A Novel Approach to Distributed Routing by Super-AntNet

Saeed Saffari-A., Mohammad-R. Akbarzadeh-T. and Mahmoud Naghibzadeh

Abstract — **Various forms of swarm intelligence are inspired by social behavior of insects that live collectively. AntNet is a form of such social algorithms, but it has a scalability problem with growing network size. If every node sends only one ant to each destination node and there are *N* nodes in the network, the total number of ants that are sent is *N(N-1)*. In addition with increasing overhead for large networks, most of the ants are often lost for distant destinations. Furthermore, due to long travel times, ants that do arrive may carry outdated information. In this paper, a novel hierarchical algorithm is proposed to resolve this scalability problem of AntNet. The proposed Super-AntNet divides a large scale network into several small networks that are chosen based their internal traffic patterns. A separate ant colony is then assigned to each of these networks. A Super-Ant Colony is then responsible to coordinate data routing among the colonies. Performance of Super-AntNet is compared with those of standard AntNet as well as two other conventional routing algorithms such as Distance Vector (DV) and Link State (LS) in terms of end-to-end delay, throughput, packet loss ratio, increased overhead, as well as jitter. Application to a 16-node network indicates the superiority of the proposed algorithm.**

I. Introduction

A wide variety of routing algorithms exist for communication networks. In traditional routing, routing tables are updated by exchanging routing information among the routers. In Distance-vector (DV) based routing, for instance, routing tables are exchanged, and in Link-State (LS), link state information is flooded over the network. More recently, mobile agents have been used to network routing by inspiration from ant routing algorithms. As Dhillon and Van mieghem [1] gave the general name ANTRAL (ANT Routing Algorithms) to hop-by-hop routing algorithms based on the stigmergic communication found in natural ant colonies. Stigmergy is a form of indirect communication by modifying the environment [2,4].

AntNet is a type of ANTRAL that was first proposed by Di Caro and Dorigo [4] in 1998. AntNet is a promising routing paradigm due to its complete distributed nature of information dissemination and is shown to provide good adaptation during network failures. But AntNet has a growing scalability problem with larger scale networks, because each node has to generate many ants for updating its routing table. In large networks, both over head and ant loss grow for distant destinations. Furthermore, ants may carry outdated information for long travel times.

To solve the above problem, Kassabalidis and his colleagues [6] divided the network into several clusters based on Euclidean distance/geographical distribution of nodes. They created two probability tables for routing at each node, one for within the cluster and one for in-between clusters. They also based their work on AntNet 1.0 which was the first implementation of AntNet for connection-less best-effort networks. In this work, authors propose the use of network traffic for clustering of the network. This new measure allows the cluster adaptation of the network based on its actual traffic. Furthermore, the utilized AntNet 2.0 is more reactive that its AntNet 1.0 predecessor allowing better matches with the routing requirements in high-speed connection-oriented networks where best-effort services are provided concurrently with Quality-of-Sevice (QoS) sessions.

In the proposed approach, the resulting hierarchical routing also simplifies the routing tables of the nodes. In other words, each node in a cluster maintains only two small routing tables, one for the nodes in the cluster (local) and one for other clusters (global). As a result, the size of these routing tables does not grow significantly with the growth of the network. Routing performance is hence improved by two separate ant colonies at each node, one (a regular AntNet) that operates within the cluster and one (Super-AntNet) that operates among clusters. Simulation analysis reveals that performance of this algorithm is often fast and is able to make up for AntNet's shortcomings.

The rest of this paper is organized as follows. Section 2 discusses the ant-colony based routing algorithms in general. The proposed algorithm is shown in Section 3. In Section 4 simulation results are presented and compared with LS and DV [9] as well as traditional AntNet as originally proposed in [4]. Finally, Section 5 presents the conclusions.

II. Routing Technology in ANTRALs

Swarm intelligence is often promoted in difficult and uncertain optimization problems, particularly in distributed systems [3]. In network routing, for instance, most of the current research takes inspiration from the behavior of natural ants such as: AntNet [4,5], Ant Colony Optimization for routing [2], swarm intelligence for routing [6] and helping ants for adaptive network routing [10].

