Backpressure Multicast Congestion Control in Mobile Ad-Hoc Networks

Björn Scheuermann* Matthias Transier† Christian Lochert*
Martin Mauve* Wolfgang Effelsberg†

*Heinrich Heine University, Düsseldorf, Germany
{scheuermann,lochert,mauve}@cs.uni-duesseldorf.de

†University of Mannheim, Mannheim, Germany
{transier,effelsberg}@informatik.uni-mannheim.de

ABSTRACT

In mobile ad-hoc networks, the multicast paradigm is of central importance. It can help to save scarce medium bandwidth if packets are to be delivered to multiple destinations. We consider the problem of congestion control for multicast traffic in wireless multihop networks. We propose to apply a congestion control concept which is tailored to the very special properties of the wireless multihop medium: implicit hop-by-hop congestion control. The idea, so far only having been considered for unicast traffic, is here generalized to multicast. We implement it in the Backpressure Multicast Congestion Control (BMCC) protocol, with a focus on how to realize it in combination with geographic multicast routing in the Scalable Position-Based Multicast (SPBM) protocol. Our evaluation points out a number of highly desirable properties of the proposed scheme. In particular, it achieves and maintains high throughput and high packet delivery ratios at low packet latencies, even in the presence of significant network load.

1. INTRODUCTION

Mobile ad-hoc networks (MANETs) use a shared broadcast medium. All the nodes within a collision domain share the medium capacity, which is therefore a scarce resource. Multicast communication is thus of particular interest in these networks, since it helps saving resources when delivering data to multiple destinations. This is further strengthened by the fact that group communication is an inherent feature of many proposed applications for MANETs. A shared broadcast medium is, however, also much more prone to network congestion than, for example, traditional wireline networks. Reducing the number of transmissions required to deliver the data to all receivers is therefore only half the battle. It is, especially in wireless multihop networks, also absolutely vital to perform efficient congestion control, to avoid a congestion collapse.

In this paper, we propose a novel congestion control scheme for multicast in mobile ad-hoc networks. While generally exhibiting very competitive performance, it focuses particularly on the most demanding class of applications: those which depend on very low packet latencies in combination with high packet delivery ratios.

Our protocol is based on *implicit hop-by-hop congestion control*, a paradigm introduced in the Cooperative Cross-layer Congestion Control (CXCC) protocol [15], but so far only having been considered for unicast traffic. CXCC combines simple packet scheduling rules and medium overhearing to implement backpressure with extremely short queues in the intermediate nodes. Here, we apply these concepts to multicast traffic. Since implicit backpressure is the key concept in our protocol, we call it Backpressure Multicast Congestion Control (BMCC).

We focus on an implementation of BMCC in combination with Scalable Position-Based Multicast (SPBM) [20], a geographic multicast routing protocol. Implementing congestion control over such a scheme is particularly challenging: the source node has neither information on the continuously changing topology of the multicast tree nor on the number of group members. BMCC can be used whenever forwarders know their set of next hop nodes in the multicast distribution tree. This holds for SPBM, but also for a large variety of other multicast routing approaches. Therefore, the ideas and concepts introduced here are not specific to SPBM.

In our evaluation, we assess the performance obtained with BMCC using ns-2 simulations. We compare it to plain SPBM, to a variant of SPBM which is also introduced here, and to ODMRP [7], a well-known topology-based multicast

*CoNEXT'*07, December 10-13, 2007, New York, NY, U.S.A.

protocol for MANETs. The results of the simulations underline the very good performance of our approach.

The remainder of this paper is structured in the following way: after reviewing related work in the subsequent section, we briefly outline SPBM in Section 3 and CXCC in Section 4. Thereafter, we describe our multicast congestion control approach in Section 5. The performance of the proposed approach is evaluated in Section 6, before we conclude our paper in Section 7.

2. RELATED WORK

While multicast routing for mobile ad-hoc networks has received some attention over the last years, congestion control for this type of traffic in a wireless multihop environment has only been studied sporadically.

