

ANSI: A Unicast Routing Protocol for Mobile Ad hoc Networks Using Swarm Intelligence

Sundaram Rajagopalan Chien-Chung Shen
Department of Computer and Information Sciences

University of Delaware
Newark, DE 19716

Tel: (302) 831-1951, Fax: (302) 831-8458
{rajagopa, cshen}@cis.udel.edu

Abstract— We present a reactive routing protocol for mobile ad hoc networks which uses the mechanisms of Swarm Intelligence to select next hops. Our protocol, Ad hoc Networking with Swarm Intelligence (ANSI), is a congestion-aware routing protocol, which, owing to the self-configuring mechanisms of Swarm Intelligence, is able to collect more information about the local network and make more effective routing decisions than traditional MANET protocols. Once routes are found, ANSI maintains routes along a path from source to destination effectively, without high MAC layer overheads, by using Swarm Intelligence techniques, and is able to gauge the slow deterioration of a link and restore a path along newer links as and when necessary. ANSI is thus more responsive to topological fluctuations. In addition, owing to ANSI's lower delay jitter characteristics at high loads, ANSI is inherently more suited for certain types of multimedia applications over mobile ad hoc networks.

Our simulation study compared ANSI with AODV and the results show that ANSI is able to achieve better or comparable results at lower loads and better results at higher traffic loads as compared to AODV. In addition, ANSI achieves this performance with comparable or better delay and jitter characteristics along with significantly lower consumption of MAC layer resources. Lastly, ANSI is able to perform more consistently, considering the lower variation (measured as the width of the confidence intervals) of the observed values in the results of the experiments.

Index Terms— MANET, swarm intelligence, unicast routing, congestion-aware routing

1. Introduction

Mobile ad hoc networks consist of a group of mobile nodes which autonomously establish connectivity via multi-hop wireless communications. Without relying on any existing, pre-configured network infrastructure or centralized control, they are useful in many situations where impromptu communication facilities are required such as battlefield communications and disaster relief missions.

A number of ad hoc routing protocols have been proposed, for example, [1], [2], [3], [4], [5]. In *proactive* protocols such as [5], nodes in the network maintain routing information to all other nodes in the network by periodically exchanging routing information. Nodes using *reactive* protocols, such as [1], [2], delay the route acquisition until a demand for a route is made. *Hybrid* protocols, like [4], [6], use a combination of both proactive and reactive activities to gather routes to the destinations in a network – nodes using ZRP, for example, proactively collect routes in their zone, and other routes are collected reactively. In

[6], on the other hand, the level of proactive activity and reactive activity are chosen autonomously by the nodes in the network, and proactive activity is only seen around favorite destination nodes. In most traditional reactive protocols, like [1], [2], only when a route breaks irreparably does the protocol mechanisms repair the damage. In reality, route deterioration is most often not sudden but gradual, and most often available routes get better/deteriorate gradually and not suddenly. So the routing protocol should continuously maintain information about the nodes in the local area to perform effectively and avoid too many link breakages.

In this paper, we present a routing suite for mobile ad hoc networks which uses the mechanisms of Swarm Intelligence [7] to select good routes to destinations. We use Swarm Intelligence (SI) because SI mechanisms allow for self-configuring systems and maintain state information about the neighboring network better than traditional MANET routing mechanisms. The combination of the SI mechanisms allow a node to change routing information quickly and efficiently to adjust to an ever-changing local topology and route deterioration, initiating fewer link breakages and incurring lower MAC layer overheads.

Our protocol, ANSI, uses a highly flexible costing function which allows it to use the information collected from the local ant activity, such as the congestion status of the neighboring nodes, in useful ways. In addition, the ant-like working of our protocol allows the maintenance of multiple routes to a destination, and when one route fails, others may be used. Our motivation comes from the fact that different networks face different conditions, and thus a protocol suite should allow for various configurations as the network conditions dictate. Furthermore, supporting multiple routes simultaneously is essential to ensure *survivability* of the network [8]. ANSI facilitates ad hoc unicast routing by exploiting route finding behaviors that are *emergent* from ant packets working collectively, rather than explicitly coding them to cope with the problem. We formulate the routing problem at node i as a set of “food foraging” problems from nest i , where each “food source” is a destination d in the network. In this formulation, next hops are evaluated on the basis of the strength of the *pheromone trail*¹ on the link connecting a node and a next hop.

The remainder of this paper is organized as follows – In the next section, we discuss a number of approaches and protocols which are related to our research. In Section 3, we describe in detail the components of ANSI unicast routing protocol, and follow it with a brief specifics of our implementation in Section 4.1 and the simulation results and comparison with AODV in Section 4.2. We conclude in Section 5 with a brief overview of our future research effort.

This work is supported in part by National Science Foundation under grant ANI-0240398.

¹The computational equivalent of the chemical deposited on the forest floor in the study of ants.

2. Related work

The main ingredients of SI are positive/negative reinforcement, amplification of fluctuations, achieved in an environment with multiple interaction among nodes [7]. The benefits of using SI-based algorithms are not fully accrued if the any of the above aspects are not present in the protocol. This is because in any SI-based system, these aspects work together in learning about the network.