There are two principal components in performance of ANTRALs. First, *exploration* for shortest path and then *update* of routing tables. Exploration and routing table updates are coupled, because the mobile agents use the same

Saeed Saffari Aman is with the Switching Center 1 (SC1) of Mashhad, Korasane Razavi Telecom. (e-mail: sa_saf_amn@yahoo.com).

Mohammad-R. Akbarzadeh-T. (IEEE Senior Member) is with the Department of Electrical Engineering, Ferdowsi University of Mashhad. (e-mail: akbarzadeh@ieee.org).

Mahmoud Naghibzadeh is with the Department of Computer Engineering, Ferdowsi University of Mashhad. (e-mail: naghibzadeh@um.ac.ir).

978-1-4244-1823-7/08/$25.00©2008 IEEE

routing table values for exploration which they use for update. Hence, improving on exploration and routing table update can affect the rate of convergence.

In all algorithms presented thus far [1-5], each node has a probability table of next hop to each destination, as shown in Table I.

TABLE I
ROUTING TABLE OF ANTRALS

Current Node = 1		Next Hop	
		0	3
Dest. Node	0	0.9	0.1
	2	0.15	0.85
	3	0.1	0.9

Each row of this table corresponds to a destination and each column to a neighbor. The values of each row are the probabilities of choosing the corresponding neighbors as the next hop. In swarm based algorithms, these values lead ants to explore the network and find better paths. By discovering a better path, probability values of tables are updated. Hence, the data packets transit more through this path, leading to higher probabilities to next hop.

A. Data Structures at Nodes

In AntNet, mobile agents communicate indirectly, through update of the two data structures T_k and M_k stored at each network node k. A routing table T_k is a matrix with probabilistic entries as shown in Figure 1. The table of each destination d and every neighbor node n, T_k stores a probability value p_{nd} that is the probability of choosing n as the next hop.

$$\sum_{n \in N_k} p_{nd} = 1 \qquad N_k = \{neighbors\ (k)\} \qquad d \in [1, N]$$

As seen by local node k, each destination d's table $M_k(\mu_d, \sigma_d^2, W_d)$ contains a moving observation window W_d, of size W_{max}. W_d is used to compute the best last trip times t_{best_d}. The average μ_d and variance σ_d^2 represent the mean and variance of the trip times experienced by forward ants to move from node k to destination node d.

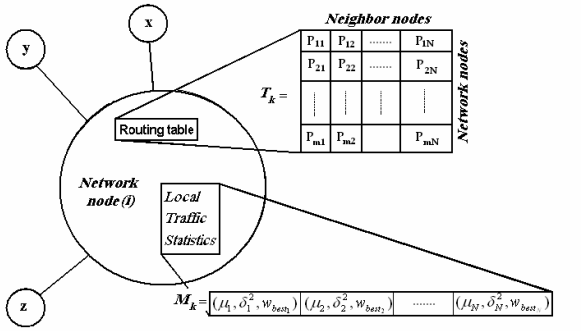

Fig. 1. The data structures of node k with neighbors x, y and z and a network with N nodes: routing table (T_k) and statistics table (M_k) [1].

In Equations (1) and (2), $t_{k \to d}$ represents the new trip times of ants that come recently from node k to destination node d and the factor η weighs the number of most recent samples that will really affect average and W_{max} is the maximum allowed size of the observation window [4,10].

$$\mu_d \leftarrow \mu_d + \eta(t_{k \to d} - \mu_d) \tag{1}$$

$$\sigma_d^2 \leftarrow \sigma_d^2 + \eta((t_{k \to d} - \mu_d)^2 - \sigma_d^2) \tag{2}$$

B. AntNet Description

AntNet can be described as follows [5, 7, 8]:

1) At regular intervals, from each source node s, a forward ant $F_{s \to d}$ is launched randomly to each destination node d.
2) The forward ants store their paths and traffic information to their stack $S_{s \to d}$ while traveling to destination nodes.
3) At each node k, each forward ant chooses the next node as follows:
   - If any of the neighboring nodes have not been visited, the next hop is one of the nodes that are not already visited.
   - The selection is based on the routing table and the size of queue of the neighbor using (3) and (4).

$$p'_{nd} = \frac{p_{nd} + \alpha l_n}{1 + \alpha(|N_k| - 1)} \tag{3}$$

$$l_n = 1 - \frac{q_n}{\sum_{n'=1}^{|N_k|} q_{n'}} \tag{4}$$

   N_k is the set of neighbors of the node k and l_n is the availability factor which is calculated according to Equation (4). The q_n in Equation (4) is the length of the queue of messages to be forwarded from node k to its neighbor node n. The value α in (3) weighs the importance of the instantaneous state of the node's queue with respect to the probability values stored in the routing table.
   - If all the neighbors have been already visited, then next node is selected with equal probability.
4) If a cycle is detected, all nodes composing the cycle are popped from the ant's stack.
5) After forward ant $F_{s \to d}$ is reached to node d, it produces backward ant $B_{d \to s}$. The backward ant returns to the source node by using the same path as the forward ant in opposite direction.
6) When the backward ant arrives at node k from neighbor h, it can update the two data structure T_k and M_k of node k.

For more information see our earlier work mention in [10].

III. THE PROPOSED ALGORITHM

Every network is characterized by a certain traffic pattern among its member nodes. Considering this traffic pattern and its corresponding topology can greatly influence routing performance. For example, the internal network of a given

department is designed for transporting data and administrative works within it. Hence, it has a lot of communication within itself. On the other hand, it has little communication with other networks that are outside of it. In such case, by dividing the network to several local clusters, we can do local routing and it does not need to know about the other network nodes.

In a large planar network, when a part of the network is changed, it is more important to issue this variation to all of the nodes in the network. But this variation is more important for nodes that have more communication with it. Therefore, network clustering allows nodes of a certain cluster to only store the information of that cluster, and the nodes become free of the rest of the network's routing information. Even if parts of the network alter such as due to link or node failures, these variations will have effect mainly on that specific part of the network, and other parts of the network are less affected by it. Hence, nodes can continue routing into the colonies, without being aware of these variations. Moreover, the speed of routing increases, because they have little interaction between routing data and external nodes.

In this research, use of intelligent agents at two levels of local and global is proposed to significantly reduce the size of routing table and hence routing is simplified. The proposed algorithm has 3 steps. The first step is dividing the set of network nodes into clusters based on their traffic pattern. For example, the nodes that use same language have more data exchange with each other (relation Spanish nodes with each other). The Second step is routing by mobile agents, and the third step is sending traffic to the destination via discovered routes by the ants. The first step is only done when there is a change in general topology. And so it is not frequent. The second and third steps are the usual routing algorithms that are operated in real time.

A. *Network Clustering*

AntNet has a scalability problem. If every node sends only one ant to each destination node and we have *N* nodes at the network, the total number of ants that have to be sent is *N(N-1)*. This amount of overhead for a big size network is too high. Furthermore, for distant destinations, most of the ants are lost. In addition, long travel times make that ant's information outdated. This problem is solved by network clustering into colonies. I. Kassabalidis and the other authors of [6] chosen k-means clustering, where the distance function is the Euclidean distance between nodes. But, our clustering is done based on the relation of nodes to each other, as mentioned above.

B. *Routing*

After network clustering, two kinds of agents (ants) are identified for network, "*local ant*" and "*super ant*". Local ants are used to discover routes within their clusters, while super ants are used to find routes among clusters. Hence, each node has two kinds of routing tables; a *local routing table* and a *super routing table*, as shown in Table II, III.

