Available online at www.sciencedirect.com
SCIENCE DIRECT®
Ad Hoc Networks xxx (2004) xxx–xxx
Ad Hoc Networks
www.elsevier.com/locate/adhoc

Application versus network layer multicasting in ad hoc networks: the ALMA routing protocol ☆

Min Ge, Srikanth V. Krishnamurthy, Michalis Faloutsos *

Department of Computer Science and Engineering, University of California, Riverside, CA 92521, USA

Received 1 March 2004; received in revised form 19 September 2004; accepted 8 October 2004

Abstract

Application layer multicasting has emerged as an appealing alternative to network layer multicasting in wireline networks. Here, we examine the suitability of application layer multicast in ad hoc networks. To this effect, we propose a flexible receiver-driven overlay multicast protocol which we call Application Layer Multicast Algorithm (ALMA). ALMA constructs an overlay multicast tree in a dynamic, decentralized and incremental way. First, ALMA is receiver-driven: the member nodes find their connections according to their needs. Second, it is flexible, and thus, it can satisfy the performance goals and the needs of a wide range of applications. Third, it is highly adaptive: it reconfigures the tree in response to mobility or congestion. In addition, our protocol has the advantages of an application layer protocol: (a) simplicity of deployment, (b) independence from lower layer protocols, and (c) capability of exploiting features such as reliability and security that may be provided by the lower layers. Through extensive simulations, we show that ALMA performs favorably against the currently best application layer and network layer protocols. In more detail, we find that ALMA performs significantly better than ODMRP, a network layer, for small group sizes. We conclude that the application layer approach and ALMA seem very promising for ad hoc multicasting.

Keywords: Ad hoc networks; Multicast routing

1. Introduction

The high level goal of this work is to examine wether the advantages of application layer multicasting as seen in wireline networks can carry over in in ad hoc networks. To do this, we develop an application layer protocol and we study its performance extensively. Application layer multicasting

☆This work was partially supported by grants from Telcordia Technologies and ARL no: 100833196, from DARPA FTN Grant no: F30602-01-2-0535 and from UC Micro and HRL Laboratories, and NSF CAREER grant ANIR 9985195.

* Corresponding author. Tel.: +1 951 827 2480; fax: +1 951 827 4643.

E-mail addresses: mge@cs.ucr.edu (M. Ge), krish@cs.ucr.edu (S.V. Krishnamurthy), michalis@cs.ucr.edu (M. Faloutsos).

1570-8705/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2004.10.002

in wireline networks has received a lot of interest. However, many new challenges arise in using application layer multicasting in ad hoc networks. On the one hand, application layer protocols provides simplicity in terms of deployment and interoperability with existing infrastructure and lower layer protocols. On the other hand, such an approach may overload the already contention-ridden wireless links.

The majority of the proposed multicast protocols are network or IP layer multicast protocols. Such protocols require the cooperation of all the nodes in the network. We can group the protocols into two main categories according to wether they create a tree [10,18,2] or a mesh [11,20,5], which increases the routing robustness. For comparison purposes, we identify ODMRP as arguably one of the most efficient protocols network layer protocols [5,19,20]. ODMRP creates a mesh via periodic flooding of the network and then pruning of the unwanted branches.

Application layer or *overlay* multicasting is an attractive alternative to network layer multicasting, but has received little attention in the ad hoc networks domain. The main advantages of an application layer solution are the following. First, application layer multicast is easy to deploy, since it does not require changes at the network layer. Second, the construction of a logical structure hides routing complications such as link failure instances, which are left to be taken care of at the routing layer. Third, intermediate nodes do not have to maintain *per group state* for each multicast group. Maintaining per group state has always been a problem in multicasting even in the Internet [25,26]. Finally, application layer multicast can exploit the capabilities of lower layer protocols in providing reliability, congestion control, flow control or security according to the needs of the application. If the application requires reliability, it can choose, at run time, to use TCP between group members, otherwise it can choose UDP. In addition, secure group communications are reduced to secure unicast communications, which avoid the use of complex protocols for group key management [30].

The price to pay for the above advantages of an application layer solution is the routing efficiency. First, the use of application layer multicast can result in the transmission of multiple copies of multicast data packets over each physical link. This is exactly because non-multicast group members cannot make copies of multicast packets. This effect is especially visible when there are a large number of multicast group members and/or if the network load is high. Second, with mobility, using logical links may lead to sub-optimal paths, since the communicating member nodes are not aware of increases to their possibly small initial physical hop count distances from the source. Reconfiguring the logical connections is possible, but it introduces overhead. Furthermore, the frequency of these reconfigurations has not been examined yet.

In this paper, we want to evaluate the efficiency of application layer multicasting in an ad hoc setting. To do this, we develop an application layer multicast protocol, which we call Application Layer Multicast Algorithm or ALMA. ALMA is an adaptive receiver-driven protocol that creates a tree of logical links between the group members. Our goals is to design ALMA to overcome the inefficiencies of an application layer approach. We propose methods to reconfigure the tree upon mobility or congestion so as to reduce the *cost* of each logical link in the tree. We study the factors that affect the performance of ALMA and the sensitivity of its performance to various system parameters.

Our work can be summarized in the following points:

- ALMA outperforms the previous best-performing application layer multicast protocol in terms of goodput and reliability.
- ALMA performs favorably even when compared with network layer multicast protocols, more specifically ODRMP [20]. ALMA exhibits better goodput than ODMRP for moderately sized to reasonably large [1] group sizes where 20% to 40% of nodes are a part of the group.

[1] ODMRP performs better for extremely large multicast groups (when 60% of the nodes in the network in the group). Note that multicasting to groups that consist of more than 50% of the nodes in the network is close to performing a broadcast, since, most nodes will most likely overhear packets, independently of whether they are a part of the group or not.

- We examine the sensitivity of our protocol to different scenarios and we show how we can fine-tune its performance by choosing appropriate values for various system parameters.

This work extends our earlier work which introduced ALMA [6]. In this paper, we provide some more detailed analysis of the sensitivity of our protocol to some parameters. In more detail, we examine the reconfiguration thresholds which dictate when a node should attempt to switch parents, and the extent of the scope of the search. In addition, we provide some more detail on the protocol behavior, specifically on failure recovery and loop detection. Finally, we provide some more detail on previous work, specifically some work that compares broadcasting to multicasting.

The rest of the paper is organized as follows. In Section 2 we discuss relevant related work. We describe ALMA in detail in Section 3. In Section 4 we describe our simulation model, list the basic assumptions that we make and discuss the metrics of interest. In Section 5 we present our performance results, and our comparisons with the other aforementioned multicast protocols and discuss our observations. In Section 6 we present our conclusions and possible future work.

2. Background

In this section, we look at relevant background work in brief, including prior work on network layer multicasting in ad hoc networks, overlay multicast in wire-line networks and overlay multicast in ad hoc networks.

Network layer multicast protocols for ad hoc networks: Multicasting in ad hoc networks has primarily received attention in terms of designing efficient protocols at the network layer [10,2,11,5,16–18]. The Adhoc Multicast Routing Protocol utilizing Increasing id-numbers (AMRIS) [10] constructs a shared multicast delivery tree to deliver data. A multicast extension of the Ad Hoc On Demand Distance Vector(AODV) [1] is presented in [2] and is called MAODV for Multicast AODV. MAODV establishes on-demand multicast routes and uses these routes for delivery of multicast data. The Core-Assisted Mesh Protocol (CAMP) [11] builds and maintains a multicast mesh using a receiver-initiated approach and is based on the use of core nodes to maintain the mesh. The On-Demand Multicast Routing Protocol (ODMRP) [5,19] also uses a creates mesh of nodes for forwarding multicast data. Since we compare our protocol ALMA with ODMRP later in this paper we descibe ODMRP in some detail.