In [9] Baras et. al. describe a swarm intelligence based reactive ad hoc routing protocol called PERA. PERA uses broadcast *forward ants* as exploratory agents sent out on demand to find new routes to destinations. Each ant holds a list of nodes that were visited while exploring the network, and since these ants are broadcast at each node, a forward ant can result in several *backward ants* – ants sent by destination nodes in response to forward ants. This uncovers several routes for each forward ant sent, and at each node these multiple routes found to the destinations are maintained as probability values. As with AntNet [10], the routing table R_i at node i is a probability matrix with a probability entry P_{ijd} as the probability that a data packet at i 's FIFO queue will take the next hop j to be routed to d . Positive reinforcement is managed in PERA using forward/backward ants and negative reinforcement is implicit – no explicit aging of the pheromone trails is done. After a route has been established, PERA regularly uses forward ants to find newer routes to destinations. This is wasteful, considering the fact that forward ants cause a lot of network resources to be consumed and should not be sent when not necessary.

In [11], Camara et al. outline a source routing scheme in which the network relies on location information and support from fixed infrastructure. Owing to a source routing approach, the algorithm relies heavily on a source–destination route which is available at the time of message creation. New nodes in the network start with using their neighbor's routing table. The routing table, generated using shortest path algorithms, on the other hand, may contain information which is outdated. Ants, unicast from a source to specific destinations, for example, the node with the oldest information in the routing table, are used to make sure that the routing information in the source is updated and recent. Thereby, ants are used in [11] with the semantics of routing information updates, like classical distance vector protocols such as DSDV or DBF – ants are not used as feedback agents to reinforce routes positively (in the case when a route is still good), negatively (when a route is no longer good) or explore new routes randomly – ants in this approach are unicast to specified direction, not allowing for amplification of fluctuations, and depending on known metrics such as timestamp of a route in the routing table.

The approach used in [12] by Heissenbttel et al. also relies on location information, and is a purely proactive routing approach based on dividing the network into logical zones and assigning logical routers to each. Ants – forward ants and backward ants – are used by logical routers in this approach to periodically check if the logical links connecting it to a randomly chosen destination are functional and reflect on the current state of the network surrounding the logical router. Positive and negative reinforcement are achieved by means of multiple interactions and pheromone additions (by forward and backward ants) and pheromone aging respectively. Random amplification of a new good route in the face of topological fluctuations is possible by random dissemination of ants to destinations.

In [13], Güneş et al. outline ARA, a multipath, purely reactive scheme. ARA uses forward ants and backward ants to create fresh routes from a node to a destination. When routes to a destination D are not known at S , a forward ant is broadcast, taking care to avoid loops and duplicate ants. When a forward ant is received at an intermediate node X via node Y , the ant reinforces the link XY in X to route to all the nodes covered

so far by the forward ant. When a forward ant is received at D , a backward ant is created which backtracks the path of the corresponding forward ant. At each node the backward ant is received, the link via which the backward ant is received is reinforced, like the forward ant does, for all nodes which have been visited by the backward ant. In ARA, data packets perform the necessary (positive) reinforcement required to maintain routes. When a path is not taken, it subsequently evaporates (negative reinforcement) and cannot be taken by subsequent data packets. Under the described scheme, amplification of topological and network fluctuations is not possible except under extreme conditions when routes break often.

In [14], [15], Di Caro, Ducattelle, and Gambardella describe AntHocNet, a hybrid, stochastic approach to the routing problem in MANET. AntHocNet is a congestion-aware protocol which only finds routes on-demand, but once a route is established, it is proactively maintained. We feel that this approach, argued by the authors to be more ant-like [15] than other competing ant-based protocols, will fail to reduce overheads in very high traffic scenarios (such as our experiment scenarios), owing to the rate at which proactive ants are potentially unicast. However, we do agree with the comment that the authors make regarding the repeated path sampling, and ANSI manages to steer clear from repeated path sampling by carefully choosing when to engage in route discovery activity.

Some characteristics seen in traditional on-demand routing protocols can be seen in ANSI. For example, an optimization used in AODV, expanding ring search, is also used in ANSI, albeit used more efficiently, owing to the use of history information.

3. ANSI unicast routing protocol

1. Protocol overview

The outline of the process of ANSI routing is as follows:

- 1) When a route to a destination D is required, but not known at S , S broadcasts a *forward reactive ant* to discover a route to D .
- 2) When D receives the forward reactive ant from S , it source-routes a *backward reactive ant* to the source S . The backward reactive ant updates the routing table of all the nodes in the path from S to D , allowing for data transfer from S to D .
- 3) When a route fails at an intermediate node X , X first checks if there are other routes which can be used to route the packet to D . If not, then ANSI buffers the packets which could not be routed and initiates a route discovery to find D by using a *forward reactive ant* to perform local route repair. Additionally, X sends a route error message back to the source node S .
- 4) When a route at D is known at S , ANSI *deterministically* chooses the best next hop to reach the destination.

As we can see, the basic working of ANSI, as shown above, is very similar to most traditional reactive MANET routing protocols, like AODV [1]. The better performance of ANSI, however, lies in the working of the SI mechanisms at each node, which maintain routing information and local information more effectively than traditional reactive MANET routing protocols. In addition, the congestion-awareness of ANSI also helps in controlling the extent of congestion in high traffic scenarios. In Section 3.3.1, 3.3.2 and 3.3.3, we explain the details of the above actions, and show how the SI mechanisms work at each node in maintaining routing information.

2. Protocol model

1) *Data structures:* Data structure (1) below is the ant structure, carried by all ants, and data structures (2) and (3)

below are maintained at each node, and are updated every time an ant arrives at the node.