TABLE II
LOCAL ROUTING TABLE

Current Node =1		Next Hop	
		0	3
Dest. Node	0	0.9	0.1
	2	0.15	0.85
	3	0.1	0.9

TABLE III
SUPER ROUTING TABLE

Current Node =1		Next Hop		
		0	3	4
Dest. Cluster	B	0.05	0.15	0.8
	C	0.05	0.9	0.05
	D	0.2	0.75	0.05

In the super routing table, all the neighbors of the nodes contain a nodes list of next hop. But, in the local routing table, only local nodes that belong to the same cluster can be chosen as a next hop. For example, according to Fig. 3, when we are in node 1 at cluster A, the next hop for destination node 2 is specified by local routing table as node 3. When the destination node is 11 in cluster C the next hop is specified by the super routing table as node 3. See the pseudo code of Super-Ant in Fig. 2 for more conception. The probabilities of these tables are computing just like AntNet algorithm, while adding some features as below:

```
t := Current time;
t_end := Time length of the simulation;
Δt := Time interval between ants generation;
foreach (Node)   /*Concurrent activity over the network */
  M= Cluster traffic model;
  T = Super routing table;
  while (t ≠ t_end)
    in parallel   /* Concurrent activity on each node */
      if (t mod Δt = 0)
        destination node := SelectDestinationNode_of_another_cluster();
        LaunchSuperForwardAnt(destination_node, source_node);
      end if
      foreach (ActiveSuperForwardAnt[source node, current node, destination node])
        while (current_node ≠ each_node_of_the_destination_cluster)
          next hop node := SelectLink(current_node, destination_node, T, link_queues);
          PutAntOnLinkQueue(current_node, next_hop_node);
          WaitOnDataLinkQueue(current_node, next_hop_node);
          CrossTheLink(current_node, next_hop_node);
          PushOnTheStack(next_hop_node, elapsed_time);
          current node := next hop node;
        end while
        LaunchSuperBackwardAnt(destination_node, source_node, stack_data);
        DieSuprForwardAnt();
      end foreach
      foreach (ActiveSuperBackwardAnt[source_node, current_node, destination_node])
        while (current_node ≠ destination_node)
          next hop node := PopTheStack();
          WaitOnHighPriorityLinkQueue(current_node, next_hop_node);
          CrossTheLink(current_node, next_hop_node);
          UpdateLocalTra_cModel(M, current_node, source_node, stack_data);
          reinforcement := GetReinforcement(current_node, source_node, stack_data, M);
          UpdateLocalRoutingTable(T, current_node, source_node, reinforcement);
        end while
      end foreach
    end in parallel
  end while
end foreach
```

Fig. 2. The pseudo code of Super-AntNet.

1) At a regular time, one local ant is sent to the node of the same cluster and one super ant is sent to other clusters.
2) Local ants find next hop within local routing tables and super-ants find it within super-routing tables.
3) Both of them store routing information and corresponding arrival times in their stack.
4) If an ant comes to a node that it has visited previously, it removes all loop information from its memory and chooses next hop randomly.
Notice that, ants have a specified hopcount for traveling. If they are local ants, their hopcount is equal to their cluster's number of nodes. If they are super-ants, their hopcount is equal to the total number of nodes in the network.
5) When an ant reaches to its destination, it produces a backward ant and returns it to the source node within the same path. It also has the permission to update the routing tables of intermediate nodes. Note that, super ants produce backward super ants as well, immediately after reaching the first node of their destination cluster.

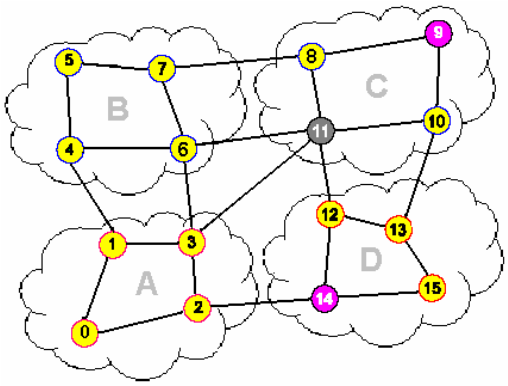

Fig. 3 Topology of the 16 node networks with 4 clusters.

