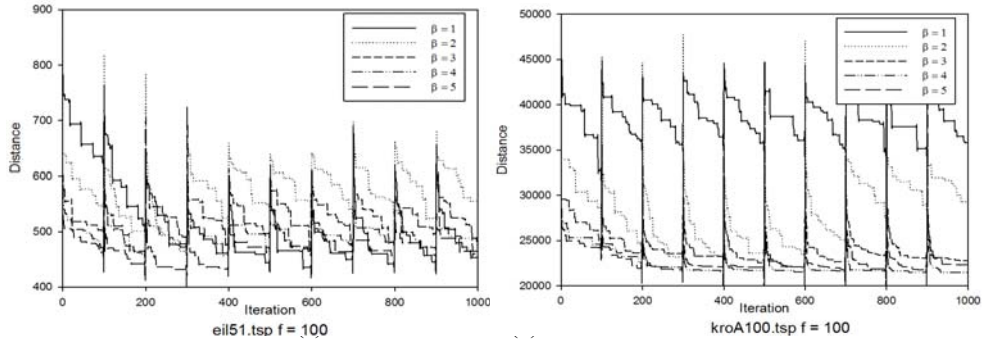
(a) eil51 dataset. (b) kroA100 dataset.

(c) d198 dataset. (d) lin318 dataset.

Fig. 1. Distance convergence under different α value.

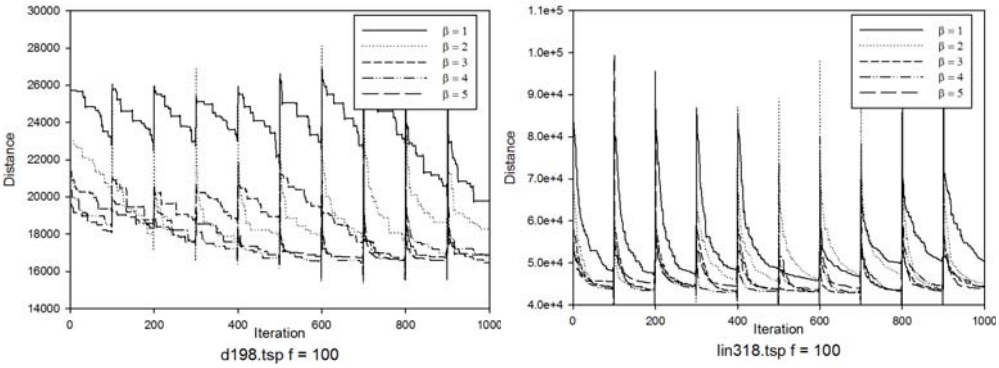
The parameter turning results for parameter α are shown in Fig. 1. In Fig. 1, X-axis represents the number of iterations, Y-axis represents the *Distance* of solution. From Fig. 1, we can see that, the algorithm performs better when α is set smaller value, such as α = 0.5 or α = 1. That means the importance of pheromone information is relatively low. For the benchmark eil51 dataset including 51 cities, the quality of the solution is better and the convergence speed of A-MMAS is faster when parameter α is set as α = 0.5. For the benchmark data set including more cities, α = 1 is better. On average, when parameter α is set as α = 1, the whole performance of A-MMAS is much better for all the data set.

The parameter turning results for β are shown in Fig. 2.



(a) eil51 dataset. (b) kroA100 dataset.

Fig. 2. Distance convergence under different β value.



(c) d198 dataset. (d) lin318 dataset.

