AJSTD Vol. 25 Issue 1 pp. 81-93 (2008)

ANT INTELLIGENCE ROUTING

Chye Ong Gan, K. Daniel Wong, and Wei-Lee Woon

Malaysia University of Science and Technology

Received 30 October 2006

ABSTRACT

We introduce AIR, a new ant-based routing protocol for ad hoc wireless networks. AIR incorporates recent technology advances in ad hoc routing protocols, such as ring-based searching and third-party reply, while retaining the characteristics and benefits of ant-based routing protocols, including probabilistic routing tables and pheromone evaporation. In addition, we combine the best features of previous ant-based routing protocols with new features like update ants and two-way route establishment, to further improve performance. We compare the performance of AIR against AODV and DSR using appropriately chosen simulations, the results of which are presented and discussed. We find that AIR is especially useful for ad hoc networks requiring low end-to-end delay where mobility rates are high.

Keywords: ad hoc networks, routing.

1. INTRODUCTION

Two of the most prominent ad hoc routing protocols are Ad hoc On-Demand Distance Vector (AODV [1]) and Dynamic Source Routing (DSR [2]), both of which find routes reactively, i.e., they are on-demand routing protocols.

A class of ad hoc routing protocols has recently emerged that borrows ideas from "swarm intelligence." Finding routes in an ad hoc network is similar in some ways to the problem of swarms (of ants, bees, *etc.*) finding routes to food sources. While individual agents (e.g., an ant, analogous to a route request packet) are dumb, their collective behavior appears intelligent – as a group, they can find routes to desired locations. However, ants' paths to food sources are specified based on levels of pheromones cumulatively deposited by ants traversing a particular path. These pheromones evaporate with time, providing a graceful decay mechanism for the established routes. Ad hoc routing protocols have been recently developed that incorporate such characteristics (probabilistic routes, route decay, *etc.*). These routing algorithms form the class of ant-based routing algorithms for ad hoc networks. Ant-based routing algorithms are known for their good throughput and end-to-end delay performance.

In the meantime, traditional ad hoc routing protocols have continued to be refined with the latest ideas, such as the expanding ring search procedure. Features like the expanding ring search procedure improve the scalability and performance of ad hoc routing protocols [4]. The authors of Reference [4] conjecture that the expanding ring search procedure should be beneficial for other on-demand routing protocols as well, although they only simulate its performance for AODV. Thus, our first main contribution is to propose a new ant-based ad hoc routing protocol that expanding ring search while retaining the beneficial aspects of ant-based routing protocols like probabilistic routing tables and pheromone evaporation. Our second main contribution is to combine the most attractive features of earlier ant-based routing algorithms with new features,

including update ants and two-way route establishment, which have not been tried in earlier ant-based routing algorithms as far as we know. The combination of our contributions is in the form of our new protocol, AIR (Ant Intelligence Routing).

Prominent ant-based ad hoc routing protocols include AntNet [5], ARA [6], ANSI [7] and Termite [8]. Typically, ant-based ad hoc routing protocols utilize *forward ants* to go seek for a particular destination and *backward ants* in that return from the destination to the source. Usually, if there are paths to a destination, one of them is chosen randomly, where the probability distribution of the choice is proportional to the pheromone levels for each path. AntNet [5] is pro-active, periodically trying to find routes to randomly selected destinations (although weighted by the volume of traffic previously sent to that destination). Forward ants only collect information on trip time between each pair of intermediate nodes. Backward ants inherit route information from the forward ants and use it to update the pheromone values in the routing tables in intermediate nodes' routing tables. The pheromone updates are larger for shorter trip time (of the forward ants from source to destination).

In ARA [6], route discovery is achieved by flooding forward ants to the destination, as is the case with AntNet. The key difference in ARA is that routes are maintained primarily by data packets as they flow through the network. Instead of using periodic ants as with AntNet, the path to the destination is reinforced by increasing the pheromone value in the routing table as data packets travel along. ARA implements exponential pheromone decay.

ANSI [7] is a hybrid of both proactive and reactive routing. It incorporates the benefits of proactive routing with the flexibility and scalability of reactive routing. Each node proactively maintains and manages connectivity with neighbors located within a zone defined by a pre-specified radius. The actual value of the radius will depend on the network size and other characteristics such as node mobility. For larger networks with highly mobile nodes, the zone radius is set to a larger value so that the amount of proactive activity can be increased. It is usually acceptable to set the zone to include all nodes within two-hops.