(1) **Ant structure:** The following information is carried by an ant π :

- a) The ant ID of the ant, which is the (*node ID, sequence number*) pair,
- b) The number of nodes, m , which π visits, including the node π originated from,
- c) The nodes visited stack (adapted from [10]), \mathcal{S}_π , containing information about nodes $V = \{v_1, v_2, \dots, v_m\}$, that can be reached by backtracking the ant π 's movement (using the nodes visited stack), and
- d) Pheromone amount at $v \in V$, p_v .

(2) **Ant decision table at node i , \mathcal{A}_i :** (adapted from [16]) An ant decision table is a data structure that stores pheromone trail information for routing from node i to a destination d via k possible next hop nodes $J = \{j_1, j_2, \dots, j_k\}$. The link ij in the ANSI network, between two nodes i and j are assumed to be bidirectional. Routing tables are computed from ant decision tables. Each ant decision table entry \mathcal{A}_{ijd} for node i maintains a row for the destination-next hop pair (d, j) along with the $\tau_{ijd}(t)$, η_{ijd} , ψ_{ijd} and a_{ijd} values described below:

- a) $\tau_{ijd}(t)$ is the pheromone trail concentration left on a trail ij used as a first hop to destination d at current time t due to all the ants that have traversed the trail, taking into consideration the pheromone evaporation (see Equation 4). τ is thus a *weighted* measure of how many times the trail ij was traversed by packets intended to d and is thereby a measure of the *goodness* of trail ij .
- b) η_{ijd} is the heuristic value of going from j to i . In our mapping, η is a measure of the distance to the destination i from d .
- c) $\psi_{ijd} \in [0, 1]$ is the value of the congestion status at node j . If $\psi_{ijd} = 1$, then, node j is considered not congested, and if $\psi_{ijd} = 0$, then the node j is considered congested. The value of ψ at a node j is measured as the ratio of empty space in packets in the IP queue size to the number of packets already in the IP queue at j .
- d) The a_{ijd} value for a destination d is computed using the following formula:

$$a_{ijd}(t) = \frac{[\tau_{ijd}(t)]^\alpha [\eta_{ijd}]^\beta [\psi_{ijd}]^\gamma}{\sum_{l \in J} [\tau_{ild}(t)]^\alpha [\eta_{ild}]^\beta [\psi_{ild}]^\gamma} \quad (1)$$

where α , β and γ are chosen appropriately (see Section 4.2). The above formula was adapted from Ant Colony Optimization techniques outlined in [16]. The intuition behind this equation is that we want to use the metrics of hop distance and path goodness and allow some flexibility as to how much we rely on either metric by varying α , β and γ values.

As soon as an ant π is received at a node i via neighbor node j , i has the information about j 's congestion status from \mathcal{S}_π . The pheromone τ_{ijv}^π deposited by an ant π and the heuristic η_{ijv} to a destination v in the ant π traversing from node j to node i via nodes $v \in V$ are given by the equations:

$$\tau_{ijv}^\pi = \frac{1}{p_j - p_i} \quad (v, i, j \in V) \quad (2)$$

and

$$\eta_{ijv}^\pi = \frac{1}{\text{depth}(v)} \quad (v, i, j \in V) \quad (3)$$

where p_i and p_j are the pheromone amounts of ant π at nodes i and j , respectively, and $V = \{v_1, v_2, \dots, v_m\}$ denotes the set of m nodes visited by π . The value $\text{depth}(v)$ is the depth of the node v in π 's nodes visited stack.

All τ values in \mathcal{A}_i are evaporated according to Equation 5 each time another ant, π' , visits node i . Let us say π' traverses the same trail ij at time $(t + \Delta)$ as traversed by π at time t . π' then *positively reinforces* the trail $ij \forall v \in V$ in \mathcal{S}_π . All other trails iJ' , (where J' is the set of all possible next hops from i except j) in the ant decision table \mathcal{A}_i are not positively reinforced, and in the event no ant traverses through any of the other trails iJ' , the trails iJ' eventually become invalidated (*negatively reinforced*) owing to pheromone evaporation. The new τ_{ijv} at time $(t + \Delta)$ is calculated as follows:

$$\tau_{ijv}(t + \Delta) = \text{evaporate}(\tau_{ijv}(t), \Delta) + \tau_{ijv}^{\pi'} \quad (4)$$

where $\tau_{ijv}^{\pi'}$ is the pheromone deposited on the trail by π' over ij (see Equation 2). The function $\text{evaporate}(\tau_{ijv}(t), \Delta)$ returns the pheromone amount left on trail ij for destination v (after evaporation) due to the ants which traversed ij before π' . The pheromone evaporation model used to calculate how much of the earlier pheromone trail, $\tau_{ijv}(t)$ is left behind at $(t + \Delta)$ when π' traverses the trail ij is as follows:

$$\text{evaporate}(\tau_{ijv}(t), \Delta) = \frac{\tau_{ijv}(t)}{2^{\Delta/c}} \quad (5)$$

where c is an arbitrary constant.

After all the τ values in the ant decision table are evaporated (including the $\tau_{iJ'V'}$ values on the trails iJ' that were negatively reinforced, i.e., no ants that traveled $V' = \{v'_1, v'_2, \dots, v'_m\}$ were received) and recalculated, the a_{ijv} values for all entries V in \mathcal{S}_π are recomputed and the new best next hops to destinations V are computed again. This is followed by an update of the routing table at node i . Negative reinforcement of routes also happens when a route is explicitly invalidated by a route error message.

