Egyptian Informatics Journal (2012) **13**, 111–121

Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

Threshold based AntNet algorithm for dynamic traffic routing of road networks

Ayman M. Ghazy, Fatma EL-Licy, Hesham A. Hefny *

Department of Computer and Information Sciences, Institute of Statistical Studies and Research, Cairo University, Giza, Egypt

Received 2 November 2011; revised 6 May 2012; accepted 13 May 2012
Available online 12 June 2012

KEYWORDS
Swarm intelligence;
Dynamic traffic routing;
AntNet;
Forward ant;
Backward ant;
Check ant

Abstract Dynamic routing algorithms play an important role in road traffic routing to avoid congestion and to direct vehicles to better routes. AntNet routing algorithms have been applied, extensively and successfully, in data communication network. However, its application for dynamic routing on road networks is still considerably limited. This paper presents a modified version of the AntNet routing algorithm, called "Threshold based AntNet", that has the ability to efficiently utilize a priori information of dynamic traffic routing, especially, for road networks. The modification exploits the practical and pre-known information for most road traffic networks, namely, the good travel times between sources and destinations. The values of those good travel times are manipulated as threshold values. This approach has proven to conserve tracking of good routes. According to the dynamic nature of the problem, the presented approach guards the agility of rediscovering a good route. Attaining the thresholds (good reported travel times), of a given source to destination route, permits for a better utilization of the computational resources, that, leads to better accommodation for the network changes. The presented algorithm introduces a new type of ants called "check ants". It assists in preserving good routes and, better yet, exposes and discards the

* Corresponding author. Address: Department of Computer and Information Sciences. Institute of Statistical studies and Research Cairo University, 5 Tharwat St. ORMAN GIZA, P.O. 12613, Egypt. Tel.: +20 2 3351499/2 3351223; fax: +20 2 7482533.
E-mail addresses: aymghazy@yahoo.com (A.M. Ghazy), Why2fatma@yahoo.com (F. EL-Licy), hehefny@ieee.org (H.A. Hefny).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.
http://dx.doi.org/10.1016/j.eij.2012.05.002

Production and hosting by Elsevier

degraded ones. The threshold AntNet algorithm presents a new strategy for updating the routing information, supported by the backward ants.

1. Introduction

Finding best routes based on dynamic data have been studied extensively in literature. Old approaches used traditional techniques such as Dijkstra algorithm, which, have high computational complexity. In the last decade, population-based searching methodologies, such as genetic algorithms (GAs) and artificial immune systems (AISs), are proposed as efficient parallel searching algorithms that can successfully be adopted for finding short paths over dynamic networks [1–3]. Swarm Intelligence (*SI*) based techniques have been also investigated as a solution for finding good routes in a dynamically changing network. Ant routing algorithms is one of the most promising *SI* techniques for finding near optimal solutions at low computational cost.

AntNet is a distributed agent based routing algorithm inspired by the behavior of natural ants [4]. It has been shown that under varying traffic loads, AntNet algorithm is amenable to the associated changes. It shows better performance than that of shortest path routing algorithms [5]. Several enhancements have been made to the AntNet algorithm. Baran and Sosa [6], proposed to initialize the routing table at each node in the network. The proposed initialization reflects previous knowledge about network topology rather than the assumed uniform distribution probabilities (as in standard AntNet algorithms) [6]. Kassabalidis et al. [7], have shown that good routing solutions can be achieved by the combined use of network clustering autonomous system and ant colony algorithm. A new type of helping ants has been introduced in [8] to increase cooperation among neighboring nodes, thereby reducing AntNet algorithm's convergence time. Tekiner et al. [9], produced a version of the AntNet algorithm that improved the throughput and the average delay. In addition, their algorithm utilized the ant/packet ratio to limit the number of used ants [9]. Radwan et al. [10], proposed an adapted AntNet protocol with blocking–expanding ring search and local retransmission technique for routing of MANET (Mobile ad hoc network).

Most of the research efforts and applications of Ant routing algorithms are directed towards finding best routes over data communication networks. However, its application for dynamic routing on road networks is still considerably limited. Dynamic routing algorithms play an important role in road traffic routing to avoid congestion and to direct cars to better routes. An Ant Based Control (ABC) algorithm has been applied in [11], for routing of road traffic through a city. In [12] a city based parking routing system (CBPRS) that used Ant based routing has been proposed. In [13] a modification to the Ant Based Control (ABC) and AntNet has been presented for routing vehicle drivers using historically-based traffic information. In [14], a version of the AntNet algorithm has been applied to improve traveling time over a road traffic network with the ability to divert traffic from congested routes.

In this paper, a new modified version of the AntNet algorithm is proposed for dynamic routing of road traffic networks. A pre-known information about good travel times among different nodes is exploited to increase the performance of the algorithm by reducing much of the computations. This enforces the agility in exploring more routes in a given network, therefore, guaranties fast coverage of the changes in the road states. The proposed algorithm, not only accelerates the discovery of good routes, but it, also, conserves it. A software agent (called check ants) were employed to monitor the conserved good routes, in such a way to discover and reject the derogated ones.

The main benefits of the proposed modifications are:

- Reducing computation time, which guaranties agility of exploring new routes.
- Conserving the discovered good routes, shrinks the size of searching problem, (for a pre-determined time) and accelerating the explorations of more routes. This would permits rapid cycles of covering the changes in the dynamic road network.
- Discovering the derogated good route enforces fast recovery to obtain better routs.

The adopted performance measures for evaluating the proposed modified algorithm with the traditional AntNet are:

- The average travel time per simulation period for each node
- The average travel time at each simulation minute for all nods.

For the purpose of this paper, the standard Antnet algorithm is introduced in Section 2. While, the proposed modified version of the algorithm is presented in Section 3. Section 4 presents the simulation experiment. Section 5 concludes the paper.

2. Description of AntNet algorithm

The AntNet routing algorithm [4], utilizes two types of software agents, namely, forward and backward ants. A forward ant is responsible for gathering information about the state of the network while moving from a source node to a destination node. A backward ant, however, is responsible for manipulating these information to update the routing tables of each node while moving backward from destination node to source node.

