ARTICLE IN PRESS

Available online at www.sciencedirect.com

SCIENCE DIRECT

Ad Hoc Networks xxx (2005) xxx–xxx

Ad Hoc Networks

www.elsevier.com/locate/adhoc

Protocol-independent multicast packet delivery improvement service for mobile Ad hoc networks ☆

Chien-Chung Shen *, Sundaram Rajagopalan

Department of Computer and Information Sciences, University of Delaware, 103 Smith Hall, Newark, DE 19716, USA

Received 24 October 2004; received in revised form 9 August 2005; accepted 2 November 2005

Abstract

This paper addresses the issue of improving multicast packet delivery in mobile ad hoc networks and proposes an adaptive mechanism called Protocol-Independent Packet Delivery Improvement Service (PIDIS) to recover lost multicast packets. PIDIS provides its packet-delivery improvement services to any multicast routing protocol for mobile ad hoc networks by exploiting the mechanism of swarm intelligence to make intelligent decisions about where to fetch the lost multicast packets from. PIDIS is a gossip protocol, and nodes using PIDIS are only concerned with which neighbor nodes to gossip with to recover the most lost packets, rather than which member nodes to gossip with. Thus, it does not rely on membership information in a multicast scenario, which is often difficult to get. PIDIS employs the beneficial aspects of probabilistic routing and adapts well to mobility. PIDIS achieves probabilistic improvement in multicast packet delivery and, unlike other gossip-based schemes, does not need to maintain information about group members from which lost multicast packets are retrieved. Further, the operations of PIDIS do not rely on any underlying routing protocol or primitive, and can be incorporated into any ad hoc multicast routing protocol. We incorporated PIDIS over ODMRP [On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks, Kluwer Mobile Networks and Applications, 2000], and compared it against Anonymous Gossip (AG) [International Conference on Distributed Computing Systems (ICDCS 2001) Phoenix, Arizona, April 2001] implemented over ODMRP, and ODMRP itself. Our simulation results show that ODMRP + PIDIS is more efficient and performs better than ODMRP + AG and ODMRP in terms of multicast packet delivery, end-to-end delay, and MAC layer overheads. We attribute the better performance and lower MAC overheads of ODMRP + PIDIS to the efficient gossiping made possible by using the swarm intelligence techniques.

Keywords: Multicast; Mobile Ad hoc networks; Packet delivery improvement; Swarm intelligence

1. Introduction

Ad hoc networks consist of mobile nodes that autonomously establish connectivity via multihop wireless communications. In many ad hoc applications, nodes need to collaborate to achieve common goals and are expected to communicate as a group rather than as pairs of individuals (point-to-point).

☆ A portion of the work was presented as a poster paper in the 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS), Ft. Lauderdale, Florida, October 24–27, 2004.

* Corresponding author. Tel.: +1 302 831 1951; fax: +1 302 831 8458.

E-mail addresses: cshen@cis.udel.edu (C.-C. Shen), rajagopa@is.udel.edu (S. Rajagopalan).

1570-8705/$ - see front matter
doi:10.1016/j.adhoc.2005.11.001

For instance, a group of soldiers roaming in the battlefield listen to their group commander (point-to-multipoint), and a group of commanders exchange current mission scenarios with one another (multipoint-to-multipoint). Multicast communication is a critical capability to support these applications.

However, severe operating constraints such as mobility of nodes, limited energy, memory and wireless bandwidth, jamming, interference and other environmental impairment prevent reliable packet delivery and result in high variation in the number of packets received by different member nodes. Although several multicast routing protocols have been proposed for mobile ad hoc networks, for example, [12,18,13,15,22], improving packet delivery has been a challenge. There have been efforts made to provide *reliable multicast* for ad hoc networks, however, these schemes, based on either ACK/NACK [21] or adaptive flooding [20], could either easily congest the networks and degrade throughput when network topology changes frequently, or may need to maintain state information about other members in the network. There have also been *transport* layer approaches to the problem of reliable multicasting in mobile ad hoc networks. In ReACT [17], the authors outline a transport layer protocol which achieves very high reliability by the use of transport layer mechanisms with end-to-end purviews.

Recently, two gossip-based approaches have been proposed to facilitate the notion of reliable multicast for ad hoc networks. Anonymous Gossip (AG) [6] provides a reliability improvement *service* that runs atop unreliable multicast protocols, and Route-Driven Gossip (RDG) [14] is a reliable multicast protocol. In contrast to [20,21] which suffer from the tradeoff between reliability and scalability, gossip-based approaches exploit the non-deterministic nature of mobile ad hoc networks to provide probabilistic reliability in a scalable manner [14].

In this paper, we address the problem of improving multicast packet delivery in mobile ad hoc networks via a protocol-independent, packet delivery improvement *service* that could be incorporated into any ad hoc multicast routing protocol. The service, Protocol-Independent packet Delivery Improvement Service (PIDIS), uses the mechanisms of swarm intelligence to decide where to recover lost packets from. Notice that PIDIS itself is not a reliable multicast *protocol*, but a *service* which improves multicast packet delivery of an ad hoc multicast protocol that incorporates it.

Swarm intelligence (SI) [3] refers to complex behaviors that arise from very simple individual behaviors and interactions, which is often observed among social insects such as ants and honeybees. Although each individual (for instance, an ant) has little intelligence and simply follows basic rules using local information obtained from the environment, (globally) optimized behavior[1] *emerges* when they work collectively as a group. In essence, swarm intelligence incorporates the following three components:

- positive/negative feedback, which search for good solutions and stabilize the results,
- amplification of fluctuation, which discovers new solutions and adapts to changing environment, and
- multiple interaction, which allows distributed entities to coordinate and self-organize.

Together, these components comprise an adaptive search mechanism that facilitates PIDIS to quickly converge to good candidate *routes* (leading to other group members) through which lost multicast packets could be recovered with the greatest probability, while discovering alternate routes for packet recovery to adapt to changing packet delivery patterns and network topology.

PIDIS is a gossip protocol and is adaptive to network usage and may gossip several times for lost packets. PIDIS continuously gauges the network conditions to control the extent of gossiping and number of gossips sent for a lost packet. PIDIS does not depend on membership views, either partial or total. Also, PIDIS is concerned with learning which neighbor next hop nodes give better packet recovery ratios when gossiped with, rather than learning which member nodes (when gossiped with) help recover the most packets (such as the use of *member_cache* in AG). Thus, in PIDIS, the extent and number of gossip packets are highly restricted by choosing from a highly focused set of next hop nodes as gossip partners. In PIDIS, gossip messages are treated as ants; valuable information in the gossip collected during the gossip request phase is processed when the gossip returns as a gossip reply. The effectiveness of PIDIS, as shown in simulation

[1] An example of these is that ants often find a shortest path from their nest to a food source.

results, attributes to the efficient learning capability of swarm intelligence.

The rest of this paper is organized as follows. In Section 2, we give an overview of PIDIS, followed by a detailed description of PIDIS and the implementation details of PIDIS over ODMRP in Section 3. In Section 4, we present both a detailed simulation study of PIDIS. Work related to PIDIS is reviewed in Section 5. We conclude the paper in Section 6.

2. Overview of PIDIS

PIDIS is a persistent packet recovery protocol: packet recovery attempts in PIDIS may be made more than once, and the number of attempts a packet recovery may be made is bounded and daptive.

A provision of PIDIS over a multicast routing mechanism works as follows:

1. An (unreliable) multicast routing protocol, χ, delivers packets to a node i, and
2. PIDIS service "kicks in" at node i to fetch the packets which χ has not been able to deliver to node i.

PIDIS, thus, provides services to the multicast protocol directly, i.e., at the network layer.

In this paper, we describe an implementation of PIDIS over ODMRP [12]. However, PIDIS can be easily implemented over any other multicast routing protocol with minimal changes from the implementation over ODMRP.

ODMRP is an on-demand, mesh-based multicast protocol that attempts to establish a *forwarding group* only when a source of the group has data to send. The nodes in the forwarding group form a mesh that connects the group members together. When a multicast source has data to send for the first time, it broadcasts to its neighbors a JOIN QUERY packet, which is a data packet with the *query flag* set. Upon receiving a non-duplicate JOIN QUERY, each node stores the upstream node ID in its routing table and rebroadcasts the packet. When a member of the multicast group receives a JOIN QUERY, it constructs and broadcasts a JOIN REPLY packet containing the source ID and the upstream node ID to all of its neighbors. Upon receiving a JOIN REPLY, a node whose ID matches the upstream ID in the JOIN REPLY packet realizes that it is on the path between the source and a member, so it becomes a forwarding node for the group by setting its *FG_FLAG* (Forwarding Group Flag). This node then constructs and broadcasts its own JOIN REPLY using its corresponding upstream node ID. The broadcasting of JOIN REPLY packets therefore propagates the information from all the members back to the source on the reverse paths. Once the source has sent out a JOIN QUERY, it sends all subsequent data packets normally with no *query flag* set. This will allow only nodes that are currently in the forwarding group to rebroadcast these data packets, thus reducing data forwarding overhead. To deal with dynamics of the network topology and group membership, each source floods the network with JOIN QUERYS every *REFRESH_INTERVAL* as long as it still has data to be sent to the group. The *FG_FLAG* on each node will be reset if it has not been refreshed by JOIN REPLY for some period of time, which implies that the source has no data to send, or it is no longer needed as a forwarding node. If nodes are equipped with GPS, a mobility prediction method can also be used to adaptively adjust the value of *REFRESH_INTERVAL* to suit the current mobility condition.