(3) **Routing Table:** The routing table at node i is a table containing an entry for each destination d reachable from node i along with the *best next hop*, \mathcal{J}_i^d , to d . The best next hop, \mathcal{J}_i^d , to a destination d is the next hop that contains the largest a_{ijd} value in \mathcal{A}_i . The value of \mathcal{J}_i^d is thereby updated every time an ant visits a node i . The routing table also contains the distance of d from i in hops, and this information is used to set the number of hops for route discovery when the routing table entry to d in i becomes defunct.

2) **Amplification of fluctuations:** The process of broadcasting ants during reactive route discovery finds new routes to nodes and alters the information in the ant decision table accordingly. Because of the nature of broadcast in the wireless medium, the routes found as a result of forward reactive ant activity reflect the current status of the network and accordingly amplify the current fluctuations in the topology. Another mechanism amplifies the fluctuations in the local area: when a node receives a unicast packet, it notes the neighbor node ID and reinforces the path to the source of the packet via the neighbor. In addition, when a data packet is sent along a next hop, the node reinforces the next hop as a valid next hop to the destination. This mechanism also amplifies local fluctuations of network and topological characteristics and see to it that the nodes in the ANSI network use up-to-date network and topological information.

Some protocols, for e.g., [9], using SI mechanisms for MANET argue for unicasting forward reactive ants along one randomly chosen path to the source and destination to amplify the fluctuations in the network. Yet others, for e.g., [15], argue for sending forward ants at regular intervals from the source while the source is sending packets to the destination. We feel that the above methods will work in low traffic scenarios and in wired networks, where there is little or no mobility, but not in highly mobile MANET with high loads. Besides, we feel that the right model for amplification of fluctuations in a MANET using SI mechanisms is the model we use: that of broadcasting forward ants *only* when absolutely needed, both at the source and intermediate nodes (to perform local repair), and using these mechanisms with a neighbor discovery mechanism, and applying the rules of SI on the data collected (viz., Equations 1 – 5). By using the SI mechanisms appropriately in ANSI, we are able to reduce the number of MAC layer resources used, as well as be responsive to a network with high traffic rates, along with better packet delivery rates and lower delay jitter characteristics.

3. Protocol description

A trail ij to destination d , τ_{ijd} , is positively reinforced in ANSI when (a) a new route to a destination d is found (via ant activity) at i via next hop (neighbor node) j , (b) when i uses an already known nexthop node j again to route a packet to d . A trail ij is negatively reinforced when (a) the trail ij to destination d is subjected to evaporation (as per Equation 5), and (b) when next hop node j to d is no longer available (owing to MAC layer errors, route errors, or congestion at j). In the following sections, we describe the various reinforcement mechanisms at work in ANSI.

1) *Local route management – reinforcement by data packets and the use neighbor discovery HELLO messages.*: When a data packet arrives at a node i via a neighbor node p and is sent to the destination along next hop j , both the trail to the previous hop, ip , and the trail to the next hop, ij , are reinforced by the SI mechanisms at i .

In addition, nodes in ANSI periodically broadcast a HELLO message. This message can contain a variety of information about the node sending the message, such as congestion status. In ANSI, hello packets are used to perform local route management by positively reinforcing previously known neighbors and new neighbors. The advantage of using this mechanism can be explained as follows: If a direct route to a destination d is known at i via this process, then a previously known indirect route to d is less favored than the direct route by the reinforcement mechanisms in ANSI.

2) *Non-local route management and explicit positive reinforcement*: Reactive route discovery is performed by *forward reactive ants*, π_f , and *backward reactive ants*, π_b . Reactive route discovery can be used both at the source of a data packet and at an intermediate node looking for an alternate route to the destination in the event that previously known routes to the destination have proved ineffective. A route request is sent by deploying a forward reactive ant π_f and the route reply is sent using a backward reactive ant, π_b . Even though multiple routes can be gathered by a source sending forward reactive ants (by allowing the destination to send backward reactive ants to all copies of the forward reactive ants sent out), we allow the destination to send a backward reactive ant only for the first forward reactive ant received. This is because we found that in a network with too many routes to the destination at the source, the packet delivery can suffer invariably owing to favoring load balancing over quick data delivery. Regardless, multiple routes are collected owing to the interaction of the ant information from the nodes in the network and HELLO beacons, and are

used as and when older routes become defunct. Also, note that regardless of collecting information about multiple routes via other mechanisms, ANSI is a *deterministic* protocol. This is because we found that stochastic approaches in ANSI are not suited to high data delivery in high traffic scenarios.

Consider a node S which needs to route data packets to D , but does not have a route to D . Node S buffers the data destined for D and broadcasts a forward reactive ant, π_f^{SD} (with a nodes visited stack \mathcal{S}_f), intended to discover the route to D . Because there is a good chance that D is not in the local area of S , the current implementation of ANSI sets the number of hops, ϕ_f , for the forward reactive ant to be a few hops larger than the last known distance of i from d , which can be obtained from the routing table at S . If S receives data intended to D after π_f^{SD} has been broadcast, S buffers the data. When D receives π_f^{SD} , D copies the nodes visited stack, \mathcal{S}_f , into a new backward reactive ant, π_b^{DS} , and kills π_f^{SD} . D then sends π_b^{DS} to S . π_b^{DS} is not broadcast, and it just backtracks the path back to the source S by using the nodes visited stack \mathcal{S}_b in π_b^{DS} . The ant, π_b^{DS} , when visiting a node i along the path to S *positively reinforces* the route to all nodes $v \in \mathcal{S}_b$ upstream from i to D , and adds an entry in \mathcal{A}_i to D via the next hop immediately upstream (in the path from i to D). An intermediate node thereby knows what next hop to use to route to D . In this way, backward reactive ants perform explicit positive reinforcement of routes to destination D . When S receives π_b^{DS} from D , S sends the buffered packets intended for D over the newly discovered route and flushes S 's buffer.