2.1. Data structures

Ants communicate through the information they concurrently read and write in the data structures stored at each network node. The data structures for a given node k over a network of N nodes are the two tables T_k and M_k, representing routing table and local traffic table respectively [4], as shown in Fig. 1. Each row in T_k corresponds to one destination in the network, whereas, each column corresponds to one of the neighbors of node k. For each destination d and each neighboring node n,

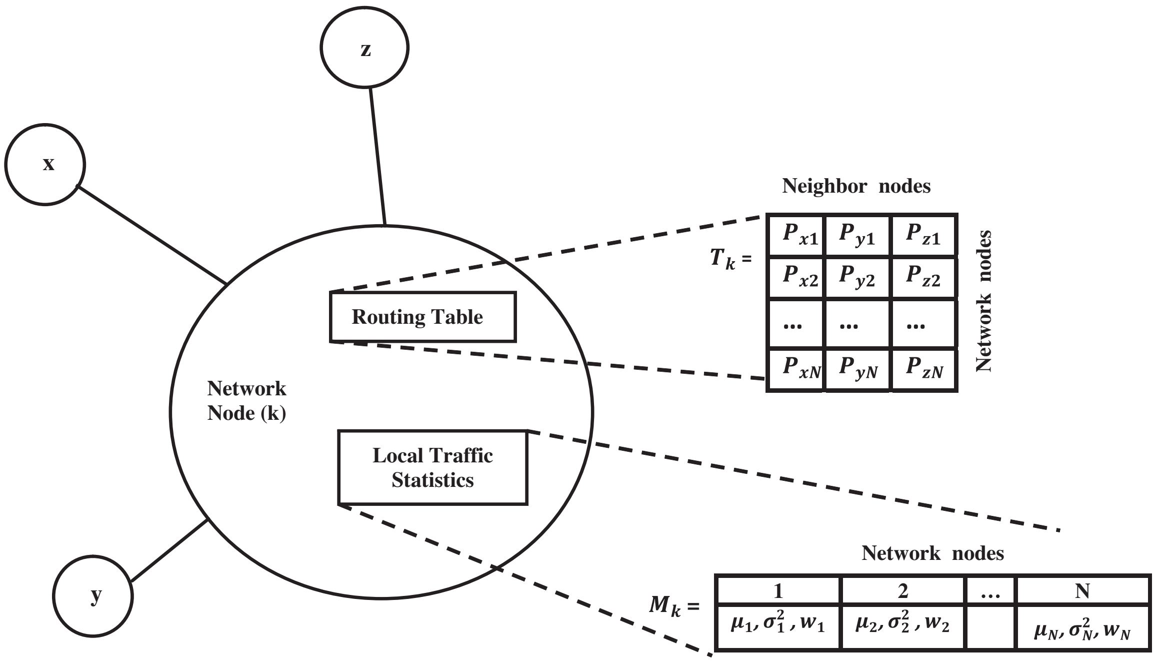
z
x
y
Network Node (k)
Routing Table
Local Traffic Statistics
Neighbor nodes
T_k =
P_{x1} P_{y1} P_{z1}
P_{x2} P_{y2} P_{z2}
P_{xN} P_{yN} P_{zN}
Network nodes
Network nodes
M_k =
1 2 … N
μ_1, σ_1^2, w_1 μ_2, σ_2^2, w_2 μ_N, σ_N^2, w_N

Fig. 1 A network node k with its routing table (T_k) and local traffic table (M_k).

the probability p_{nd} of choosing n as the next node when the destination is d, is stored in T_k such that:

$$\sum P_{nd} = 1, \quad n \in N_k \tag{1}$$

where $d \in [1, N]$ and $N_k = \{\text{neighbors }(k)\}$.

The local traffic table M_k stores statistical information about road network corresponding to node k. The Jth column of the table holds statistical information about the travel times consumed by all the forward ants that had been launched from node k to reach destination node j. It is represented in the form $M_k(\mu_d, \sigma_d^2, W_d)$, where:

μ_d is the mean travel times calculated as:

$$\mu_d \leftarrow \mu_d + \eta(t_{kd} - \mu_d) \tag{2}$$

σ_d^2 is the variance of the travel times calculated as:

$$\sigma_d^2 \leftarrow \sigma_d^2 + \eta((t_{kd} - \mu_d)^2 - \sigma_d^2) \tag{3}$$

where t_{kd} is the trip time observed by the current forward ant (launches from node k to destination d). $\eta \in (0, 1]$, weighs the number of recent samples that will affect μ_d and σ_d^2. W_d The moving observation window of size W_{max} which is used to save the trip time of the last W_{max} ants that launched from node k to node d. W_d is used to find the best trip time t_{bestd}. η is related to W_{max} as follows:

$$W_{max} = 5c/\eta, \quad where c < 1 \tag{4}$$

2.2. The AntNet algorithm

The AntNet algorithm can be described as follows [4]:

1. At regular intervals, a random destination d is chosen, and a forward ant F_{sd} is launched.
2. While F_{sd} is traveling towards the destination d, it keeps a memory about its path. When an ant arrives at a node k coming from node j the identifier of the visited node k and the travel time needed to reach k coming from j are pushed into a memory stack S_{sd}.
3. At each visited node k, the next node is selected from the neighbors that have not been visited according to the probability P_{nd}. If all the neighboring nodes have been visited previously, then the next node is chosen among all the neighbors.
4. If a cycle is detected, i.e. if the ant is forced to return to an already visited node, the cycle's nodes are popped from the ant's stack and all memory about the cycle is destroyed.
5. When the destination node d is reached, a backward ant B_{ds} is generated and the forward ant dies after transferring its memory contained in the stack S_{sd} to the backward ant.
6. The backward ant takes the opposite direction of the same path as the corresponding forward ant. At each node k, the backward ant pops up the stack S_{sd} to know the next node.
7. When arriving at a node k coming from a neighbor node h, the backward ant updates both of routing table T_k and local traffic table M_k for all the entries corresponding to the destination node d. Updates also are performed on the entries corresponding to every node $k' \in S_{sd}$, $k' \neq d$, on the sup path followed by F_{sd} when the elapsed trip time is less than $\mu + I(\mu,\sigma)$, where $I(\mu,\sigma)$ is an estimate of a confidence interval for μ.
8. The mean μ_d and variance σ_d^2 of the model M_k are updated using the formulas (2) and (3).

