Application-Layer Multicast in MANETs: To Broadcast or not to Broadcast?

Peter Baumung
Institute of Telematics
Universität Karlsruhe (TH)
eMail: baumung@tm.uka.de

***Abstract*—Mobile ad-hoc networks allow wireless devices to freely communicate without the need of any fixed infrastructure. While many applications rely on group communication, providing a multicast service on the network layer turns out to be difficult because of diverging application requirements. Research in the past year has thus focused on application-layer multicast protocols: These can easily be deployed among group members and flexibly be customized to meet an application's requirements. As group members handle packet duplication they are required to repeatedly access the medium for packet forwarding. Especially in areas of increased group member density this process can result in heavy performance degradation: Indeed, as in this area the wireless medium will be accessed respectively often for forwarding one single packet, the achievable multicast throughput will drop. Broadcast transmissions can improve the situation since one single medium access will forward a data packet to an arbitrary number of group members located within transmission range. With common MAC layers such as IEEE 802.11, broadcasts, however, are not covered by retransmissions and thus show to be more prone to packet errors than unicasts. Depending on the nature of emitted traffic and a multicast group's size, we in this paper analyze in which situations broadcasting data pays for application-layer multicast protocols and in which situations it hurts.**

I. Introduction

Mobile ad-hoc networks (MANETs) consist of mobile devices, that communicate via the wireless medium without any fixed infrastructure. While two devices located in one another's transmission range communicate directly, intermediate devices bridge distances between farther nodes. As multi-hop communication is thus enabled, a complex and potentially highly dynamic wireless network arises. Here, many application share their need for multicast communication. Indeed, cooperating and thus communicating groups will be present in the majority of educational, touristic, rescuing or military scenarios.

Depending on the actual application scenario, a multicast protocol will be confronted with heavily diverging requirements. Operating multicast protocols on the network layer of wireless devices thus shows to be hardly feasible, since these approaches require a protocol's network-wide deployment. Consequently, current research increasingly pushes multicast functionality to the application-layer of wireless devices: Here, the tasks of packet duplication, packet forwarding and group management are handled by the group's members themselves and not by the underlying network. As a result, application-layer multicast protocols need only to be deployed among a multicast group's members, where they can easily be tuned to the application actually used.

For organizing packet forwarding in an efficient way, application-layer multicast protocols use a so-called *overlay network*. The latter is set up by interconnecting group members using (potentially multi-hop) unicast transport links. While, depending on a protocol's mechanisms, overlays show different topologies and integrate varying routing strategies, they all use unicast messages for disseminating data packets between group members. Wherever an increased number of group members gather, this forwarding scheme will cause the achievable multicast throughput to drop. Indeed, as can be seen in figure 1.a), forwarding one single packet within such an agglomeration (or *cluster*) requires the medium to be accessed once for each group member the packet is forwarded to. In other words: The more group members are located within a small area, the more bandwidth is required for forwarding one single multicast packet inside this area. Since in MANETs bandwidth is a very scarce resource, such clusters of group members thus quickly become bottle necks regarding the achievable multicast throughput.

This situation can be improved by making use of the wireless medium's (semi-)broadcast capability: Indeed, a broadcasted packet requires only one medium access, but can be received by a theoretically arbitrary number of devices located within transmission range. By organizing data forwarding as shown in figure 1.b), the overhead of packet forwarding inside a cluster could be kept at a constant level, regardless of the number of group members located inside the cluster. As shown by the figure, a multicast packet is received by group member A via a standard (multi-hop) unicast transport link. A, which is part of the cluster, broadcasts the received packet and thus forwards it to all other group members inside the cluster. Afterwards, A relays the packet to some more distant group members using a standard (multi-hop) unicast transport link.

Although it reduces the overhead for packet forwarding, broadcasting data packets, on the downside, happens to be more prone to packet errors, e.g. because of radio interference. When looking at the most widely adopted MAC for MANETs, i.e. IEEE 802.11, broadcasts, in contrary to unicasts, are not covered by packet retransmissions. As a consequence, one single collision on the medium suffices for causing a broadcast packet to be lost, while in case of unicasts up to seven retransmissions are performed. Whether broadcasting