Reactive routing is utilized only when the destination node is not located within the zone radius of the source node. In this case, ANSI uses the same route discovery mechanism as AntNet where forward ants are broadcast in search of the destination node.

Because each node maintains routes to every other node within its zone radius, the network does not suffer from high delays during the route discovery or route setup phases. Nevertheless, due to the proactive aspect of ANSI, periodic updates are frequently broadcast which creates high overhead, resulting in higher power consumption and bandwidth utilization. However, in general this is not as severe as in the cases with fully proactive approaches.

"Termite" adopts the analogy of termites instead of ants. Each network node is analogous to a termite hill. The more termites passing through a node, the more pheromone would be collected at the node, making it a preferred next-hop node for other packets. There are five types of packets used in Termite. These are data, hello, RREQ, RREP and seed. Data packets are routed based on the pheromone levels of each outgoing link. On the arrival of a data packet, a given node will increase the pheromone levels associated with the node which emitted the packet.

There is no flooding involved in Termite. A limited number of RREQ packets perform a random walk over the network in search of the destination. The reply packet is routed by following the pheromone trail left by the request packet [8]. However, such random walk may sometimes results in the destination not being found or in the inadvertent discovery of a sub-optimal route.

Termites also utilize a hello mechanism. Hello packets are broadcast at regular intervals until a reply is received. A reply is sent by all nodes who hear the hello. Seed packets are used to actively spread a node's pheromones throughout the network. Seeds propagate via "random walks" through the network and serve to advertise a node's existence.

The main drawback of Termite is the high rate of dropped packets. This is especially pronounced in highly mobile networks. It does not define any route failure handling mechanism. Packets are dropped immediately if no alternate routes can be found.

We explain the features of AIR in Section [2]. To evaluate the effectiveness of the proposed approach, we have performed a number of simulations aimed at comparing the performance of AIR with AODV and DSR. The methodology employed and the results obtained from these simulations are discussed in Section [3].

2. OUR PROPOSED NEW ROUTING PROTOCOL

2.1 Route discovery strategy

Like other ant-based routing protocols, AIR (Ant Intelligence Routing) utilizes a collection of routing agents called forward ants to search for the destination host. When a path to an unknown destination is required, the source node initiates a route discovery procedure. However, existing ant-based algorithms suffer from a number of performance issues in particular high overhead and route discovery latencies. To address these problems, AIR incorporates a combination of enhancements that hitherto have not been attempted in ant-based ad hoc routing algorithms. In the route discovery phase, these include the expanding ring search procedure and two-way route establishment.

2.1.1 Expanding ring search model

Because flooding a large network can be a great waste of resources, there is a need for efficient distribution of forward ants in the network. The goal of the ring search model, borrowed from AODV, is to provide a mechanism whereby forward ants may be efficiently propagated to each node without incurring excessively high overhead. In a ring search procedure, forward ants are flooded in batches, first in a small radius, and only to progressively larger distances if the destination can not be found within the initial, smaller search area. The search radius is controlled by setting a maximum hop count. Ants exceeding the maximum number of hops are terminated. Only when there is no reply from the destination within a certain time limit, will the search radius be progressively increased so that a larger set of nodes can be reached. In this way, flooding and unnecessary overhead will be reduced on the average [4].

To quicken the process of route discovery and to further reduce flooding, AIR adopts a third-party reply model. Any intermediate node with knowledge of an existing route to a requested destination can take the initiative to generate a backward ant in reply to a forward ant, instead of continuing to propagate forward ants whose destination it knows how to reach. This eliminates the need for the forward ants to rediscover a previously known route, thus utilizing existing information in the network to avoid redundant effort. This strategy works efficiently with the ring search model to significantly reduce routing overhead.

AIR does not require the extra overhead of sequence numbers to prevent loops as is the case with AODV. For each node visited by the forward ant, the node's unique address is appended to the ant stack. A node receiving the forward ant would make sure that it has never seen this particular ant before by checking if its own address is included in this stack. Any ants found to have taken a circuitous route are redundant and will hence be discarded.

2.2 Route establishment strategy

Each forward ant carries in its header a stack listing the address of each node it has visited. Every node visited by a forward ant appends its own address to the ant's stack before forwarding the ant to other nodes. When a route is found, this stack is copied to the backward

ants so that they can trace the reverse route to reach the source node in order to initiate data packet transmission.

