

---

## Time-dependent pheromones and electric-field model: a new ACO algorithm for dynamic traffic routing

---

Biao-bin Jiang\*

Student Innovation Base of Computer Science and Technology,  
School of Computer Science and Technology,  
Beijing Institute of Technology,  
Beijing, 100081, China  
E-mail: ambitiousjbb@gmail.com  
\*Corresponding author

Han-ming Chen and Li-na Ma

School of Computer Science and Technology,  
Beijing Institute of Technology,  
Beijing, 100081, China  
E-mail: hanming@bit.edu.cn  
E-mail: marina19870807@sina.com

Lei Deng

School of Information Science and Technology,  
Beijing Institute of Technology,  
Beijing, 100081, China  
E-mail: dleidlei@bit.edu.cn

**Abstract:** In this paper, we present a dynamic ant colony optimisation (ACO) algorithm to solve dynamic traffic routing problem. The main objective of this work is to search out the least-time-cost route in a variable-edge-weight graph. We introduce time-dependent pheromones and electric-field model as two heuristic factors to improve the basic ACO. The simulation results show that the proposed dynamic ACO algorithm can effectively reduce time cost by avoiding the dynamic congestion areas. Finally, this proposed heuristic algorithm is verified to be steady-going by repeated testing.

**Keywords:** dynamic traffic routing; ant colony optimisation; ACO; time-dependent pheromones; electrostatic-field model; directional angle.

**Reference** to this paper should be made as follows: Jiang, B-b., Chen, H-m., Ma, L-n. and Deng, L. (2011) 'Time-dependent pheromones and electric-field model: a new ACO algorithm for dynamic traffic routing', *Int. J. Modelling, Identification and Control*, Vol. 12, Nos. 1/2, pp.29–35.

**Biographical notes:** Biao-bin Jiang, BEng of Beijing Institute of Technology, was the Group Leader of Multi-agent Technology Group in the Student Innovation Base of Computer Science and Technology. He currently focuses on complex network and bioinformatics.

Han-ming Chen is the Dean of Software School in Beijing Institute of Technology and is also the Director of Student Innovation Base of Computer Science and Technology.

Li-na Ma is currently a candidate of BEng in Computer Science and Technology in Beijing Institute of Technology. Her research interests include software development and computer vision.

Lei Deng, BEng of Beijing Institute of Technology, is currently a PhD candidate in Dept. of Automation, Tsinghua University, Beijing, China. His research interests are pattern recognition and image processing.

## 1 Introduction

In many metropolises, it is easily seen that hundreds of vehicles stand still in a long queue and roar noisily for a tiny gap to pass. Increasing car ownership and limited road loads lead to serious traffic congestion nowadays. The centralisation of working time even gets the congestion worse in rush hour. Therefore, researchers devoted to intelligent transportation system (ITS) declared that traffic congestion is the phenomenon that too many cars appear in a wrong place at a wrong time. How should we tackle the worsening congestion problem? Obviously, we cannot assign a certain car to pass a fixed intersection at a time. But we are able to guide drivers to travel along the fastest path and avoid the crowded intersections, which requires taking advantage of spare roads in order to disperse congestion.

During the recent decade, advanced traveller information system (ATIS) is presented and inspired to tackle with the traffic congestion problem. Dynamic route guidance, as part of the ATIS, is developed to provide users with real time information of global traffic networks and assist them to pass via the fastest path and round the crowded intersections. The algorithm supporting the route guidance system must be able to adapt the dynamic information collected from the sensors in road surface or above the signal lights which can count the vehicles and measure their speed in real time (Tatomir and Rothkrantz, 2004). Thus, this algorithm is expected to update dynamically the traffic information and to work out an optimal path in a tolerable computational time.

In this paper, a new ant colony optimisation (ACO) algorithm based on time-dependent pheromones and electric-field model is presented to solve the dynamic routing problem. In Section 2, several related papers on dynamic routing and ACO will be reviewed. The graph model abstracted from traffic networks is depicted in Section 3. The implemental details of proposed ACO algorithm will be presented in Section 4 and then, a simulation as the application of the algorithm in traffic environment is displayed in Section 5. Finally, the strengths, weaknesses and future developments of the proposed ACO will be concluded in Section 6.