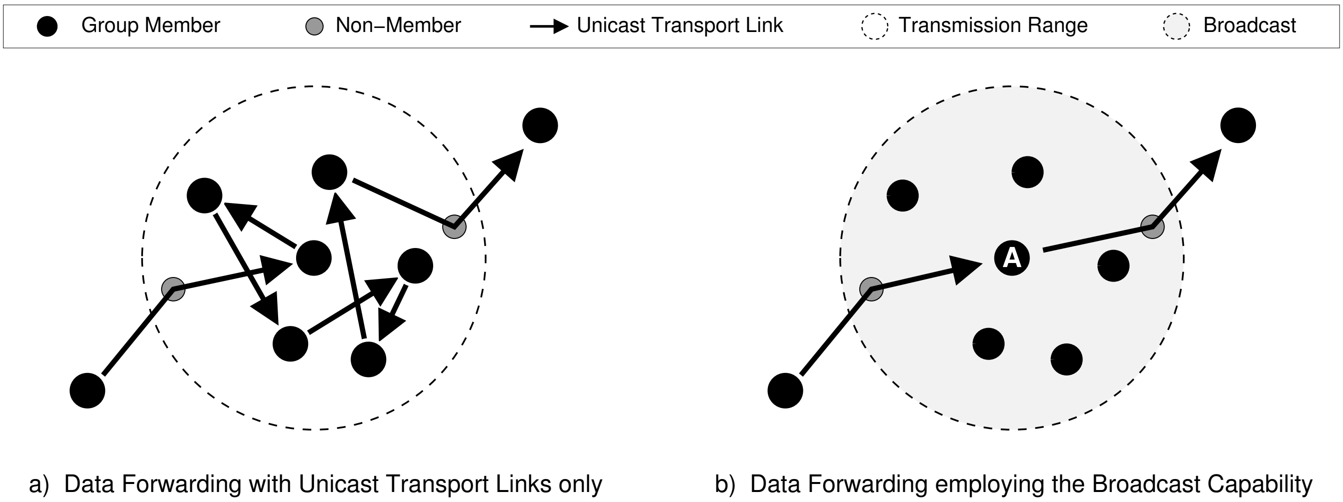

Group Member
Non−Member
Unicast Transport Link
Transmission Range
Broadcast
A
a) Data Forwarding with Unicast Transport Links only
b) Data Forwarding employing the Broadcast Capability

Fig. 1. Data Forwarding with and without using the Wireless Medium's Broadcast Capability.

data packets pays or whether it hurts mainly depends on the nature of the emitted traffic and of a multicast group's size.

We in this paper evaluate several application-layer multicast protocols using different application scenarios. By optionally extending the protocols with a generic mechanism that transparently adds broadcast data delivery similar to figure 1.b) to an overlay topology, we obtain direct comparisons how broadcasting data affects communication.

The remainder of this paper is organized as follows: We in section II first present our mechanism of *Local Broadcast Clusters*, which enables highly efficient broadcast data delivery for arbitrary overlay topologies. In section III we briefly outline the application-layer multicast protocols evaluated in this paper and highlight their pros and cons. While section IV contains the actual evaluations, section V concludes the paper by giving a short summary.

II. Local Broadcast Clustering

Local Broadcast Clusters (LBCs), as seen in figure 2, enhance the scalability of application-layer multicast protocols by making use of the wireless medium's broadcast capability. A LBC exists around every member of the multicast group that has joined the overlay. Additionally to all operations usually associated with the overlay's maintenance, the overlay nodes periodically broadcast dedicated heartbeat messages signaling the LBC's presence to nearby group members. By doing so, the overlay node becomes *leader of its LBC*.

A LBC's dimension is limited by the transmission range of its leader. Members of the multicast group that are located inside a LBC (i.e. within transmission range of an overlay node) do *not join the overlay* as long as they notice a LBC leader's presence through received heartbeat messages. Instead, these nodes join the nearby LBC and by this become *locally joined nodes*. In order to reduce total overhead required for the multicast group's management, locally joined nodes do not exchange any control flow information with members of the overlay.

As it might be located within transmission range of different overlay nodes, a locally joined node can receive heartbeats from multiple LBC leaders. This situation, which is called *Overlapping LBCs*, is exemplarily depicted on the right of figure 2. Using information about the loss rate of heartbeats, the locally joined node then computes the best quality leader and assigns itself to the respective LBC. Using the computed quality, locally joined nodes also detect the loss of LBC leaders. As soon as a LBC leader's quality drops below a certain threshold, it is declared as lost: We, here, speak of a *LBC Loss*, visible on the left of figure 2. Such events usually occur when a locally joined node moves out of its LBC leader's transmission range. In order to receive further multicast data and participate in the multicast group's activities, the locally joined node is required to join the overlay. It thereby becomes leader of its own, newly created LBC.

In analogy to the creation of a LBC by a specific node joining the overlay, a LBC's dissolution can be defined by letting the respective leader leave the overlay. Such a mechanism is required to achieve the most effective use of the wireless medium. LBCs indeed become redundant as soon as two LBC leaders find themselves within one another's transmission range: We, here, speak of a *LBC Collision*, visible on the bottom right of figure 2. This is avoided by letting a node retire from the overlay after having received a certain number of heartbeats from a nearby LBC leader with a higher IP address. The retiring node thereby abandons its own LBC and joins the other overlay node's LBC. Obviously, group members that had locally joined the former LBC now either need to assign themselves to another nearby LBC, or, if no more LBC leaders are know, are required to join the overlay in order to further participate in the group's activities.

According to the above definition of LBCs, a specific member of the multicast group finds itself in one of two possible states during its entire group membership: It has joined either the overlay, or a LBC. While transitions between both states are possible at every time, a group member handles

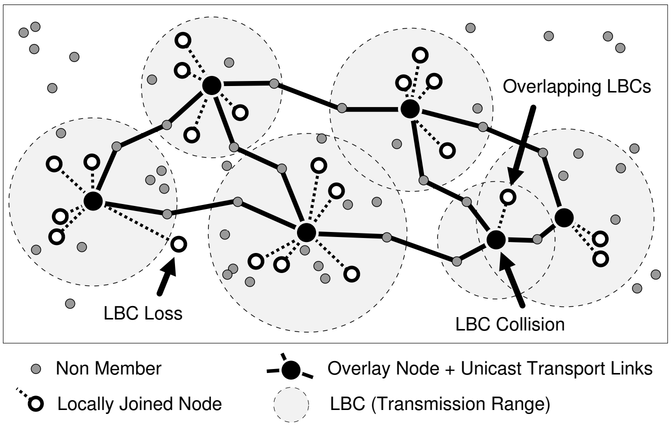


Fig. 2. Overlay Topology extended by Local Broadcast Clusters (LBCs).

the sending and forwarding of multicast data depending on its current state. Overlay nodes send and forward data through unicast tunnels to nearby overlay nodes. This forwarding is done according to the overlay network's routing protocol. Additionally, overlay nodes broadcast data to their locally joined nodes. They hereby reach any of the multicast group's members within their transmission range but require to access the medium only once. A locally joined node which acts as a multicast source simply unicasts its data to its LBC leader which then takes care of both data forwarding to overlay nodes according to the overlay's routing protocol and broadcasting to other locally joined nodes.