The routing table T_k is updated as follows:

The probability $P_{hd'}$ is increased by the reinforcement value r as:

$$P_{hd'} \leftarrow P_{hd'} + r(1 - P_{hd'}) \tag{5}$$

While the other neighbors $P_{nd'}$ are decreased by the negative reinforcement as:

$$P_{nd'} \leftarrow P_{nd'} - rP_{nd'}, \quad n \neq h, \quad n \in N_k \tag{6}$$

where h is the next node to k in the chosen path by the forward ant, d' the destination or sub path destination and N_k is the set of neighbors of node k.

Every path that is generated by a forward ant receives a positive reinforcement, $r \in (0, 1]$

$$r = c_1(t_{bestd}/t_{kd}) + c_2\{(t_{sup} - t_{bestd})/((t_{sup} - t_{bestd}) + (t_{kd} - t_{bestd}))\} \tag{7}$$

where t_{kd} is the observed forward ant's trip time from node k to the destination d. t_{bestd} is the best trip time experienced by the forward ants traveling towards the destination d over the observation window W_d.

The coefficients C_1 and C_2 weight the importance of each term in Eq. (7). C_2 is a correction coefficient, a typical upper limit for C_2 is observed by Caro and Dorigo [4], not to exceed 0.35.

The value of t_{sup} is calculated as:

$$T_{sup} = \mu_d + \sigma_d/(\surd((1-\gamma)|w_{max}|) \tag{8}$$

where γ is the selected confidence level. It is observed that best results are obtained for $\gamma \in [0.75, 0.8]$.

The value of r calculated in (7) is finally transformed by means of a squash function $s(x)$ defined as:

$$s(x) = 1/(1 + exp(a/x)), \quad x \in (0, 1], \quad a \in R^+ \tag{9}$$

$$r \leftarrow s(r)/s(1) \tag{10}$$

The squash function $s(x)$, dilutes the values of, r, while updating the routing tables.

3. The proposed threshold-based AntNet

The main strategy of the standard AntNet algorithm [4], is to search for better routes via launching ants that fetch for routes among the network nodes. Yet, even after discovering good routes, the algorithm does not account on it, but rather, seeks for ever better routes. This is, not only rational, but it is also, necessary, for a high speed data communication networks, where best routes among different hosts may change in a few milliseconds. Road-traffic conditions, however, change with a much slower rate. However dynamic, it sustain its flow or congestion conditions for a considerable period of time. This property influenced the idea behind the proposed threshold-based AntNet algorithm. The idea is constituted in the rules:

- A discovered good route at time t, is assumed to be good until proven otherwise.

A threshold values for travelling times between the nodes of a given road network, could be estimated or obtained, see Section 3.3. Those threshold values are the seeds for the typical good travel time among the network node. Thus, the rules:

- At time $t + \Delta t$, a discovered good route at time t, is assumed to be good if its travel times were bounded by a given threshold values.

Due to the dynamic changes of the state of the road traffic network, a periodical checking is necessary to examine the integrity of the assumption about the route.

Table 1 The routing table (T_k) utilized to implement the modified AntNet.

P_{X1}	P_{Y1}	P_{Z1}	T_Good_1	G_1
P_{X2}	P_{Y2}	P_{Z2}	T_Good_2	G_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
P_{XN}	P_{YN}	P_{ZN}	T_Good_N	G_N

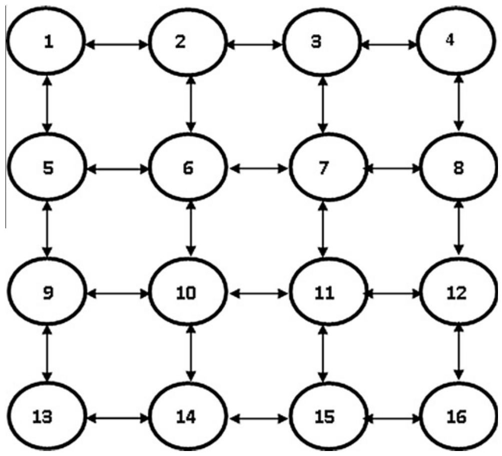

Fig. 2 The topology of a road network with 16 nodes.

- At time $t + \Delta t$, a check is on action for each discovered good route at time t. A failed check derogates the route.

The value of Δt, ranges from several minutes to several hours, depending on the network geographical location (affected by weather and road type), the time of day (peek traffic time, regular or night), the calendar time (vacation, national, religious and social events), managerial and road maintenance factors.

Utilizing this set of rules in the AntNet algorithm reduces much of the computations involved in route discovery, leading to better utilization of the computation power. Therefore, permits for discovering and monitoring more routes in the network, counterbalancing better with dynamic road states. This was the policy that inspired the enhancement in the *Threshold-Based* AntNet algorithm.

3.1. Threshold enhancement theory

Given a network of n nodes, $p_1, p_2, \ldots, p_n$, a good route R_{sd} between a source node sp_1 and destination node sp_d is defined to be the route $\{sp_1, sp_2, \ldots, sp_d\}$, such that the travel time between each consecutive nodes, sp_i and sp_{i+1}, $(i = 1, \ldots, d-1)$, in the route, is bounded by the corresponding threshold

Table 2 Number of launched ants and the average travel time.

Algorithm	Average no. of ants	Average travel time
AntNet	1667 ± 69	34.50 ± 1.82
Modified AntNet	1865 ± 163	33.42 ± 1.87
Percentage of improvement (%)	11.88	3.13

Average ± standard deviation.

values. The enhancement rules are integrated into the original AntNet as follows:

When a good route R_{sd} is discovered, the algorithm performs the following:

1. Assigns a maximum probability values for the nodes sp_i ($i = 1,\ldots,d$) in the routing table, rather than computing the ordinary update Eq. (5).
2. Stops searching for another good route from sp_i to sp_d.
3. Checks the integrity of the assumed good route after a pre-specified period of time (Δt)

To apply this enhancement a new type of ants, called " *check ants*" is introduced. Check ants are software agents that are responsible of periodically checking the travel time of the discovered good route against the corresponding threshold.