ODMRP + PIDIS works as follows. When packets belonging to a source/group pair, s/g, are lost at a member node i,

1. Node i transmits a gossip request packet (GREQ) to recover lost packets. The GREQ is transmitted to a chosen one-hop destination, the *gossip nexthop* λ, given by Algorithm 3.2 (Section 3.2).
2. At any intermediate node X which receives GREQ,
   - (a) The id of node X is recorded in the GREQ.
   - (b) If node X is not a mesh node,[2] node X discards the received GREQ.
   - (c) If node X is a forwarding group node, but not a member or source node, node X forwards GREQ to a newly chosen λ as per Algorithm 3.2.
   - (d) If node X is a member or source node, then node X checks if it has the lost packets. If node X does not have any of the lost packets, node X forwards the gossip request to a newly chosen node λ as per Algorithm 3.2.

[2] Since we implement PIDIS over ODMRP, we use the notion of a mesh node to mean either a member node, a source node, or any of the forwarding group nodes.

(e) If node X is a member or source node, and if it has any of the lost packets which were reported lost by node i, then node X recovers the packets from its cache (see Section 3.1) and prepares a gossip reply packet (GREP) for *each* packet found. Each GREP backtracks the path of the GREQ to node i. The GREQ is then discarded at node X.

3. At each node y the GREP visits on the route back to node i, the hop previous to node y, node z, is remembered (at node y) as a useful hop to gossip with when there are lost packets from the source/group pair s/g. This process maintains the *Gossip Table* (described in detail in Section 3), a data structure used to maintain the information about which next hop nodes were useful in fetching GREPs.
4. If GREP packets intended to the source/group pair s/g are not received at node i after a timeout period, *PIDIS_GREQ_TIMEOUT*, node i may initiate another GREQ for the lost packets depending on a value c_s^g, which is the number of times node i will gossip for lost packets from s/g. The value of c_s^g is chosen according to the methods described in Section 3.5.

Fig. 1 illustrates the gossip process in PIDIS. The figure shows forwarding group nodes with letter IDs, the source node S, and member nodes with number IDs, and the non-participating nodes are hollow. Member node 1 has lost some packets, and hence a GREQ is sent from member node 1 to gossip for lost packets. This GREQ (eventually) reaches node 5 via several hops in the mesh. Node 5 is a member node and has cached some of the packets which member node 1 has lost. Member node 5 responds to the GREQ by sending a GREP back to member node 1. This GREP backtracks the path taken by the GREQ from node 1.

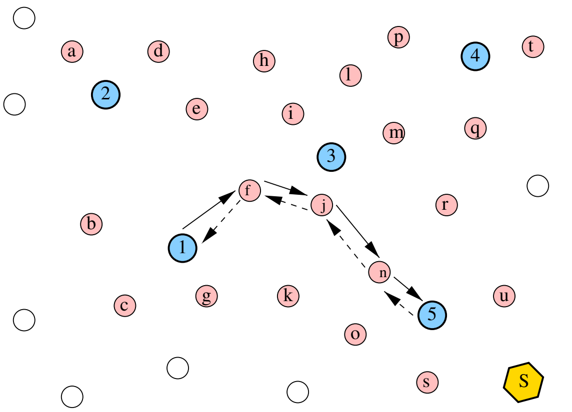


Fig. 1. The path of the GREQ from member node 1 is shown in solid arrows and the path of the GREP from 5 is shown in dashed arrows.

In PIDIS, the mechanism of swarm intelligence is exploited as follows. GREQs and GREPs work as ants—packets traversing the network, collecting information about the nodes they visit, which search and reinforce the (good) route(s) (leading to other group members) where lost packets could be recovered from. Information about the nodes a gossip traverses is recovered from a GREP, which backtracks the path of the GREQ. Since GREPs are only sent in response to GREQs sent, the overheads of PIDIS due to ant activity is highly controlled.

When no information about choosing next hops to gossip is available, only the neighboring mesh nodes information is used—the *gossip next-hop* node, λ, is picked *randomly* from the neighboring nodes. On the other hand, if information from previous gossip replies is available in the Gossip Table, the choice of λ is made *intelligently*—by choosing λ from the Gossip Table. Choosing λ from the Gossip Table improves the efficiency of gossiping, and there is a good chance that the gossip request sent to λ results in a gossip reply.

In addition to being able to choose a next hop intelligently, PIDIS eventually converges to find the best (next hop) node, in terms of number of GREPs fetched, to gossip with. Furthermore, owing to the amplification of fluctuation mechanism of SI, PIDIS adapts well to mobility by reacting to the topology change locally and exploiting other (or better) nodes to gossip with. Lastly, if several choices are available to gossip from the Gossip Table, then a choice of gossip next-hop is made probabilistically to better distribute (load-balance) the recovery efforts.

3. PIDIS description

This section describes the implementation of PIDIS over ODMRP in detail. Section 3.1 describes the local data structures used. Section 3.2 describes the selection process of gossip next-hop, λ. Section 3.3 describes the message cache functionalities. We describe the maintenance process of Gossip Table in Section 3.4, and in Section 3.5, we describe the adaptive characteristics of PIDIS. Finally, we describe our justification for the model we choose in Section 3.1 in Section 3.6.

3.1. Local data structures

3.1.1. Gossip Table at node i ($\mathcal{G}_i$)

A gossip table, containing the information collected from ant activity, is maintained at each node i which is either a member node or a forwarding node. The format and usage of the information in the gossip table is modeled from the ant decision table and algorithms described in [3]. At node i, $\mathcal{G}_i$ maintains the information about GREP receipts. A node j unicasting a GREP to member node i will result in an entry for node j in $\mathcal{G}_i$. The information contained in $\mathcal{G}_i$ is used to choose a next hop when a GREQ is to be sent out in search of lost packets. $\mathcal{G}_i$ stores the possible next hops for each multicast source/group pair, s/g, along with next hop j, the pheromone level τ_{ijsg}[3] and a heuristic η_{ijsg},[4] along with a probability value ρ_{ijsg} calculated from τ_{ijsg} and η_{ijsg}.

The value of ρ_{ijsg} is also a measurement of *goodness* of a particular next hop for gossiping. Intuitively, a next hop which has larger values for τ has "higher goodness" (i.e., better) as compared to another next hop with the same η, and likewise, a next hop that has higher η for the same τ has "higher goodness." The value of ρ for each of the next hops is then a measure of the *composite* goodness of the next hop, taking both pheromone level and the heuristic into account.

In the presence of multiple possible next hops for gossip, the value of ρ_{ijsg} is the probability of node i choosing node j as a next hop to gossip with when there are lost packets corresponding to s/g pair at node i. Note that at the time of node i sending the GREQ to node j, node j must be a member of the forwarding group, otherwise a GREQ sent from node i to node j is discarded and not propagated at node j. The quantity ρ_{ijsg} is computed as follows:

$$\rho_{ijsg} = \frac{\tau_{ijsg}^2 \times \eta_{ijsg}^2}{\sum_{j \in J} \tau_{ijsg}^2 \times \eta_{ijsg}^2} \quad (1)$$

where $J = \{j_1, j_2, \ldots, j_m\}$ is the set of m next hops available to contact for lost packets for a s/g pair at node i.

The value of τ_{ijsg} is a measure of how many GREPs (corresponding to a s/g pair) have reached node i via next hop node j, and the value of η_{ijsg} a measure of the closeness of node i to the gossip reply sender.

The pheromone level τ in $\mathcal{G}_i$ *evaporates* at a predictable rate. In this case, a *half life* model[5] is used for pheromone evaporation. If the value of ρ_{ijsg} gets below a threshold, node j's entry is purged from $\mathcal{G}_i$. The evaporation half time should be chosen carefully; otherwise the likelihood of choosing a gossip next-hop λ which does not belong in the current mesh for the multicast group increases.

Let us say a pheromone trail was first laid between nodes i and j for s/g at time t_1 due to the first gossip reply seen traversing the link ij. At time t_2, another gossip reply for the same s/g traverses the link ij. Then, $\mathcal{G}_i$ is modified as follows:

1. The old pheromone concentration is evaporated to reflect the current concentration due to the old trail:

$$\tau_{ijsg} = \frac{\tau_{ijsg}}{2^{(t_2 - t_1)/t_{0.5}}}, \quad (2)$$

where $t_{0.5}$ is the half life for evaporation.