AIR implements a two-way route establishment technique. Forward ants and backward ants also have a common task that is to established routes. Unlike other ant-based protocols, the role of forward ants in AIR is not restricted to route discovery but also includes establishing reverse routes. As forward ants travel forward in finding the destination, they establish reverse routes from visited nodes towards the source. While backward ants are agents that establish the route to the destination node, forward ants establish reverse routes from the "destination" to the "source". Using this two-way route establishment model, the forward and reverse route can be established. This is especially important for transport protocols that require bidirectional paths, such as TCP.

This strategy enables the source node to learn routes to each intermediate node in addition to the intended destination. Similarly, these intermediate nodes are also able to learn routes to every other node on the route.

Ants in AIR establish multiple routes to a given destination. Due to rapid changes in network topology, routes may suffer from frequent breakdowns. Maintaining a list of alternate routes reduces delays and overhead by not always requiring another flooding of ants when a particular path is broken. In AIR, through random search and broadcasting of forward ants in the network, several different routes can be established between the source and destination. Different forward ants travel over different paths between the sources and the destination, establishing a number of alternate routes.

One problem with having multiple paths is that the algorithm now needs to select the optimal route for data delivery. Again, this can be addressed by referring to the metaphor of ant foraging activity. Studies of ant behavioral patterns 9 indicate that ants are capable of finding the shortest path from the nest to the food source. Ants use an intelligent yet simple strategy of laying pheromone that helps the colony decide on which path is the shortest. Fortuitously, this mechanism can also be adopted for use in mobile ad hoc networks to find the optimal path for routing.

For AIR, each route in the node's routing table is assigned a score or "pheromone" value that represents the quality of the route. When an ant reaches a node, the initial pheromone value is calculated based on the information collected by the ant. This value is then assigned to the route entry in the node's routing table. It depends on two parameters:

- Time taken to travel from its source to the present node
- Number of hops needed to reach the node

The length of the path is given by the number of hops in the path, whereas delays due to congestion or longer processing times at the intermediate nodes are quantified by the time parameter. For the time parameter to be accurately measured, the ants are forwarded with the same queuing priority as data packets, instead of being given special priority.

The pheromone value is calculated based on the following exponential decay expression:

$$\text{Pheromone, } p_{id} = \mu e^{-hT} \qquad \text{i)}$$

where h is the number of hops traversed, T is the trip time and μ is a constant parameter.

$$T = CURRENT\_TIME - time\_stamp \qquad \text{ii)}$$

CURRENT_TIME is the time when the ant reaches the current node and *time_stamp* is the time when the ant was created.

Pheromones in AIR serve a dual purpose. Firstly, it directs data packets to the optimal route based on the amount of pheromone associated with each outgoing link. Secondly, in conjunction with the third-party reply model, forward ants often do not need to actually reach the destination in order to establish a route. Rather, it only needs to discover the beginning of a pheromone trail to the destination. This results in faster route discovery.

2.3 Route maintenance strategy

Similar to AODV, AIR uses a table-based approach to route the data packets. Each entry in the routing table contains only the next hop and the distance to the destination.

In keeping with the biological analogy, as more ants travel along a path, each ant deposits pheromones causing the cumulative strength of pheromone scent along that path to increase. Over time, all ants in the colony would converge to that optimal path.

In AIR, data packets would also travel via the route with the highest pheromone value in view of the objective of fully utilizing the optimal route. Thus, once the ants have established the pheromone tracks for the source and destination, subsequent data packets are used to maintain the path.

All pheromone values in the routing tables decrease over time. In this way, the pheromone values also show the utilization rate of a route. When the pheromone entry reaches a minimum threshold, it is considered a stale route and will be discarded from the routing table. The evaporation function is defined as:

$$p_{new} = p_{id.} - \delta p_{id} \qquad \text{iii)}$$

where p_{new} is the updated value and p_{id} is the previous pheromone value. δ is an evaporation scaling factor.

Based on the example from **Error! Reference source not found.**, the routing table of node 0 after route enforcement and pheromone evaporation at τ seconds will look like this:

Table 1: *Routing table of node 0 after route enforcement and evaporation at τ seconds.*

Next Hop	Destination	Pheromone
1	*D*	$P_1 - \delta P_1$
3	*D*	$(P_3 + \varphi P_3) - \delta (P_3 + \varphi P_3)$
5	*D*	$P_5 - \delta P_5$

It is important to ensure that the network does not go into saturation mode where data packets only transmitted over a particular path. Network saturation is a disadvantage for many other ant-based routing because alternate routes will disappear after some time due to evaporation. To avoid network saturation, AIR uses *update ants* to update and refresh the routing tables. Periodically, the source node will send forth update ants but only to active destinations in its routing table (NB: this does not make AIR a proactive routing protocol: it only sends update ants for active destinations, whereas a proactive routing protocol would periodically update routes to all nodes). The operation of update ants is somewhat similar to forward and backward ants in which they explore and quickly reinforce newly discovered paths in the network. It replaces the existing pheromone value with the newly computed value based on the current network condition.