3.2. The proposed data structure

The proposed data structures for implementing the threshold AntNet algorithm is an extension to that of the standard AntNet, shown in Fig. 1. Two tables T_k and M_k, are reserved for each node k, in the given network of N nodes. What is more, the routing table, T_k is enlarged by adding two columns, as shown in Table 1. The first column, (T_Good_i), contains the exploited good travel time between the source node k and every destination node, i, in the road network. Thus, T_Good_i, $i = 1\cdots N$, are the threshold values for the travel times. The second column, (G), acts as a flag to mark those destinations for which a good routes were discovered. The value of G_i, indirectly, discriminates the type of ant to be launched as follows:

- (G_i = 'yes') → launch a check ant
- (G_i = 'no') → launch a forward followed by backward ant

3.3. Setting of threshold travel times

In road networks, good travel times among different nodes can be obtained by various ways such as:

1- Consulting traffic road experts.
2- Using the information about the available maximum speed for each link on the road network by which a route of shortest travel time can be computed between every source and destination.
3- Running the classical AntNet routing system, without any pre-assumption about good travelling times between any source and any destination. In other words, good travel time is considered to be a zero value. After running the routing system for a pre-determined period of time, best routes found by the launched ants are saved together with their good travel times in the various routing tables at every considered node.[1] This process can, also, be applied for several intervals of time during the day. Also, it can be used to provide real accurate data during critical and peak traffic day/seasons times.

[1] *This method is used in our testing experiment simulation in Section 4.*

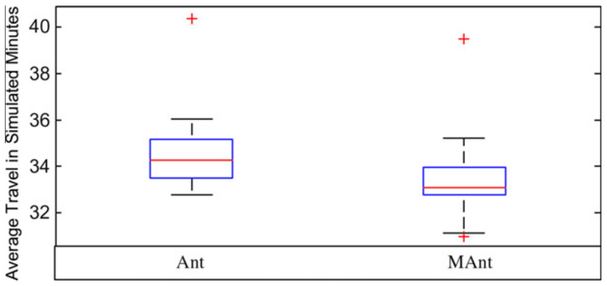

Fig. 3 Boxplot comparison between average ants'travel time for both of AntNet and modified AntNet for travelling times.

Exploitation of the pre-known good travel times as threshold values combined by the strategy of periodical check of the discovered routes achieves the following benefits:

- Reducing computation cost.
- Conserving the discovered good route.
- Accelerating convergence of a good route discovery.
- Accelerating recovery from a derogated route.

3.4. Threshold AntNet algorithm

The pseudo code for the threshold AntNet algorithm for dynamic road traffic network is illustrated bellow.[2] The lines of codes appear in bold font represent the new modifications compared with the standard AntNet algorithm.

It is worth to mention that the standard AntNet as a distributed agent based algorithm can be implemented on a parallel processing system. In such a case, each processing unit can be responsible for finding routes from one or more pre-specified nodes to all other nodes in the network. It is true, of course, that the computational complexity can be greatly reduced, in this case, from the order of O(n^2) down to order O(n). However, the argument here is that the newly proposed algorithm is still better than standard AntNet even in case of a parallel processing environment. This is due to its ability to converge fast to good route solutions and to conserve on the good discovered ones. More discussions about our modified AntNet algorithms are given in [15].

4. Experiment

A simulation program was designed and implemented (in visual basic) to test the newly Threshold-based AntNet algorithm and to compare its performance with the standard AntNet. The network under study has 16 nodes with the lattice topology shown in Fig. 2. The objective of the algorithm is to get best routes R_{1j}, $j = 2\cdots16$ between the source node 1, and the network nodes, during a certain period of time.

[2] *The modified AntNet algorithm reverts to the standard AntNet by setting the values in the column T_Good in all routing tables T_k to zero and setting all the values of the G column to "No". Such a setting does not permit the launching of check ant.*

Algorithm: Threshold-based AntNet

```
/* Main loop */
FOR each source Node (s)                    /*Concurrent activity*/
  Set t to be current time
  WHILE $t \le T$          /* T is the total experiment time */
        Select destination node to be d;
        Set $T_{sd}$ to zero                     /* $T_{sd}$ travel time from s to d */
        IF ($G_d$ = yes)
            Launch Check Ant (s, d);        /* From s to d*/
        ELSE
            Launch Forward Ant (s, d);      /* From s to d*/
            IF ($T_{sd}$ <= $T\_Good_{sd}$)           /* extracted from T-Good table */
                Set $G_d$ to 'yes'
            END IF
        END IF
  END WHILE
END FOR
Launch CHECK ANT ( source node: s , destination node: d)
      $T_{sd}$ = 0
      WHILE (current_node ≠ destination_node)
        Select next node using routing table;        /* node with highest probability */
        Get travel_time from the routing table of the source node;
                                                     /* from current node to next_node */
        Set $T_{sd}$ to be ( $T_{sd}$ + travel_time);
        Set current_node to be next_node;
      END WHILE
      IF ($T_{sd}$ > $T\_Good_{sd}$) Set $G_d$ to be 'No'
      END IF

END CHECK ANT
Launch FORWARD ANT ( source node: s , destination node: d)
      WHILE (current_node ≠ destination_node)
        Select next node using routing table
        Push on stack (next_node, travel_time);
        Set current_node = next_node;
      END WHILE
      Launch backward ant (d, s);
      Die
END FORWARD ANT

Launch BACKWARD ANT ( source node: s , destination node: d)
      WHILE (current node ≠ source node)
        Choose next node by popping the stack
        Update the traffic model
        Update the routing table ($T_{sd}$)
      END WHILE
END BACKWARD ANT
UPDATE THE ROUTING TABLE ($T_{sd}$)
      IF ($T_{sd}$<=$T\_Good_{sd}$)
      $P_{hd'} \leftarrow 1$
      $P_{nd'} \leftarrow 0, \forall n \neq h, n \in N_k$     /* h is the node "come from", k is the current node, $N_K$ is
                                                 the set of neighbors nodes and   is the path or sub
                                                 path destination */
      ELSE
      $P_{hd'} \leftarrow P_{hd'} + r(1- P_{hd'})$
      $P_{nd'} \leftarrow P_{nd'} - r P_{nd'}, \forall n \neq h, n \in N_k$       /* where r is the reinforcement value*/
      END ELSE
    END IF
END UPDATE THE ROUTING TABLE
```