## 2 Related works

Essentially, dynamic traffic routing is a dynamic shortest path (SP) problem. For static SP problem, the length of edges is a constant. The classical algorithm for SP problem is Dijkstra algorithm from graph theory (Dijkstra, 1959). It has proved to be an effective method to find out an SP from one source to multiple destinations in a directed graph. However, it has to traverse all the vertexes in order to search out the global optimal path, which brings to the computational complexity  $O(n^2)$ . In addition, the dynamic edge weights violate the rule during the Dijkstra search that the sub-path must be global optimal at each iteration. Thus, it is ineffective in solving the dynamic SP problem or in the

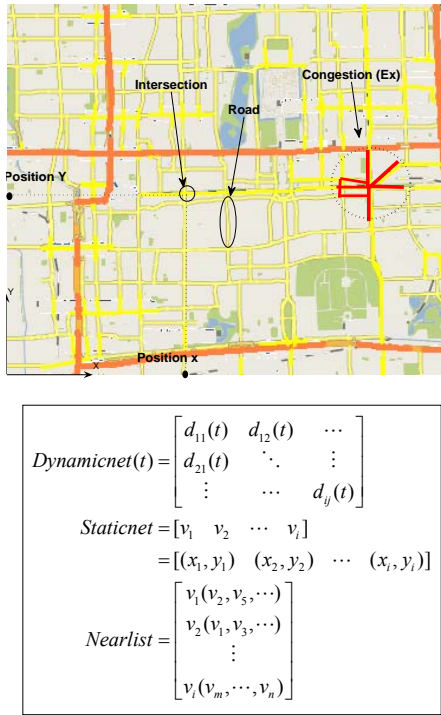
application of searching in large-scale graph such as urban traffic networks within the maximal tolerance of computational time.

For reducing the complexity of computational time, several heuristic methods have been presented and inspired to solve non-polymer (NP) problem or overcome the dynamic uncertainty. One of the well known methods is the genetic algorithm (GA) presented by John M. Holland in 1975 (Holland, 1975). Ahn and Ramakrishna (2002) made good use of GA and achieve the gene encoding with uncertain length to solve static SP problem. A new evolutionary algorithm based on swarm intelligence, ant algorithm, was presented by Dorigo (1991). The classical application of ant colony algorithm is solving the travelling salesman problem (TSP). Gambardella and Dorigo (1995) first applied ACO to solve the TSP using reinforcement learning approach. Hsiao et al. (2004) successfully used ACO technique to search the SP from a fixed original to a desired destination in a random given map. Tatomir and Rothkrantz (2004) effectively tackled the problem of dynamic routing for urban transportation based on the AntNet algorithm. But their experiment is made on a  $4 \times 4$  net in which all the edges have the same length.

In this paper, we construct an incomplete undirected graph in which the weights of all edges are time-dependent variable and each vertex stores its actual location with geographical coordinates. When a user asks for help by sending his current location and desired destination to the service centre of route guidance, our ACO algorithm will work out a fastest route based on the prediction of traffic flow distribution. In the design of ACO algorithm, we encode the pheromones with time sequences rather than cumulative variables according to the time-dependent variable edge length. Meanwhile, we introduce electric-field model to set up the heuristic oriented search mechanism for the promotion of search efficiency.