AIR assumes that there is a link-layer failure signal, e.g., from the wireless device driver that decides on link failure based on wireless-technology specific criteria such as a sequence of

missing acknowledgments. Upon reception of a link-layer failure signal, AIR removes the associated routing table entry. If one or more alternate routes to the destination exists, data transmission will shift to the one with the highest pheromone (excluding the failed route). In case there are no alternate routes, the packets will be buffered at the current node. Then, forward ants will be sent forth from the current node in search for the destination that may be located in a new position. Once found, the buffered packets will be delivered instantly along the new route.

Table 2: *Comparison of AIR with existing ant-based routing protocols.*

Criteria	**AntNet**	**ARA**	**ANSI**	**Termite**	**AIR**
Scheme	Proactive	Reactive	Hybrid	Proactive	Reactive
Discovery mechanism	Flood	Flood	Flood	Random	Flood
Loop Prevention	None	Sequence Number	Visited node stack	Message identification	Visited node stack
Ring Search Model	No	No	No	No	Yes
Third-Party Reply Model	No	Yes	No	Yes	Yes
Table-Based	Yes	Yes	Yes	Yes	Yes
Table updating	Unidirectional	Bidirectional	Unidirectional	Unidirectional	Bidirectional
Two Way Route Establishment	No	No	No	No	Yes
Multiple route	Yes	Yes	Yes	Yes	Yes
Optimal path criteria	Time travel	Time travel	Amount of transmission energy	Time travel	Time travel and number of hops
Pheromone Enforcement	By forward and backward ants	By data packets	By forward and backward ants	By data packets	By data packets
Pheromone Evaporation	Yes	Yes	Yes	Yes	Yes
Remove stale routes	Yes	Yes	Yes	Yes	Yes
Network Saturation	Yes	Yes	Yes	Yes	No
Update and Maintenance	Yes. Periodic forward ants	No	No	Yes. Periodic seed packets	Yes. Update ants
Hello mechanism	No	No	No	Yes	No
Route failure Notification	Not define	Link layer	Not define	Missing Hello packets	Link layer
Route failure	None	Backtracking	Local repair Error message	None	Local repair

Any special route error messages are not necessary because a source node would periodically do an update on all its routes. Route update enables a source node to find a new optimal route and purge old stale routes.

Table 2 summarizes the similarities and differences between AIR and other ant-based routing algorithm for MANET. As can be seen in the table, only our algorithm (AIR) implements the ring search model to reduce the flooding of ants throughout the network. Other ant-based approaches use a full flooding of ants whenever a route discovery or periodic update is required. The second innovation introduced by AIR is two-way route establishment. Intermediate nodes visited by the ants can establish routes among themselves as mentioned earlier in the chapter.

For the path selection criteria, we do not merely select the path associated with the first backward ant that arrives from the destination. Instead, we also take into consideration the number of hops the ants have to travel. A shorter length path is desired because it requires less delivery period and less amount of transmission energy to deliver the data packets. Hence, the optimal path is depicted as both the fastest and shortest route.

In addition, AIR avoids the network saturation problem by utilizing update ants to proactively update routes to destination with which active communications are in progress. No error message is required to inform other nodes about link break because the network automatically updates its routing environment.

3. SIMULATION STUDIES ON AIR

~~A.~~3.1 Simulation model

Formatted: Bullets and Numbering

We used ns-2 to study the performance of AIR and compare it with AODV and DSR. The physical radio characteristics of each mobile are based on that of IEEE 802.11 radios using direct sequence spread spectrum. The communication range of each mobile node is about 50 m. For medium access control, the IEEE 802.11 MAC protocol has been used. Each link has bidirectional connectivity. The mobility model used for node movement is the Random Waypoint Model [10].

Table 3: *Simulation parameters.*

Parameter	Value
Area size	1000 m × 1000 m
Number of nodes	Variable (10, 30, 50)
Node speed	Variable (up to 60 m/s)
Pause time	5 s
Simulation time	100 s
Data rate	4 packets per second
Packet size	512
Buffer size	50 packets
Communication range	50 m

Traffic sources are from Constant Bit Rate (CBR) generators using User Datagram Protocol (UDP) to transfer data packets. The data traffic rate is set at 2 kbps, where each source sends four 512-byte packets per second. Each node has a queue for packets awaiting transmission by the network interface that holds up to 50 packets and managed in a drop-tail fashion. The drop-

tail fashion maintains exactly one FIFO queue. Packets sent by a routing layer are queued at the interface queue until the MAC layer can transmit them. Routing packets has the same priority as data packets in order that accurate information about the network condition can be collected and disseminated by the ants.