2. Then the pheromone deposit and the heuristic due to the new gossip reply are set:

$$\tau_{ijsg} = 1 + \tau_{ijsg} \quad (3)$$

and

$$\eta_{ijsg} = \frac{1}{D}, \quad (4)$$

where D is the distance of node i from the gossip sender in number of hops. That is, each new GREP traversing link ij strengthens the pheromone trail ij by one unit for gossiping for packets corresponding to s/g.

Thereafter, the ρ values are recalculated at node i according to Eq. (1).

3.1.2. Neighbor table at node i ($\mathcal{N}_i$)

A neighbor table, $\mathcal{N}_i$, is maintained at each member/forwarding node i. $\mathcal{N}_i$ contains the list of neighboring mesh nodes (which may be member nodes, sources or forwarding group nodes) of node i for a particular multicast group. The neighbor table is used to select next hops when $\mathcal{G}_i$ cannot be used for selecting next hops. To reflect the status

[3] The pheromone level of a link ij is proportional to the number of times ants travel the link, and hence is one measure of goodness of the path for recovering lost packets.

[4] As again, higher values of η have "higher goodness".

[5] In the half-life model, a quantity halves its current value in a time duration given by the model's *half-life*.

of the current mesh, $\mathcal{N}_i$ is kept up-to-date by collecting information from the join replies of the ODMRP protocol. Thus, this information is maintained without any extra protocol overheads.

We note here that when implementing PIDIS over other types of multicast protocols, for example, in a multicast protocol which uses a multicast tree, the notion of a neighbor is modified to mean an adjacent node in the multicast tree.

3.1.3. Gossip request and reply

A gossip request message with sequence number node k initiated from member node i, GREQ_i^k, contains the following information: (a) The gossip sender address, i, (b) the sequence number of the gossip request, k, (c) the multicast source/group information, s/g, (d) the expected sequence number, e, (e) the number of packets, n, from s, lost since packet number e at i, and (f) the nodes-visited stack, S_i^k, where S_i^k is the record of the nodes which the gossip request visited.

At a node x receiving GREQ_i^k, if node x contains some or all the messages in GREQ_i^k, one gossip reply is potentially created for each lost message in GREQ_i^k, subject to lost message availability at node x. Hence, potentially, n separate gossip replies are generated for a gossip request that reports n lost messages at node i. Because each gossip reply, GREP_x^q (q is the sequence number of GREP sent) is source-routed, it must contain the nodes-visited stack from the corresponding GREQ_i^k. Hence, GREP_x^q contains: (a) The gossip reply sender address, node x, (b) the nodes-visited stack S_i^k from GREQ_i^k, (c) a message with sequence number $l \in [e, e+n)$ which was lost at node i, and (d) the s/g pair for the message.

Only members are concerned about lost packets, and hence, only members can generate GREQ. Also, because only members/sources cache messages, only members/source nodes can send gossip replies. As mentioned earlier, both gossip requests and replies are treated as ants, and the information contained in GREP_x^q is used to update the gossip table at all intermediate nodes.

3.2. Selecting a gossip next-hop (λ)

A GREQ with sequence number k from node i, GREQ_i^k, is broadcast, unicast or not sent at all depending upon two parameters: (a) the broadcast probability, P_{bi}, and (b) the neighborhood information. A gossip request is broadcast with a probability of P_{bi}. The value of P_{bi} has to be controlled so the network is not overwhelmed by broadcast packets. In the event GREQ_i^k is to not to be broadcast, the gossip table $\mathcal{G}_i$ or the neighbor table $\mathcal{N}_i$ is used for picking a next hop for unicasting the gossip request. At all times during the transmission of the gossip, care is taken not to gossip with a node which has already been visited by comparing the node ids in the nodes visited stack against a chosen next hop node. This prevents cycling of gossips. The algorithm for the process of next hop selection in PIDIS is shown in Algorithm 3.2.

Algorithm 1. Next hop selection for PIDIS

Require: λ, gossip next-hop for group G and source S
 $x \Leftarrow rand()$; {Generate a random number}
 if $x <= P_{bi}$ **then**
  $\lambda \Leftarrow -1$; {Broadcast}
 else if $(|\mathcal{G}_{isg}| > 0)$ & (A new node to gossip with) **then**
  $\lambda \Leftarrow$ *chooseFromGossipTable*( ); {A node is probabilistically chosen, depending on ρ value}
 else if $(|\mathcal{N}_{isg}| > 0)$ & (A new node to gossip with) **then**
  $\lambda \Leftarrow$ *randomlyChooseFromNeighborNodes*( );
 else
  $\lambda \Leftarrow 0$; {Don't send GREQ}
 end if
 return λ;

3.3. Message caching

To enable the process of retrieving messages, each member/source node x in the PIDIS scheme stores a finite number $|\mathcal{C}_x|$ of the most recently received data packets it receives/sends respectively. When a gossip request GREQ_i^k is received at a member/source node x, node x checks its cache $\mathcal{C}_x$ for the lost messages. If a message is in $\mathcal{C}_x$, node x retrieves it and sends a gossip reply message containing that message to node i.

3.4. Maintaining the gossip table

To illustrate how SI and the gossip table help in the gossip next-hop selection, consider the example in Fig. 1. The GREP (sent from node 5) is treated as an ant collecting information about the regions of the network it traverses. At each intermediate node x which the GREP traverses, the information in the GREP is used to update the gossip table $\mathcal{G}_x$ of the node x. When the GREP reaches node 1, the GREP is used to update the gossip table $\mathcal{G}_1$ at node 1. The information in the gossip table at $\mathcal{G}_x$ is used to choose gossip next-hops for future gossip requests from node x.

Suppose that no other GREP has reached member node 1 yet. The receipt of the new GREP, which we know traveled a total of 4 hops including nodes 1 and 5, results in the following entry in $\mathcal{G}_1$:

Mcast Gp	Mcast Src	Next hop	τ	η	ρ
g	s	f	1	1/4	1

In this way, the *positive feedback* mechanism of SI in PIDIS positively reinforces next hop f as an effective gossip next-hop. Positive reinforcement of next hops can also occur at forwarding nodes and other members when a gossip reply is forwarded at these nodes to the intended lost packet recipient.

In addition to finding an effective gossip next-hop, a gossip is broadcast instead of being unicast with a probability of P_{bi}. This *amplification of fluctuations* mechanism of SI allows to discover alternate (better) routes. When a gossip reply is received in this case, another entry is made into the gossip table.

For instance, in the example above, let us say a GREQ was broadcast, and a corresponding GREP traversing 3 hops was received via node b approximately $t = 5$ s after the first GREP is received. Then, all the pheromone trails are first updated using Eq. (2) (assume $t_{0.5} = 10$ s):

$$\tau_{1fsg} = \frac{\tau_{1fsg}}{2^{t/t_{0.5}}} = \frac{1}{2^{5/10}} = 0.70$$

(only one entry exists at $\mathcal{G}_1$)

Then an entry for the GREP receipt via node b is made, using Eqs. (3) and (4), into the gossip table $\mathcal{G}_1$ at node 1, and the ρ values are recalculated using Eq. (1):

Mcast Gp	Mcast Src	Next hop	τ	η	ρ
g	s	f	0.70	1/4	0.22
g	s	b	1	1/3	0.78

Thereafter, when a GREQ is to be made for packets from source s, $\lambda = b$ is chosen with a probability $\rho_{1bsg} = 0.78$, and $\lambda = f$ is chosen with a probability $\rho_{1fsg} = 0.22$. Evaporating a pheromone trail, the *negative reinforcement* mechanism of SI, carries the semantics of reducing the importance of information that is old.

Note that further receipts of GREPs at node 1 via node f update the entry corresponding to node f in $\mathcal{G}_1$ according to Eqs. (2)–(4). So, if another GREP were received at node 1 from member node 3 via node b (two hops) 5 s later, after the receipt of this GREP, $\mathcal{G}_1$ would be:

Mcast Gp	Mcast Src	Next hop	τ	η	ρ
g	s	f	1.49	1/2	0.91
g	s	b	0.70	1/3	0.09

where τ_{1fsg} was calculated as follows:

$$\tau_{1fsg} = 1(\text{due to the new GREP}) + \frac{0.70}{2^{5/10}} \text{ (remaining pheromone due to all old gossip replies)}$$

Furthermore, in PIDIS, the *multiple interactions* mechanism of SI allows nodes to collaboratively interact between each another via gossiping messages (GREQ and GREP, which are used as ants) to discover routes. By combining the effects of positive/negative reinforcement, amplification of fluctuations, and multiple interactions, the mechanisms of swarm intelligence make PIDIS to adapt quickly to mobility and at the same time discover information about the best possible routes to recover lost packets. In this way the information maintained in the gossip table greatly improves the choice of gossip next-hop λ used for gossiping.