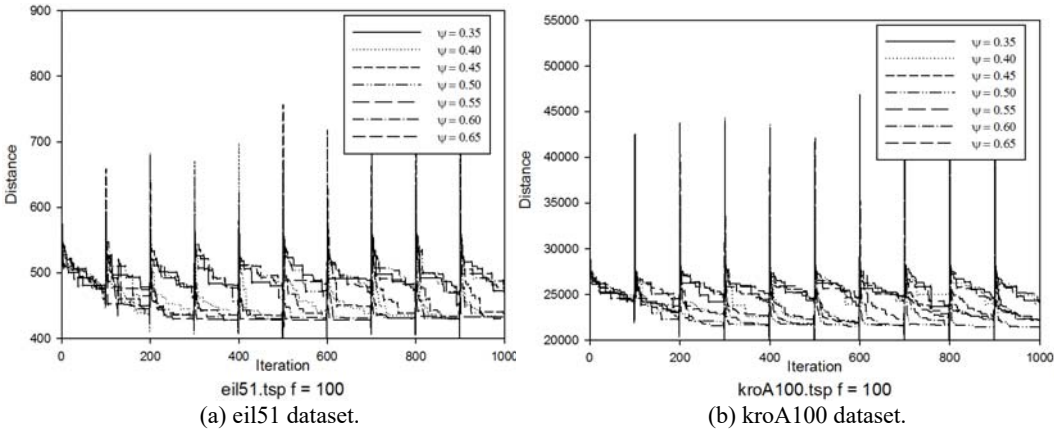
Fig. 2. (Cont'd) Distance convergence under different β value.

From Fig. 2, we can see that, the algorithm performs better when β is set larger value, such as $\beta = 4$. That means the importance of heuristic information is relatively higher than pheromone information in Eq. (1). For the benchmark kroA100.tsp, d198.tsp and lin318.tsp, the quality of the solution is better and the convergence speed of A-MMAS is faster when parameter β is set as $\beta = 4$. For the benchmark eil51.tsp, the performance of A-MMAS is not best, but it is still ranked front. Therefore, the comparatively better setting is $\beta = 4$.

In the same way, we find that the average performance of A-MMAS is better and the convergence speed of A-MMAS is faster when parameter ρ is set as $\rho = 0.25$. That means, low pheromone evaporation rate will lead to high algorithm performance.

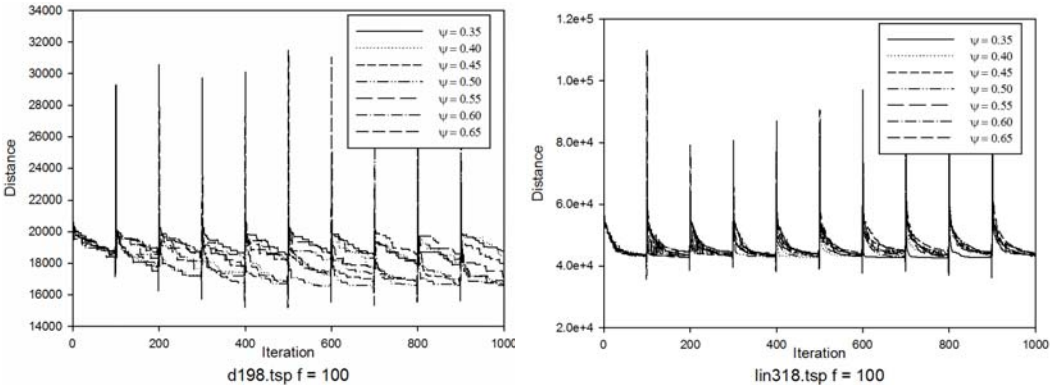
The parameter turning results for pheromone adaptive updating threshold ψ are shown in Fig. 3.

From Fig. 3, we can conclude that, the algorithm performs better when pheromone adaptive updating threshold ψ is set larger value, such as $\psi = 0.50$. That means, it is necessary to reinitiate pheromone information only when environment information changes lot. For the benchmark including less cities such as eil51.tsp, the performance of



(a) eil51 dataset. (b) kroA100 dataset.

Fig. 3. Distance convergence under different ψ value.



(c) d198 dataset. (d) lin318 dataset.

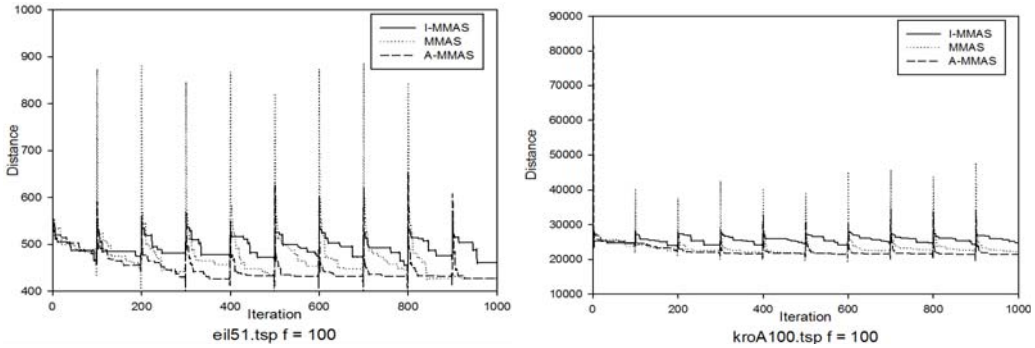
Fig. 3. (Cont'd) Distance convergence under different β value.