ODMRP: The mesh created by ODMRP is called the forwarding group. Multicast data is forwarded on the shortest path on the mesh between any member pair. Sources establish and update multicast routes on demand. They broadcast *Join-Query packets* to the entire network periodically. When a node receives a Join-Query, it records the ID of the upstream node that forwarded the query and rebroadcasts the packet. When the query packet reaches a multicast receiver, the receiver creates a *Join Table* and broadcasts this table to its neighbors. Upon receiving such a Join table, each neighbor checks to see if it is designated to be the next hop of one of the entries. If so, the node knows that it is on the path to one of the sources and now becomes a member of the forwarding group. It then broadcasts its own Join Table to its neighbors. Thus, Join Tables are propagated back to the source via the shortest path and accordingly the routes from source are thus constructed. In a fairly recent work, it is shown that ODMRP outperforms most of its competitor network layer multicast protocols for ad hoc networks [5].

Overlay multicast in wire-line networks: Overlay multicasting in wired networks has received a lot of attention [12,4,3,13]. Yoid [12] proposes a distributed logical tree building protocol between the end-hosts. It also creates a mesh used for dissemination of control data, and for fault tolerance. ALMI [4] uses a centralized algorithm to create a minimum spanning tree rooted at a designated single source. The Overcast protocol [3] organizes a set of proxies to form a distribution tree rooted at a central source. The NICE protocol[13] establishes a hierarchical clustering of multicast endhosts peers. Narada [8]creates a mesh and then builds delivery trees over the mesh using a DVMRP-like approach.

Overlay multicast in ad hoc networks: To the best of our knowledge, there are only two proposals for doing application layer multicasting in wireless ad hoc networks: (a) the Adhoc Multicast Routing protocol (AMRoute) [9] and (b) the Progressively Adapted Sub-Tree in Dynamic Mesh (PAST-DM) [7]. AMRoute uses bi-directional unicast tunnels to interconnect the multicast group members into a logical mesh. It then builds a shared tree for data delivery and maintains the tree within the mesh. Studies in [5] show that AMRoute performs well under static conditions, but it suffers from loops and creates inefficient trees even in scenarios of low mobility. PAST-DM also constructs a logical mesh connecting all group members. We describe PAST-DM in greater detail since we compare the performance of ALMA with that of this protocol.

PAST-DM: In PAST-DM, initially, each member initiates a search within a limited hop count to discover its logical neighbors. Each member records its logical neighbors and exchanges link state information with its neighbors. By doing so, each member obtains the topology map of the logical mesh. A Source-Based Steiner tree is then constructed upon this mesh. The tree is then periodically refreshed. During the construction process, the source makes all its logical neighbors its *first-level children* in the multicast tree and divides the remaining members into sub-groups. Each of these sub-groups forms a sub-tree rooted at one of the first-level children. The source includes the sub-group information in each packet header to let the first-level children know as to who belong to their sub-group. Each of the source's first-level children then repeat the Source-Based Steiner tree algorithm to establish their own subtrees and forward the data packet to the subtrees. The process continues. Both the logical mesh topology and the multicast tree gradually adapt to the changes of underlying network topology. Simulations [7] show that PAST-DM is more efficient than AMRoute.

It is interesting to note that overlay schemes can improve their routing efficiency by exploiting the broadcast nature of ad hoc networks. For example, one packet broadcast can be received simultaneously by two neighboring group members. However, in this paper, we do not consider such improvements in order to provide a fair comparison between pure overlay and routing layer schemes.

Broadcasting: Broadcasting has been proposed as an alternative to multicasting. This approach is desirable when the multicast group is a large percentage of the network or in highly mobile scenarios. Clearly, the use of broadcasting is inefficient when the group is a small percentage of a really large network.

There is one main work that compares multicast and broadcast protocols [22]. The authors compared two multicast protocols with flooding over a wide range of mobilities and traffic load conditions. It was shown that multicast protocols do not perform well in extremely dynamic networks. However, they considered only the scenario in which all nodes are receivers. To the best of our knowledge, there is no direct related research on evaluating the trade-offs between multicasting and intelligent broadcasting schemes. The comparison of broadcast protocols has been done in [21] and [29]. In [21], the authors discuss the problems with flooding and evaluate five modified broadcast protocols that they proposed. These five broadcast protocols are the counter-based scheme, the probabilistic scheme, the location-based scheme, the distance-based scheme and the cluster-based scheme. They showed that via simple heuristic modifications to the flooding protocol one can significantly reduce the number of wasteful rebroadcasts in the network while maintaining coverage. In [29], broadcast protocols are categorized into 4 different classes: simple flooding, probability based methods, area based methods and neighbor knowledge methods. A representative protocol was chosen for each class and the performance of the candidate protocols was evaluated. From the results in that work, the neighbor knowledge method appeared to have the best performance among the all broadcast approaches.

3. The ALMA protocol

In this section, we describe our protocol in detail. We provide an overview of the architecture, and highlight several interesting properties.

ALMA creates a logical multicast tree between the multicast members. We have considered the creation of a logical mesh, but a tree induces less maintenance overhead. Furthermore, the main advantage of a mesh, reliability, can be taken care of with the use of a reliable transport layer protocol such as TCP. Each edge of this tree represents a logical link, which corresponds to a path at the network layer. As an example in Fig. 1, there is a single logical link between nodes C and D. However, note that this logical link contains four underlying physical links, from C to Y, from Y to Z and from Z to D.

Receiver-driven Approach: Each group member finds a parent node on its own and once it joins, can decide to facilitate zero or more children. The parent of a node is the first node on the logical path from the node to the root along the tree. When a node receives a packet from the source, it makes multiple copies of the packet and forwards a copy to each of its children. Members are responsible for maintaining their connections with their parent. If the performance drops below a user or application defined threshold, the member reconfigures the tree locally, either by switching parents or by releasing children.

In the rest of this section, we describe the functions and mechanisms for the creation and maintenance of the logical tree. In the rest of this section, the term *link* refers to a logical link, and the term *node* refers to a member node, unless otherwise stated.

We specifically discuss the following issues with regards to ALMA:

- Creating and maintaining the multicast tree, i.e., member joins and leaves.
- Improving efficiency by reconfigurations of the multicast tree in mobile scenarios or when congestion is experienced at certain nodes.
- Ensuring packet level reliability during reconfigurations.
- Protecting against loops during reconfigurations.
- Ability to inter-operate with any lower layer protocol suite.

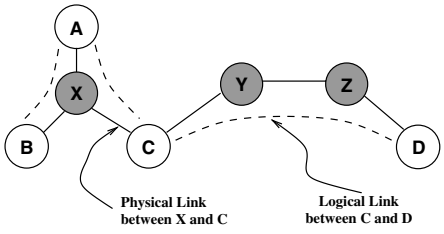

Fig. 1. Logical links versus physical links.

Joining a group: A new node finds a subset of the existing members preferably the ones that are close to it. A rendezvous point or a local search can provide such a list. A new member joins the group by sending join messages to possibly multiple existing members. An existing member that is willing to "take" a new child responds to this message. Note that an overloaded node (e.g. one with too many children already) can refuse to reply to a request. If a new node receives multiple replies, it can pick the one that seems best according to various criteria. Here, we assume that the node picks the member whose reply arrives first. Clearly, the parent selection can be based on other criteria as well. The first reply rule, which we use here, suggests but does not guarantee good performance, either because of proximity to that member or due to light load experienced by that member.

When a member wants to leave the group, it is required to send an explicit leave message to both its parent and its children. The parent will delete the node from its list of children, and its children then attempt to rejoin the multicast group. Unannounced member departures are equivalent to node failures that we discuss below.

Dealing with failures and partitions: Our protocols follows a soft state approach to deal with disruptive events such as node failures, network partitions, or unannounced departures from the group. Each member sends periodic hello messages to its parent and receives a response in return. If a node does not hear from its parent for a preset time-out period, [2] it assumes that the parent has

[2] The preset time-out should be a few "typical" round-trip times. This parameter controls the speed of reaction of the protocol to delays and failures.

failed and attempts to rejoin the multicast group. Clearly, the timely arrival of data packets is an indication that the parent is alive. Hello messages also provide an indication of the quality of the path, which is used in tree reconfiguration that we discuss next. We can have two courses of action when a failure is detected. First, the node that detects the loop can try to find a suitable parent without notifying its descendants nodes, in an effort to minimize the control overhead. Second, the node can let all its descendants know of the failure and each node tries to find a new parent separately. The last approach seems more wasteful, but it is simpler to implement, and it is commonly used in failure recovery in Internet multicast protocols [14].