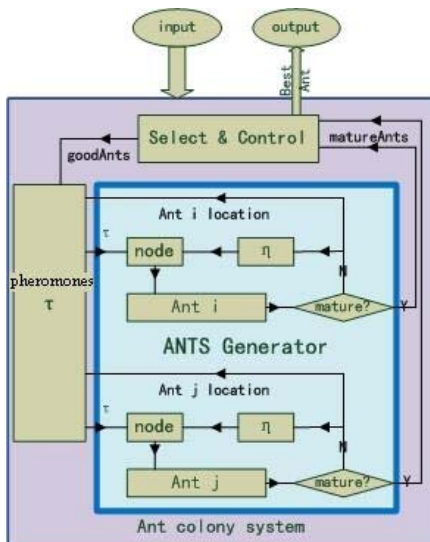
## 3 Network model

Usually, the real distribution of road network is abstracted into an incomplete undirected graph  $G(V, E)$  based on the graph theory. In this graph, the edge  $e(v_i, v_j)$  represents the road from the intersection  $v_i$  to next linked intersection  $v_j$ , and its weight  $d_{ij}(t)$  denotes the time interval during which a vehicle spends in passing through this road. This duration measured by the sensors on the roads in real time is stored in form of time-dependent element in an adjacency matrix  $Dynamicnet(d_{ij}(t))$ . And the vertex (or node)  $v_i$  represents the intersection associated to the crossing roads at it, and is portrayed by the coordinates  $v_i(x_i, y_i)$  of its location in a row vector  $Staticnet(v_i)$ . The values of these located coordinates can be accessed from e-map. The congestion at an intersection is quantised by increasing the weights of the edges crossing at that intersection. In addition, a 2-D array *Nearlist* records each  $v_i$  and all the nodes  $(v_m, \dots, v_n)$  linked with  $v_i$ . The structure of this abstracted traffic network and the symbols mentioned above is illustrated in Figure 1.

**Figure 1** Network structure (see online version for colours)


#### 4 Proposed ACO algorithm

In this section, the details of ACO routing algorithm are introduced by three following parts. We first outline the entire ant colony system (ACS) using a diagram (Figure 2) to illustrate the system structure and each module. Then we develop a basic ACO for static SP problem. In addition, we set up a new dynamic and heuristic search mechanism by improving pheromone variables and introducing heuristic factor.

**Figure 2** Structure of ACS (see online version for colours)


#### 4.1 Description of ACS

At the beginning of system operation, the data of traffic network besides the origin and destination sent from the user are input into the system. Then initialise the pheromones  $\tau$  at each edge and heuristic factor  $\eta$ . The ants start to select next node and move forward according to the both ingredients until they reach the destination or confront a fact that all selectable nodes are visited, which means that the ant is 'mature' in system diagram. The select and control module afterwards selects 'good' ants which successfully find out the destination from the 'mature' ants and then records the 'best' ant which reaches the destination using the fastest way among all 'good' ants. At the end of this iteration, the system updates the pheromones on the roads which the 'good' ants travel along.

#### 4.2 Basic ACO for SP

It is well known that the ACO is good at solving the TSP. We develop a basic ACO for SP based on that for TSP according to the following rules.

- *Modify the story structure of the path:* The path store in TSP is determined by the total number of cites before running the algorithm, but the path length is uncertain at each iteration in SP.
- *Change the terminative condition:* Let ants search the path until they find the destination rather than return the origin.

The procedure of a basic ACO for SP is portrayed as below:

*Procedure:* BasicACOforSP.

- Step 1* Input origin, destination and network matrix; initialise parameters and pheromones.
- Step 2* Construct path solutions node by node for each ant; the ant firstly judges the selectable nodes not visited then computes the selection probabilities of next node using current pheromones, finally select one of the selectable nodes as next node randomly.
- Step 3* Update the pheromones; the ants who can find out the destination have the rights to deposit pheromones on the roads they have just passed through.
- Step 4* If terminate condition is true, end the procedure; otherwise go to the *Step 2*.

Where several important functions and conditions are specified as below:

- *Terminate condition:* Only if the iteration count  $I$  reaches the upper limit  $I_{\max}$  or the current global optimal solution is better than a given promising threshold, the BasicACOforSP procedure can be terminated.

- *NotVisited()*: This function is used to judge and select the nodes not visited in foregone path from all the nodes linked with the current located node.
- *ComputeProbability()*: This function is used to compute the selection probabilities with the deposited pheromones of each selectable node by the following formula,

$$P_{ij} = \frac{\tau_{ij}}{\sum_j \tau_{ij}}$$

where  $\tau_{ij}$  is the element of matrix pheromone $[n][n]$  storing the current pheromone values on each edge.

- *RandomSelect()*: this function is used to do a random selection for next node to visit based on the probabilities  $P_{ij}$ .
- *DepositPheromone()*: this function in UpdatePheromones procedure is used to update the pheromones on the edges along which the corresponding ants can achieve the search in higher probability from the origin to the destination. The updating mechanism is formulated as below,

$$\tau_{ij} = \rho \cdot \tau_{ij} + \sum_{h=1}^m \Delta_h \tau_{ij}$$

$$\Delta_h \tau_{ij} = \begin{cases} \frac{1}{L_h} \\ 0, \text{ if ant}[h] \text{ fails to reach destination} \end{cases}$$

where the  $L_h$  denotes total time cost in the path which is found out by the ant $[h]$  at the iteration  $I$ . And the constant  $\rho$  indicates the pheromone loss by evaporating.