Figure 2 shows the resulting network topology which consists of LBC leaders connected through the overlay's unicast transport links. As can be seen, LBCs might be of different size depending on the transmission range of overlay nodes. The key benefits of LBCs, on the one hand, are, that locally joined nodes do not join the overlay and thus reduce the latter's size as well as the control flow required for its maintenance. On the other hand, overhead involved in data forwarding is drastically reduced. Furthermore, the *total overhead* (control flow information and data forwarding) is now limited by the area group members occupy and not by the number of group members. Indeed, using LBCs the overlay becomes comparable to a kind of "application-layer MANET infrastructure" which is used for data forwarding across multi-hop distances. This infrastructure is highly flexible and can be extended or released depending on a multicast group's needs.

III. Application-Layer Multicast Protocols

In this section we outline the application-layer multicast protocols evaluated and compared in this paper. As we cannot in detail discuss each protocol's features, we limit ourselves to briefly discussing a overlay topology's aspects and highlighting each protocol's pros and cons. Please refer to the given references for more information about the single protocols.

A. TrAM

TrAM (*Tree-based overlay-Architecture for Manets*) features a very lightweight overlay topology: As it avoids unnecessary unicast transport links, it connects group members using a simple and flexible tree structure [1]. For setting up the overlay topology and for the latter's adaption to the physical network, TrAM relies on a highly efficient mechanism for locating nearby group members using limited network flooding. Since the protocol has been designed for reducing potential cross-layer effects (such as route discoveries performed by reactive routing protocols on the network layer), it comes with a very low protocol overhead. On the downside however, TrAM's tree topology will have to be shared by a potentially high number of multicast sources. Depending on the number of active multicast sources and emitted traffic, packet collisions may thus become frequent.

Pros: *X-Layer Optimized Overhead, Topology Adaption*
Cons: *One Single (Shared) Multicast Tree*

B. Narada

Although Narada has initially been developed for the fixed Internet [2], its lightweight mesh topology and its topology adaption mechanisms make it interesting for operation in MANETs. Indeed, as group members can potentially set up an overlay link to any other group member, Narada's topology shows a high degree of flexibility: Narada is, thus, supposed to cope well with node mobility in MANETs. On the downside however, the protocol periodically probes the link quality to *all* other group members. In the context of cross-layer effects, this results in additional overhead. The latter will turn out to be unnecessary in most cases, since probing very distant group members most likely will not lead to topology optimizations.

Pros: *Lightweight, Topology Adaption*
Cons: *X-Layer Effects*

C. NICE

Similarly to Narada, NICE also has been developed for the fixed Internet [3]. It has been designed with respect to scalability in terms of a multicast group's size: As a result the overhead involved in state information maintained in group members is constant for the average group member and logarithmic in the worst case. While this is achieved by organizing group members in fully-meshed and hierarchical clusters, the low overhead makes the protocol interesting for operation in MANETs. Here, however, the protocol is opposed to node mobility and thus frequently changing link quality between group members. Although NICE includes mechanisms for topology adaption, its seems questionable whether the topology's degree of flexibility is sufficient for MANETs. In addition, the fully-meshed nature of clusters can lead to undesired cross-layer effects such as frequent route discoveries on the network layer.

Pros: *Scalability*
Cons: *Questionable Topology Adaption, X-Layer Effects*

D. PAST-DM

PAST-DM is a protocol designed for providing application-layer multicast in MANETs [4]. The protocol features a mesh-like overlay topology which is periodically adapted to the physical network and thus copes with node mobility. PAST-DM specifically was developed for *small* groups for which it features a simple source-routing approach: The latter for each multicast source computes steiner trees on top of link-state information which is periodically exchanged between neighboring overlay nodes. While the cost of multicast trees is kept low, the protocol overhead resulting from exchanged link-state information degrades the protocol's scalability in terms of a multicast group's size.

Pros: *Topology Adaption, Reduced Cost of Multicast Trees*
Cons: *Scalability, Protocol Overhead*

*E. n * Unicast*

While not precisely an established protocol, the simplest way of multicasting packets on the application-layer is to let the multicast source unicast them to each group member. While the approach seems justified for small groups and low traffic, its performance is likely to drop with an increasing number of group members and traffic volume: Indeed, with an increasing number of unicasts necessary in each multicast source's vicinity, i.e. an increasing link stress around each multicast source, the wireless medium's capacity is expected to quickly exhaust. In addition, each group member performs packet duplication only for its own multicast packets. As a consequence, unicast transport links show an increased length, which results in more frequent packet drops and increased network load.

Pros: *Simple*
Cons: *Scalability, Link Stress, No Topology Adaption*

IV. Evaluation

For evaluating the application-layer multicast protocols in combination with LBCs, we use the *Modular Architecture for Application-Layer Multicast* (the *MAAM* architecture, [5], [6]). By, on the one hand, implementing the respective protocols as modules within the MAAM, and by, on the other hand, operating the MAAM itself on top of the network simulator GloMoSim, we oppose the protocols to highly identic situations and, thus, obtain well comparable results. Before proceeding to the evaluation of two different application scenarios, i.e. a CBR and a chat application, we in the following sections first present the simulator's configuration as well as simulated scenarios and define measured values used for protocol rating.