Parent selection and tree reconfiguration: ALMA has a decentralized, receiver-driven reconfiguration scheme to avoid sub-optimal tree configurations. Children monitor the "quality" of the path to their parents, and switch parents when necessary. Even with an initial good tree, the multicast structure becomes inefficient, as members join, move or leave. For instance, the length of the path that forms the logical link may increase over time and this will degrade the performance.

We want the scheme to operate at the application layer to avoid any dependencies with lower layers. For this reason, we use the round-trip time (RTT), measured at the application layer, as an estimate of the quality of the link. Using lower-level metrics, such as hop count, may give better results, but violate the application level nature of the protocol. Our scheme works as follows: a member sends a hello message to its parent periodically. Upon the receipt of a response sent by the parent, it estimates the average RTT. When the average RTT exceeds some preset threshold (a protocol parameter), it will attempt to search for a new potential parent that can deliver data with a a smaller RTT in its neighborhood. If such a new parent is identified, the child can switch.

How "far" should a node look for a new parent? The extent of the search for a new parent may depend upon the RTT experienced. To identify the right extent, we introduce multiple thresholds. If the RTT exceeds the smallest threshold, the node would search for a new parent within a range of two logical hops of itself, that is, it polls its grand-parent and siblings on the tree. If the RTT exceeds next largest threshold, then the node would increase the scope of the search to three logical hops. This algorithm may be represented as:

```
  If estimated RTT < threshold_level1
stay with its current parent;
  If threshold_level1 < estimated RTT
      < threshold_level2,
search within a range of 2 logical
hops;
  If threshold_level2 < estimated RTT
      < threshold_level3,
search within a range of 3 logical
hops;
```

The reconfiguration capability of our protocol algorithm is shown in Fig. 2. When node C starts moving away from its parent B, it will experience an increase in the average estimated RTT. At some point in time, the first threshold level is exceeded and then C would start polling nodes that are at a distance of two logical hops from itself (The structure of the logical tree can be periodically piggybacked onto multicast data or can be obtained from the rendezvous host on demand.). Thus, C would periodically ping nodes A and D and estimate the RTT experienced in communicating with each of them. Let us for example say that the RTT does not change by much (is still greater than `threshold_1`). C continues to receive multicast data from B. After each attempt to reconfigure, C refrains from further attempts for a preset peri-

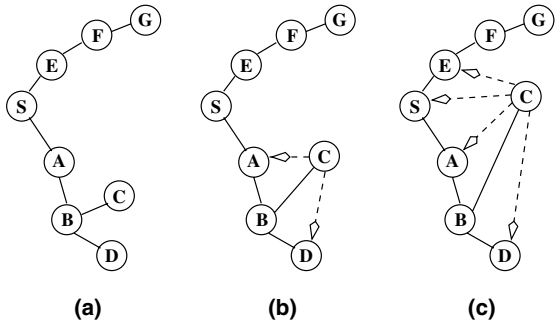

Fig. 2. Reconfiguration due to mobility. (a) Initial overlay tree, (b) C searches within one hop, (c) C searches within 3 hops.

od in time. At a later time, C now finds that the estimated RTT is now larger than the next threshold limit. At this time, it attempts to find a new parent by polling nodes that are within three logical hops; in this case those nodes are A,D,S and E. If C finds multiple parents that can deliver multicast data with a lower average RTT, it chooses the one that delivers data with the lowest average RTT. Upon deciding to switch parents a node sends a *switch* message to the newly chosen parent.

The use of RTT as a metric hides some subtleties. The average RTT may not capture the "quality" of the path towards the parent. If the node itself has many children, there is a competition for bandwidth between receiving data from the parent and delivering data to the children, in the node's vicinity. In such cases, attempts to switch parents may not be fruitful. We examine the effects of local congestion on RTT in our simulations.

Considering other end-to-end performance criteria: In our protocol, we can rather easily consider other end-to-end performance criteria while establishing or reconfiguring the tree. We can construct a more complex objective function of what is considered a good parent and a good path. This objective function may depend on several factors: (a) the RTT between the node and its parent; (b) the end-to-end delay as the sum of the average RTTs of all links between the source and the member and (c) the degree of the parent. In addition, nodes could keep statistics of packet losses at the application layer. However, this would require us to exchange a large amount of state and in order to keep our design simple, we do not examine this in our current work.

Detecting and avoiding loops: To avoid loops, a member should not be allowed to select one of its descendants as a parent. For this, each member is required to know the entire logical path on the tree towards the source, which we call **source-path**. Note that the source-path consists of group member nodes only. This approach has two advantages. First, when a node receives a switch request from another node, it checks to see if this node is on its source-path, and if so, it does not respond to the switch request. Note here that the path information needs to be updated only when a node joins the group or when a parent switch takes place. In fact, the path information can be easily refreshed with the periodic hello messages, which nodes exchange anyway for reliability purposes. Each member checks its path information, and if a loop is detected, the member will relinquish its connection with its parent and rejoin the group.

Loops due to synchronous switching: Loops can be detected, can they be avoided? Loops may still occur when two members simultaneously decide to switch parents and in the process, select each other or a descendant of the other as a parent as depicted in Fig. 3. In the first case, nodes A and B lose their logical connection to their parent, here the source, and they attempt to join each other. This case is trivial to detect and avoid. When a node sends a parent request to a prospective parent, the node can keep track of the identity of the prospective parent. If a parent request comes from that node the request is denied. In addition, even if we assume that the parent-child links are established, it is trivial to detect and break the cycle.

In the second case, nodes A and B select as parents each others children. The switch can happen at the same time, which makes this case more challenging to detect. To "break" the synchronization, we require that, when a member switches to a new parent, it has to wait for a short preset period of time, before accepting switch requests from other members. Furthermore, it can notify its descendants of the parent-change and they can also delay accepting new children. If nodes connect to each others descendants, the loop may not be prevented, but it can be detected as we explain below.

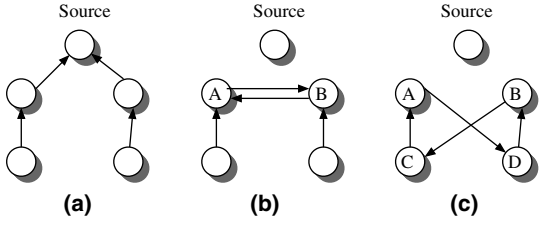

Fig. 3. Loops due to simultaneous changes in parents. We assume that nodes A and B lose their current connection to the source. The arrows are logical links, and the direction is from the child to the parent. (a) The initial tree, (b) A and B chooses each other as parents, (c)A and B chooses each other's descendants as parents.

In any case, the loop can be detected using the logical path to the source. The nodes in the loop will eventually find that their logical path to the source, never reaches the source. Furthermore, they will find that their path to the source includes themselves. When this happens, the node which detects the cycle, will abandon its current parent, and attempt to find a new parent. A loop detection can be considered as a failure and it is handled as one in the way we have discussed above.

Ensuring the continuity of the multicast upon reconfiguration: Switching parents can create "holes" in the received data stream. The new parent may have be forwarding packets that are further down the stream. For this reason, we provide two mechanisms: (a) delaying the connection to the old parent, and (b) data caching on every member node. Note that whether or not it is important for node A to obtain the missed packets depends on the application. For an application, such as voice or video that uses the user datagram protocol (UDP) such losses may be acceptable. However, for applications that require reliable delivery, recovery of these packets is important.

First, when a node switches parents, it maintains its connection with both parents, if possible, until the stream continuity is ensured. Since the position of each node, in the multicast logical tree, with respect to the source is different the new parent might have already received and distributed packets (to its children) that were not yet either received or distributed by the old parent. For instance, let node X switch from parent P to a parent P′; suppose the latest packet that X has received from P is packet with an application sequence number (if used) 20 while upon switching, the first packet that X gets from P′ is packet 32. If X simply breaks the connection with P upon switching to P′, it can no longer receive packets 21 to 31. However, if X does maintain its connection with P, X can receive packets 21 to 31 from P and then relinquish the connection.