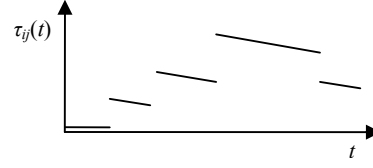
### 4.3 Development of search mechanism

#### 4.3.1 Time-dependent pheromone

In dynamic routing problem, the length of edges is a time-dependent variable rather than a constant. Thus, the story structure of road net is modified into Dynamicnet $[n][n][t]$  instead of net $[n][n]$ . The dynamic variation of road length will cause such a problem that the pheromone deposited by previous ant cannot effectively enlighten the latter ant because the dose of pheromone is closely associated with the dynamic road length. Therefore, the pheromones should be also dynamic as the road length. It can be stored by the 3-D matrix pheromone $[n][n][t]$  and be symbolised as  $\tau_{ij}(t)$ .

Then how does the system update pheromones at the end of each iteration? When an ant ever spent  $\Delta t$  passing through an edge, the pheromone  $\tau_{ij}(t)$  will be accumulated with dose  $\Delta \tau_{ij}$  deposited at the interval  $\Delta t$ , while is evaporating and decreasing, which is shown as Figure 3.

**Figure 3** Time-dependent pheromones



Afterward, another ant will select the edge according to the pheromones at latter iterations. The system first accesses the moment  $t_0$  when the ant arrives at the node associated with the edges. Then it judges which edges should be selected with the pheromone  $\tau_{ij}(t_0)$  that involves the selection probability  $P_{ij}$ . The updating mechanism is formulated as below,

$$\tau_{ij}(t_0) = \rho \cdot \tau_{ij}(t_0) + \sum_{h=1}^m \Delta_h \tau_{ij}(t_0)$$

$$\Delta_h \tau_{ij}(t_0) = \begin{cases} \frac{1}{L_h}, & \text{if ant}[h] \text{ reach } e(v_i, v_j) \text{ within } [t_0 - \delta, t_0 + \delta] \\ 0, & \text{otherwise} \end{cases}$$

#### 4.3.2 Heuristic factors

The search process is undirected and exhausted if the ant system computes the selection probability just by means of the pheromones. The searching efficiency can be improved by introducing the geographical locations of the nodes in order to make the ants search towards the destination node. Thus, a heuristic factor  $\eta_{ij}$  will be introduced into the selection probability which is formulated as below, where the two constants  $\alpha$  and  $\beta$  are used to scale the weights of the both ingredients.

$$P_{ij} = \tau_{ij}(t)^\alpha \cdot \eta_{ij}^\beta$$

Here, the heuristic factor will be developed by means of electrostatic field model and directional angle.

#### Electrostatic-field model

Suppose the whole traffic net is an electrostatic field. In this field, a driver in vehicle is considered to be a dynamic positive charge  $+q$ , the destination node is a static negative charge  $-Q$ , and the congestive intersections is regarded as a static positive charge  $+cQ$ , where  $c$  is a time-dependent fraction  $c = c(t)$  in order to make the charge in destination more than that in congestive intersection. It is easily imagined that the dynamic positive charge  $+q$  will be attracted by the negative charge  $-Q$  in the destination and move towards it. Meanwhile, the  $+q$  will be also repelled by the positive charge  $+cQ$  in the congestive intersection and pass around it, where the fraction  $c = c(t)$  can be determined by the extent of time-dependent congestion. The entire model is illustrated as Figure 4.

The value of electrostatic force is associated with the distance  $r_{ij}$  between the both charges. So it is necessary to introduce the geographical position of each node  $p_i(x_i, y_i)$ . The distance from node  $i$  to node  $j$  is formulated as below,

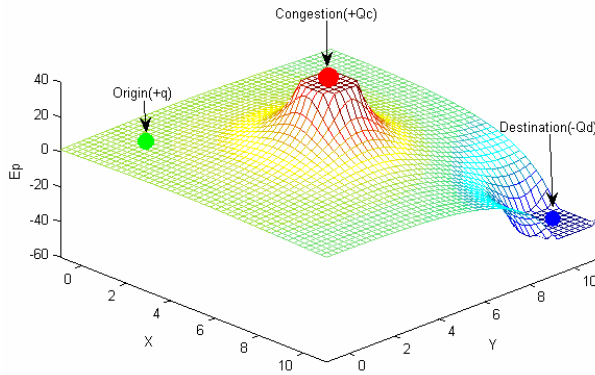
$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

And the electrostatic force can be formulated according to the electrostatic field theory as below,

$$\varphi_{ij} = k \frac{Q}{r_{ij}} - k \frac{c(t) \cdot Q}{r_{ij}}$$

where  $k$  is a constant for scaling the contribution  $\varphi_{ij}$  in the factor  $\eta_{ij}$ .