The performance metrics used in the analysis and evaluation phase are:

- Throughput, *T*
- Packet delivery ratio, PDR
- Average End-to-End Delay, *D*
- Routing Overhead, *R*

The performance metrics are defined as follows:

$$T = \frac{\sum_i p_i(t)}{t} \quad \text{iv)}$$

where *T* is throughput, *t* is the time variable, and $p_i(t)$ is number of packets arriving at node *i* between time 0 and *t*, and the summation is over all the nodes in the network

$$\text{PDR} = \frac{\sum_i p_i(t)}{\sum_j p'_j(t)} \text{ x } 100\% \quad \text{v)}$$

where $p_i(t)$ is as before, and $\sum_j p'_j(t)$ is number of packets *sent* from node *j* between time 0 and *t*, and the summation is also over all the nodes in the network

$$D = \frac{\sum_j \sum_n L_j(n)}{\sum_i p_i(t)} \quad \text{vi)}$$

where $p_i(t)$ is as before, and $L_j(n)$ is the "lifetime" of the n^{th} packet arriving at node *j*, the lifetime of a packet being the difference between the time when the packet is received at its destination and the time when the packet was created at its source (thus, only successfully received packets are included in the summation).

$$R = \sum_j \sum_n P_j(n) \quad \text{vii)}$$

where $P_j(n)$ is the n^{th} overhead packet sent by node *j* and the summations are over all overhead packets in all the nodes.

3.2 Simulation results

3.2.1 Throughput

As illustrated in Fig. 1, the throughput comparison shows that the three algorithms have similar throughput performance. However, it clearly shows that the throughput for AIR in all scenarios is higher than for AODV and DSR. All three algorithms reflect the degradation of throughput as mobility increases. Similar results were observed for other network sizes (10 and 30 nodes).

Because of their close relationship, we will discuss the throughput and PDR performance together, in Section 0 next.

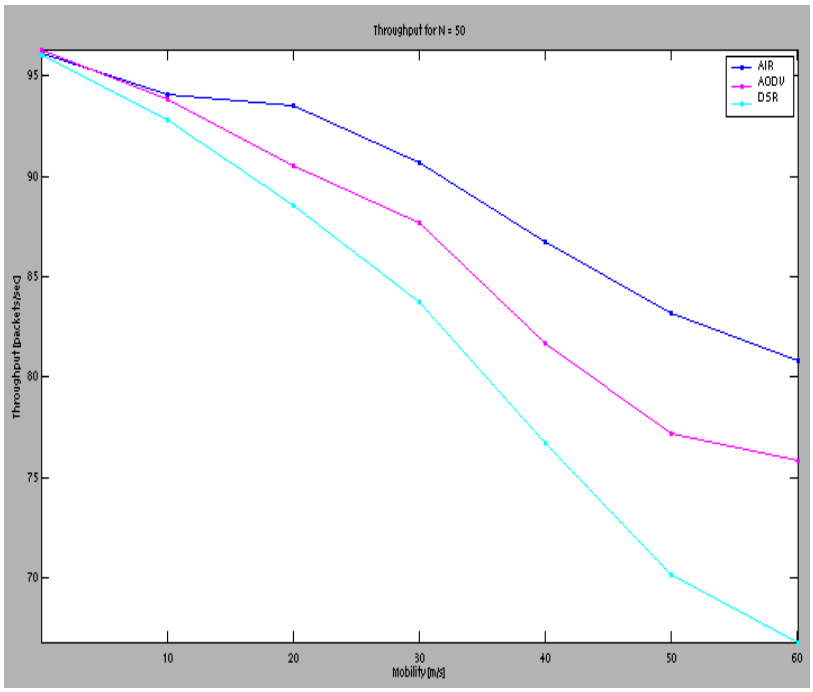

Fig. 1: *Throughput of AIR, AODV and DSR for N = 50.*

3.2.2 Packet delivery ratio

The destination records the number of data packets it received and estimates the PDR from the count of the data packets sent. Fig. 2 validates that AIR has a better PDR than AODV or DSR even in cases of high mobility. When the nodes are static, all three algorithms show a 100% success rate for packet delivery. However, the PDR for AODV drops faster than for AIR when mobility increases. DSR performs even worse.