C. Sending Traffic

After setting up the routing tables by ants, data packets choose their next hop according to the highest probable for going to destination node. In our simulation, data packets use the following procedure:
1) When a data packet is produced at the source, it knows the destination node and cluster.
2) If source and destination nodes are not at the same cluster, the data packets chooses the next hop according to the highest probability of the super-routing table.
3) Otherwise it uses the local routing table.

IV. Experimental Results

For showing the above concept, the proposed algorithm is simulated on 16 node network at NS2 [12]. According Fig. 3, the network is divided into 4 clusters. All the links are duplex and there are 2 sources for producing the Constant Bit Rate (CBR) traffic.

At first, the dynamically of the algorithm is examined with starting CBR flow from node 9 to node 11. It specifies the shortest path as 9-8-11. But after the link 9-8 fails, data is routed through 9-10-11. This procedure is examined for source node 14 and destination node 11. The results were similar in dynamically to local routing as above.

The bit rate of source node 14 is 2Mbps, and for source node 9 it is 8Mbps. The packet size is 1000 bytes for node 9 and 256 bytes for node 14. Note that, sending local traffic time is more than non-local, because we have supposed that, more of the network communication is local.

To verify the implementation of our proposed *Super-AntNet* algorithm, the simulation results are compared with three cases (standard AntNet and two traditional routing algorithms LS and DV [9]).

A. Compare Super-AntNet and AntNet

Some of network parameters such as: end-to-end delay (sec), receiving throughput (bit/sec), packet loss rate, jitter and overhead were computed for measurement of QoS.

1) End-to-End Delay: According to Fig. 4, end-to-end delay of Super-AntNet at local routing is close to that of AntNet, but for destination nodes that are not in the same cluster as the source node, Super-AntNet is significantly better than AntNet. This improvement is due to using super routing tables. When the destination node is specified at the outside of cluster, it does not need to search it from all destinations, while it searches the destination cluster of the node from the super routing table. Also, the number of clusters is much lower than the number of nodes in AntNet routing table.

The other reason for lower delay in Super-AntNet is due to the modified routing table's information; because ants do not need to travel long distances at the network. They return when they reach the first node of the destination cluster; therefore, they return the most recent variations. Fig. 5 shows a comparison of delay along the parts of simulation.

2) Receiving Throughput: As Fig. 6 indicates, the receiving throughput of Super-AntNet in the local region is approximately comparable to AntNet. But in Non-Local fields Super-AntNet results is better than AntNet, because with clustering we have fewer packet losses and more packets arriving at their destinations. Fig. 7 shows this improvement clearly.

3) Packet Loss Rate: According to Table. IV, packet loss of AntNet is significantly higher than Super-AntNet, because we have more lost packets in AntNet in the Non-local field. Moreover, as mentioned above, ant loss rate at long distant paths is high in AntNet.

TABLE IV
PACKET LOSS VALUES FOR DATA PACKET AND OVERHEAD.

Number of lost packets	Total	Local traffic	Non-Local traffic	Overheead
Super-AntNet	3	0	2	1
AntNet	3735	0	3538	197

4) Jitter Error: Jitter error or delay variation is the other parameter for showing convergence rate. As Fig. 8 shows the average of jitter of Super-AntNet is less than AntNet.

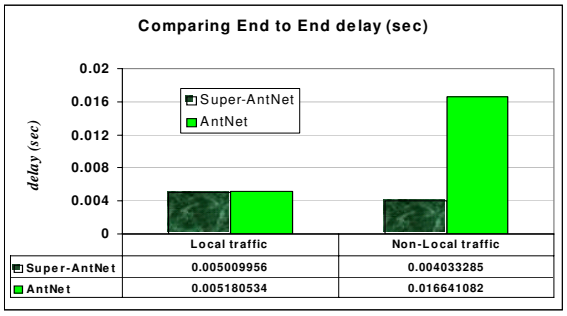

Fig. 4. End-to-End delay for Super-AntNet and standard AntNet.