3.5. Adaptive mechanisms in PIDIS

To improve the packet delivery at a multicast member node m_1, it seems intuitive to increase the number of times member node m_1 gossips for a lost packet, i.e., increase the number of times a GREQ is sent for a particular chunk of lost packets. In this case, after a timeout *PIDIS_GREQ_TIMEOUT*, if all the GREPs are not received at node m_1, another GREQ is initiated for the lost packets. The number of times a GREQ is initiated at node m_1 for packets intended to an s/g pair, c_s^g, $1 \leqslant c_s^g \leqslant l$, is adaptive, where l is the the maximum number of times PIDIS can gossip for a particular chunk of lost packets from s/g at a node. At m_1, packets intended to each s/g pair at m_1 are recorded and lost packets at m_1 corresponding to each s/g initiate GREQ. This gossip action initiated at m_1 has the effect of improving the packet delivery at m_1 for packets intented to s/g but also have the effect of increasing MAC layer demands which have the detrimental effect of decreasing the amount of resources that are needed to deliver packets to other members and also deter packet delivery to m_1 intended for other s/g pairs. Hence, the number of times a gossip is initiated must be controlled.

To achieve a balance in the overall network resources used for gossiping for lost packets, members in a multicast ad hoc network using PIDIS broadcast their packet delivery measurements maintained per s/g pair (this is maintained as as set of tuples $\langle s/g, pd_s^g \rangle$ for each s/g pair at the member) to other nodes in the network via a limited hop count broadcast. Any other member can use these measurements to decide whether or not a particular c_s^g value used for gossiping is appropriate. For instance, consider a member m_1 which receives information from another member m_2 about m_2's packet delivery measurements of all s/g pairs to which m_2 belongs. If the packet delivery at m_2 for s/g is lower than the packet delivery at m_1 for s/g, then m_1 reduces its c_s^g value to make more resources available to m_2 for gossiping and improving packet delivery. This process also ensures the reduction of the packet delivery variation index, a measure of the variation in the number of packets received across members.

3.6. Justification for the reinforcement models

In this section, we justify our use of the reinforcement equations used in Eqs. (1)–(4) for PIDIS design.

We note that the goodness of a next hop for gossip, ρ_{ijsg}, depends both on (a) the absolute value of the pheromone trail, τ_{ijsg}, and (b) the distance of the GREP sender, $\frac{1}{\eta_{ijsg}}$. A higher value of τ_{ijsg} means that the next hop was recently useful in fetching a lost packet, and higher values of η_{ijsg} means that a lost packet can be recovered in fewer hops if next hop node j is chosen to gossip. Also, if the path via one gossip next hop, j', is not as good (measured as the absolute value of ρ_{ijsg}) as the path via another next hop node, j, then the path via j should be chosen more frequently. Because GREQs should be sent *more often* via paths that have better ρ values, we choose to square the terms in our equation, because if we choose any greater an exponent for the terms in Eq. (1), then the protocol will increasingly choose better paths (i.e., paths with higher ρ values) more often than choosing paths with lower ρ values. We choose the exponent of 2 to balance the GREQ forwarding in favor of the better next hops, but perform load balancing of the GREQ propagation by occasionally choosing paths which are not as good.

Likewise, for Eq. (2), we keep the following ideas in mind: (a) a previously found good next hop node to gossip with decreases in effectiveness as a next hop to gossip with as time progresses, and so, the pheromone trail (which is measured as an absolute value of τ_{ijsg}) for a next hop should progressively decrease in goodness as time progresses, and (b) a newer next hop found for gossiping for the same s/g pair is better (i.e., has "higher goodness") than an older next hop, provided both the older and newer next hop share the same heuristics (η). In addition, in the context of mobile ad hoc network, we have to choose our negative reinforcement models such that (a) they do not become insignificant in value *too soon*, as this will trigger wastage of network resources (assuming no other next hop is available) by randomly choosing next hops, or (b) become insignificant *too late* before which a lot of next hops were used as though they would still be effective next hops. Intuitively, it seems like an exponential model will fit the MANET scenario, and the value of the base of the exponential model was chosen after observing the results from preliminary experiments.

For Eq. (3), we note that the pheromone update model was chosen to reinforce the goodness of a newly found next hop. In case that a trail for that next hop already exists, the equation simply re-iterates the goodness of the route.

The choice of the value of the heuristic in Eq. (4) is chosen to reflect that longer paths to recovery are less favored. Thus, if the GREP is sent from via two paths, one with $\eta_{ijsg} = 1/D$ and another with $\eta_{ij'sg} = 1/D'$, where $D < D'$, then the effect of this heuristic equation (Eq. (4)) is to choose next hop j over next hop j' for recovering lost packets for the s/g pair.

4. Performance studies

We implemented PIDIS over ODMRP in QualNet [1], and compared its performance with AG [6], which is also a reliability improvement service for multicast in ad hoc networks. We start our performance analysis with a brief comment on our implementation of AG.

4.1. Implementing anonymous gossip

The essence of AG is to allow a member m, which has lost packets, to recover packets from another group member m', whose identity is not known by m. This is why the authors call the protocol "anonymous." However, a few optimizations are used when AG is implemented over MAODV, namely, localizing the gossip and caching information about which members fetch gossip replies. These optimizations for MAODV+AG are tied together using a probabilistic model with several parameters. These optimizations could not be implemented in our implementation of AG owing to lack of adequate information in the AG paper regarding the appropriate parameters for the probabilistic model (for example, the value of p_{anon}, which is the probability of gossiping anonymously). Our implementation of AG was thus a "bare bones" version of AG which captured the essence of the anonymousness of AG.

In addition to the above, we had to adapt the AG protocol, which was described in the AG paper for an implementation over MAODV (a tree-based multicast protocol running over AODV), for use over ODMRP, which is a mesh-based multicast protocol, without the use of any unicast protocol. For transporting gossip packets in our implementation of AG, we used the nodes visited stack used in PIDIS (see Section 3.1.3) for AG as well. The AG mechanisms handling a GREQ avoided loops by making sure that only nodes which are not already recorded in the nodes visited stack are sent the GREQ.

4.2. Network and protocol characteristics

The mobility model was Random Waypoint with a minimum speed of 0.001 m/s and a pause time of 100 s. The MAC layer used 802.11DCF and the physical layer used omnidirectional antennas with a transmission range of 251 m using 802.11b. The propagation path loss model used was two-ray and no propagation fading model was used. The terrain size was 1000 m × 1000 m. We modeled a lossy network, and in our simulation model, the protocols dropped packets in a rectangular region defined by the cartesian coordinates (50, 50)–(250, 250) with a probability of 0.3. Our network contained 100 nodes. Both the sources and the group members were chosen randomly. The performance of the protocols ODMRP, ODMRP+AG and ODMRP+PIDIS was recorded and studied.

In addition to the parameters described above, several other parameters that were used were protocol specific. For the ODMRP simulations, we used the parameters specified in [12]. For the AG simulations, the protocol specification in [6] was used as guidelines. Several optimizations which were used in [6] could not be used in our AG implementation owing to the lack of specifications and/or lack of unicast framework in ODMRP. For AG, we used a *lost_buffer* size of 200 at each member node. For both AG and PIDIS, a message cache, $\mathcal{C}_m$, of size of 100 ($|\mathcal{C}_m| = 100$) was used. GREQs in AG, sent one per second, sends the 10 *most recently lost messages* (a list of the 200 most recently lost messages is stored in the *lost_buffer* at each node in AG). The size of the data packet was 512 bytes. Gossip replies were source-routed (using a nodes-visited stack) in both ODMRP+AG and ODMRP+PIDIS. While the GREQ used a fixed size nodes visited stack of 10 for both AG and PIDIS, AG used other information in the GREQ, such as the last 10 sequence numbers lost at the receiver. Thus, the size of the GREQ for AG was 108 bytes, and the size of the GREQ for PIDIS was 76 bytes, including the fixed nodes visited stack of size 10 for both AG and PIDIS. The size of the GREP for both AG and PIDIS were 76 bytes plus the size of the data packet recovered.

For PIDIS, probability of broadcast, P_{bi} was 0.001. In addition, the evaporation half time, $t_{0.5}$, was 12 s. This was carefully chosen to reflect the fact that a gossip next-hop chosen from the gossip table will most likely be a member of the mesh for the group in consideration. The pheromone threshold was set at 0.0625; a next hop j whose τ value fell below this threshold was removed from $\mathcal{G}_i$. Both PIDIS and AG used a fixed nodes-visited array of size 10 for keeping track of the nodes visited by a gossip request. The number was appropriate given the terrain size, node distribution, transmission range and the number of nodes in each experiment. In PIDIS, the packet delivery data for each s/g pair at the members are broadcast (with a TTL of 5 hops) every 20 s. Also, the maximum l value was 6 and the minimum was 1: there was *at least* one GREQ per lost packet.