Overall, the data throughput and packet delivery ratio declines as node mobility increases for all three algorithms in different network size scenarios. In higher mobility, network topology changes rapidly and link failures happen more frequently. Nevertheless, AIR shows that the decrease in data throughput and packet delivery ratio is less compared to AODV and DSR.

AIR minimizes the rate of dropped packets through an effective route failure handling mechanism. When packets failed to be delivered due to a broken link, traffic will quickly switch to an alternate routeFor AIR, the source periodically does an update on all its routes. This is to ensure that the traffic can quickly converge to the new optimal path depending on the network condition. AIR enables all data packets to be delivered even in cases of broken links. This results in a higher level of throughput and packet delivery ratio.

On the other hand, AODV and DSR have different route failure mechanisms. In AODV, RERRs (error messages) would be propagated throughout the network in order that all nodes using the

link will be notified about the failure and do the necessary changes on its routing table. This propagation is assumed to reach the source node to initiate a new route discovery.

DSR performs badly in high mobility because unsuccessful packets are not kept at the node. They are dropped immediately. For route repair, the node will send a route error message in a unicast fashion back to the source. When the source and the upstream nodes receive the error message, they will erase all the paths that use the broken link from their route cache. However, nodes that are not on the upstream route but use the failed link are not notified promptly, thus packets of other routes using the particular broken link will be dropped.

For AODV and DSR, once RERR reaches the source, the source node would have to initiate another route discovery. Due to late notification to the source, the source will keep on sending data packets unaware of the failure and these packets mostly ends up being dropped.

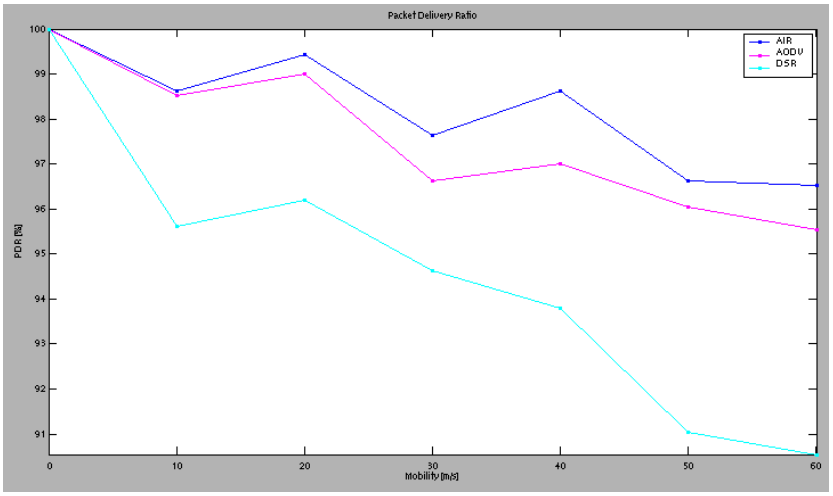

Fig. 2: *Packet Delivery Ratio for different mobility rates.*

3.2.3 Average end-to-end delay

The average end-to-end delay evaluates the efficiency of the routing methods. It includes buffering delay during route discovery, queuing delay at interface queue, propagation and transfer time delay. Low end-to-end delay is very important for certain applications like push-to-talk where users do not wish to wait any significant length of time before communications are established. Comparing the average end-to-end delay, as shown in Fig. 3, AIR generally has a lower end-to-end delay than AODV or DSR. Even in high mobility scenarios, the end-to-end delay for AIR remains at a stable level. AODV, on the other hand shows the highest delay among the three, followed by DSR. The delay increases drastically when mobility increases. Similar results were observed for other network sizes (N = 10 and N = 30).

Delays are mostly from the route discovery phase because data packets need to wait at the source buffer until a new route is discovered. In case of high mobility, AODV shows a tremendous rise of end-to-end delay due to the frequent route discovery. This is because in AODV each node only maintains a single route to a destination, so a new route discovery is needed whenever there is a broken link. On the other hand, AIR and DSR utilize alternate

routes, so a source does not need to initiate a new discovery process whenever the primary route becomes unavailable. Furthermore, with the two way route establishment model, AIR generally establishes more paths per route discovery cycle (forward and backward ant).