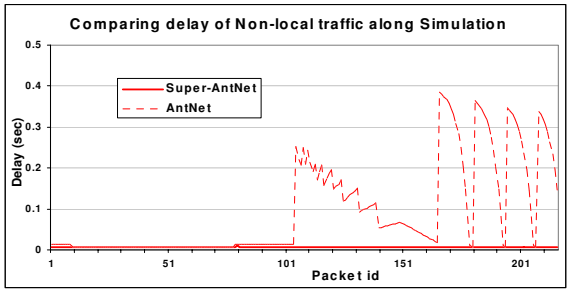

Fig. 5. Comparison of the delay along simulation.

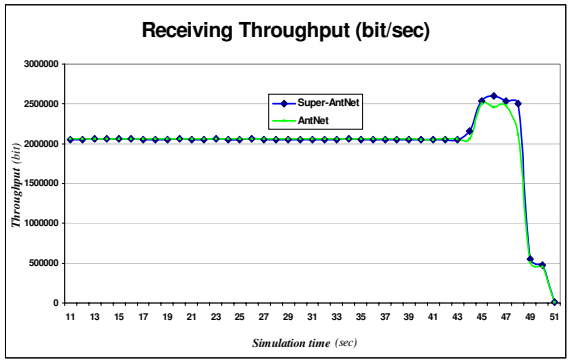

Fig. 6. Comparison of receiving throughput along the simulation time.

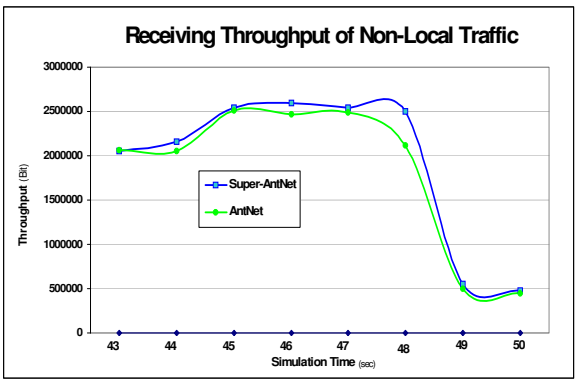

Fig. 7. Comparison of receiving throughput in non-local field.

5) Overhead: One of the suitable properties for routing algorithm is decreasing overhead, because we should have optimum use of bandwidth. As we use two kinds of ants for Super-AntNet, our ant generation rate must be half of AntNet for correct comparison of results. However, if ants have high circulation and are forwarded continuously by intermediate nodes, they impose more overhead to the network. So, overhead is computed based on sending ants at the network (including the intermediate nodes). According to Fig. 9 the number of sending overhead bytes (circulating of ants) in Super-AntNet is much lower than AntNet, because the local ants travel into clusters, firstly and super ants return to source node after reaching the first node of destination cluster. But, the question is, "what ant generation rate yields the optimal routing?" . Fig. 10 shows a comparison of the number of ant losses by increasing ant generation rate.

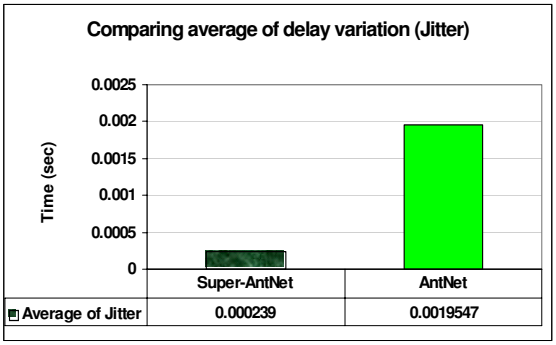

Fig. 8. Comparison of average of delay variation (Jitter Error).

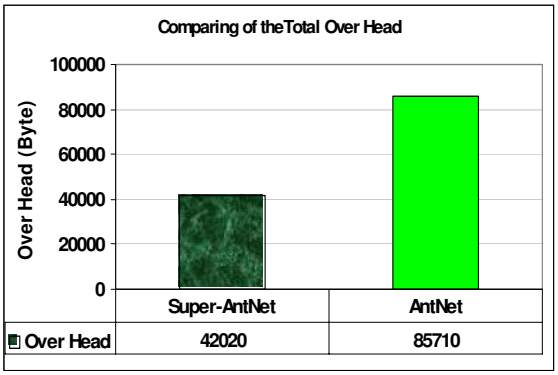

Fig. 9. Comparison of overhead between Super-AntNet and AntNet.