A. Node Placement and Simulator Configuration

For evaluations we in this work focus on node mobility as found in pedestrian areas. We model the MANET using a square area of *1000 x 1000* m^2. Inside this area, we let 100 nodes roam according to the *Random Direction Mobility* (*RDM*) model, i.e. each node chooses a random direction in which it moves for a certain time, potentially bumping of the area's limits. These nodes form a "base network" which assures connectivity for group members. While not participating in the multicast group's activities, nodes from the base network move at a speed randomly chosen between *1.25* and *1.75* $^m/_s$.

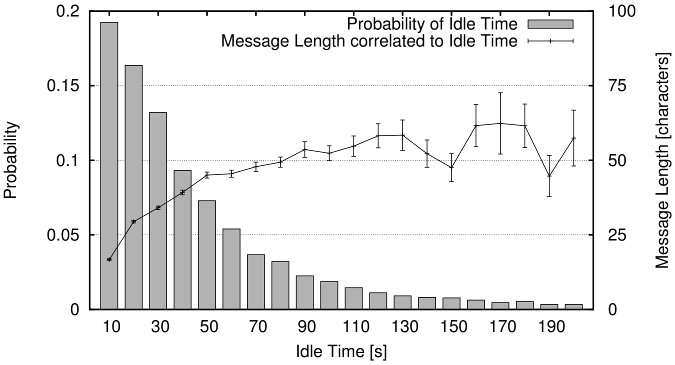


Fig. 3. Distribution Functions for Artificially Generated Chat Traffic.

On top of the base network we, depending on the scenario, add a certain number of group members nodes. These move at a speed randomly chosen between *1.0* and *1.5* $^m/_s$. We for movements of group member nodes use a clustered variant of the RDM model, which groups the respective nodes in small clusters with up to *5* nodes. Considering scenarios with only few group members, we take care of not placing all member nodes inside the same cluster: This is achieved by generating node placements and movements for a total of *50* group members, from which the required number of group members are randomly picked.

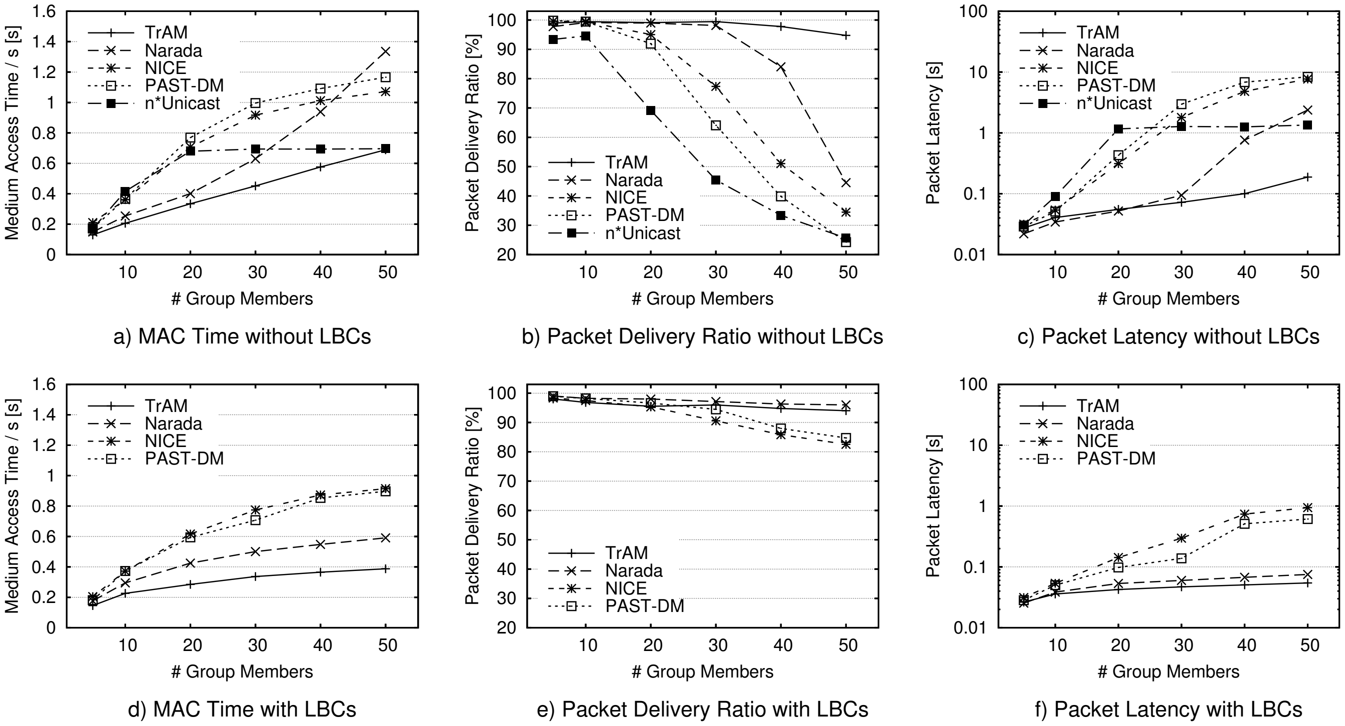
All nodes in the MANET maintain their direction for a period of time randomly chosen between *30* and *180s*. Before choosing their next direction group members make a pause of up to *60s*. Base network nodes only pause for up to *5s*.