A-MMAS is better when $\psi = 0.65$. For benchmark kroA100.tsp and d198.tsp, the performance of A-MMAS is better when $\psi = 0.50$. For benchmark lin318.tsp, ψ has little impact on the quality and convergence speed of the solution. For the benchmark eil51.tsp, when $\psi = 0.50$ the performance of A-MMAS is not best, but it is still ranked front. On average, the whole performance of A-MMAS is much better for all the data set when $\psi = 0.50$.

In summary, the comparatively better settings for these parameters are $\alpha = 1$, $\beta = 4$, $\rho = 0.25$ and $\psi = 0.5$ for the algorithm proposed.

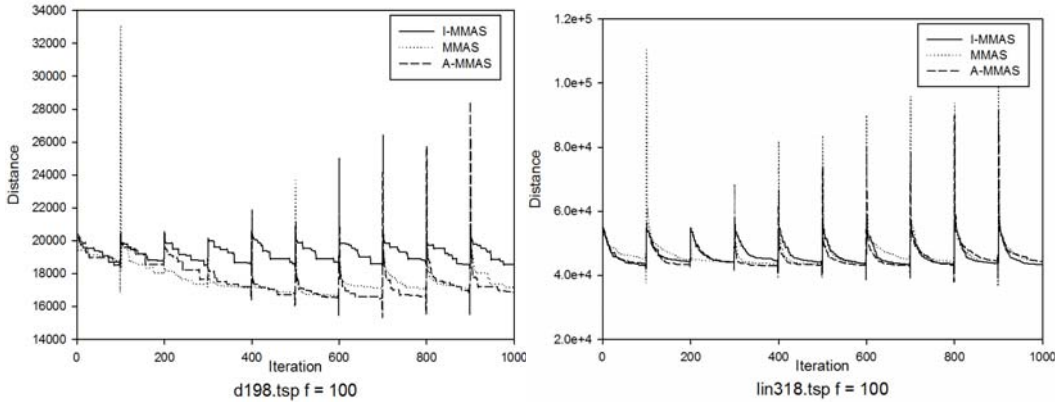
4.2 The Analysis of A-MMAS Performance

To verify the performance of A-MMAS, we compare it with the algorithm proposed in reference [5]. To compare the performance of adaptive pheromone updating strategy, we ignore the local search technique in the algorithm proposed in reference [5]. According to the different reaction method to the environment changes, the approach in reference [5] is divided into Max-Min ant system with complete pheromone initialization (called I-MMAS) and Max-Min ant system with previous pheromone (called MMAS). When environment changes, I-MMAS reinitializes pheromone information, however, MMAS still uses previous pheromone information. The experiment results are shown in Fig. 4.



(a) eil51 dataset. (b) kroA100 dataset.

Fig. 4. The analysis of A-MMAS performance.



(c) d198 dataset. (d) lin318 dataset.

Fig. 4. (Cont'd) The analysis of A-MMAS performance.

From the experimental results, we can see the advantages of the A-MMAS algorithm to deal with DTSP. For the benchmark eil51.tsp, kroA100.tsp and d198.tsp, the performance of A-MMAS is obviously superior to the other two algorithms. For the benchmark lin318.tsp, it includes too many cities. When environment changes, the number of cities affected is also getting more, the performance of A-MMAS is nearly equivalent to the I-MMAS algorithm. Therefore, we can conclude that the performance of A-MMAS is obviously superior to the other two algorithms for the small and medium sized datasets. On average, the performance of A-MMAS is superior to the other two algorithms.