As you see in Fig. 10 in AntNet, when ant generation rate is increasing, ant loss is growing, too. It shows some of the ants in high ant generation rate are losing along the distant paths, but Super-AntNet has not such problem.

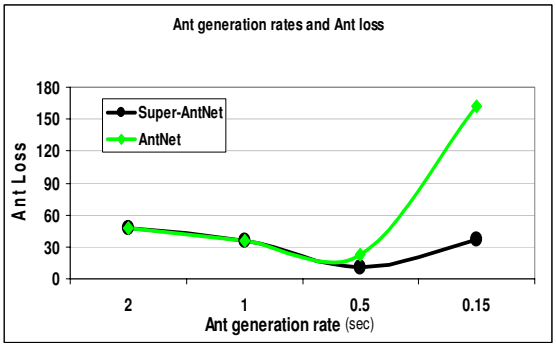

Fig. 10. Comparison of overhead between Super-AntNet and AntNet.

B. *Super-AntNet with Traditional Routing Algorithms*

The proposed algorithm is comparable on several principal measures of routing when compared with DV and LS, two traditional routing algorithms.

1) End-to-End Delay: According to Fig. 11, end-to-end delay of Super-AntNet is less than LS. It is worse than DV. This is because as in Fig 14, it has a higher overhead from DV.

2) Receiving Throughput: As Fig. 12 indicates, the receiving throughput of Super-AntNet is better than all others. So as seen in Fig. 13, it has better throughput in non-local regions than others.

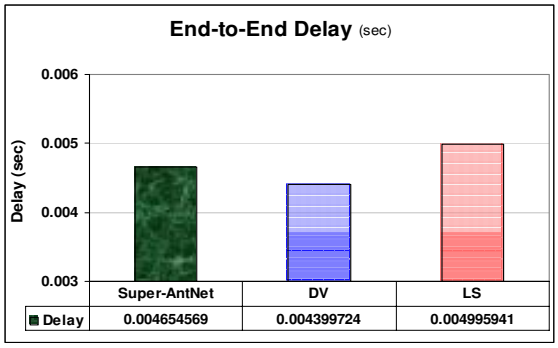

Fig. 11. Comparison of End-to-End delay (sec)

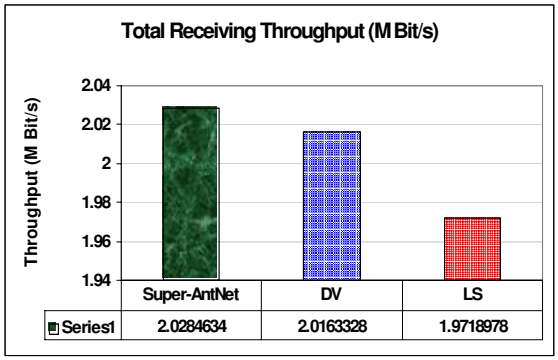

Fig. 12. Comparison of receiving throughput.

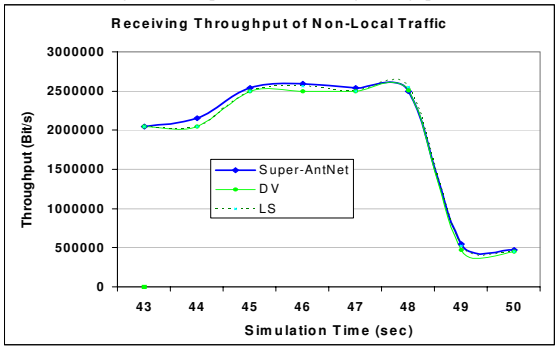

Fig. 13. Comparison of receiving throughput at Non-local field.