Every node uses IEEE 802.11b as MAC with a bandwidth of *2* $^{Mbit}/_s$. Each node has a transmission range of *175m*, which results in an interference range of *358m*. Application-layer data is sent using UDP packets, which are routed using AODV [7] on the network layer.

B. Application Scenarios and Measured Values

In order to rate an overlay's performance, we consider it as important, not to restrain evaluations to one single application scenario. We thus compare the performance of application-layer multicast protocols in the context of a single-source CBR and a multi-source chat application. While the emission of a CBR stream is simple task, the realistic simulation of a chat application requires some more complex modeling. We therefore monitored conversations in typical chat rooms for a couple of days. The "average user behavior" resulting from the obtained measurements is shown in figure 3. The *x* axis shows a user's *idle time*, which is the amount of time that elapses between two consecutive messages sent by the same user. While on the left *y* axis the diagram shows the probability of a specific idle time, the right *y* axis denotes the message length the user has generated within the respective idle time. Note that raw measurements show a much finer granularity and were condensed only for the sake of clarity.

For rating application-layer multicast protocols, we, furthermore, do not restrain ourselves to measuring standard values,



a) MAC Time without LBCs

b) Packet Delivery Ratio without LBCs

c) Packet Latency without LBCs

d) MAC Time with LBCs

e) Packet Delivery Ratio with LBCs

f) Packet Latency with LBCs

Fig. 4. Performance of a Single-Source CBR Application with and without Local Broadcast Clustering applied to the Overlay Topologies.

such as achieved delivery ratios and the observed latencies. To rate overlay topologies with respect to involved cross-layer effects, we additionally measure the *Medium Access Time* (*MAC Time*). The latter sums the durations of all medium accesses network-wide for each second of simulation time. This metric thus not only covers traffic emitted on the application-layer, but also monitors overhead generated by network routing (e.g. route discoveries) and the MAC layer (e.g. retransmissions of unicast packets because of radio interference). A brief discussion of how to interpret measured MAC time values can be found in the appendix A at the end of this document.

All diagrams were obtained by averaging the results of *20* node distribution scenarios, each being run with a different seed value for random number generation.

C. Single-Source CBR Application

For evaluating a CBR application, we let a single source multicast data packets with a size of *512 bytes*. Although we conducted numerous experiments, we in this work limit presented results to a data rate of $3\ ^{KByte}/_{s}$ (i.e. $6\ ^{1}/_{s}$ *x 512 Byte*) since it shows differences in protocol scalability best. We, instead, vary the number of members joining the multicast group and thus investigate protocol scalability in terms of group size. The results of simulation experiments are shown in figure 4 and discussed for the single protocols in the following.

Note that for latencies, we adopt a logarithmic scale on the *y* axis, because of the exponentially growing nature of the back-off window used for retransmissions on the MAC layer.

- *TrAM:* As can be seen, the MAC time required by the TrAM protocol, c.f. figures 4.a) and 4.d), is by far the lowest of all protocols. This results from TrAM's very lightweight tree topology, which, on the one hand, avoids unnecessary unicast transport links and, on the other hand, uses cross-layer optimized protocol mechanisms for topology maintenance. As a result, the MAC time required by the raw protocol grows linearly, figure 4.a), with the depicted number of group members. When combined with LBCs, figure 4.d), the plotted curve quickly flattens: Indeed, group members can locally join LBCs and, hence, do neither introduce any control flow, nor require additional medium accesses for data forwarding. Additionally, the more group members actually join the overlay, the more of the simulated area is covered by LBCs. With other words: The more group members are required to join the overlay, the more likely it gets for further joining group members to join an LBC. As a consequence, for the 50 group member scenario, TrAM's MAC time can be reduced by about *43%* when extending the protocol with LBCs.

The major benefit of saving bandwidth when using a lightweight and cross-layer optimized topology is, that the saved bandwidth can be used for transmitting additional application data. This become visible in figure 4.b), in which we plot delivery ratios achieved for data packets. As can be seen, TrAM, in its raw version, achieves a nearly flawless (*100%*) delivery for multicast groups with up to 30 group members, and shows only a slight performance drop by *5%* for the 50 group member scenario. When combining TrAM with LBCs, c.f. figure 4.e), one can notice the downside of broadcasting data, i.e. the missing MAC retransmissions in case of radio interference. Indeed, in scenarios with up to 40 group members, TrAM's performance is about *2 - 3%* lower than in its raw version. For the 50 group member scenario, the achieved performance remains the same, regardless of LBCs being used or not.

The figures 4.c) and 4.f) plot the latencies of data packets observed during simulations. Measurements for TrAM's raw version, figure 4.c), remain below *70 ms* for multicast groups with up to 30 members. They however start rising to *100 ms* and *200 ms* for 40 and 50 members scenarios respectively. As the medium gets more and more loaded, c.f. figure 4.a), packet collisions get more frequent. Consequently, the MAC layer's back-off windows start growing, resulting in increasing latencies. Combining TrAM with LBCs, figure 4.f), brings a relief: Indeed, as the overhead for packet forwarding is reduced, the medium is loaded less, making packet collisions rare. Latencies can overall be reduced and do not exceed *60 ms*, even for scenarios with 50 group members.