Second, we require each multicast group member to *cache* data packets even after they have been forwarded to all existing children. This is especially important if reliable delivery is required. Thus, it becomes more likely that a new child will not miss packets: it can explicitly request the new parent for the packets that it has not received. If these packets are still in the new parent's cache, they can be retransmitted only to the requesting node, since the logical connections are unicast anyway. There is clearly a trade-off between the number of packets cached and the efficiency in retrieving missed packets when a switch occurs. If the missed packets are no longer in the new parent's cache, the switching node might recover these packets from the old parent or from the source itself using possibly longer physical paths.

Lower layers of the protocol stack: ALMA can be effectively used with any existing protocol suite. At the transport layer, one might choose either UDP or TCP. If reliability is required one might choose TCP. Recently, there have been modifications proposed to TCP for use in ad hoc networks [24] and new transport layer protocols such as ATP [15]. ALMA can be used in conjunction with any of these modified versions or new protocols. Any inherent advantages that a transport layer protocol provides can seamlessly provide benefits in performing multicast.

At the routing layer any ad hoc routing protocol may be used. ALMA can work with both proactive and reactive routing protocols [2]. ALMA depends on the routing layer for the detection of link failures and the reconstruction of logical links. Changes to the MAC or physical layers are also transparent to ALMA. Thus, ALMA can benefit from the use of advanced lower layer features such as from the use of directional antennas [27].

4. Performance evaluation

We evaluate the performance of ALMA and other multicast protocols by performing extensive simulations with the GloMosim, a parallel simulation software developed at UCLA using PARSEC [23]. We have three series of experiments with distinctive goals.

Series I: Application layer protocol comparison. First, we first compare ALMA with the state of the art application layer multicast protocol, PAST-DM [7] and show that ALMA performs better in most of the important performance metrics. We find that ALMA creates more efficient

trees, and avoids local hotspots by keeping the multicast degree of nodes lower. We trace the improved performance in ALMA to: (a) the use of RTT in selecting paths, which can indirectly avoid congested regions, (b) the quicker reconfiguration capability given the receiver-driven approach as compared to the centralized one in PAST-DM.

Series II: Application layer versus network layer multicast. We next compare ALMA with an IP Layer multicast scheme ODMRP. The latter has been integrated into the GloMosim simulator by the team that developed it in UCLA. These comparisons help us analyze the trade-offs between choosing an overlay multicast and an IP Layer network multicast in terms of a set of chosen performance metrics.

Series III: Quantifying the effect of protocol parameters. We examine the sensitivity of ALMA to various system parameters. The first observation is that a small cache can provide significant increase in the reliability of the protocol. Second, we evaluate the *goodness* of RTT as a metric for tree reconfiguration. Initial results suggest it is quite effective; we study the effects of varying the reconfiguration thresholds on the performance of ALMA.

The simulation environment: We use two different simulation scenarios in order to include as many cases as possible. Our first scenario is geared towards long paths, while the second scenario is the more typically used scenario in the literature.

Scenario 1: The scenario has 120 wireless mobile nodes in a $1000\,m \times 1000\,m$ region. To introduce longer distances, we select the radio transmission range to be 125 m. Such a relative small range could lead to network partitions, which would obscure the results. A way around this is to guarantee a good spread of nodes. Therefore, we make 81 nodes statically positioned in a 9×9 grid. Each node in this grid, is within a single-hop distance from its neighbors. We allow the other nodes to roam at certain chosen speeds (different speeds are considered). We use the random way point model in our experiments. In some of the experiments, we set the minimum speed to be equal to the maximum speed, i.e., the speed is constant for all nodes. Our motivation for using this model was based on recent results that show that with the random way point model nodes converge to slower speeds as the simulations progress [28] and our objective was to isolate the effects of speed on the performance of the multicast protocols. The pause time is 30 s as in other similar work [5].

Scenario 2: This scenario is the one used in the performance evaluation of ODMRP [5]. In fact, we used this scenario to establish that our simulator produces the same results for ODMRP as reported in the evaluation by its creators. The simulated network consists of 50 mobile nodes that move in accordance to the random-way-point mobility model within a $1000\,m \times 1000\,m$ region. The radio transmission range is 250 m, which leads to fairly short distances (approximately 3–4 on average). The modified random-way-point model as described earlier is used with a pause time of 30 s and the chosen fixed speed is varied as before.

We assume a raw maximum achievable data rate of 2 Mbps. Each member joins the group at the beginning of the simulation and remains in the group until the end of the simulation. Mobility causes reconfigurations and therefore, nodes often disconnect from and rejoin the tree. Each simulation lasts for 1000 s of simulated time. We varied the group size from 5 to 40 and the moving speed is varied from 0 m/s to 12 m/s. The traffic generated is constant bit rate (CBR) traffic.

Performance metrics: We use the following performance metrics to evaluate ALMA and to compare it with the other multicast protocols:

Multicast tree cost: The total number of the physical links that make up the logical links in the multicast delivery tree. This metric represents the *goodness* of the structure created by the application layer multicast protocol.

Stress: The stress of a physical link is the number of identical copies of a multicast packet that need to traverse the link. This metric quantifies the efficiency of the overlay multicast scheme.

Maximum logical degree: The logical or multicast degree of a node is equal to the number children plus one for the parent (if applicable). We consider the maximum logical degree over all member nodes for a simulation run.

Packet delivery ratio: The ratio of the number of packets actually delivered to the receivers versus

the number of data packets that were actually expected. This metric is used to quantify the reliability of the multicast protocol.

Goodput: The number of *useful bytes* (excluding duplicate bytes) received by the application process at a receiver per unit time. We use this instead of throughput since this definition is appropriate for comparing protocols with retransmissions (if applicable) as we do here.

Performance analysis: We describe in detail our simulation results and provide explanations of the observed behavior.

Series I: Application layer multicast protocol comparisons: We compare ALMA against the most recent and arguably the best application layer protocol for ad hoc networks in terms of performance, PAST-DM. We use the set-up from scenario 1. The source transmits CBR data at rate of 1 kbps. For the experiments for which no group size is specified a default size of 15 group members was used. For these experiments, we use the random way point model with a minimum speed of 0 m/s and the maximum possible speed set to 20 m/s.

ALMA creates a less expensive tree than PAST-DM: Our protocol can construct a multicast tree with a lower cost in terms of the physical hop count than PAST-DM. In Fig. 4, we plot the tree cost versus the size of the group. We attribute the difference to the 'receiver-driven' and 'locally adaptive' nature of ALMA. Recall that PAST-DM creates a logical Steiner tree in a somewhat centralized way; the decisions at the source affect the creation of the tree globally. Furthermore, the information between the members is exchanged using link-state updates and the source starts the construction of the tree. In ALMA, the reconfigurations are handled by the receivers, and we think that the local decisions turn out to respond more efficiently to the effects of mobility.

We varied the rate at which link state updates are sent in PAST-DM. This did not seem to make much difference. In Fig. 5 we consider a group of size of 25 and vary the periodicity of link-state updates in PAST-DM. We see that ALMA performs better than PAST-DM over a wide range of considered update frequencies. Note that if we further reduce the update period, the overhead incurred increases tremondously since the exchange of link-state information is expensive.

ALMA creates a tree with fewer bottleneck nodes: The maximum logical node degree in a tree constructed by ALMA and the maximum stress observed (among all the physical links in the tree) are much smaller than that with PAST-DM (with update period of 20). In Fig. 6, we plot the maximum logical degree versus the group size, and in Fig. 7, the maximum stress versus the group size. In addition, the variance in the logical degree in ALMA is much lower than that in PAST-DM, but these results are not shown here due to space

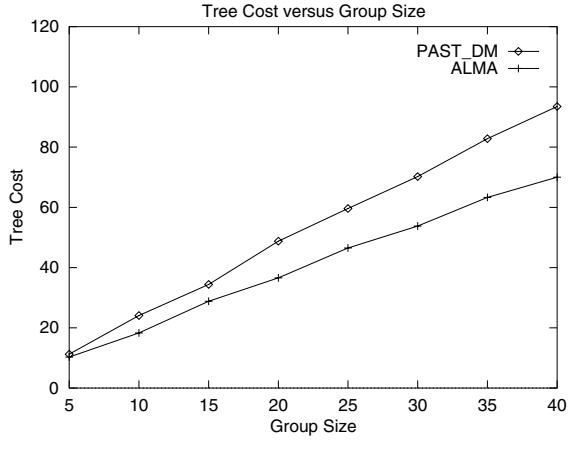

Fig. 4. Series I: tree cost versus the group size.