3) Packet Loss Rate and Jitter Error: Packet loss rate and jitter of Super-AntNet is higher than the others, but this can be acceptable considering the high overhead of LS as Fig.14.

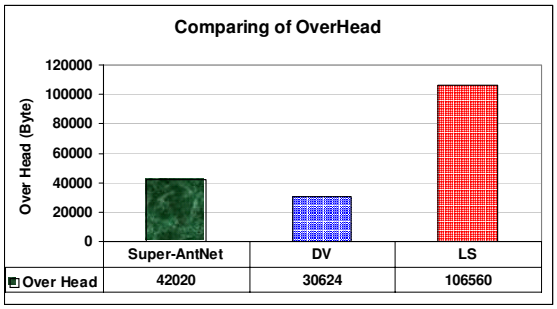

Fig. 14. Comparison of overhead among Super-AntNet, DV and LS.

V. Conclusion

In this paper, the problem of scalability in traditional AntNet is addressed by proposing hierarchical network routing. Super-AntNet improves routing by local and global routing. It divides the network into clusters by considering traffic among nodes. Routing into clusters is performed without any knowledge of routing variation outside of the clusters. So it tends to be an efficient and fast routing with less overhead and packet loss as well as higher throughput, especially in long distances.

Simulation results using NS2 confirms the advantage of the proposed algorithm as compared with standard AntNet. Super AntNet shows that it can route data packet more efficiently and to avoid going to false long paths. Results indicate that, by applying the hierarchical Super-AntNet, the total delay of the traditional algorithm is improved by 16.22% and also 24.23% for far destinations when compared with AntNet. Future directions of this research include testing the algorithm on more complex networks such as WAN.

Acknowledgment

We should thanks to Department of Research and Development of Korasane Razavi Telecommunication Company for support and also Azadeh Soltani and Raheleh Motakef for their cooperation.

References

[1] S.S. Dhillon, P. Van Mieghem, "Performance analysis of the AntNet algorithm, Computer Network," (2006), doi:10.1016/j.comnet. 2006.11.002.

[2] K.M. Sim ,W. H. Sun,"Ant Colony Optimization for Routing and Load-Balancing, " *IEEE transactions on systems, man, and Cybernetics part A:systems and humans*, vol. 33.No.5.september 2003.

[3] E. Bonabeau, M. Dorigo, and G. Théraulaz, "Swarm intelligence: from natural to artificial systems*," Oxford University Press*, 1999.

[4] G. Di Caro and M. Dorigo, "AntNet: distributed stigmergetic control for communications networks," *Journal of Artificial Intelligence Research*, vol. 9, pp. 317-365, 1998.

[5] G. Di Caro and M. Dorigo, "AntNet: a mobile agents approach to adaptive routing*," Tech. Rep*. IRIDIA/97-12, University Libre de Bruxelles, Belgium.

[6] I. Kassabalidis, M.A. El-Sharkawi, R.J. Marks II, P. Arabshahi, and A.A. Gray, "Adaptive-SDR: adaptive swarm-based distributed routing ", *Proc. IEEE World Congress on Computational Intelligence*, May 12-17 2002.

[7] Richard S. Sutton, Andrew G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.
[8] G. Di Caro, M. Dorigo, "Two ant colony algorithms for best-effort routing in datagram networks," *in Proceedings of the 10th International Conference on Parallel and Distributed Computing and Systems*, 1998.
[9] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall, Inc, Upper Saddle River, New Jersey, 1992.
[10] Azadeh Soltani, M.-R. Akbarzadeh-T and M. Naghibzadeh, " Helping ants for adaptive network routing," *Journal of the Franklin Institute* Volume 343, Issues 4-5 , July-August 2006, Pages 389-403.
[11] G. Di Caro and M. Dorigo, "Two ant colony algorithms for best-effort routing in datagram networks" *In Proceedings of the Tenth IASTED International Conference on Parallel and Distributed Computing and Systems* (PDCS'98), pages 541-546. IASTED/ACTA Press, 1998.
[12] Network Simulator, V.2., http://www.isi.edu/nsnam/ns.