**Figure 4** Electrostatic-field model (see online version for colours)



### Directional angle

For avoiding blindfold search and reducing iterations, the directional angle is added into the heuristic factor.

In the procedure of selecting next node, if the vector from located node to one selectable node approximately points to the destination node, this selectable node will be chosen as the next node in higher priority, which is illustrated as Figure 5.

According to this basic idea, the directional angle  $\theta_{ij}$  is formulated as below,

$$\theta_{ij} = \begin{cases} \frac{1}{A_{ij}}, \frac{1}{A_{ij}} < \theta_{\max} \\ \theta_{\max}, \text{otherwise} \end{cases}$$

In sum, the entire heuristic factor is combined with the force factor and angle factor as below,

$$\eta_{ij} = \varphi_{ij} \cdot \theta_{ij}$$

where the contributions of the both factors can be adjusted with the constant  $k$  in  $\varphi_{ij}$ .

In sum, the entire dynamic ACO algorithm can be described with the following pseudo-code.

---

#### Procedure: DynamicACO

Initialise, set parameters

$iteration \leftarrow 1$

**while** (not Terminate condition) **do**

    ConstructAnts

    UpdatePheromones

    Output shortest path

$iteration \leftarrow iteration + 1$

**end-while**

**end-procedure**

#### Procedure: Initialise

Dynamicnet[V][V][time]

Staticnet[V]

near\_list[V][[]]

**structure** single\_ant

    locatedV

    time

    path[]

single\_ant ant[m]

**end-procedure**

#### Procedure: ConstructAnts

clear ant[m]

**for** each ant[] **do**

$locatedV, path[1] \leftarrow origin$

**while** ( $locatedV \neq destination$ ) **do**

        Select Notvisited[] from near\_list[locatedV][[]]

**if** Notvisited[] ==  $\emptyset$  **then**

            break

**end-if**

        heuristic\_factor = ComputeHeuristic(ant[])

        select\_probability = ComputeProbability(Notvisited[], pheromone[locatedV][time], heuristic\_factor)

        nextV = RandomSelect(select\_probability, Notvisited[])

        time  $\leftarrow$  time + Dynamicnet[locatedV][nextV][time]

$locatedV \leftarrow nextV$

        path  $\leftarrow$  [path, nextV]

**end-while**

**end-for**

**end-procedure**

#### Procedure: UpdatePheromones

**for** each ant[] **do**

**if**  $locatedV == destination$  **then**

        pheromone = DepositPheromone()