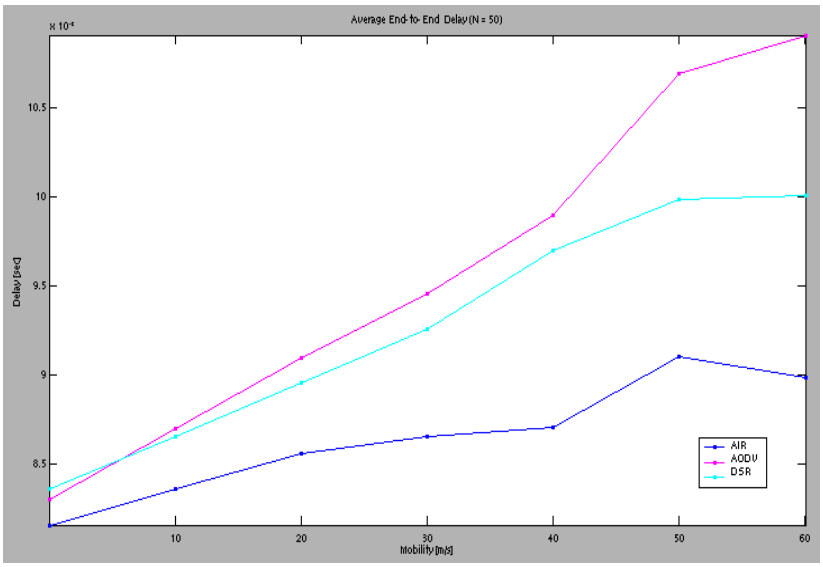

Fig. 3: *Average end-to-end delay for N = 50.*

Moreover, AIR has a lower delay than AODV and DSR because it prefers low-delay paths with low hop-counts for data traffic connectivity over merely low hop-count paths. The pheromone level, calculated from both the hop count and travel time of the ants, reflects accurately the length, the traffic conditions and the validity of the route. On the other hand, AODV and DSR are biased towards routes with a smaller hop count.

In AIR, even if the source does not have a ready route to the destination, the route discovery phase is much shorter. Nearby nodes that knows a path to the same destination can generate backward ants back to the source. The probability of it receiving backward ants quickly from nearby nodes is high after a short time period in the simulation.

Lastly, the dynamic nature in which routes are kept updated by the ants lead to traffic switching from a longer route to a newer shorter ones, thus, further reducing the end-to-end delay for active routes.

3.2.4 Routing overhead

Routing overhead is another important consideration when choosing a routing algorithm. Routing algorithms that require excessive overhead communication can result in significantly reduced throughput and increased delays. In calculating the routing overhead, we consider the following type of routing packets for each algorithm:

Routing overhead for AIR (bytes) = forward ants + backward ants + update ants

Routing overhead for AODV (bytes) = RREQ + RREP + RRER

Routing overhead for DSR (bytes) = RREQ + RREP + RRER

In Fig. 4, the observed routing overhead for AIR, AODV and DSR are reported for different network sizes and mobility. From the figure, we see that the overhead for all three algorithms increases as the network becomes more mobile. AIR demonstrates a higher routing overhead compares to AODV and DSR in all cases. However, we see that the routing overhead for AIR method increases gradually whereas AODV and DSR display a faster growth as the mobility increases.

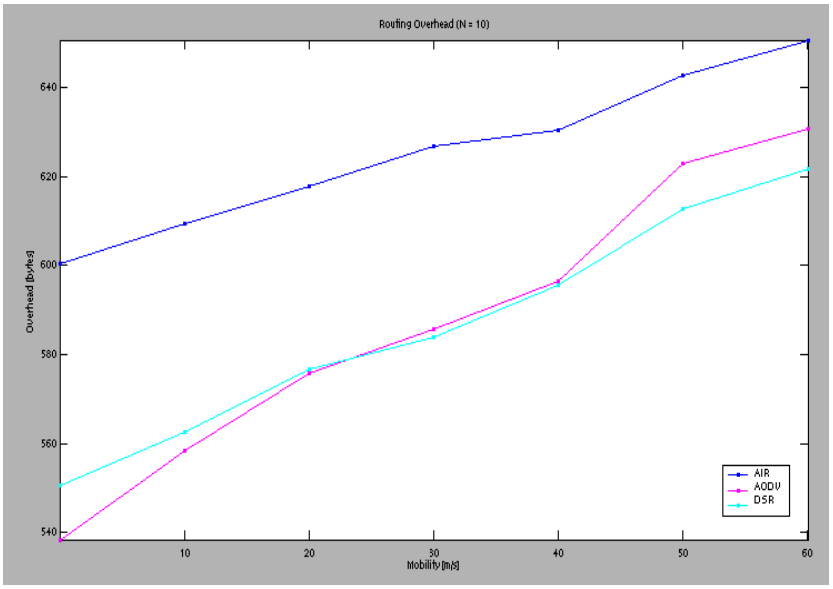

Fig. 4: *Routing overhead.*

The above observations can be explained as follows. The major contribution to AIR routing overhead is the update ants that are periodically sent to update or refresh the active routes in the network. The benefits of the update ants come at the expense of a higher overhead.