4.1. Simulation data

The simulation data is generated by initializing a *From-To* time table. This time table contains information about the travel times (in minutes) between each node in the network and its neighbors. The experiment is simulated for 20 min with 5 min intervals. The travel time between each neighboring nodes (i.e. for each of the 48 links shown in Fig. 2 are,

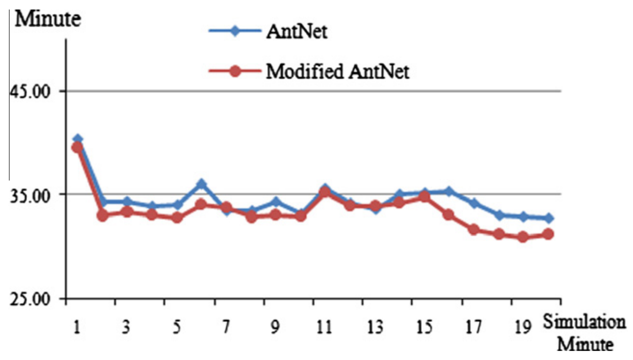

Fig. 4 The average ant's travel time at each minute.

Table 3 Average travel time at each minute.

Minute	AntNet	Threshold based AntNet
1	40.37 ± 5.55	38.99 ± 6.08
2	34.39 ± 3.97	32.44 ± 5.07
3	34.34 ± 4.68	31.97 ± 5.98
4	33.92 ± 3.92	32.02 ± 5.65
5	34.13 ± 4.08	31.8 ± 5.7
6	36.02 ± 4.60	33.86 ± 4.99
7	33.56 ± 4.63	32.87 ± 5.72
8	33.46 ± 4.28	32 ± 5.53
9	34.31 ± 4.49	32.22 ± 6.14
10	33.15 ± 5.10	32.32 ± 6.14
11	35.67 ± 4.16	35.31 ± 4.85
12	34.27 ± 3.72	33.76 ± 5.2
13	33.57 ± 4.05	33.28 ± 5.46
14	35.11 ± 4.09	33.64 ± 5.74
15	35.24 ± 4.36	33.38 ± 5.69
16	35.39 ± 3.83	32.35 ± 3.66
17	34.24 ± 3.50	31.11 ± 4.16
18	33.03 ± 2.98	30.29 ± 4.04
19	32.95 ± 3.44	30.1 ± 4.54
20	32.78 ± 3.43	29.93 ± 4.19

Average ± standard deviation.

randomly, generated in the range 3 to −15 min. Those randomly generated values are invariant during the assumed 5 min interval. However, at the end of each time interval, different set of travel times are generated, randomly, and reassigned to the corresponding entry in the *From-To* time table. Therefore, a total of 240 values for travel times are randomly generated to simulate four sudden changes in values of the travel times over the total time of the experiment.

The entries of node k routing table T_k (shown in Table 1), are initialized to be the values of the minimum travel time from that node to every destination node. For each row in T_k, the smallest of those minimums is assigned to the *T_Good* entry. The last column of T_k is preset to "*No*". It is worth to recall, as mentioned in Section 3.3, that the minimum travel times values were obtained from a previous experiment employing the standard AntNet algorithm.

4.2. The simulation process

Simulation experiments were carried out for the standard and the Threshold-based AntNet algorithms to evaluate and compare their performance. Forward or check ants are, continuously, launched from the source node 1 to some arbitrary node in the network. For each launched ant the algorithm computes the travel time to a certain destination node. Soon after, another ant is launched from the source node 1 to another arbitrary node (which may be a previously visited node). The elapsed time between the launched ants is computed (in simulation seconds). During a 60 running simulation seconds, the average of ant's travel times (from node 1) to each destination node, and that of ant's travel times to all network nodes are computed. After 20 simulation minutes, the total number of launched ants and the total averages of travel times to each destination node are computed.

This experiment was repeated with 40 different set of random data, which, were applied, each, for exercising both algorithms. The experiments were executed on the same platform with 2.2 GHz Intel core2duo processor and 2 GB RAM memory.

4.3. Experimental results

Analysis of the experiments results indicates the following observations:

1- The modified AntNet algorithm allows 11.88% increase in the number of launched ants with 3.13% decrease in their average travel time as shown in Table 2. Fig. 3 illustrates the Boxplot comparison between the whole average ants'travel time for each algorithm.
2- At each simulation minute, the average travelling time to all network nodes for the modified AntNet is less than it in the case of the standard AntNet as shown in Fig. 4 and Table 3. Detailed statistical comparisons using Boxplots are illustrated in Fig. 5, for the first 10 min, and Fig. 6, for the last 10 min.
3- Over all the simulation period, the average travelling time to each network's node for the modified AntNet is less than it in the case of the standard AntNet as shown in Table 4 and Fig. 7. Figs. 8 and 9 illustrate the Boxplot statistical comparison of the travelling time to each node.

4.4. Statistical analysis

Related t-test is utilized to ensure the significance of the achieved enhancement of AntNet algorithm. A one-tailed t-test in the positive direction was designed and applied to relate the standard AntNet to the modified algorithm. A one-tailed t-test in the positive direction is used. In our experiment, the degree of freedom (df) equals 39, then the tabulated t-value at a critical level (α) of 0.05 is found to be $t_{\text{crit}} = +1.68$. Therefore, when the computed t-value is greater than or equals to 1.68, the Null hypothesis can be rejected, which means a real significant enhancement of the proposed algorithm over the standard AntNet algorithm.