4.3. Experiments and performance metrics

We performed three experiments: In Experiment 1, one group consisting of 10 group members were sent packets at 10 packets/s from three sources, in Experiment 2, one group consisting of 30 members was sent 10 packets/s from 1 to 5 sources. In Experiment 3, 2 sources sent 10 packets/s to one group consisting of 10–50 members (in steps of 10 members).

We studied the following metrics/overheads from the simulations:

1. *Metric 1*: The packet delivery ratio, defined as the ratio of the total number of packets received at the receivers to the total number of packets which are expected to be received at the receivers.
2. *Metric 2*: The variation index of packet delivery, defined as the average Coefficient of Variation among packet deliveries at each member for all sources.
   Thus,
   $$VI = \text{mean}\left(\frac{\text{std}(PD_{m_1}, PD_{m_2}, \ldots, PD_{m_l})}{\text{mean}(PD_{m_1}, PD_{m_2}, \ldots, PD_{m_l})}\right)$$
   where *PD* is the packet delivery due to *all s/g* pairs a member belongs to, and *mean*( ) and *std*( ) are the average and standard deviation of the values respectively.
3. *Metric 3*: The end-to-end delay, in seconds, defined as the average end-to end delay experienced by all packets received at the receivers.
4. *Overhead 1*: The routing layer overheads, expressed as the total number of GREQs sent by the members in ODMRP + AG and ODMRP + PIDIS.
5. *Overhead 2*: The MAC layer overheads, represented by the total number of MAC layer unicasts sent in the entire network, and,
6. *Overhead 3*: The MAC layer overheads, represented by the total number of MAC layer broadcasts sent in the entire network.

Our aim was to show that PIDIS achieves better performance metrics (packet delivery and end-to end delay), and achieved these goals with lower overhead than a comparable protocol, AG.

Before we describe and discuss our results, we note that while AG is a gossip protocol which relies on *periodic* lost packet recovery, PIDIS *reacts* to packet loss at a multicast receiver and thus is expected to send more GREQs in search of lost packets. In fact, we will see that as per our results, PIDIS, at times, sends nearly 8 times the number of GREQs which AG sends. This should *favor* AG, but we will see how AG's randomness in choosing gossip next hops actually decreases its performance, by increasing its MAC layer resource consumption, despite its substantially lower routing overheads (which are measured as the number of GREQs sent out at the members). We note here that for AG implemented over MAODV, the chance of a gossip request randomly sent out to a neighboring multicast tree node ending at a member or source is pretty high, but in ODMRP, the chances of a gossip request sent to a randomly chosen mesh node reaching a member or source are not as high as in AG implemented over MAODV. Indeed, it appears from our simulations that the chances are substantially lower.

4.4. Performance results

Figs. 2(a)–4(f) show our results. For all graphs, the errorbars indicate 95% confidence intervals of the recorded values.

In Experiment 1, we analyzed the effect of mobility on ODMRP, ODMRP + AG and ODMRP + PIDIS. We see that ODMRP + PIDIS shows significant improvement in terms of *packet delivery* and *end-to end delays*, though in the static case, the performance benefits of ODMRP + PIDIS are not as clear as the mobile case. In terms of *variation index*, ODMRP performs very well, with ODMRP + AG and ODMRP + PIDIS showing greater variation index. In terms of the number of *gossip requests*, ODMRP + PIDIS consistently shows a larger number of gossip requests sent than ODMRP + AG.

In Experiments 2 and 3, as in Experiment 1, we see that using PIDIS over ODMRP improves the metrics significantly in terms of *packet delivery* and *end to end delay*. The *variation index* for ODMRP + PIDIS is larger than both ODMRP + AG and ODMRP, with ODMRP showing the lowest variation index, but the variation indices for ODMRP + AG and ODMRP + PIDIS are comparable. From the figures, we also see that a larger number of *gossip requests* are initiated for ODMRP + PIDIS than ODMRP + AG.

In all three experiments, we see that *MAC layer unicasts* sent by ODMRP + AG are significantly larger than ODMRP + PIDIS and ODMRP. In terms of *MAC broadcasts*, ODMRP + PIDIS shows more (or comparable) traffic (as compared to ODMRP + AG), and ODMRP shows the most broadcast traffic. Note that in ODMRP, the only MAC unicast packets are due to acknowledgement packets to JOIN REPLYS, and the protocol functions are mostly due to MAC broadcast packets.

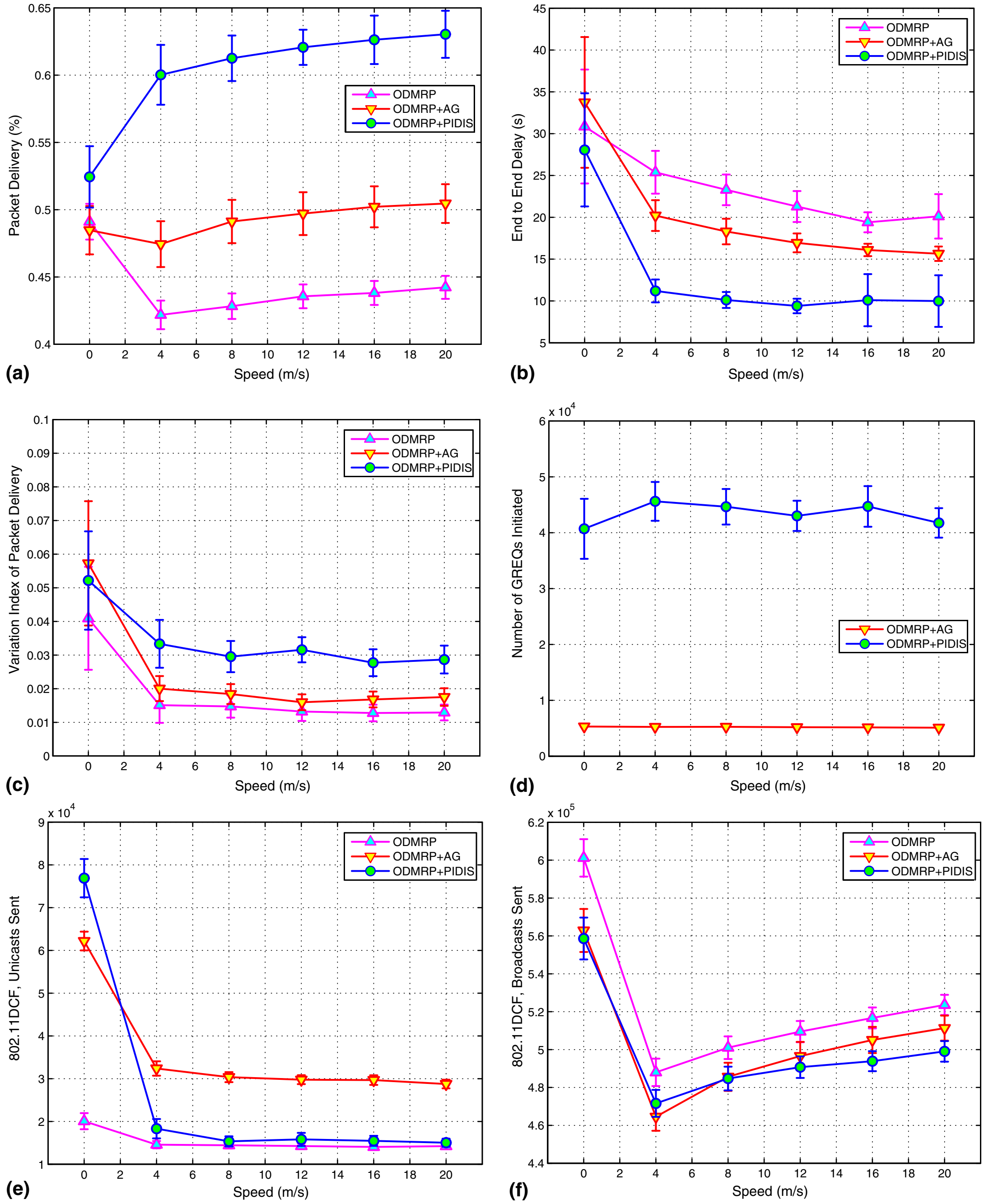


Fig. 2. Experiment 1: Effect of increasing mobility. (a) Packet delivery, (b) end to end delay, (c) variation index, (d) GREQs sent, (e) MAC layer unicasts sent and (f) MAC layer broadcasts sent.

Thus, for lost packet recovery in ODMRP, we conclude that PIDIS (a) is able to recover significantly more packets than AG, (b) recovers the same more quickly than AG, (c) uses lesser MAC layer unicast resources than AG, and lastly, (d) does the above with a variation index comparable to AG, but larger than that of ODMRP.