In the event that π_b^{DS} is not received at S within a timeout period, then the value of ϕ_f is increased by 2 more hops and the search for the route resumes again. The process of route discovery is continued again if a route is not found after the second try. ANSI retries twice for a route to destination.

To control the amount of MAC layer usage at a node i , a HELLO packet is broadcast at i only if the last broadcast forward reactive ant was sent before the last HELLO message.

3) *Route errors, and negative reinforcement*: Route errors occur at a node i when i is unable to provide a route for the destination d owing to non-availability of a routing table entry at i or due to the non-availability of the next hop suggested by the routing table entry at i . When a route error occurs at a node i in a network running ANSI, i first buffers the packet which i needs to forward and then sends a forward reactive ant to find the destination d . If i happens to be an intermediate node X , in addition to sending a forward reactive ant, X also sends a route error back to the source S of the packet. The packets buffered at i are relayed across the network after a backward reactive ant from d reaches i .

In addition, when a route error is received at an intermediate node between X and S , the node explicitly invalidates the routing table entries to d . The packets received at X before the route error is received at S are X 's responsibility (to forward), but the packets generated after the time when the route error is received at S from X are S 's responsibility – S generates a forward reactive ant to find the route to d .

4) *Driving the routing process via more desirable nodes*: By choosing higher values for α , β and γ , the process of next hop selection in ANSI favors the next hops with higher values for τ , η and ψ respectively. However, by choosing values which are too high, the route selection is too skewed towards the best next hops and it becomes very difficult for the SI mechanisms at the nodes to respond quickly to the changes in the network. Hence, a choice for α , β and γ should be made carefully to allow for responsiveness of the system. From our preliminary results, we arrived at a value of $\alpha = \beta = \gamma = 2$, and these are the values we use in our implementation.

4. Simulation results

ANSI was simulated in QualNet (Version 3.7), and the performance of ANSI was compared with a popular routing protocol, AODV [1], for the same network and load characteristics and the same version of QualNet. We chose to compare our protocol with AODV because AODV has been shown to perform well in a vast majority of ad hoc network scenarios.

1. Simulation and network model

1) *ANSI Parameters:* The current implementation of ANSI used $\alpha = \beta = \gamma = 2$. In both AODV and ANSI, the reactive route recovery is retried twice, and for ANSI, the last try uses $\phi_f = 15$. For the first two trials in ANSI, ϕ_f is determined according to the information available about the unknown destination: if the destination had a valid entry in the routing table earlier, then ϕ_f is set to one more than the earlier number of hops to the destination. Otherwise, $\phi_f = 5$. The evaporation constant, c , used in Equation 5 is 15 s.

2) *Network and application parameters:* We performed two experiments, in which we studied the performance of ANSI and AODV with increasing traffic. In Experiment 1, we varied the number of source-destination pairs and in Experiment 2, we varied the packet rate for a fixed number of source-destination pairs. In both experiments, 50 nodes were uniformly distributed initially in a terrain of $600m \times 600m$ and moved as per the Random Waypoint Model with a minimum speed of $0.001 m/s$, maximum speed of $20m/s$, with a pause time of 10s. The nodes in the network used 802.11 with omnidirectional antennas and a transmission range of 250 m at the physical layer and 802.11DCF at the MAC layer. The link bandwidth was 2 Mbps. The simulations used a two-ray pathloss model and no propagation fading model was assumed. The application used was CBR, and both source and destination were pairwise distinct and chosen randomly. All experiments were run for a simulated time of 5 min and all sources started sending packets at exactly 40s into the simulation and ended data generation at exactly 260s.

We studied the following end-to-end and network-wide characteristics:

- 1) **(End-to-end metric 1) Packet delivery fraction:** measured as ratio of the total number of packets which were sent from the sources to the total number of packets that were received at the destinations, and averaged over the number of source-destination pairs.
- 2) **(End-to-end metric 2) End-to end delay:** measured as the average delay in sending packets from source to destination and averaged over the number of source-destination pairs.
- 3) **(End-to-end metric 3) Delay jitter:** measured as the average time difference between two consecutive data packet arrivals at the destinations and averaged over the number of source-destination pairs.
- 4) **(Network-wide metric 1) Average number of packet drops due to retransmission limit at the MAC layer:** measured as the average number of packets dropped at the MAC layer owing to a lack of route and averaged over the number of nodes.
- 5) **(Network-wide metric 2) Average number of 802.11DCF MAC layer unicasts sent:** is the total number of all (successful) MAC layer unicast transmissions sent in the network. This is a measure of the network overheads, and a measure of the total number of data, route-acquisition/route error packets sent and is averaged over the number of nodes.
- 6) **(Network-wide metric 3) Average number of 802.11DCF MAC layer broadcasts sent:** is the total

number of all MAC layer broadcasts sent by all nodes in the network. This is a measure of route recovery process overheads and is averaged over the number of nodes.

- 7) **(Network-wide metric 4) Average number of route errors initiated:** is the average number of route errors generated in the network and is averaged over the number of nodes.

We analyzed the results from the two experiments and show them using graphs showing 95% confidence intervals of the measured values.