From the analysis above, A-MMAS can improve the speed of convergence and improve the quality of the solution. Besides, the stability of algorithm is also important. To test the stability of A-MMAS, considering the different change frequency f, we analyze the average value and variance value of A-MMAS algorithm.

The experiment results of A-MMAS with $f = 100$ and with $f = 10$ are shown in Tables 3 and 4 respectively.

From Tables 3 and 4, we can see that A-MMAS performs better than the other two algorithms no matter $f = 100$ or $f = 10$. The average value of A-MMAS is smaller than the other two algorithms. That means the solution quality of A-MMAS is the best among three algorithms. The variance value for $f = 10$ is lower than the other two algorithms.

Table 3. Results of A-MMAS when $f = 100$.

Category	eil51(426)				kroA100(21282)			
$f = 100$	average	variance	max	min	average	variance	max	min
I-MMAS	504	26	560	455	25912	1103	28088	24003
MMAS	515	94	885	427	25001	5531	47746	21606
A-MMAS	469	45	653	426	23708	4747	82550	21477
Category	d198(15780)				lin318(42029)			
$f = 100$	average	variance	max	min	average	variance	max	min
I-MMAS	19401	553	20573	18470	52950	1527	55851	50183
MMAS	18175	1824	33051	16666	53922	13380	115642	46168
A-MMAS	18157	1765	28457	16469	52007	7070	92762	45704

Table 4. Results of A-MMAS when $f = 10$.

Category	eil51(426)				kroA100(21282)			
$f = 10$	average	variance	max	min	average	variance	max	min
I-MMAS	546	89	893	448	29851	7680	58209	23617
MMAS	538	53	833	485	28375	4376	48197	23838
A-MMAS	533	34	572	467	27392	669	29074	25267
Category	d198(15780)				lin318(42029)			
$f = 10$	average	variance	max	min	average	variance	max	min
I-MMAS	21630	3992	35268	17990	59254	12999	115518	50329
MMAS	20503	2513	32084	18082	54387	676	56000	53044
A-MMAS	19716	386	20640	18918	53887	845	56052	51475

The variance value for $f = 100$ is between MMAS and I-MMAS. This indicates that the fluctuation of the optimal solution for A-MMAS is relatively small under the condition of guaranteeing the solution quality. This also shows that A-MMAS is relatively stable. Compare the results of A-MMAS with $f = 100$ and the results of A-MMAS with $f = 10$, the quality of solution with $f = 100$ is superior to $f = 10$. If changes occur frequently (such as $f = 10$), the algorithm has not yet searched the optimal solution when environment changes, then the algorithm must search again in the new environment. This will lead to the poor quality of the solution when f is set as small value rather than high value.

5. CONCLUSION

This paper proposes an adaptive ant colony algorithm to solve the DTSP. The algorithm designs an adaptive pheromone initialization mechanism adaptable to environment changing degree. To improve the speed of convergence, we propose optimization guidance based search mechanism and integrate it into the adaptive ant colony algorithm proposed. Experiments show that the algorithm proposed in this paper performs well and has better applicability.

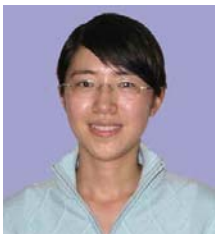
ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation Program of China (61572116, 61572117, 61502089) and by the Fundamental Research Funds for the Central Universities (N171604009). This work was funded by the Chinese State Foundation for Overseas Study.

REFERENCES

1. J. Kanda, A. D. Carvalho, and E. Hruschka, "Meta-learning to select the best meta-heuristic for the Traveling Salesman Problem: A comparison of meta-features," *Neurocomputing*, Vol. 205(C), 2016, pp. 393-406.
2. X. Chen, Y. Zhou, and Z. Tang, "A hybrid algorithm combining glowworm swarm optimization and complete 2-opt algorithm for spherical travelling salesman problems," *Applied Soft Computing*, Vol. 58, 2017, pp. 104-114.
3. C. Groba and A. Sartal, "Solving the dynamic traveling salesman problem using a