4.5. Discussion

There are two components to packet recovery using gossip—(a) interfere with the parent protocol minimally, and (b) gossip efficiently by choosing to gossip only with nodes along paths that will yield GREPs. Both of the above components go hand in

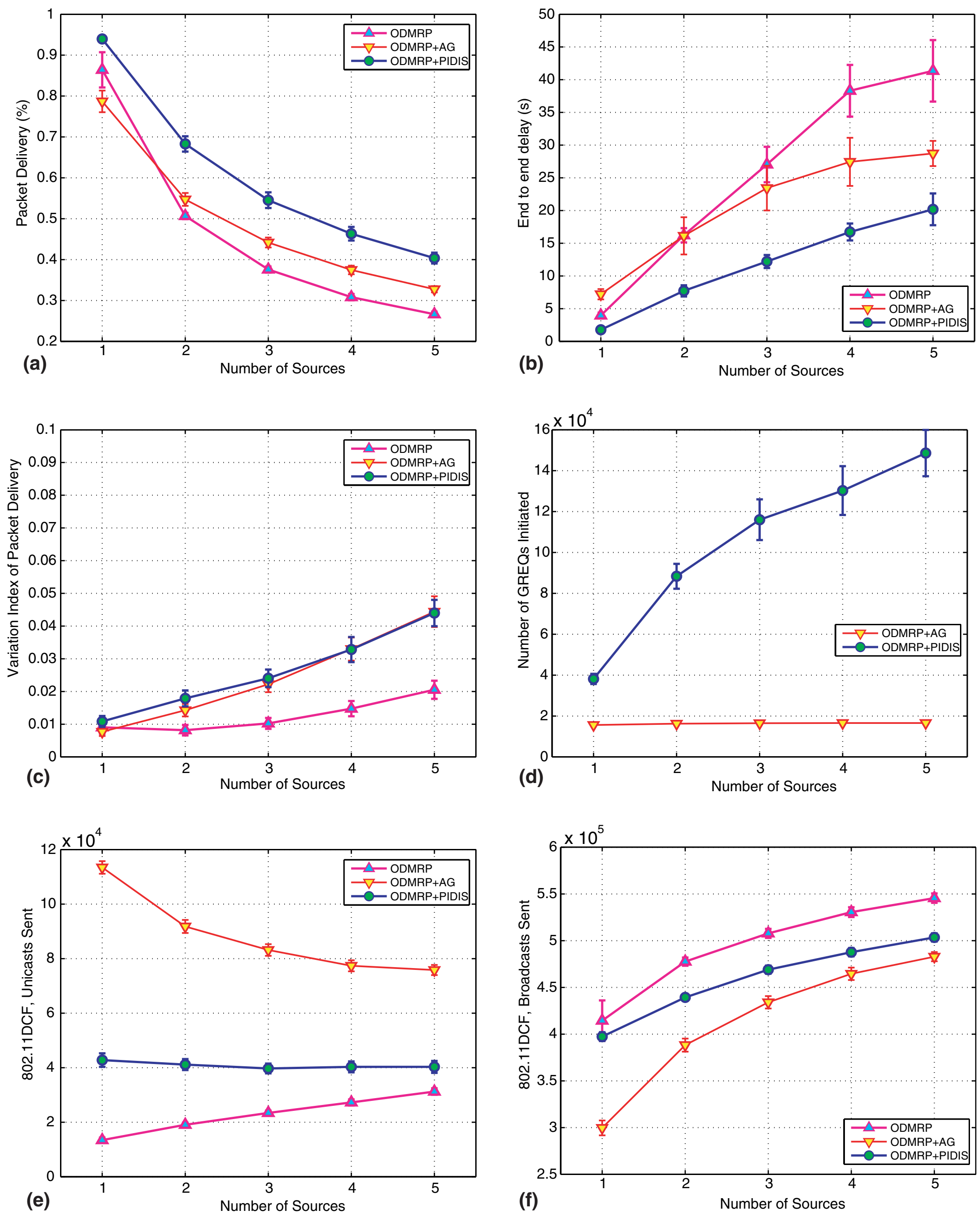


Fig. 3. Experiment 2: Effect of increasing the number of sources sending packets to a single group. (a) Packet delivery, (b) end to end delay, (c) variation index, (d) GREQs sent, (e) MAC layer unicasts sent and (f) MAC layer broadcasts sent.

hand when gossiping for lost packets. Our results can be reasoned according to one or both of these interacting properties for the protocols in question.

Our arguments in the following paragraphs stem from the above insights. In particular, we argue that:

1. PIDIS interferes *minimally* with ODMRP activity owing to carefully chosen gossip paths, and
2. AG interferes *heavily* with ODMRP activity owing to randomly chosen gossip paths.

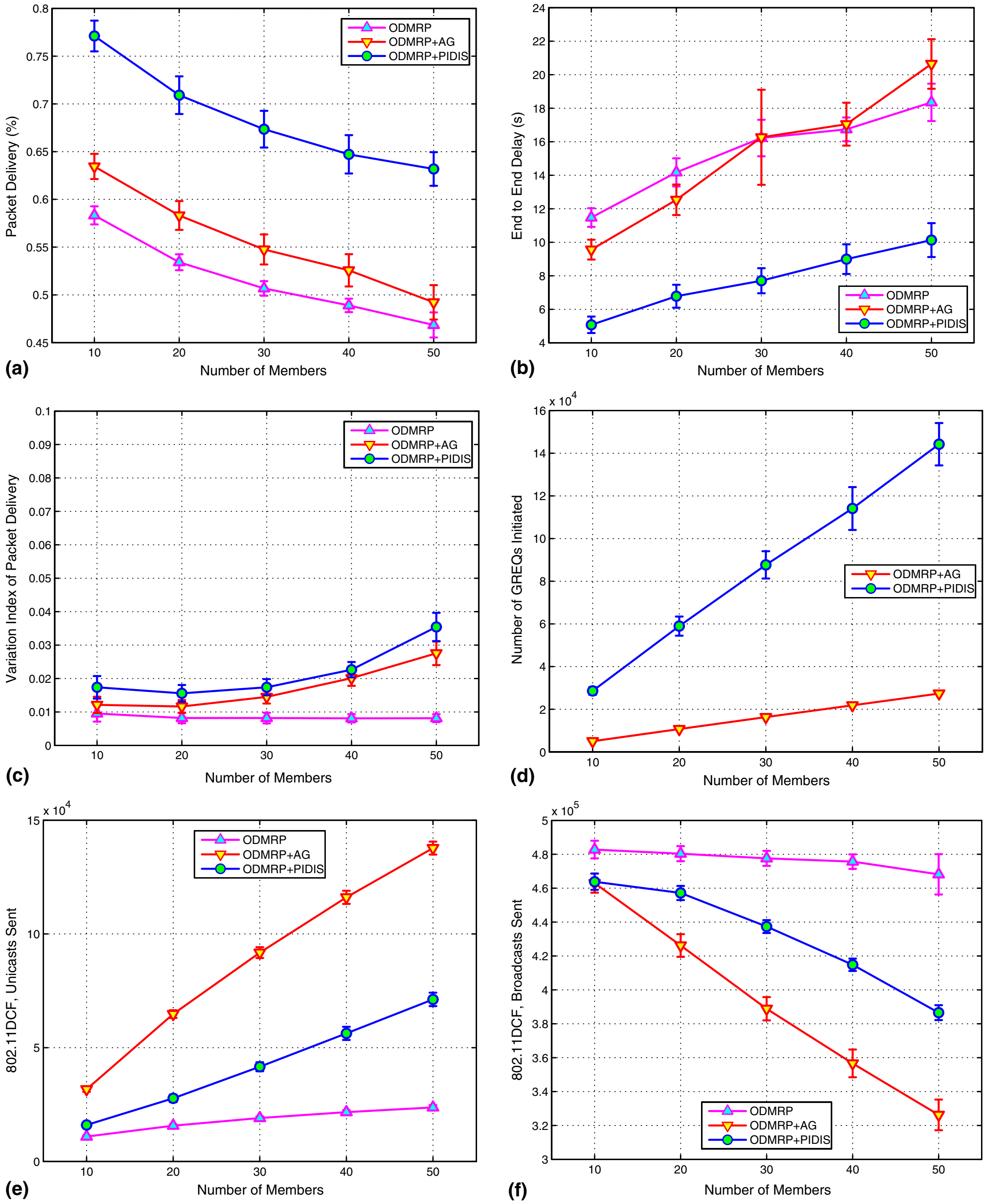


Fig. 4. Experiment 3: Effect of two sources communicating with one group with an increasing number of members. (a) Packet delivery, (b) end to end delay, (c) variation index, (d) GREQs sent, (e) MAC layer unicasts sent and (f) MAC layer broadcasts sent.

We will first discuss the overheads from the results, which should set the context for the discussion of the observed metrics (packet delivery, end-to end delay and variation index) in our experiments.

4.5.1. Routing overheads: Number of GREQs sent

The higher number of GREQs sent in ODMRP + PIDIS is because of two reasons: (a) PIDIS *reacts* to packet loss at a member, and so every time a packet is lost, a GREQ is generated, and (b) GREQs may potentially be generated l times (the adaptive nature of gossiping in PIDIS controls this number). Owing to this, a large number of GREQs are generated in ODMRP + PIDIS, which are broadcast with a probability of 0.001.