- *Narada:* For multicast scenarios with up to 20 group members, the performance of the raw Narada, c.f. figure 4.a), is comparable to TrAM: Indeed, with only few group members, the overhead involved in Narada's mesh remains low enough to compete with TrAM simple tree topology. From 30 group members on, however, MAC times start rising disproportionately: Since the mesh overlay counts more and more transport links, the overhead for topology maintenance, for the involved cross-layer effects and for data forwarding increases. For 40 and 50 group members the MAC time reaches *0.95* $^{s}/_{s}$ and *1.35* $^{s}/_{s}$ respectively, coming close to an overloaded medium (c.f. appendix A). As enabling LBCs reduces the number of overlay nodes, and thus the total overhead, results depicted in figure 4.d) show vast improvements for scenarios with a high member count: MAC time, e.g., can be reduced by *55%* in the 50 group member scenario.

The frequently accessed medium observed in scenarios with high group member count increasingly leads to dropped packets for Narada's raw version. The resulting packet delivery ratios are plotted in figure 4.b): Here, Narada achieves a ratio of *85%* for 40 group members and of only *45%* for 50 group members, heavily restraining the protocol's usefulness in the given application scenario. When combining Narada with LBCs, however, results can be drastically improved: Indeed, figure 4.e) shows that now Narada even slightly outperforms TrAM for all depicted group member scenarios.

Activating LBCs also positively affects packet latencies. As can be seen from figure 4.c) latencies abruptly rise for Narada's raw version operated in scenarios with more than 30 group members: Here, latencies reach *0.8 s* for 40 group members, and climb to about *3 s* for 50 group members. While this behavior can be ascribed to the heavily loaded medium and, hence, to the overall growing back-off windows, combining Narada with LBCs brings the expected improvement: Indeed, figure 4.f) shows that latencies can successfully be lowered to about *70 ms* for 50 group member scenarios.

- *NICE:* While NICE achieves scalability in terms of state information kept in single group members, the protocol's overhead in terms of network load appears far less scalable: For scenarios with 30 or more group members, NICE in figure 4.a) shows a MAC time of $\approx$ *1* $^{s}/_{s}$, indicating a highly loaded medium. This, on the one hand, can be ascribed to NICE's overlay topology, consisting of fully-meshed clusters. Especially in combination with the reactive AODV, NICE's topology involves a lot of cross-layer effects, i.e. route discoveries on the network layer. On the other hand, the duplication of multicast packets inside a NICE cluster is handled by its leader only, resulting in an increased link stress around the respective group member. When combining NICE with LBCs, c.f. figure 4.d), the MAC time drops below *1* $^{s}/_{s}$ for the scenarios with high member count. Note, however, that the drop's extent is rather small, i.e. *18%*, compared to Narada and TrAM.

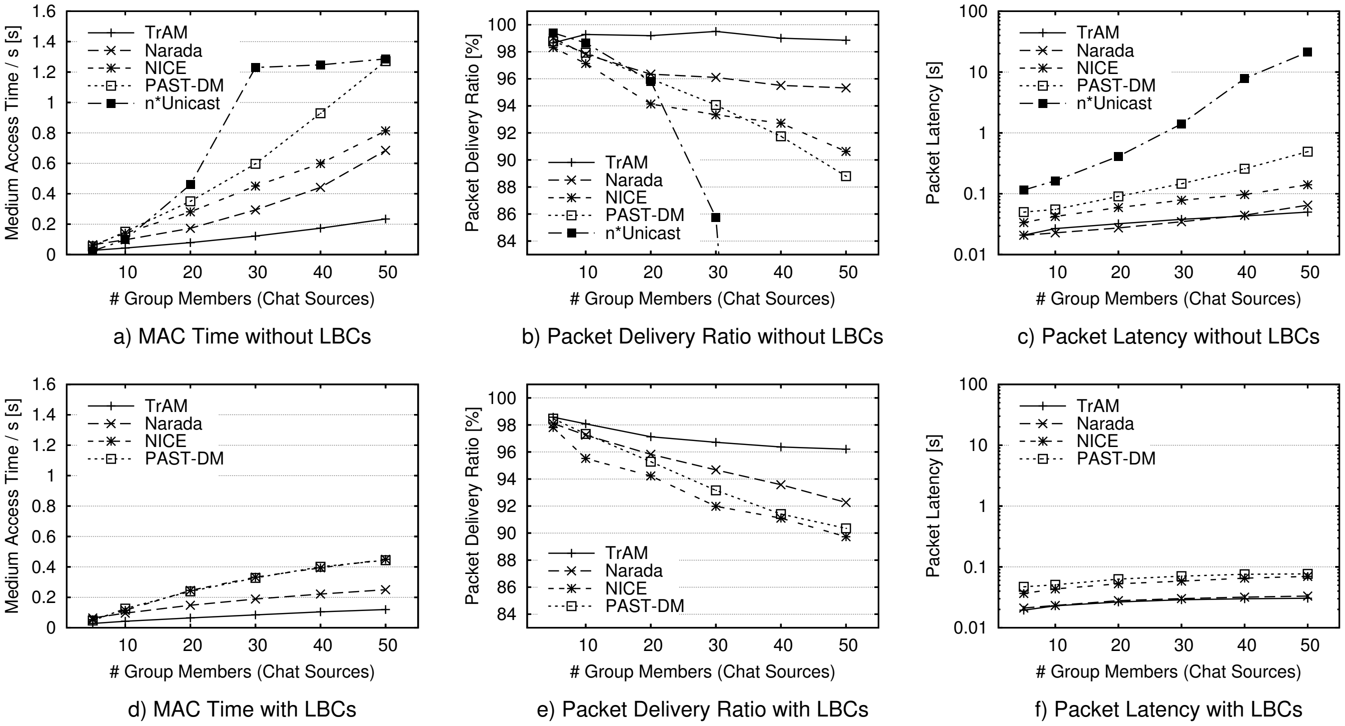
Considering NICE's increased MAC time for 30 and more group members, the expected drop of packet delivery ratios is shown in figure 4.b): For 40 and 50 group members, achieved ratios (*51%* and *35%* respectively) heavily restrain NICE's usefulness for the given CBR application. As can be seen from figure 4.e), combining the protocol with LBCs, however, brings NICE's performance back to an acceptable level: Here, even for scenarios with 50 group members, delivery ratios exceed *80%*.