For low traffic conditions, AODV performs well because it finds routes only when old route breaks and do not induce any other routing overhead. In addition, nodes in AODV do not accept multiple RREQS for the same destination. The routing table maintains at most one entry per destination. This reduces the flooding of RREQs and RREPs in the network. In contrast AIR and DSR accepts more than one route requests or forward ants so that multiple routes can be established.

Nevertheless, AODV shows a high increase of overhead for higher mobility. This is due to the flooding of route error messages whenever a broken link occurs. In cases of route failure, AIR does not flood the network with any route error message. Also, by virtue of its use of multiple paths and two-way route establishment, AIR resorts to route discovery less frequently than AODV and DSR.

Note that in calculating the routing overhead for AODV, we only consider RREQ, RREP and RERR packets. Hello packets in AODV would also take a considerable overhead and energy for all the nodes.

4. SUMMARY AND CONCLUSIONS

In this paper, we have proposed a new ant-based routing protocol for mobile ad hoc networks. Our new protocol, AIR, combines the best features of earlier ant-based routing protocols with new features, such as update ants, two-way route establishment, expanding ring search model and third party reply model, some of which were borrowed from the latest versions of traditional (not ant-based) routing protocols. The goal was to improve performance and scalability, to improve throughput and packet delivery ratio, and especially to reduce end-to-end delay, without needing to increase overhead levels by too much. We have found that AIR delivers improved throughput and packet delivery ratios, and is especially strong in providing low end-to-end delays even in large networks with high mobility rates. The main drawback of AIR compared with AODV and DSR is that it has significantly more overhead, particularly when mobility rates are low. However, as the gap closes at higher mobility rates, AIR should be particularly useful to provide good throughput and packet delivery ratios in ad hoc networks with high mobility where low end-to-end delay is critical.

REFERENCES

1. Perkins, C., Belding-Royer, C., and Das, S. (2003), Ad hoc On-Demand Distance Vector (AODV) Routing, IETF RFC, 3561, July.
2. Johnson, D., Maltz, D., and Hu, Y.-C. (2004), The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR), IETF work in progress, July.
3. Di Caro, G. and Dorigo, M. (1998), AntNet: Distributed Sigmergetic Control for Communications Networks, Journal of Artificial Intelligence Research, pp. 317-365.
4. Lee, S.J., Belding-Royer, E., and Perkins, C. (2003), Scalability Study of the Ad hoc On-Demand Distance Vector Routing Protocol, ACM/Wiley International Journal of Network Management, Vol. 13, No. 2, pp. 97-114, March.
5. Caro, G.D. and Dorigo, M. (1998), Ant Colonies for Adaptive Routing in Packet-Switched Communications Networks", 5th International Conference Parallel Problem Solving from Nature, Amsterdam, Holland, September.
6. Gunes, M., Sorges, U., and Bouazizi, I. (2002), ARA-The Ant-Colony Based Routing Algorithm for MANETs, Int'l Workshop on Ad Hoc Networking (IWAHN), Vancouver, Canada, August.
7. Rajagopalan, S., Jaikaeo, C., and Shen, C.C. (2003), Unicast Routing for Mobile Ad Hoc Networks with Swarm Intelligence, Technical Report #2003-07, University of Delaware, July.
8. Roth, M. and Wicker, S. (2003), Termite: Ad Hoc Networking with Stigmergy", IEEE Globecom 2003, San Francisco, USA, December.
9. Beckers, R., Deneubourg, J.L., and Goss, S. (1992), Trails and U-turns in the selection of the shortest path by the ant Lasius niger, Journal of Theoretical Biology, pp. 397-415.
10. Bettstetter, C., Restar, G., and Santi, P. (2003), The Node Distribution of the Random Waypoint Mobility Model for Wireless Ad hoc Networks, IEEE Transactions on Mobile Computing, Vol. 2 No. 3, pp. 257-269, July-September.
11. Stockburger, D. (1996), Introductory Statistics: Concepts, Models and Applications, RockHill Press.