2. Results

1) *Experiment 1: Effect of increasing number of source-destination pairs:* In Experiment 1, the number of source-destination pairs was increased from 10 – 45 in steps of 1 source-destination pair. Each source sent 5 packets of 512 bytes each per second to a destination. Figures 1 – 7 show the results from Experiment 1.

From Fig. 1, we see that ANSI and AODV perform comparably² at lower loads to medium loads (with ANSI performing slightly better³ than AODV in very low loads – points for 10, 11, and 12 source-destination pairs) and ANSI performs increasingly better than AODV in higher loads. That is, we see that the degradation in performance of AODV is rapid and drastic⁴, while for ANSI, it is more gradual and stable⁵. In addition, from Figures 2 and 3, we see that ANSI delivers these packets with either a comparable delay accompanied by a comparable delay jitter, or lower delay along with an increasingly lower delay jitter at higher loads. Furthermore, we also see that ANSI delivers these packets with comparable or the same MAC layer resource usage at lower loads (values for points below 25 source-destination pairs in Figures 6 – 7) and substantially lower MAC layer usage at higher loads (values for points above 25 source-destination pairs in the same figures). As before, we see that the behavior of AODV when the traffic increases in volume is rapid and drastic, judging from the jumps in the curves for AODV in Figures 6 – 7. This is also due to an increasing RERR activity, as can be observed in Fig. 5 (where we see that ANSI *consistently* generates fewer route errors), and the corresponding route acquisition activity in AODV. We see how this affects the network in Fig. 4. At very high loads, we see that AODV drops *nearly 4 times* as many packets as ANSI does, partly owing to the fact that ANSI does congestion-aware routing, and AODV does not, and partly due to the fact that AODV seems to spend more MAC resources with basic protocol procedures. We will re-visit this phenomenon when discussing the results of Experiment 2. Overall, we see that the AODV network reacts very drastically to the increasing traffic, but the ANSI network reacts more gradually, and any degradation in performance of ANSI seems to be due to the physical limits of the capacity of the network owing to high traffic.

2) *Experiment 2: Effect of increasing the packet rate.:* In Experiment 1, we saw how AODV and ANSI reacted to high traffic scenarios and commented that as compared to ANSI, AODV probably spent a lot more MAC resources on route acquisition procedures, but we could not conclude effectively as to whether it was ANSI's congestion aware routing that contributed to lower MAC layer resource usage or the fact that ANSI performs route discovery procedures conservatively.

²We mean to say “comparably” statistically – i.e., the corresponding points for AODV and ANSI have overlapping confidence intervals.

³By “better” we mean that (a) the confidence intervals of the corresponding points do not overlap, and (b) the average of the “better” protocol is better.

⁴The variation of the observed metric for AODV increases, as can be seen by the wider confidence intervals for AODV as the number of source-destination pairs increases.

⁵The width of the confidence intervals for ANSI does not show appreciable increases as the number of source-destination pairs increases.

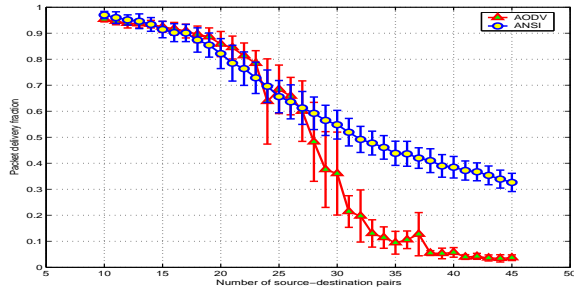


Fig. 1. Experiment 1: Average packet delivery fraction.

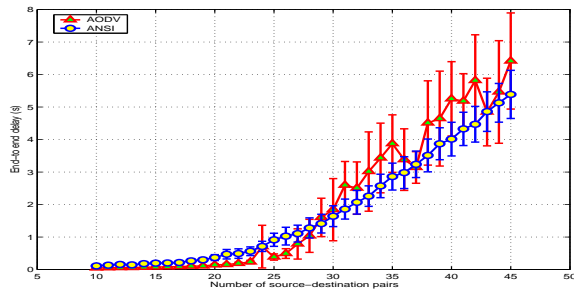


Fig. 2. Experiment 1: Average end-to-end delay.

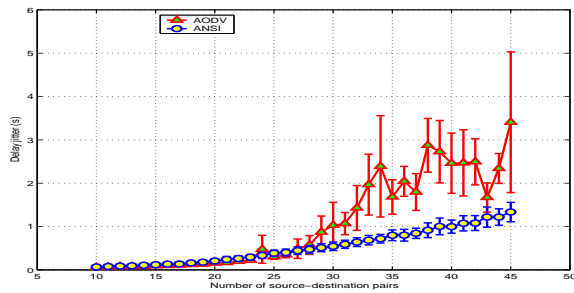


Fig. 3. Experiment 1: Average delay jitter.

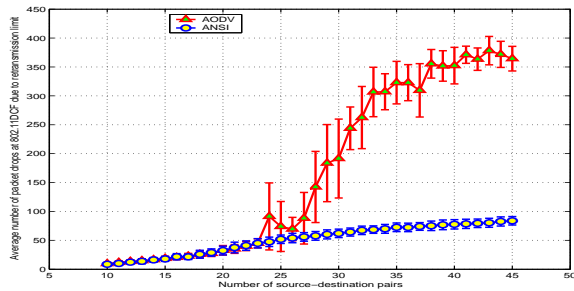


Fig. 4. Experiment 2: Average number of packet drops due to retransmission limit at the MAC layer.