As before, note that the routing overhead for AG is much lower than PIDIS, but the protocol mechanisms in networks using AG do not guide the gossips correctly, leading to a large number of MAC layer unicasts, as we will soon see.

4.5.2. MAC overheads: Number of unicasts/broadcasts and number of unicast bytes

The only packets broadcast in ODMRP + AG are due to ODMRP packets, but in ODMRP + PIDIS, they are due to (a) ODMRP packets, (b) GREQs which are broadcast, and (c) s/g packet delivery measurements at each member which are broadcast from each member every 20 s. MAC layer broadcasts due to gossip activity have the detrimental effect of increasing collisions in the network, and reducing the effectivness of the mesh-wide flooding activity due to ODMRP activity. We observe this phenomenon in Figs. 2(f), 3(f) and 4(f). Both ODMRP + AG and ODMRP + PIDIS show decreases in the number of broadcasts as compared to ODMRP. In addition, ODMRP + AG shows the greatest reduction in the number of broadcasts. This is because of gossip-induced unicasts interfering with the ODMRP activity, as we will see soon. At any rate, the reduction in the number of broadcasts for both ODMRP + AG and ODMRP + PIDIS is due to the MAC layer broadcasts of ODMRP colliding with the gossip activity of ODMRP + AG and ODMRP + PIDIS. Thus, gossip activity, though intended to *reduce* packet loss, should accommodate for both the loss due to the gossip, and then recover the lost packets, for it to be an effective lost-packet recovery protocol.

MAC layer unicast packets have the detrimental effect of (a) increasing the average IP queue size at each node, and (b) increasing the propagation delay at each node. With increasing traffic rates, this leads to increasing packet drops at both the IP layer (owing to buffer overflows) and the MAC layer (due to reaching the maximum retransmission limit). In addition, the packets that do get delivered have had to wait in the IP queue for a long durations. We see from Figs. 2(e), 3(e) and 4(e) that ODMRP + AG has significantly more MAC layer unicasts as compared to ODMRP + PIDIS, Thus, we expect ODMRP + AG to have larger average IP queue sizes and a lot of packet drops due to both reaching re-transmission limits at MAC and IP buffer overflows. We note that this is owing to unguided gossips in ODMRP + AG.

We would also like to comment on the number of MAC layer unicast *bytes* that are consumed in AG and PIDIS. Despite the difference in the sizes of the GREQ in AG and PIDIS, and even though we compare only the *number* of unicasts sent (which is a measure of the gossip activity) in AG and PIDIS and not the number of MAC layer *bytes* sent for all gossip activity, our comparison is legitimate owing to the fact that the size of the GREQ of PIDIS is lower than the size of the GREQ in AG. In our results, we have shown that the number of unicasts in PIDIS is lower than AG, thus, the number of MAC layer bytes consumed in ODMRP + AG is larger than the number of MAC layer bytes consumed in ODMRP + PIDIS by a factor of $\frac{108}{76} \times \frac{\text{(The number of MAC layer unicasts in ODMRP + AG)}}{\text{(The number of MAC layer unicasts in ODMRP + PIDIS)}}$, which is clearly greater than 1. Given the fact that ODMRP + PIDIS sends fewer MAC layer unicast packets in all our experiments, we can reasonably conclude that the MAC layer utilization in ODMRP + PIDIS is more efficient.

4.5.3. Packet delivery and end-to end delay

In a nutshell, the better packet delivery and delay characteristics of ODMRP + PIDIS are due to the fact that the GREQs are better guided in ODMRP + PIDIS than in ODMRP + AG. Owing to this feature, the GREQs in PIDIS take shorter trips to nodes in the network that fetch GREPs, and along paths that are *more likely* to yield GREPs, rather than traversing the network randomly, which is what GREQs in AG do. The side effect of the better guided gossips in PIDIS is that gossip activity interferes minimally with ODMRP flooding activity. This is evident from Figs. 2(f), 3(f), and 4(f). The beneficial effects due to carefully guiding the GREQs in ODMRP + PIDIS is significant enough to guarantee that even successive retrials for fetching the lost packets do not increase the MAC layer unicasts to such an extent that the network bogs down due to gossip activity in ODMRP + PIDIS. We saw earlier how the MAC layer unicasts in ODMRP + AG affect its packet-delivery.

The reason why PIDIS does not perform as expected in the static case is because of a lean mesh in the static case. In the static case, ODMRP activity does not create as many new forwarding nodes as in the mobile case because the relative positions of the nodes do not change in the static case. Owing to this, there are a fewer number of mesh nodes in the static case. This translates to a few gossip next hops being chosen over and over again from the gossip table, which in turn results in hotspots in the network (owing to gossip activity along those paths). This reduces the performance of ODMRP + PIDIS in the static case.

When the mobility increases, the mesh becomes progressively denser, as more and more (new) nodes are added into the mesh. Note that some mesh nodes also drop out of the forwarding group if they do not receive JOIN REPLY messages for a specified period. If more mesh nodes are added into the forwarding group as are dropping out of the forwarding group, as is the case when the nodes are mobile, the mesh progressively increases in size as the mobility increases. With a larger mesh, more effective next hops are found by the gossip activity and are used, thus improving packet delivery for ODMRP + PIDIS when mobility increases.

From Section 4.5.2, we deduce that ODMRP + AG has a larger average IP queue size as compared to ODMRP + PIDIS. This large queue size is the reason why AG experiences high end-to end delay measures. The large number of unicasts in ODMRP + AG also significantly interferes with the flooding activity of ODMRP. Thus, the larger MAC unicast traffic in ODMRP + AG significantly affects the performance of the system. Conversely, the relatively smaller IP queue sizes in ODMRP + PIDIS allow it to deliver packets more quickly and with lower delay characteristics. This is why ODMRP + PIDIS is able to achieve better performance while incurring lower unicast overheads.

4.5.4. Variation index

Owing to flooding over the mesh structure, ODMRP is inherently geared to provide packet delivery with very low variation across members. Disturbing this mesh-wide flooding by including unicast traffic (by means of GREQ and GREP by ODMRP + AG or ODMRP + PIDIS) have the detrimental effect of increasing the packet delivery variation between members. As MAC layer unicast activity increases, the network becomes more and more "clogged" from a MAC broadcast stand point. This is because broadcasts do not require RTS/CTS shandshakes, nor are they retransmitted. Hence, ODMRP has almost no contention in the network for broadcasts—broadcasts are queued at a networks' IP queue and then transmitted—they may result in collisions, but the redundant nature of the flooding technique in ODMRP, resulting from the use of a mesh, recovers the packets lost during the collisions.

The increased MAC layer unicast activity for both ODMRP + PIDIS and ODMRP + AG is seen in Figs. 2(e), 3(e) and 4(e) then explains why ODMRP + AG and ODMRP + PIDIS shows higher variation indices as compared to ODMRP—interfering with the flooding activity of ODMRP with gossip activity has its drawbacks by cutting the number of packets received at members due to ODMRP activity alone.

Also, the variation index of ODMRP + PIDIS is (mostly) higher than both ODMRP and ODMRP + AG because of the large amount of wireless interface activity at the members—note that a very large number of GREQs (routing layer overheads) are initiated for ODMRP + PIDIS which, when sent out through the member's wireless interfaces, affects the packet delivery due to ODMRP flooding activity alone at the members. Regardless, owing to the efficient guiding of GREQ in ODMRP + PIDIS, the variation index is still comparable to that of ODMRP + AG.

PIDIS is thus able to significantly improve metrics for ODMRP, along with a variation index close to that of AG. This is owing to the adaptive persistence model of PIDIS.

5. Related work

The notion of reliable multicast for ad hoc networks has attracted different approaches to the problem. There are two categories of protocols—reliability improvement *services*, that run atop unreliable protocols, like AG [6] and PIDIS, and reliable *protocols*, like RDG [14], ReACT [17] and the reliability extension to ODMRP [20]. Unlike the reliability extension to ODMRP, RDG is an example of a reliable multicast protocol implemented over a unicast protocol (DSR [11]). Both RDG and the reliable extension to ODMRP, however, concentrate on being reliable without using any *repair* services, like AG or PIDIS. We compare the three gossip-based protocols, AG, RDG and PIDIS in Table 1.