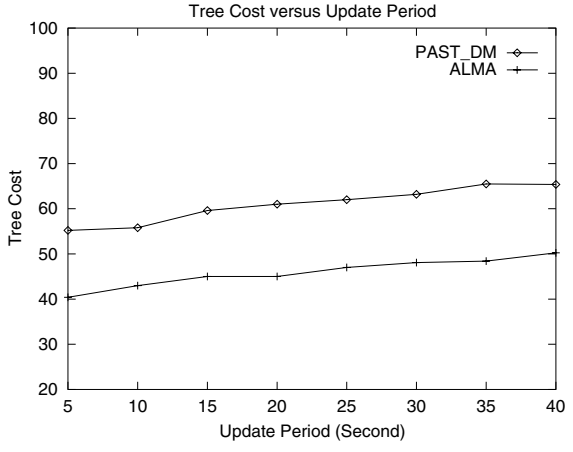

Fig. 5. Series I: tree cost versus the update period in PAST-DM: group size = 25).

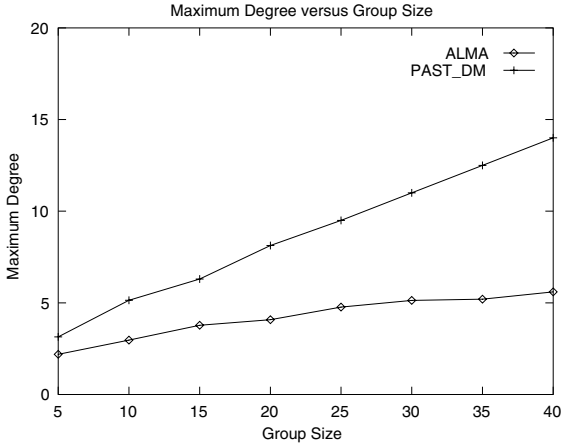

Fig. 6. Series I: maximum logical degree versus the group size.

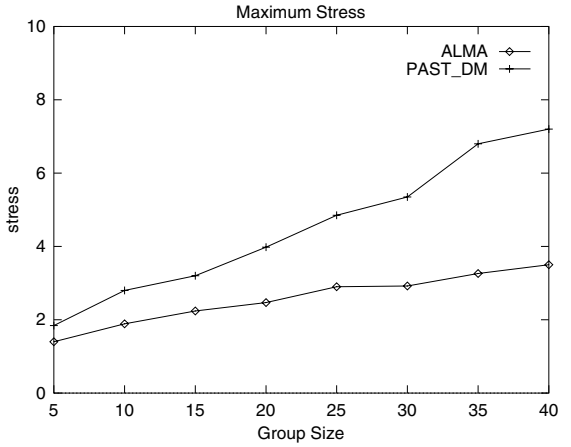

Fig. 7. Series I: maximum stress versus the group size.

constraints. Consequently, the possibility of bottlenecks are much lower in ALMA than in PAST-DM.

We attribute this behavior to the fact that ALMA uses a dynamic metric, such as RTT, while PAST-DM uses a static metric such as hop count to estimate the goodness of a logical link. When nodes choose parents, or decide to reconfigure, ALMA is actually responding not only to path length but also to congestion. [3] Thus, a new node will be less likely to pick a closer but congested node (e.g. due too many children) for its parent, if it can find another less congested parent, even if the second candidate parent node is further away in terms of hop count.

ALMA achieves a much better goodput as compared to PAST-DM for both TCP and UDP: We plot the goodput of the two protocols versus the speed of the mobile nodes in Fig. 8. For PAST-DM an update period of 40 s was chosen. In the following experiments we set the maximum speed to be equal to the minimum speed for the reasons stated earlier. ALMA consistently outperforms

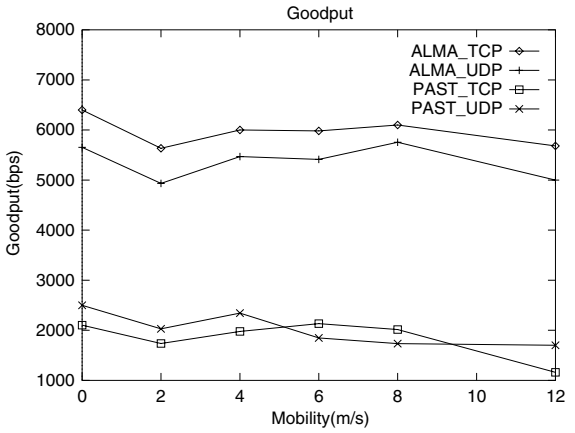

Fig. 8. Series I: goodput versus speed.

PAST-DM by almost three times in terms of the achieved goodput. We attribute this large difference to the cost efficiency of the tree, and the reduced contention that ALMA provides. When the update period was reduced to 20 s, we find that the performance of PAST-DM improves a bit. However, ALMA still outperforms PAST-DM by about 40% in terms of the packet delivery ratio and more than twice in terms of goodput.

ALMA and guaranteed reliable packet delivery: Initially, we thought that using TCP on each logical link would guarantee the delivery of all packets. However, this is not true. Packet losses can occur when the child node switches from its cur-

[3] In PAST-DM, the problem of high logical degree is identified and discussed. Adding a constraint to the number of children that a node can adopt can improve the performance of PAST-DM, but this could lead to the need of wider searches for neighbors and thus to higher overhead. In any case, this option was not further explored in [7].

rent parent to a new parent, and the new parent has already forwarded the packets that the new child still expects.

In Fig. 9, we plot the packet delivery ratio versus the speed of the nodes. In PAST-DM, when a reconfiguration occurs and a node switches from one parent to another, it might lose packets in the process. PAST-DM does not have any features that enable a node to recover these lost packets. Furthermore, when a node wishes to join the group, there is a delay incurred between when it sends a join message and when the source actually receives the link-state update from the nodes that receive the join message. In the interim, there is no reliable delivery of data to the new node. Furthermore, the loss of link-state updates could exacerbate this effect. In ALMA, reliable data delivery is not driven by a central directive from the source. Furthermore, group members cache packets in order to facilitate reliability during reconfigurations. These features allow ALMA to perform much better than PAST-DM in terms of the packet delivery ratio. Clearly, using caches provides a significant advantage, which we quantify at the end of this section. We conclude that caching is a critical component in ensuring reliability in ALMA; we examine the sensitivity of ALMA's performance to the size of the cache later.

In conclusion, these experiments show that ALMA performs better than PAST-DM in terms of the metrics chosen. In [7], the authors argue that PAST-DM performs better than AMRoute [9], which is also an application layer multicast protocol. Furthermore, ALMA here has not been carefully optimized: we have identified a number of functions and parameters that could help improve the performance. Some of these parameters we examine later in this section.

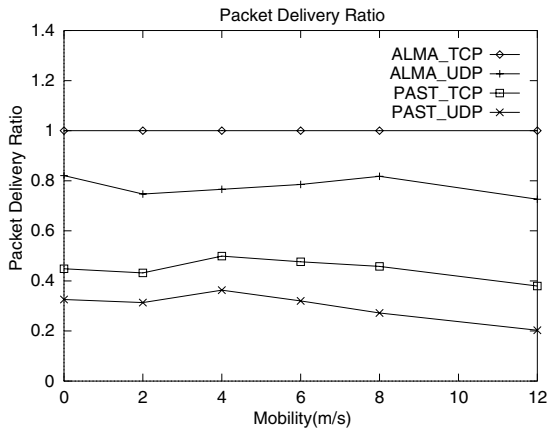

Fig. 9. Series I: packet delivery ratio versus speed.

Series II: ALMA performs favorably as compared with ODMRP: We compare the performance of application layer multicasting with that of a network layer multicast protocol. Naturally, we pick the two most promising protocols in each class: ALMA and ODMRP; the latter was shown to have a very competitive performance as compared with other network layer multicast protocols for ad hoc networks [5].