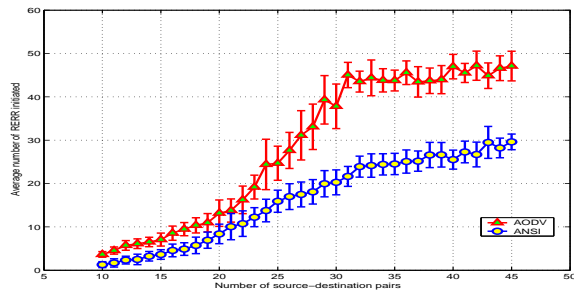


Fig. 5. Experiment 1: Average number of RERR initiated.

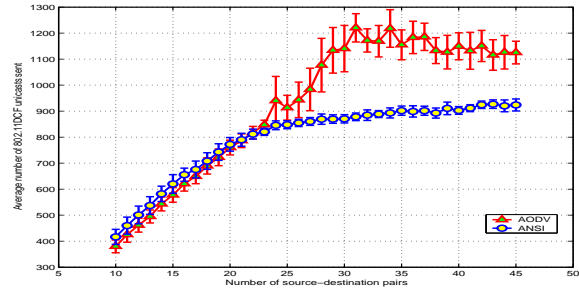


Fig. 6. Experiment 1: Average number of 802.11DCF unicasts sent.

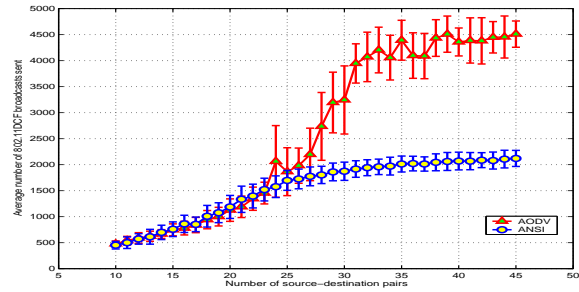


Fig. 7. Experiment 1: Average number of 802.11DCF broadcasts sent.

In Experiment 2, we conducted experiments where we expect almost no congestion to occur, i.e., low traffic scenarios, thus eliminating the possibility of ANSI's congestion aware routing.

We increased the packet rate of sources from 1 packet a second to 5 packets a second. There were 25 source-destination pairs in the network. The sources sent 64 byte packets. Figures 8 – 10 show the results for Experiment 2. We show only packet delivery, delay jitter and the number of RERRs for these experiments as the other graphs do not show a different behavior than observed in Experiment 1.

From the figures for Experiment 2, we see that ANSI performs with substantially better packet delivery than AODV in lower loads (see Fig. 8), with comparable or lower delay jitter (see Fig. 9), and about half the number of route errors (see Fig. 10). Note that the number of route errors is averaged. Also note that the loss percentage in ANSI is very low (packet delivery very close to 100%). Hence, it is reasonable to conclude that ANSI is able to achieve better performance as a combination of better managed route acquisition activities *and* congestion-aware routing.

3. Discussion

From the results, we see that ANSI performs better in cases where traffic is very low (owing to better managed routing information) or when the traffic is very high (owing to a combination of both congestion-aware routing and better managed routing

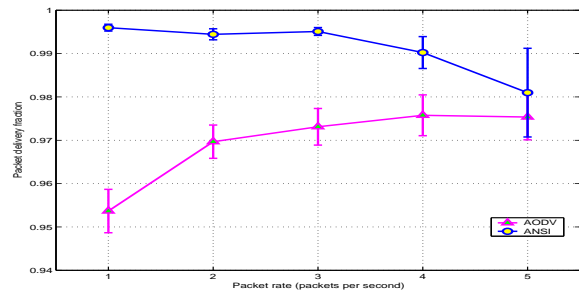


Fig. 8. Experiment 2: Average packet delivery fraction.

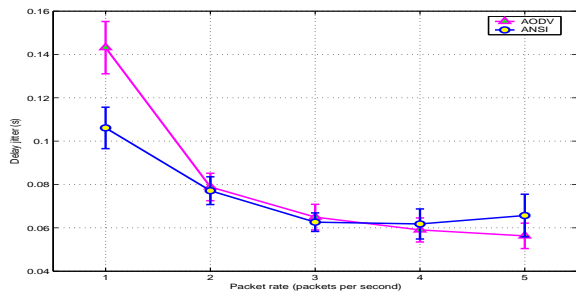


Fig. 9. Experiment 2: Average delay jitter.

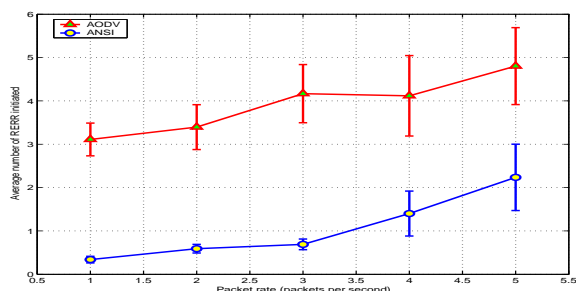


Fig. 10. Experiment 2: Average number of RERR initiated.

information), and performs comparably to AODV when the traffic is in-between. In addition, ANSI degrades more gradually than AODV. Also, judging from the size of the confidence intervals from all the results, ANSI shows lower variation in the measured values (i.e., the confidence intervals for ANSI are less wider) as compared to AODV for the same settings. This is because ANSI protocols contain more information about the local area around nodes, and thus adapt more quickly to local network and topological activity, by maintaining all possible next hop information at every node, and thus are more responsive to topological change than AODV. AODV generates more RERRs owing to the fact that it reacts only when a route breaks beyond repair, in comparison to ANSI, which continually repairs its local view of the network, choosing from a pool of next hops to avoid link breakages. Thus, AODV suffers from a RERR explosion, which in turn reduces the network resources to perform data delivery, especially when these RERRs are not received at the sources to notify the sources of a bad route.