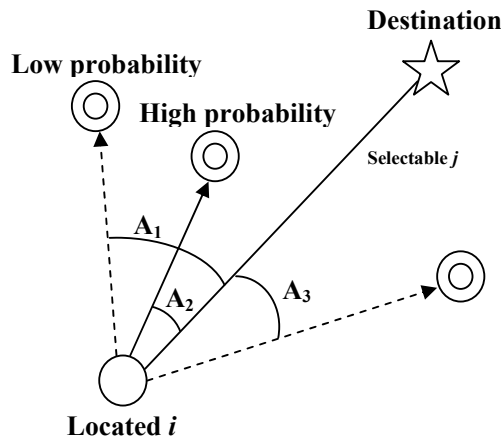
**end-if**

**end-for**

**end-procedure**

---

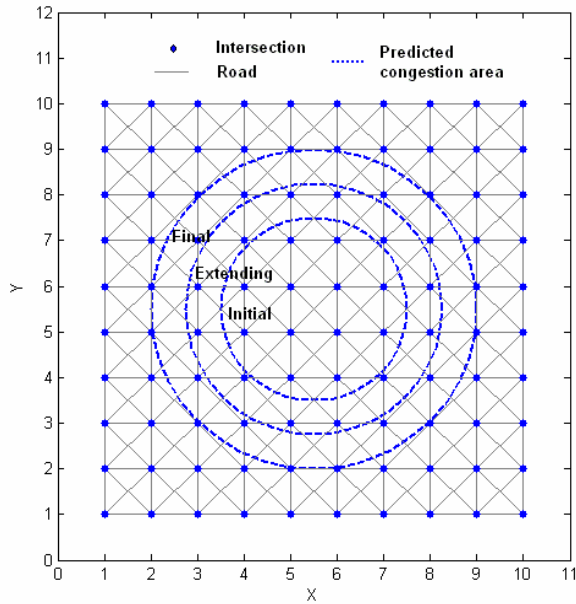
Figure 5 Directional angle illustration



5 Simulation and analysis

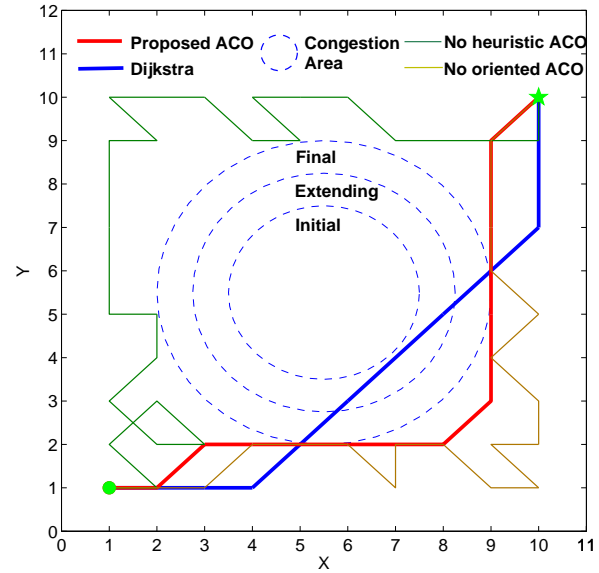
For testing the efficiency and robustness of the proposed ACO algorithm, a 10 × 10 net (Figure 6) is setup to simulate actual traffic environment in this section.

Figure 6 Simulation environment (see online version for colours)



We test the proposed dynamic ACO algorithm using Matlab 2007a in this simulation environment, and compare the result with that from Dijkstra algorithm. The proposed ACO is also run without the heuristic factor or oriented factor in order to demonstrate their key roles in the stochastic search mechanism. The visual results from the four programs mentioned above are shown together in Figure 7.

Figure 7 Visual results (see online version for colours)



It is obviously seen from the Figure 7 that the proposed dynamic ACO algorithm achieves the route that can effectively avoid the congestion range expanding dynamically. However, the classical Dijkstra algorithm fails to judge the dynamic trend of congestion and its yielding route goes across the congestion area, which implies that the driver using this route will be caught by traffic jam. In addition, the route searched out by the ACO without heuristic factor includes loop segments and shows that this search mechanism neglects the position of destination and causes blindfold search. And the route worked out by the ACO without the oriented factor indicates that the ant in this route is easily repelled by the expanding congestion for the sake of the lack of the orientation from the destination.

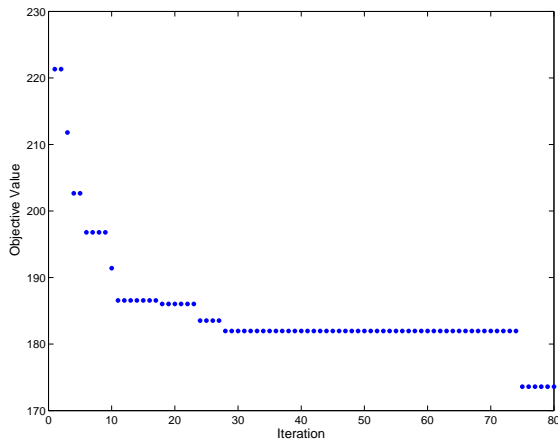
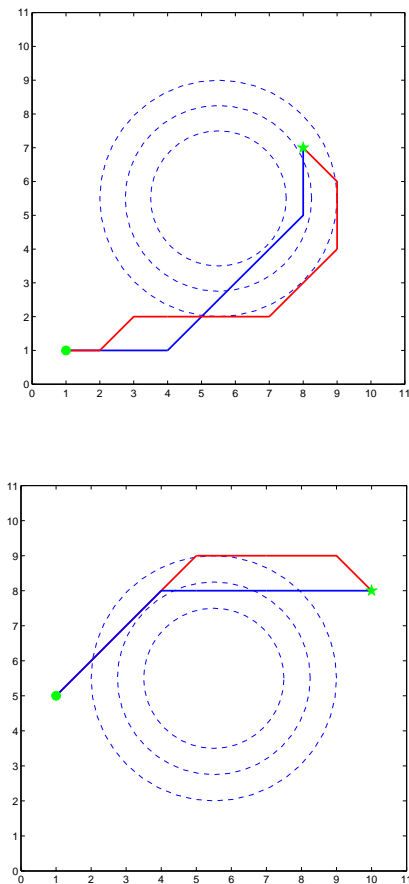
The quantitative results on the four algorithms in the simulation are listed in Table 1.