For fairness, we restrict our studies to unreliable data delivery and we use UDP for the logical links in ALMA. ODMRP does not support guaranteed packet delivery like most known network layer multicast protocols for ad hoc networks. We stress that it is an advantage of ALMA that it can exploit the reliability of TCP. In this series, UDP is used in all of the following experiments with the set-up of Scenario 2. We compare the performance of ALMA and ODMRP in terms of the packet delivery ratio and goodput. The results are shown in Figs. 10–13, which we discuss below.

For moderate group sizes, ALMA exhibits excellent goodput compared to ODMRP. A key parameter here is the **group density** of the multicast group which is the percentage of nodes that are multicast group members. [4] In Fig. 10, we plot the packet delivery ratio versus the speed of the nodes. We observe that for a moderate size group of 10 members (20% group density), ALMA outperforms ODMRP by about 15%. We attribute this to the ability of ALMA to avoid nodes that are highly congested by suitably reconfiguring the tree when the observed RTTs become large. ODMRP on the other hand, attempts to minimize the hop-count from the source to each receiver. This can

[4] We consider multicast groups of size 10, 20 and 30 which correspond to group density (ratio of the number of multicast group members to the total number of nodes in the network) of 20%, 40%, 60%.

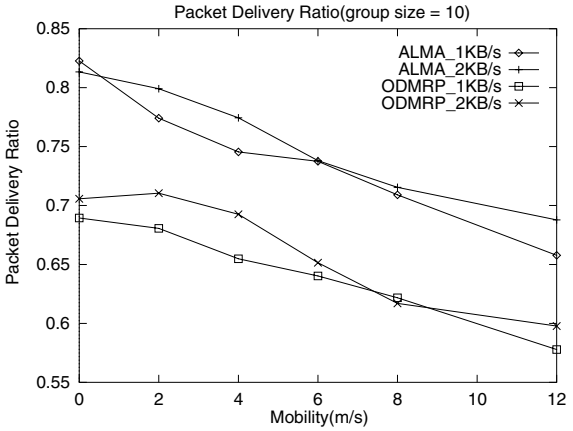

Fig. 10. Series II: packet delivery ratio versus speed (group size = 10).

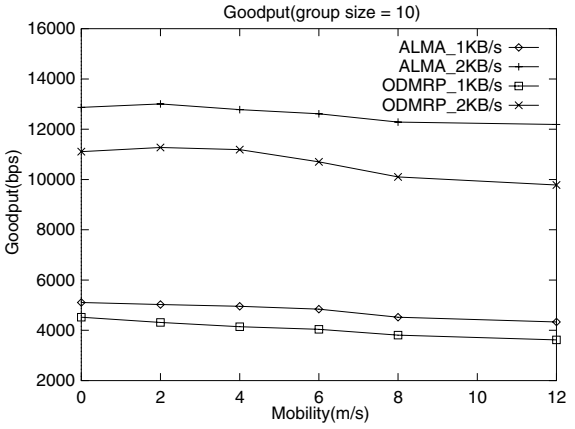

Fig. 12. Series II: goodput versus speed (group size = 10).

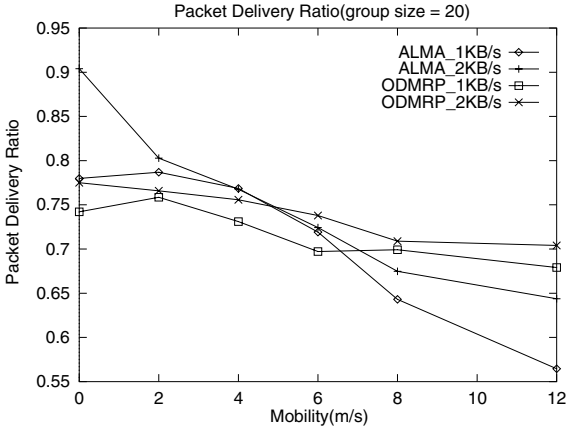

Fig. 11. Series II: packet delivery ratio versus speed (group size = 20).

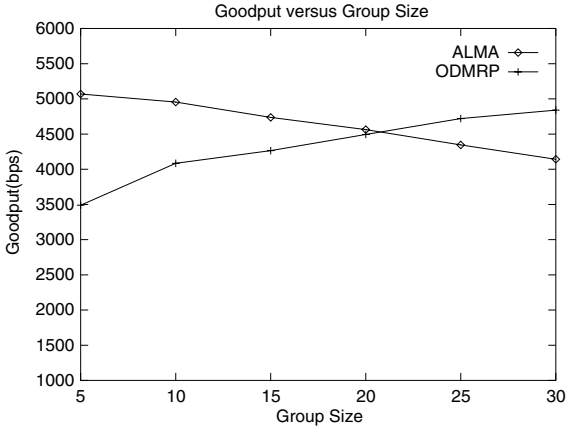

Fig. 13. Series II: goodput versus the group size (speed = 6 m/s).

cause congestion at certain bottleneck nodes that happen to be on the shortest path towards the source. We note a similar behavior when we plot the goodput versus the speed (Fig. 12). ALMA outperforms ODMRP by about 20% for this group size.

ALMA compares favorably with ODMRP for large group sizes: Next we repeat the experiments with a large group size (20 nodes). This corresponds to a group density of 40%. We observe from Fig. 11 that ALMA performs favorably with ODMRP. The performance for low mobilities is almost identical. The performance of ALMA degrades much more rapidly than ODMRP with group size since the number of multicast copies that traverse a single physical link now increases i.e., the stress increases. This in turn, increases congestion and causes the performance to degrade. Furthermore, with mobility, the performance worsens due to an increased frequency of reconfigurations which causes an increased number of control packets as well. However, in spite of these effects, the performance of ALMA is only worse than ODMRP by about 5% when the data rate is 2 kbps in terms of the packet delivery ratio. The goodput for ALMA and ODMRP are almost identical over

the range of speeds considered. The result is not presented here due to space limitations.

Effects of extremely large group sizes: Our simulation results show a further degradation of ALMA performance as we increase the group size further (a group density of 60% was considered). ODMRP on the other hand continued to perform well (packet delivery ratio of about 80%). With a 2 kbps source rate, ODMRP outperformed ALMA by about 18% when a speed of 6 m/s was considered. The reasons for this degradation in the performance of ALMA were again due to an increased number of copies of multicast packets. Note however, that with these extremely large group sizes the multicasting to the group approaches the function of achieving a broadcast. Clearly, ODMRP is still a very good protocol under these scenarios. We plot the goodput achieved by ODMRP and ALMA versus the group size in Fig. 13. We see that ALMA outperforms ODMRP if the group densityis below 46% (group size of approximately 23). Beyond this, ODMRP outperforms ALMA. In the scenario in Fig. 13, all nodes move in accordance to the mobility model described earlier with a speed of 6 m/s.

In conclusion, we believe that ALMA performs well even when compared with ODMRP, one of the best network layer protocols. It is a viable candidate for deployment given that it is simple to deploy, can exploit the ability of the transport layer in terms of providing reliability, and can be made secure with relatively simpler mechanisms. It even performs very well, unless the group membership becomes extremely large. In such cases, a network layer protocol, in particular ODMRP seems to be a apt choice, if the performance in terms of goodput or packet delivery ratio is the only metric of interest.

Series III: Sensitivity of ALMA to system parameters: In these set of experiments, we examine the sensitivity of ALMA to various system parameters that have been chosen; specifically, we look at the quantum of multicast packets that a member has to cache at any given time so as to support seamless reconfigurations. We also investigate the appropriateness of using increases in RTT as an index to represent the need for reconfiguration.

Cache size: The packet delivery ratio of ALMA can be improved, if members cache the multicast packets. Upon reconfiguration, a child node might find that the new parent is ahead in terms of the multicast data transmission schedule as compared to the old parent. With a cache, the new parent can retransmit past packets to the new child. In absence of such cached packets, the child would be forced to maintain the long inefficient connection to the old parent. Clearly, the efficiency of the reconfiguration may be improved if the new parent were to already have these packets in its cache. We want to quantify the effects of varying the size of the cache on ALMA's performance. We use a random way point model with a maximum speed of 20 m/s to simulate mobility.