ReACT is a transport layer multicast protocol, working with a multicast and unicast (the paper uses ODMRP and AODV) protocol, and has *end-to-end* purviews. ReACT, using cross-layer mechanisms, performs "typical" transport layer actions such as

Table 1
Comparing PIDIS with Route-Driven Gossip and Anonymous Gossip

Anonymous Gossip (AG)	Route-Driven Gossip (RDG)	PIDIS
Reliability improvement *service*	Reliable multicast *protocol*	Packet delivery improvement *service*
Gossip *pull* mechanism	Gossip *push* and *pull* mechanisms	Gossip *pull* mechanism
Persistent—periodically initiates recovery. A recovery may potentially take an unbounded number of gossips	*Persistent*—periodically initiates recovery	*Persistent*, non-periodic and adaptive. A recovery gossip is bounded—l times max.
Uses membership information (for the optimized version over MAODV)	Uses membership information	Does *not need* membership information
Uses unicast routing primitives (for the optimized verion over MAODV)	Needs unicast routing primitives	Does *not need* unicast routing primitive
Uses a randomly chosen recovery path (for the "bare bones" version)	Uses a randomly chosen recovery path	Uses a *smartly* chosen recovery path *probabilistically*

congestion control and error-receovery. These actions are *outside* the scope of PIDIS, which is designed purely as a packet-recovery protocol, and will kick in only if packets are lost at a receiver. In addition, PIDIS has (a) no control over the source (because PIDIS does not have an end-to-end purview) to prevent congestion and perform flow control (by decreasing flow rate), and (b) no control over congestion in the network. Furthermore, PIDIS is not concerned with the interaction across layers, and interacts only with a multicast protocol (i.e., strictly routing layer interaction).

We feel there are two approaches to reliability, (a) designing a packet delivery improvement scheme, such as PIDIS and AG, and (b) designing cross-layer and transport layer multicast protocols with flow control, congestion control and error-recovery, such as ReACT. These are two different approaches to the problem of addressing lack of reliability in multicast routing protocols in MANETs.

PIDIS uses Swarm Intelligence (SI) mechanisms to perform its services efficiently. There have been a number of SI approaches to perform unicast routing in mobile ad hoc networks. In addition, there is also a multicast routing protocol for mobile ad hoc networks which uses SI [19]. We discuss these protocols briefly below.

In [16], we describe a unicast routing protocol called ANSI for mobile, ad hoc networks which uses SI mechanisms to outperform AODV for high traffic scenarios. ANSI uses SI mechanisms to efficiently manage information about the neighboring nodes so route discovery overheads are not incurred too often. In [2] 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 [5], 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 [4], 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 e.g., 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 [4] 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 [10] 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 [9], 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 [8,7], 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. This approach is argued by the authors to be more ant-like [7] than other competing ant-based protocols.

In [19], the authors apply SI mechanisms to the problem of multicast routing in mobile ad hoc networks. MANSI adapts a core-based approach which establishes multicast connectivity among members through a designated node (core). In MANSI, an initial multicast connection is rapidly set up by having the core flood the network with an announcement so that nodes on the reverse paths to the core will be requested by group members to serve as forwarding nodes. In addition, each member which is not the core periodically deploys a small packet that behaves like an ant to opportunistically explore different paths to the core. This exploration mechanism enables the protocol to discover new forwarding nodes that yield lower total forwarding costs, where cost is abstract and can be used to represent any metric to suit the application.

6. Conclusions

In this paper, we designed and implemented a packet delivery improvement *service* for multicast routing in mobile ad hoc networks called PIDIS and studied its performance with an implementation over ODMRP. PIDIS is an adaptive, persistent packet recovery mechanism which uses Swarm Intelligence to gossip for lost packets effectively. PIDIS adapts to network conditions and adjusts the number of times lost message recovery attempts are made. PIDIS exploits the positive and negative feedback mechanisms of swarm intelligence to quickly search for good candidate routes from which lost packets could be recovered. PIDIS also utilizes the amplification of fluctuation mechanism of swarm intelligence to discover alternate, and possibly better, routes to adapt to changing packet delivery patterns and network topology. ODMRP + PIDIS is shown to have better performance characteristics—packet delivery, end-to-end delay, and overheads as compared to ODMRP + AG, in most simulation scenarios, because ODMRP + PIDIS is able to guide the gossip process effectively thus controlling the number of messages traversing the network.

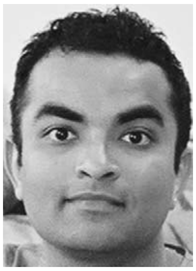
Acknowledgment

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

References

[1] QualNet, Version 3.6. Available from http://www.scalable-networks.com.
[2] J.S. Baras, H. Mehta. A probabilistic emergent routing algorithm for mobile ad hoc network, in: WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, March 3–5, 2003.
[3] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence—From Natural to Artificial Systems, Oxford University Press, New York, 1999.
[4] D. Camara, 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.
[5] G. Di Caro, M. Dorigo, AntNet: A Mobile Agents Approach to Adaptive Routing. Technical Report IRIDIA/97-12, Universite Libre de Bruxelles, Belgium, 1997.
[6] R. Chandra, V. Ramasubramanian, K. Birman, Anonymous gossip: improving multicast reliability in ad-hoc networks, in: International Conference on Distributed Computing Systems (ICDCS 2001), Phoenix, Arizona, April 2001.
[7] G. Di Caro, F. Ducatelle, L.M. Gambardella, AntHocNet: an ant-based hybrid routing algorithm for mobile ad hoc networks, in: Proceedings of PPSN VIII—Eight International Conference on Parallel Problem Solving from Nature, number 3242 in Lecture Notes in Computer Science, Birmingham, UK, September 2004. Springer-Verlag. Best paper award.
[8] F. Ducatelle, G. Di Caro, 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) 5 (2) (2005) 169–184.
[9] M. Gunes, U. Sorges, I. Bouazizi, ARA—The Ant-Colony Based Routing Algorithm for MANETs, in: International Conference on Parallel Processing Workshops (ICPPW'02), Vancouver, BC, Canada, August 18–21 2001.
[10] M. Heissenbttel, T. Braun, Ants-Based Routing in Large Scale Mobile Ad-Hoc Networks. Technical report, University of Bern, 2003.
[11] D.B. Johnson, D.A. Maltz, Y.-C. Hu, 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.
[12] S.-J. Lee, W. Su, M. Gerla, On-demand multicast routing protocol in multihop wireless mobile networks, Kluwer Mobile Networks and Applications (2000).
[13] M. Liu, R.R. Talpade, A. McAuley, AMRoute: Adhoc Multicast Routing Protocol. Technical Report 99, The Institute for Systems Research, University of Maryland, 1999.
[14] J. Luo, Patrick Th. Eugster, J.-P. Hubaux, Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks, in: INFOCOM 2003, San Francisco, March 30–April 3 2003.
[15] E.L. Madruga, J.J. Garcia-Luna-Aceves, Scalable multicasting: the core assisted mesh protocol, ACM/Baltzer Mobile Networks and Applications, Special Issue on Management of Mobility in Distributed Systems 6 (1) (2001).
[16] S. Rajagopalan, C.-C. Shen, ANSI: A unicast routing protocol for mobile ad hoc networks using swarm intelligence, in: International Conference on Artificial Intelligence (ICAI), Las Vegas, Nevada, June 2005.
[17] V. Rajendran, K. Obraczka, Y. Yi, S.-J. Lee, K. Tang, M. Gerla, Combining Source- and Localized Recovery to Achieve Reliable Multicast in Multi-Hop Ad Hoc Networks. May 2004.
[18] R. Royer, C. Perkins, Multicast using ad-hoc on demand distance vector routing. In *Mobicom'99*, Seattle, WA, August 1999, pp. 207–218.
[19] C.-C. Shen, C. Jaikaeo, Ad hoc multicast routing algorithm with swarm intelligence, ACM Mobile Networks and Applications (MONET) Journal 10 (1–2) (2005) 47–59.
[20] K. Tang, M. Gerla, Reliable on-demand multicast routing with congestion control in wireless ad hoc networks, in: SPIE 2001, Denver, Colorado, August 2001.
[21] K. Tang, K. Obraczka, S.-J. Lee, M. Gerla, Reliable adaptive lightweight multicast protocol, in: IEEE ICC, Anchorage, AK, May 2003.
[22] C.W. Wu, Y.C. Tay, AMRIS: a multicast protocol for ad hoc wireless networks, in: IEEE Military Communications Conference (MILCOM), Atlantic City, NJ, November 1999, pp. 25–29.

Chien-Chung Shen received his B.S. and M.S. degrees from National Chiao Tung University, Taiwan, and his Ph.D. degree from UCLA, all in computer science.
He was a senior research scientist at Bellcore Applied Research working on control and management of broadband networks. He is currently an associate professor in the Department of Computer and Information Sciences of the University of Delaware, and a recipient of NSF CAREER Award. His research interests include ad hoc and sensor networks, peer-to-peer and distributed object computing, control and management of broadband networks, and simulation. He is member of both ACM and IEEE.



Sundaram Rajagopalan obtained his undergraduate degree in Civil Engineering at the Indian Institute of Technology, Madras (IIT-M), India in 1995. After a brief stint as a software consultant, he returned to school in 1999 for graduate study in Computer Science at the University of Delaware. He obtained his Masters degree in Summer 2001 and has since been working on his Ph.D. His areas of interest include routing in ad hoc and space networks, peer-to-peer networks and content distribution, routing under extreme conditions and disruption-tolerant networks.