Table 1 Quantitative results

	Total time cost	Running time/s
Dijkstra	204.5	-
Dynamic ACO	177.0	8.4
No heuristic ACO	321.7	15.9
No oriented ACO	247.2	14.2

The quantitative results display that the proposed dynamic ACO algorithm yields the best solution with the least time cost among the four algorithms. And its running time is also tolerable in the application of this small-scale network.

The iteration process of the proposed ACO algorithm is depicted in Figure 8, which shows that this algorithm can converge within tolerable iterations ( $T_{max} = 80$ ).

**Figure 8** Iteration process (see online version for colours)**Figure 9** Test of robustness (see online version for colours)

Due to the uncertainty in the heuristic search process, the heuristic algorithms need to be tested repeatedly. We use different origin-destination pair to test our ACO algorithm. The test results (Figure 9) show that the performance of our algorithm is steady-going and cannot be perturbed as the

OD pair changes, which verifies the robustness of our ACO algorithm.

## 6 Conclusions and future work

In this paper, we successfully achieve the dynamic ACO algorithm to solve the dynamic traffic routing problem. We have then setup the search mechanism using time-dependent pheromones and electric-field model. These both improvements help our algorithm yield the optimal route that can effectively avoid the congestion areas and cost least time to reach the desired destination. By the comparison with the classical Dijkstra algorithm, our ACO algorithm performs better in solving the dynamic SP problem. Finally, the robustness of this heuristic algorithm is verified by repeated testing.

For future works, the improvement in the story of dynamic data is inspired to reduce the running time of the algorithm. And the running time can also be reduced by implementing of parallel computing for each ant. Furthermore, the ACO algorithm involving multi-objective optimisation is worthy of more researches.

## Acknowledgements

We are grateful for the computing resources provided by Multi-agent Group at Student Innovative Base of Computer Science and Technology, Beijing Institute of Technology (BIT). We thank Prof. Bai-hai Zhang at Dept. Automation, BIT, for discussing the research direct of this paper with us. This work was sponsored by Hi-tech Research and Development Program of China (2006AA04Z209).

## References

- Ahn, C.W. and Ramakrishna, R.S. (2002) 'A genetic algorithm for shortest path routing problem and sizing of populations', *IEEE Trans. on Evolutionary Computation*, December, Vol. 6, No. 6, pp.566–579.
- Dijkstra, E.W. (1959) 'A note on two problems in connection with graph', *Numerische Mathematic*, pp.269–271.
- Dorigo, M. et al. (1991) 'Distributed optimization by ant colonies', *Proceedings of the 1st European Conference on Artificial Life*, pp.134–142.
- Gambardella, L.M. and Dorigo, M. (1995) 'Ant-Q: a reinforcement learning approach to the traveling salesman problem', *Proceedings of the 12th International Conference on Machine Learning*, pp.252–260.
- Holland, J.H. (1975) 'Adaptation in natural and artificial systems', *University of Michigan Press*, Ann Arbor, Michigan.
- Hsiao, Y.T. et al. (2004) 'Ant colony optimization for best path planning', *International Symposium on Communications and Information Technologies*, Sapporo, Japan, October, pp.109–113.
- Tatomir, B. and Rothkrantz, L. (2004) 'Dynamic traffic routing using ant based control', *IEEE International Conference on Systems, Man and Cybernetics*, pp.3970–3975.