5. Conclusions and future work

This paper describes the design, implementation and performance of a new reactive routing protocol, ANSI, for mobile ad hoc networks. We simulated ANSI and carried out a performance comparison with AODV, and the results show that ANSI either performs better than AODV with respect to packet delivery, end-to-end delay, delay jitter and MAC layer resource consumption or comparably with AODV. Regardless, we see that the variance of the observed values (as measured by the width of the confidence intervals) is always lower in ANSI. We also show that owing to the SI mechanisms in ANSI, it is able to abstract the activity around the local area of node better than AODV, and thus more responsive to topological fluctuations as compared to AODV. Hence, ANSI conserves network resources by generating lower number of RERRs.

We believe the basic ANSI structure provides for implementing a more general purpose routing protocol incorporating autonomously adaptive characteristics that enable it to behave well under all network and traffic conditions. For example, the nodes visited stack used in the ants in ANSI can be used

to collect a wide variety of information at the nodes the ant visits, such as energy reserves at a node, which in turn can be used to make next hop selection. In particular, we are currently analyzing a *hybrid* version of the protocol which will enable ANSI to work over hybrid networks with fixed and more capable infrastructure and adapt seamlessly in Mesh networks. In our hybrid version, we allow proactive routing and load balancing by choosing to perform stochastic routing in the fixed infrastructure.

References

- [1] C. E. Perkins, E. M. Royer, and S. R. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing, Internet Draft (draft-ietf-manet-aodv-09.txt)," November 2001, work in Progress.
- [2] D. B. Johnson, D. A. Maltz, Y.-C. Hu, and J. G. Jetcheva, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR), Internet Draft (draft-ietf-manet-dsr-07.txt)," February 21 2002.
- [3] C. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, 1994, pp. 234–244. [Online]. Available: citeseer.nj.nec.com/perkins94highly.html
- [4] Z. J. Haas, M. R. Pearlman, and P. Samar, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," July 2002, iETF Internet Draft, draft-ietf-manet-zone-zrp-04.txt.
- [5] C. Adjih, T. Clausen, P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized Link State Routing Protocol," INRIA Rocquencourt," Internet Draft (draft-ietf-manet-olsr-09.txt), April 15 2003, work in Progress.
- [6] V. Ramasubramaniam, Z. J. Haas, and E. G. Siler, "SHARP: A Hybrid Adaptive Routing Protocol for Mobile Ad Hoc Networks," in *The ACM Symposium on Mobile Adhoc Networking and Computing (MobiHoc 2003)*, Annapolis, Maryland, June 1–3 2003.
- [7] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence – From Natural to Artificial Systems*. New York: Oxford University Press, 1999.
- [8] J. P. Sterbenz, R. Krishnan, R. R. Hain, A. W. Jackson, D. Levin, R. Ramamathan, and J. Zao, "Survivable Mobile Wireless Networks: Issues, Challenges, and Research Directions," BBN Technologies, Cambridge, MA, September 2002.
- [9] J. S. Baras and H. Mehta, "A Probabilistic Emergent Routing Algorithm for Mobile Ad hoc Networks," in *WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, March 3-5, 2003.
- [10] G. D. Caro and M. Dorigo, "AntNet: A Mobile Agents Approach to Adaptive Routing," Universite Libre de Bruxelles, Belgium, Tech. Rep. IRIDIA/97-12, 1997.
- [11] D. Camara and A. A. F. Loureiro, "A GPS/Ant-Like Routing Algorithm for Ad Hoc Networks," in *IEEE Wireless Communications and Networking Conference (WCNC'00)*, Chicago, IL, September 2000.
- [12] M. Heissenbttel and T. Braun, "Ants-Based Routing in Large Scale Mobile Ad-Hoc Networks," University of Bern, Tech. Rep., 2003.
- [13] M. Gunes, U. Sorges, and I. Bouazizi, "ARA – The Ant-Colony Based Routing Algorithm for MANETs," in *International Conference on Parallel Processing Workshops (ICPPW'02)*, Vancouver, B.C., Canada, August 18–21 2001.
- [14] F. Ducatelle, G. Di Caro, and L. M. Gambardella, "Using Ant Agents to Combine Reactive and Proactive Strategies for Routing in Mobile Ad-Hoc Networks," *International Journal of Computational Intelligence and Applications*, Special Issue on Nature-Inspired Approaches to Networks and Telecommunications, 2005, to appear. Also Technical Report IDSIA 28-04.
- [15] G. Di Caro, F. Ducatelle, and L. M. Gambardella, "AntHocNet: an ant-based hybrid routing algorithm for mobile ad hoc networks," in *In Proceedings of PPSN VIII - Eight International Conference on Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science, no. 3242. Birmingham, UK: Springer-Verlag, Sept. 2004, best paper award.
- [16] M. Dorigo, G. D. Caro, and L. M. Gambardella, "Ant Algorithms for Discrete Optimization," Universite Libre de Bruxelles, Tech. Rep. IRIDIA/98-10, 1999.