The simulation results are shown in Fig. 14, where we plot the fraction of requested packets due to reconfigurations that were recovered from the cache of the new parent. When the source transmits at a low data rate (1 kbps), with a cache of 40KB, the fraction of the packets that can be recovered from the cache upon reconfiguration is as much as 70%. However, when the source increases its data rate (4 kbps), the average number of missed packets tends to increase. Note that, for a fixed reconfiguration time, as the data rate increases, the number of missed packets during a reconfiguration increases. Now, in order to ensure that 70% of the missed packets may be available in

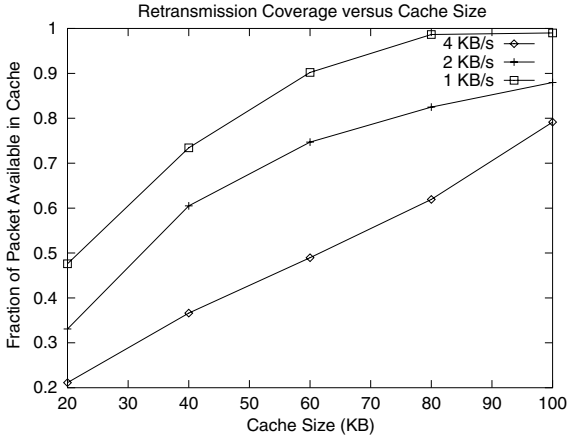

Fig. 14. Series III: fraction of packets available in cache versus group size.

the new parent's cache, the cache size has to be increased to about 90 kb. Thus, depending upon the data rate of the multicast session and the mobility, we need to choose the cache size appropriately for a desired level of performance. The overall observation is that a small cache in the order of Kbytes can provide significant performance improvement for streams in the order of kbps. As real world case study, recall that required bandwidth for voice is in the 8–64 kbps range, and thus a cache of a few hundreds Kbytes could be very beneficial.

RTT as a reasonable measure for reconfigurations: We evaluate the suitability of the use of the round trip time (RTT) to determine the quality of the path from a node to its parent. In ALMA, members have to rely on end-to-end measurements of the round trip time (RTT) to determine the quality of the path to one's parent. Each member measures the RTT experienced by the connection between itself and a candidate new parent when it perceives a need to reconfigure, as described earlier. In order to do so, it polls the new candidate parent for a short period of time and estimates the average RTT experienced (we simply use the `ping` function to perform this).

Our simulation results show that the RTT, thus obtained, is stable as compared to the RTT experienced at the transport layer and can reflect the application to application performance. Fig. 15 shows the observed RTT (at the application layer) of 5 separate static unicast TCP connections of different hop counts. The source polls the destination every second. Without contention or mobility, the RTT increases linearly with the hop count.

We want examine the sensitivity of the RTT to logical degree, which is a contributing factor to local congestion. In Fig. 16, we plot the mean RTT observed versus the logical degree of a node. To get interpretable results, we keep the hop count between the source and the destination fixed at 2 hops and the nodes static so as to eliminate the fluctuations due to hop count changes. We observe that at light loads (1 kbps, 2 kbps), for a fixed hop count, the RTT is insensitive to changes in logical degree. However, the higher load (4 kbps) increases the inherent local congestion, and we observe that the RTT increases at higher logical degrees. Thus, at these loads, if a node has a large number of children, changing its parent is not going to help in terms of reducing its observed RTT. Thus, with ALMA we impose a restriction on the number of children that a parent can adopt (to 4).

Fine tuning the reconfiguration thresholds. The reconfiguration thresholds dictate when a node should attempt to switch parents, and the extent of the scope of the search. We vary the set of RTT thresholds in ALMA with values shown in Table 1. We use three scenarios, P1, P2 and P3. We performed the simulation using multicast groups of 15. The source rate is 2 kbps. The choice

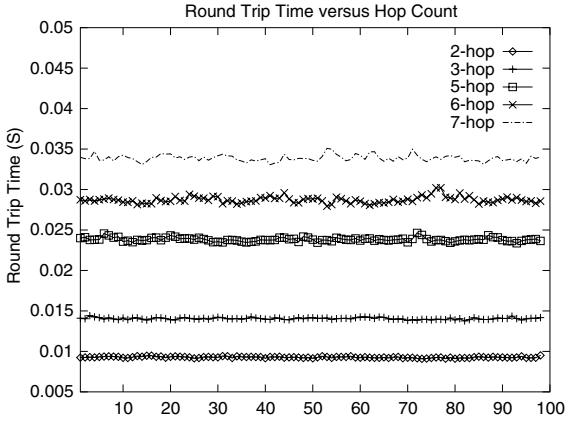

Fig. 15. Series III: round trip time versus hop count.

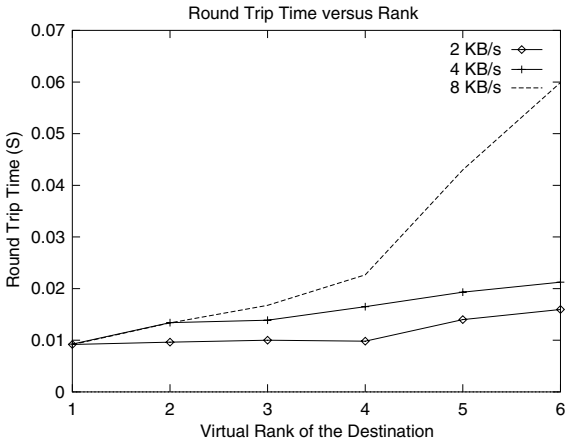

Fig. 16. Series III: round trip time versus logical degree (hop count = 2).

Table 1
The values of the threshold levels we examine in milliseconds

	P1	P2	P3
T_LEVEL1	20	30	40
T_LEVEL2	40	45	60
T_LEVEL3	60	65	80
T_LEVEL4	80	80	120

of parameters was based on the fact that with low traffic loads, the observed RTT was less than 30 ms. As we increased loads, the RTT could be as high as 100 ms. Note that if a node were to observe an RTT of 30 ms indicates (at light loads) that the parent is possibly 6 hops away and it would probably benefit if the node were to switch parents. The goodput achieved with the three chosen scenarios is shown in Fig. 17. As one might expect, with lower threshold levels (P1) one would expect the tree structure to be reconfigured more often and thus, the tree is more likely to be smaller in terms of cost. This in turn translates into a higher goodput as compared with the scenarios P2 and P3, wherein the reconfigurations occur less frequently and therefore have more costly transient trees. Note that ALMA with P1 outperforms ALMA with P3 by about 10% at a speed of 4 m/s. The average incurred overhead with P1 for this case was about 5% higher than that incurred with P3 in this scenario. The average incurred overhead is defined as the ratio of the amount of control data transmitted to the achieved goodput. The overall overhead incurred with P1 in this scenario is about 18%. As one might expect, with an increased rate of reconfigurations, a higher overhead is incurred. Table 2.

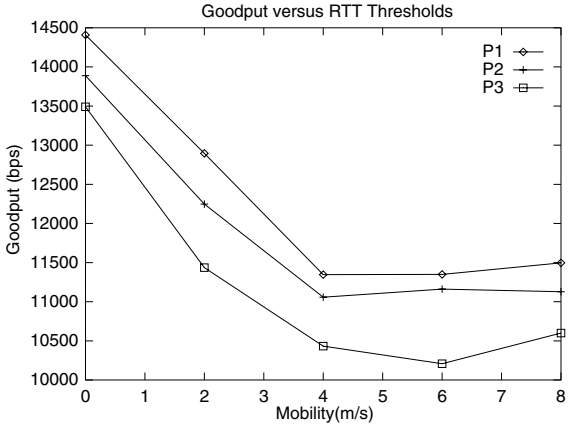

Fig. 17. Series III: goodput versus mobility speed.

Table 2
Series III: comparison of average overhead with different RTT threshold set

	P1	P2	P3
Average Overhead(%)	15.12	11.87	10.12

5. Conclusions

In this paper, we investigate the benefits of using application layer or overlay multicasting in ad hoc networks. We propose the Application Layer Multicast Algorithm (ALMA) and show that it is arguably the best application layer protocol in terms of most of the metrics that we consider in ad hoc networks. We also show that it performs favorably as compared with ODMRP which is, in turn, arguably one of the best network layer multicast protocols in ad hoc networks.