The declared t-test was applied for the data shown in Tables 2–4. Its results indicate that the Threshold-based AntNet algorithm provides:

1- A significant enhancement of the average travel time during the simulation period, over that of the standard

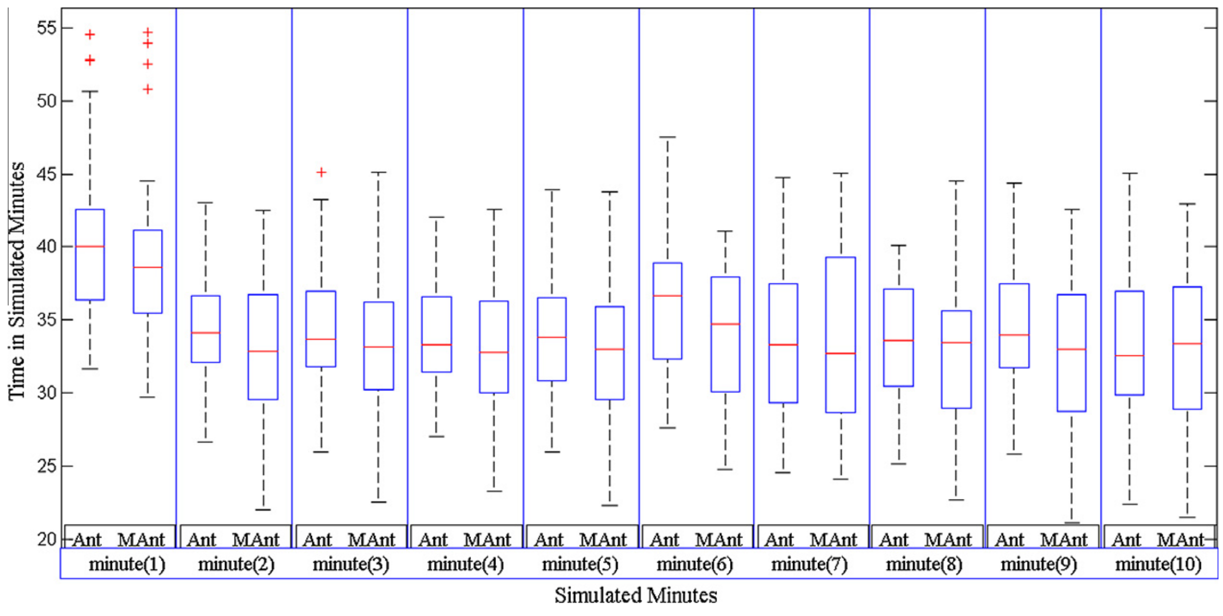

Fig. 5 Boxplot comparison between AntNet and modified AntNet for travelling times for minutes 1 up to 10.

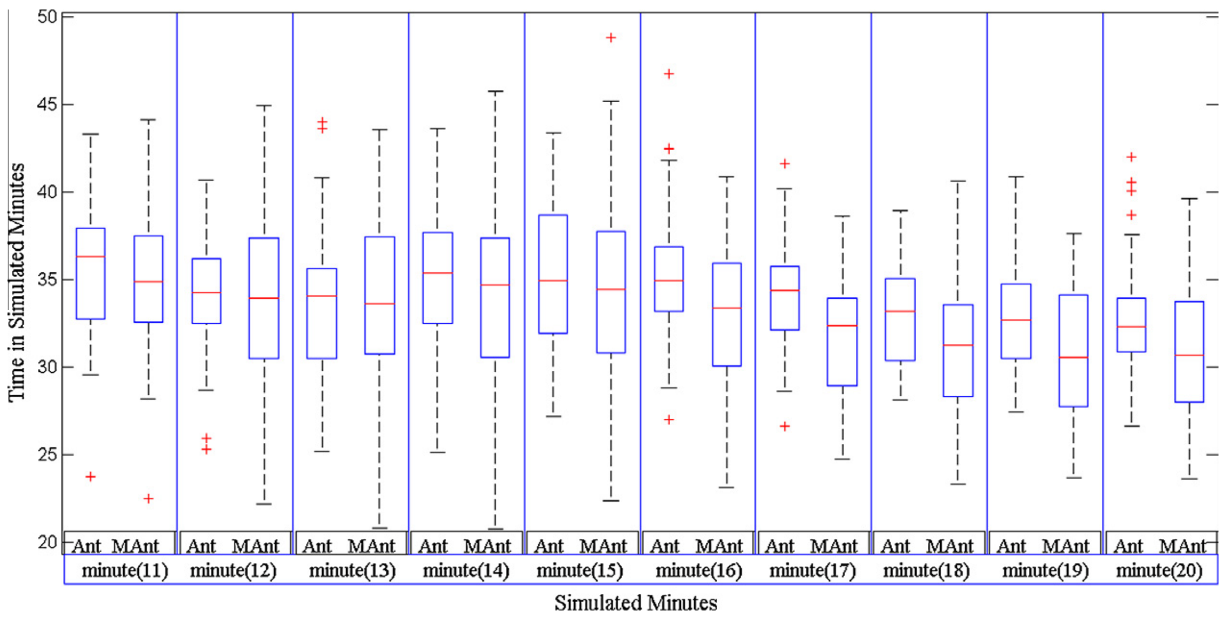

Fig. 6 Boxplot comparison between AntNet and modified AntNet for travelling times for minutes 11 up to 20.

AntNet. It is found that the computed *t*-value is 12.62 which is pretty higher than +1.68 and surely indicates a significant performance enhancement.

2- Significant enhancement during 60%, of simulation time, of the average travel time at each minute, as illustrated in Table 5. It shows a significant decreases in average travel times for 12 min, shown in red color, of the 20 simulation minutes.
3- Significant performance enhancement for 80% of the network nodes, as shown in Table 6. It shows a significant decrease in average travel times achieved for 12 nodes, shown in red color, out of the 15 visited nodes.

5. Conclusion

In this paper, a Threshold-based version of the AntNet algorithm is presented as a promising approach to be applied for dynamic traffic routing of road networks. The algorithm exploits the practical pre-known information about good travel times among various points over road network.

Exploitation of the pre-known good travel times as threshold values combined by the strategy of periodical check of the discovered routes accelerates, not only, convergence of a good route discovery, but also, recovery from a derogated route.

Table 4 Average travel time for each node.