genetic algorithm with trajectory prediction: an application to fish aggregating devices," *Computers & Operations Research*, Vol. 56, 2015, pp. 22-32.
4. M. Mavrovouniotis, F. M. Muller, and S. Yang, "Ant colony optimization with local search for dynamic traveling salesman problems," *IEEE Transactions on Cybernetics*, Vol. 47, 2016, pp. 1743-1756.
5. M. Mavrovouniotis, S. Yang, and F. M. Muller, "An ant colony optimization based memetic algorithm for the dynamic travelling salesman problem," in *Proceedings of Genetic and Evolutionary Computation Conference*, 2015, pp. 49-56.
6. M. Guntsch and M. Middendorf, "Pheromone modification strategies for ant algorithms applied to dynamic TSP," *Applications of Evolutionary Computing*, 2001, pp. 213-222.
7. C. J. Eyckelhof and M. Snoek, "Ant systems for a dynamic TSP: ants caught in a traffic jam," in *Proceedings of International Workshop on Ant Algorithms*, 2002, pp. 88-99.
8. M. Mavrovouniotis and S. Yang, "Adapting the pheromone evaporation rate in dynamic routing problems," *Lecture Notes in Computer Science*, Vol. 7835, 2013, pp. 606-615.
9. L. Melo, F. Pereira, and E. Costa, "Multi-caste ant colony algorithm for the dynamic traveling salesperson problem," in *Proceedings of International Conference on Adaptive and Natural Computing Algorithms*, 2013, pp. 179-188.
10. M. Gendreau, A. Hertz, and G. Laporte, "New insertion and post optimization procedures for the traveling salesman problem," *Operations Research*, Vol. 40, 1992, pp. 1086-1094.
11. C. J. Eyckelhof and M. Snoek, "Ant systems for a dynamic TSP," in *Proceedings of International Workshop on Ant Algorithms*, 2002, pp. 88-99.
12. M. Mavrovouniotis and S. Yang, "Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors," *Applied Soft Computing*, Vol. 10, 2013, pp. 4023-4037.
13. M. Guntsch and M. Middendorf, "Applying population based ACO to dynamic optimization problems," *Lecture Notes in Computer Science*, Vol. 2463, 2002, pp. 111-122.
14. M. Guntsch, M. Middendorf, and H. Schmeck, "An ant colony optimization approach to dynamic TSP," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2001, pp. 860-867.
15. S. Salhi, "Handbook of metaheuristics (2nd edition)," *Journal of the Operational Research Society*, Vol. 65, 2014, pp. 320-320.
16. M. Mavrovouniotis and S. Yang, "A memetic ant colony optimization algorithm for the dynamic travelling salesman problem," *Soft Computing*, Vol. 15, 2011, pp. 1405-1425.
17. S. Yang and X. Yao, "A benchmark generator for dynamic permutation-encoded problems," in *Proceedings of International Conference on Parallel Problem Solving from Nature*, 2012, pp. 508-517.
18. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.
19. T. Liao, K. Socha, D. Montes, T. Stutzle, and M. Dorigo, "Ant colony optimization for mixed-variable optimization problems," *IEEE Transactions on Evolutionary Computation*, Vol. 18, 2014, pp. 503-518.



An-Xiang Ma (馬安香) received the M.S. degree and Ph.D. degree in Computer Application Technology from Northeastern University, China. She is currently a Lecturer at Northeastern University. Her research interests include evolutionary computation, service computing and cloud computing.

Xiao-Hong Zhang (張曉紅) received the M.S. degree in Computer Application Technology from Northeastern University, China. She is a Lecturer in the College of Computer Science and Engineering at Northeastern University. Her research interests include evolutionary computation, cloud computing.

Chang-Sheng Zhang (張長勝) received the Ph.D. degree in Computer Science and Technology from Jilin University in 2009, China. He is a Professor in the College of Computer Science and Engineering at Northeastern University, China. His research interests include evolutionary computation, edge computing and distributed constraint programming.

Bin Zhang (張斌) received the Ph.D. degree in Computer Application Technology from Northeastern University, China. He is a Professor in the College of Computer Science and Engineering at Northeastern University, China. His research interests include service computing, and intelligent traffic.

Yan Gao (高岩) received the M.S. degree and Ph.D. degree in Computer Application Technology from Northeastern University, China. He is an Associate Professor in the College of Computer Science and Engineering at Northeastern University, China. His research interests include intelligent algorithm design, service computing and intelligent traffic.