Since a heavily loaded medium also means increased latencies, NICE's data delivery is supposed to get slower for scenarios with 30 or more group members. Indeed, figure 4.c) shows quickly growing latencies, which rise to about *2 s* for medium sized scenarios. With more members joining the group, latencies further grow, attaining about *8 s* in scenarios with 50 group members. Again, combining the protocol with LBCs, c.f. figure 4.f), drastically pushes NICE's performance: Latencies for 50 group members are e.g. diminished by a factor of *10*.

- *PAST-DM:* Since this protocol has specifically been developed for supporting small multicast groups only, one expects its performance to drop beyond a certain number of group members. Figure 4.a) shows that this point is reached with about 30 group members, since from this point the medium closely reaches saturation. Indeed, the link states periodically exchanged between group members results in a high volume of control flow information. As can be seen from figure 4.d), combining PAST-DM with LBCs pushes MAC time back below *1* $^{s}/_{s}$.

As can be expected and as shown in figure 4.b), packet delivery ratios for PAST-DM heavily start dropping for 30 and more group members. In analogy to NICE, PAST-DM's usefulness in the given application scenario seems questionable for 40 or more group members. Interestingly, although raw PAST-DM shows a performance worse than raw NICE, combining both protocols with LBCs turns the tables: Indeed, as shown by figure 4.e), PAST-DM now outperforms NICE in terms of achieved delivery ratios.

This also applies for achieved latencies: While in figure 4.c) raw PAST-DM performs somewhat worse than NICE, PAST-DM starts outperforming NICE in figure 4.f) when combining



a) MAC Time without LBCs

b) Packet Delivery Ratio without LBCs

c) Packet Latency without LBCs

d) MAC Time with LBCs

e) Packet Delivery Ratio with LBCs

f) Packet Latency with LBCs

Fig. 5. Performance of a Multi-Source Chat Application with and without Local Broadcast Clustering applied applied to the Overlay Topologies.

both with LBCs.

- *n * Unicast:* While it is a very simple protocol, n*Unicast comes with an overall low performance in the investigated CBR application. The MAC time shown in figure 4.a) quickly rises, but then suddenly flattens with 20 group members and remains at an almost constant level beyond this point. This behavior can be explained with the duplication of multicast packets: Indeed, the latter is handled by the multicast source *only*, which then *directly* unicasts the resulting packets to *all* recipients using a separate transport link for each recipient. Because of the large interference range, c.f. appendix A, a medium access in one transport link is likely to block medium accesses in almost all other transport links. This especially is true, since the application scenario features only one multicast source, thus causing all transport links to start from the same node inside the network. With 20 group members, the medium's capacity around the multicast source is exhausted. As a consequence, MAC time from this point remain on a constant level, regardless of further joining group members.

As the medium's capacity gets exhausted around the multicast source, the latter will not be able to process packets from the IP queue quickly enough. Since the CBR application inserts 6 packets per recipient and second in the queue, the queue's limited capacity (100 packets) is not sufficient, causing packets to be dropped: This can observed in figure 4.b).

Latencies, visible in figure 4.c), show the same characteristics as the MAC time: They quickly rise and stay at a constant level for 20 and more group members.

D. Multi-Source Chat Application

We, in this section, now oppose the different protocols with and without the LBC extension to the chat traffic as defined by the distribution functions in section IV-B. In analogy to the previously investigated CBR application, we vary the number of members joining the multicast group. Note, however, that for the chat application *each* group member acts as a multicast source, resulting in up to *50* chat sources in the respective scenarios. The results of simulation experiments are shown in figure 5 and discussed for the single protocols.

- *TrAM:* In analogy to the CBR application, TrAM shows a very high performance in the chat scenario. Measured MAC time, shown in figure 5.a), only grows very slowly with the number of active multicast sources. As visible in figure 5.d), enabling LBCs decreases MAC time by about *50%*.

LBCs however come at the cost of worsening packet delivery ratios. As can be seen in figure 5.b), raw TrAM achieves a delivery ratio of about *99%*, for all investigated group sizes. Activating LBCs results in dropping performance: Indeed, figure 5.e) shows that delivery ratios decrease with the group's size, attaining about *96%* for the 50 group member scenarios: As, here, more chat messages are generated, overall traffic rises, causing the fragile broadcasts to be increasingly destroyed by radio interference.

Considering latencies, TrAM achieves a very fast data delivery: As shown in figure 5.c), latencies range between *20* and *50ms*, depending on the number of group members. Extending TrAM with LBCs, c.f. figure 5.f), halves latencies.

- *Narada:* In terms of MAC time, measurements in figure

5.a) for Narada show the same disproportionate growth as in figure 4.a), which, again, can be ascribed to the protocol's mesh topology and its involved cross-layer effects. Using LBCs, c.f. figure 5.d), MAC time can be reduced by up to *65%* for the 50 group member scenarios.