Node	AntNet	Threshold based AntNet
2	12.39 ± 2.46	11.76 ± 2.64
3	28.56 ± 3.67	25.36 ± 4.03
4	42.97 ± 5.07	39.88 ± 4.24
5	12 ± 2.44	11.81 ± 2.59
6	19.6 ± 2.48	19.05 ± 2.24
7	32.31 ± 3.78	30.12 ± 2.96
8	44.72 ± 3.66	42.31 ± 3.74
9	23.82 ± 3.66	22.73 ± 3.72
10	29.21 ± 2.57	27.77 ± 2.59
11	41.53 ± 2.57	38.94 ± 2.59
12	50.33 ± 3.09	48.9 ± 3.78
13	37.84 ± 4.03	35.43 ± 4.33
14	40.47 ± 3.36	37.61 ± 3.84
15	46.18 ± 3.09	44.78 ± 3.26
16	55.85 ± 3.03	53.55 ± 3.00

Average ± standard deviation.

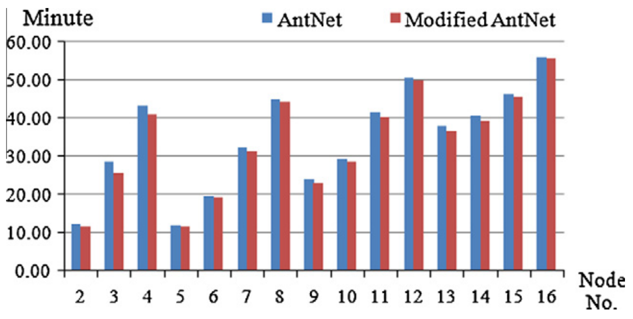

Fig. 7 The average travel time from node 1 to all other nodes during the simulation period.

In theory, the performance of the threshold algorithm for a given network of n nodes was computed to be at worse case O(n^2). Yet, the processing time for a discovered route decreases significantly to become a constant that depends, only, on the number of nodes in the given route (as it is monitored, only, by the check ant). This performance last until the route became derogated. This behavior of the algorithm permits for a rapid coverage of the network changes (more forward ants will traverse and discover good routes for the remaining destinations), therefore, counterbalancing better with dynamic road states.

The standard AntNet algorithm is amenable to the associated changes in traffic load. Dhillon and Mieghem [5], showed that it has better performance than that of the shortest path routing algorithms. In that context, the influence of the modification were measured by comparing the modified algorithm performance with that of the standard Antnet. It was tested through a designed simulation experiment. Experiment results indicated 11.88%, increase in the number of launched ants and a 3.13% decrease in the average travel time, as shown in Table 2. These results were validated, even, further by applying a one-tailed *t*-test. In general it proved the significance of the enhancement provided by the Threshold-based AntNet algorithm. In particular, it validated the significant performance of ant's travel time during 60% of the simulation time, as shown in Table 5. Also, the *t*-test confirmed the significance of enhancement performance for 80% of the network nodes, as shown in Table 6.

Experimental results ensure that the efficiency of the Threshold-based AntNet is significant with respect to the standard AntNet algorithm. The improvement in the performance was highlighted in the time efficiency that reduces computational time. Whereas, reduction in the average travel time, allows significant increase in the number of launched ants which accelerates the search for new good routes. The presented

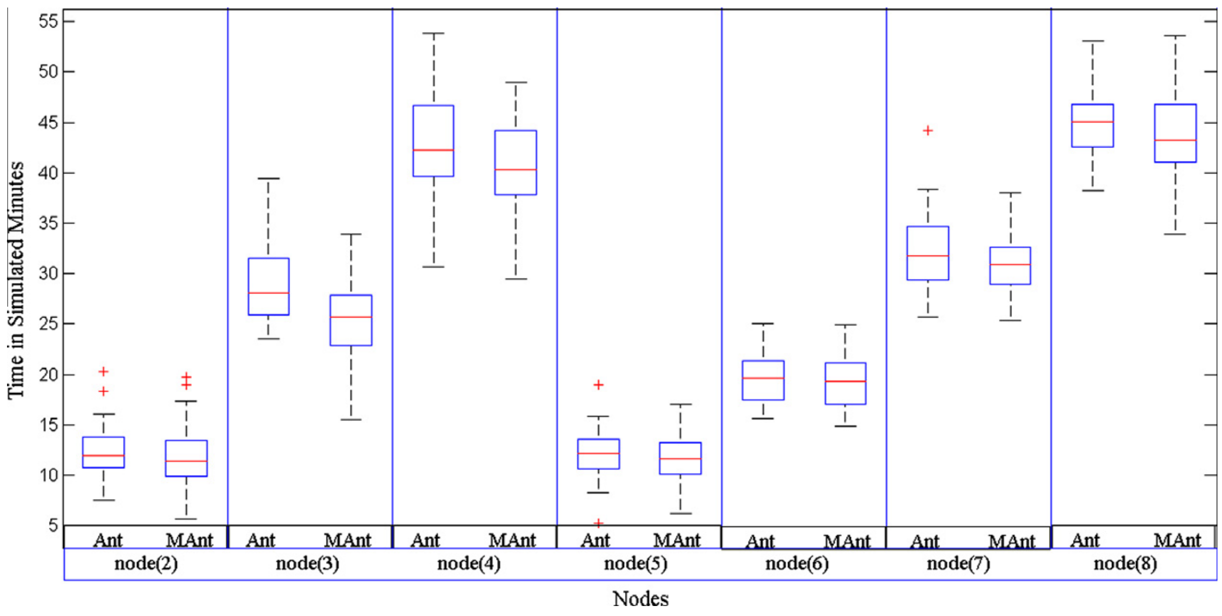

Fig. 8 Boxplot comparison between AntNet and modified AntNet For travelling times for nodes 2–8.