ALMA is based on constructing a logical tree in the ad hoc network. It is decentralized and is receiver driven. It has mechanisms that facilitate the reconfiguration of the logical tree in scenarios of mobility or congestion. These features especially benefit ALMA in terms of performance. In addition, ALMA offers all the benefits of an application layer multicast protocol i.e., simplicity of deployment, independence from lower layer protocols and an ability to exploit features that are available at lower layers such as reliability from TCP.

We perform extensive simulations to evaluate ALMA and to compare it with the best application layer multicast protocol for ad hoc networks, PAST-DM and ODMRP. We show that ALMA performs better than PAST-DM in terms of most of the metrics considered. We also show that in terms of UDP performance, ALMA performs favorably as compared with ODMRP for moderately sized or reasonably large multicast groups. However, beyond a certain group size, due to an increase in the number of multicast data copies in-

jected into the network, the performance of ALMA degrades and is worse than ODMRP. We conclude that ALMA, and in general application layer multicast, is a viable choice for multicasting in ad hoc networks if the application needs reliability or any other special requirements. Furthermore, it is a good choice if the group size is small even for unreliable multicast of UDP data.

Acknowledgments

The authors would like to thank P. Mohapatra and C. Gui of U.C. Davis for discussions on PAST-DM and for providing us with their simulation code.

References

[1] E. Royer, C.-K. Toh, A review of current routing protocols for ad hoc wireless networks, IEEE Personal Communications Magazine (1999).
[2] E. Royer, C.E. Perkins, Multicast operations of the ad-hoc on-demand distance vector routing protocol, in: Proceedings of ACM/IEEE MOBICOM'99, August 1999.
[3] J. Jannotti, et al., Overcast: Reliable multicasting with an overlay network, in: Proceedings of the Fourth Symposium on Operating Systems Design and Implementation, October 2000.
[4] D. Pendarakis, et al., ALMI: an application level multicast infrastructure, in: 3rd USENIX Symposium on Internet Technologies and Systems, March 2001.
[5] S.J. Lee, et al., On-demand multicast routing protocol in multihop wireless mobile networks, ACM/Baltzer Mobile Networks and Applications, special issue on Multipoint Communications in Wireless Mobile Networks 7 (6) (2002).
[6] M. Ge, M. Faloutsos, S.V. Krishnamurthy, Overlay multicasting for ad hoc networks, in: The Third Annual Mediterranean Ad Hoc Networking Workshop, Bodrum, Turkey, 2004.
[7] C. Gui, P. Mohapatra, Efficient overlay multicast for mobile ad hoc networks, in: Proceedings of IEEE WCNC 2003, March 2003.
[8] Y.H. Chu, et al., A case for end system multicast, in: Proceedings of ACM SIGMETRICS'00, June 2000.
[9] J. Xie, et al., AMRoute: ad hoc multicast routing protocol, ACM/Baltzer Mobile Networks and Applications, special issue on Multipoint Communications in Wireless Mobile Networks 7 (6) (2002).
[10] C.W. Wu, Y.C. Tay, AMRIS: A multicast protocol for ad hoc wireless networks, in: Proceedings of IEEE MILCOM '99, November 1999.
[11] J.J. Garcia-Luna-Aceves, et al., The core-assisted mesh protocol, IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks 17 (8) (1999).
[12] P. Francis, Yoid: Extending the internet multicast architecture, Technical report, AT&T Center for Internet Research at ICSI (ACIRI), April 2000.
[13] S. Banerjee, et al., Scalable application layer multicast, in: Proceedings of ACM SIGCOMM'02, Aug. 2002.
[14] S. Deering, et al., The PIM Architecture for Wide-Area Multicast Routing, IEEE/ACM Transactions on Networking (1996).
[15] K. Sundaresan, et al., ATP: A reliable transport protocol for ad-hoc networks, in: Proceedings of ACM MOBIHOC'03, June 2003.
[16] S.K. Das, et al., Dynamic core based multicast routing protocol, in: Proceedings of ACM MOBIHOC'02, Jun. 2002.
[17] C. Cordeiro, et al., Multicast over wireless mobile ad hoc networks: present and future directions, IEEE Network 17 (1) (2003).
[18] K. Chen, K. Nahrstedt, Effective location-guided tree construction algorithms for small group multicast in manet, in: Proceedings of IEEE INFOCOM'02, June 2002.
[19] S.J. Lee, W. Su, A performance comparison study of ad hoc wireless multicast protocols, in: Proceedings of IEEE INFOCOM'00, March 2000.
[20] S.J. Lee, et al., On-demand multicast routing protocol, in: Proceedings of IEEE WCNC '99, March 1999.
[21] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, J.-P. Sheu, The broadcast storm problem in a mobile ad hoc network, in: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, 1999, pp. 151–162.
[22] K. Obraczka, G. Tsudik, K. Viswanath, Pushing the limits of multicast in ad hoc networks, in: Proceedings of the The 21st International Conference on Distributed Computing Systems, 2001, p. 719.
[23] R. Bagrodia, et al., PARSEC: A parallel simulation environment for complex systems, IEEE Computer 31 (10) (1998).
[24] K. Chandran, et al., A Feedback-based scheme for improving tcp performance in ad hoc wireless networks, IEEE Personal Communications Magazine (2001).
[25] A. Fei, et al., Aggregated multicast with inter-group sharing, Networked Group Communications (NGC) (2001).
[26] D. Thaler, M. Handley, On the aggregatability of multicast forwarding state, in: Proceedings of IEEE INFOCOM00, Mar. 2000. Control in Ad Hoc Networks, Proceedings of ACM MOBICOM, 2002.
[27] R. Ramanathan, On the performance of ad hoc networks with beamforming antenanas, in: Proceedings of ACM MOBIHOC, 2001.
[28] J. Yoon, et al., Random waypoint considered harmful, IEEE INFOCOM (2003).
[29] B. Williams, T. Camp, Comparison of broadcasting techniques for mobile ad hoc networks, in: Proceedings

of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), 2002, pp. 194–205.

[30] C.K. Wong, et al., Secure group communications using key graphs, IEEE/ACM Transactions on Networking (TON) 8 (2000).

Min Ge is a Master's student in UC Riverside. Her interests in the networks area include multicast routing, reliable multicast protocols, and ad hoc networks.

Srikanth V. Krishnamurthy received his Ph.D degree in electrical and computer engineering from the University of California at San Diego in 1997. He received his B.E. (Hons.) degree in electrical and electronics engineering and the M.Sc. (Hons.) degree in physics with distinction from Birla Institute of Technology and Science, Pilani, India in 1992 and the Master of Applied Science degree in electrical and computer engineering from Concordia University, Montreal, Canada in 1994. From 1994 to 1995, he was a Graduate Research Assistant at the Center for Telecommunications Research, Columbia University, New York. From 1998 to 2000, he was a Research Staff Scientist at the Information Sciences Laboratory, HRL Laboratories, LLC, Malibu, CA. Currently, he is an Assistant Professor of Computer Science at the University of California, Riverside. His research interests span CDMA and TDMA technologies, medium access control protocols for satellite and wireless networks, routing and multicasting in wireless networks, power control, the use of smart antennas and security in wireless networks. He has been a PI or a project lead on projects from various DARPA programs including the Fault Tolerant Networks program, the Next Generation Internet program and the Small Unit Operations program. He is the recipient of the NSF CAREER Award from ANI in 2003. He has also co-edited the book ''Ad Hoc Networks: Technologies and Protocols'' published by Springer Verlag in 2004. He has served on the program committees of INFOCOM, MOBIHOC and ICC and is a area editor for ACM MC2R.

Michalis Faloutsos is a faculty member at the Computer Science Department in the University of California, Riverside. He got his bachelor's degree at the National Technical University of Athens and his M.Sc. and Ph.D. at the University of Toronto. His interests include, Internet protocols and measurements, multicasting, cellular and ad-hoc networks. With his two brothers, he co-authored the paper on powerlaws of the Internet topology (SIGCOMM'99), which is in the top 15 most cited papers of 1999. His work has been supported by several NSF and DAPRA grants, including the prestigious NSF CAREER award. He is actively involved in the community as a reviewer and a TPC member in many conferences and journals.