In analogy to TrAM, combining Narada with LBCs causes a comparable drop of the achieved packet delivery ratios: While, in figure 5.b), they reach *95%* with the highest group member count, they drop to *92%* in figure 5.e).

As indicated by the figures 5.c) and 5.f), Narada is on a level about equal to TrAM in terms of achieved latencies. This is true regardless of extending the protocol with LBCs or not.

- *NICE & PAST-DM:* Especially when combined with LBCs, both protocols achieve a very similar performance in the context of a chat application: Indeed, as shown by the figures 5.d-f), measurement for both protocols are hardly distinguishable. Considering the raw protocols however, PAST-DM comes with a higher MAC time than NICE, c.f. figure 5.a), and the resulting higher latencies, c.f. figure 5.c).

- *n * Unicast:* When used inside a chat application, this protocol shows a reasonable performance for up to 30 group members. Beyond this point, especially packet delivery ratios get unacceptable. As a consequence we preferred to set the *y* axis' scale of figure 5.b) so that it allows a good differentiation of the *true* application-layer multicast protocols. The achieved packet delivery ratios for n*Unicast are *26.5%* for 40 group members and *10.1%* for the 50 group members respectively. Since each group member directly unicasts its chat messages to all other group members, bad results here can be explained by cross-layer effects, i.e. a massive amount of route discoveries on the network layer. Note that MAC time, c.f. figure 5.a), now only flattens beyond 30 group members: Since the scenario features *multiple* multicast sources which are distributed over the simulation area, medium accesses can partially be performed simultaneously, resulting in increased MAC time.

V. Conclusion

In this paper we analyzed a couple of application-layer multicast protocols (TrAM, Narada, NICE, PAST-DM, n*Unicast) in the context of different application scenarios, i.e. a CBR and a chat application. For rating a protocol, we chose to measure the network load (in terms of medium access time) actually involved by the protocol, its achieved data packet delivery ratios as well as the latencies of data packets. Since all protocols use standard unicast transport links for data forwarding, their performance is likely to drop when opposed to high traffic and an increased number of group members.

Using the LBC concept we presented in this paper, we offer a generic approach for extending arbitrary overlay topologies in a way that enables them to use the wireless medium's broadcast capability. LBCs thus can effectively avoid situations in which areas with increased group member density appear as bottlenecks. By opposing the LBC-extended application-layer multicast protocols to the same application scenarios as the raw protocols, we investigated whether broadcasting data pays or is likely to hurt communication.

Whether to use broadcast messages for data delivery eventually depends on the desired degree of reliability, on the traffic's volume and on potential restrictions considering latencies. For chat traffic, which demands high reliability and shows only low volume, broadcasting data rather hurts the application's performance, although latencies can be improved. CBR applications, however, can very well profit from data delivery using broadcasts: Indeed, depending on the exact volume of emitted traffic, packet delivery ratios can drastically be increased while latencies are lowered. Broadcasting data, thus, especially becomes interesting for applications such as VoIP or multi-player games, which require low latencies and can tolerate single packet losses.

Appendix A
Interpreting the Measurements of MAC Time

The most important feature to keep in mind when interpreting measured MAC time is, the transmission range of each node and the resulting interference range. In this work, we e.g. use a transmission range of *175m* which gives an interference range of *358m*. Assuming a node located in the center of the simulation area, i.e. at *(500m, 500m)*, accesses the medium, the latter will appear busy at any point located less than *358m* away from the accessing node. This area corresponds to a disc with a diameter of *716m*, thus making up about $402000m^2$ $\approx$ *40%* of the simulation area. It, hence, is not very likely to have two simultaneous and undisturbed medium accesses. Consequently, a measured MAC time of $\approx 1\ {}^{s}\!/\!{}_{s}$ means a heavily loaded medium. Values $> 1\ {}^{s}\!/\!{}_{s}$ are likely to represent simultaneous medium accesses, which can either be successful (if accessing nodes are located far enough from each other) or result in packet collisions.

On the other hand, since accessing the medium also is a highly energy consuming process, measuring the network-wide MAC time also allows a protocol's rating with respect to its energy consumption. Although interesting, we do not further investigate this aspect in this contribution.

References

[1] Peter Baumung, "TrAM: Cross-Layer Efficient Application-Layer Multicast in Mobile Ad-hoc Networks," in *Proceedings of The IEEE Wireless Communications and Networking Conference 2007 (WCNC 2007)*, Hong Kong, China, Mar 2007.

[2] Y. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," in *ACM SIGMETRICS 2000*, Santa Clara, California, USA, June 2000.

[3] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *ACM SIGCOMM*, Pittsburgh, PA, USA, June 2002.

[4] Chao Gui and Prasant Mohapatra, "Efficient overlay multicast for mobile ad hoc networks," in *The Wireless Communications and Networking Conference (WCNC)*, New Orleans, Louisiana, USA, Mar. 2003.

[5] Peter Baumung, "On the Modular Composition of Scalable Application-Layer Multicast Services for Mobile Ad-hoc Networks," in *Proceedings of The 2006 International Workshop on Wireless Ad-hoc and Sensor Networks (IWWAN 2006)*, New York, USA, Jun 2006.

[6] Peter Baumung et al., "The Modular Architecture for Application-Layer Multicast," http://maam.pcb-net.org, 2005.

[7] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir R. Das, "Ad hoc on-demand distance vector (AODV) routing," Feb. 2003.