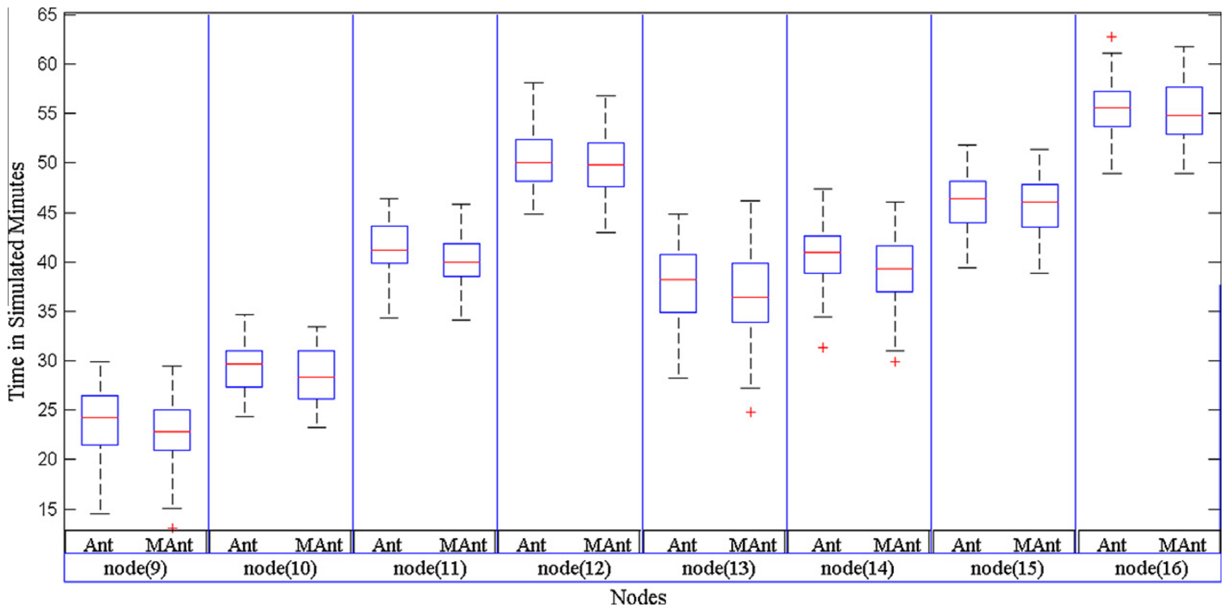

Fig. 9 Boxplot comparison between AntNet and modified AntNet for travelling times for nodes 9–16.

Table 5 Related *t*-test at each minute.

Minutes	*t*-Value	Minutes	*t*-Value	Minutes	*t*-Value	Minutes	*t*-Value
1	2.3	6	3.83	11	<u>0.94</u>	16	3.34
2	3.5	7	<u>−0.29</u>	12	<u>0.80</u>	17	5.30
3	2.04	8	<u>1.56</u>	13	<u>−0.85</u>	18	3.82
4	1.89	9	2.16	14	1.60	19	4.77
5	2.86	10	<u>0.48</u>	15	0.73	20	3.04

Table 6 Related *t*-test for each node.

Node	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Computed *t*-value	2.39	8.68	3.71	<u>1.58</u>	<u>1.63</u>	2.42	<u>1.62</u>	2.66	2.75	3.36	1.74	3.63	3.81	2.73	1.77

modifications, allowed the algorithm to preserve the discovered good routes and to, rapidly, converge toward good routes. The inspired agent, '*check ant*', assist in preserving good routes and, better yet, exposes and discards the degraded good route.

References

[1] Kanoh H, Kozuka H. Evaluation of GA-based dynamic route guidance for car navigation using cellular automata. In: IEEE intelligent vehicle symposium, 17–21 June 2002, vol. 1. p. 178–83.

[2] Yang L, Lin J, Wang D, Jia L. Dynamic route guidance based on artificial immune system. J Control Theory Appl 2007;5(4): 385–90.

[3] Hamed AY. A Genetic algorithm for finding the *k* shortest paths in a network. Egyptian Inform J 2010;11:75–9.

[4] Di Caro G, Dorigo M. AntNet: distributed stigmergetic control for communications networks. J Articial Intell Res (JAIR) 1998;9:317–65.

[5] Dhillon SS, Van Mieghem P. Performance analysis of the AntNet algorithm. Comput Networks 2007;51:2104–25.

[6] Baran B, Sosa R. AntNet routing algorithm for data networks based on mobile agents. Inteligencia Artificial, Revista Ibero-americana de Inteligencia Artificial 2001;12:75–84.

[7] Kassabalidis I, El-Sharkawi MA, Marks RJ, Arabshahi P, Gray AA. Adaptive-SDR: adaptive swarm-based distributed routing. In: Proceedings of the international joint conference on neural networks, Honolulu (HI), vol. 1; 2002, p. 351–4.

[8] Soltani A, Akbarzadeh-T M-R, Naghibzadeh M. Helping ants for adaptive network routing. J Franklin Inst 2006;343:389–403.

[9] Tekiner F, Ghassemlooy FZ, Al-khayatt S. The AntNet Routing Algorithm-Improved Version. In: Proceedings of the international symposium on communication systems networks and digital signal processing (CSNDSP), Newcastle (UK), July 2004. p. 22–6.

[10] Radwan A, Mahmoud T, Houssein E. AntNet-RSLR: a proposed ant routing protocol for MANETs. In: Proceedings of the first Saudi international electronics, communications and electronics conference (SIECPC'11), April 23–26, 2011. p. 1–6.

[11] Kroon R, Rothkrantz LJM. Dynamic vehicle routing using an ABC-algorithm. In: Transportation and telecommunication in the 3rd millennium, Prague, 2003. pp. 26–7.

[12] Boehlé J, Rothkrantz L, van Wezel M. CBPRS: a city based parking and routing system. Technical report ERS-2008-029-LIS, Erasmus Research Institute of Management, ERIM, University Rotterdam; 2008.

[13] Suson A. Dynamic routing using ant-based control. Master thesis, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology; 2010.

[14] Tatomir B, Rothkrantz LJM. Dynamic traffic routing using Ant based control. In: IEEE international conference on systems, man and cybernetics (SMC 2004) on impacts of emerging cybernetics and human-machine systems, October, 2004. p. 3970–3975.

[15] Ghazy A. Enhancement of dynamic routing using ant based control algorithm. Master thesis, Institute of Statistical Studies and Research, Cairo University; 2011.