In [16] the MANET multicast protocol ODMRP is evaluated with a different MAC protocol than IEEE 802.11. In this approach congestion control is performed in an end-to-end fashion. Explicit notifications inform the sender about the average load on the used links. The authors argue that a backpressure mechanism would react too slowly. Our protocol, however, proves the opposite, reacting virtually immediately if the forwarding of a packet is delayed.

Similar to the above approach, Tang et al. [17–19] introduce an end-to-end congestion control protocol for multicast traffic. The authors propose to use negative acknowledgments to infer congestion. The sender reacts by reducing its rate until one affected receiver acknowledges a reception explicitly. Rajendran et al. [14] also use end-to-end rate adaptation. In addition, they anticipate upcoming congestion by a local repair strategy, reducing the amount of explicit congestion notifications. Both approaches, however, depend on feedback from the group members, causing a substantial amount of feedback traffic. Our protocol builds up backpressure immediately and locally, and avoids explicit feedback.

In [1,2], Baumung proposes a congestion controlled multicast overlay for MANETs. Hierarchical aggregation of acknowledgments provides feedback on the progress of the worst receiver to the source. This feedback is then leveraged for congestion control. This approach is well-suited for overlay multicast abstracting from the underlying network. However, detecting packet loss at the receivers and propagating the aggregated feedback may require significant time. This is avoided by our approach.

Peng and Sikdar propose a congestion control scheme for layered multicast in MANETs [12]. In their protocol, multicast layers are blocked and released in intermediate nodes, based on the observation of per-link output queue lengths and throughput measurements. Adjustments finer than a whole layer are thus not possible. The scheme also does not take all aspects of the shared medium into account: links are considered heterogeneous and lossy, but independent in terms of capacity. The packets of a blocked layer are still delivered to the blocking intermediate node, and dropped there.

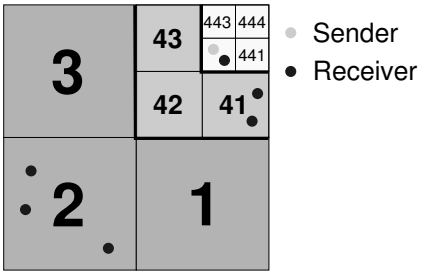


Figure 1: An example for multicast forwarding in SPBM.

This wastes valuable shared medium capacity in the bottleneck area. These problems do not exist in our protocol.

There is substantial existing work in the area of MAC layer multicast in wireless environments. These do not deal with congestion control for multihop multicast traffic, but with delivering a packet to multiple (local) receivers. As one amongst other aspects, our scheme also has to tackle this question. A typical example is the Multicast MAC (MMAC) protocol [3]. In MMAC, the receivers of a transmission are listed in the packet header. Each of them acknowledges the successful reception, in the order given by their index position in the header. In [6], a scheme is introduced which transmits a data packet to up to four receivers at once, and collects acknowledgments from them; for more than four addressees, clusters of at most four nodes are formed, and the packet is transmitted to each cluster separately. This paper also provides a broader overview of the area. We consider single-hop delivery to multiple addressees in a larger context, conjointly with multihop backpressure. This allows for a different view of the problem. All previously proposed approaches result in significant control overhead like, e. g., round-robin polling of all destinations, or many additional feedback fields. This is not necessary in our approach.

3. SCALABLE POSITION-BASED MULTICAST

Scalable Position-Based Multicast (SPBM) [20] is a position-based multicast routing protocol for MANETs. It consists of two main components: a group management protocol and a multicast forwarding protocol. Both use a subdivision of the network area according to a hierarchical quadtree, as shown in Figure 1. The whole area is divided into four sub-squares, which are in turn again subdivided, and so on. This is continued until the resulting so-called level-0 squares are small enough so that each node is able to communicate with all the other nodes within the same level-0 square directly, i. e., they are in a one-hop distance.

The group management provides every node in the ad-hoc network with aggregated group membership information. For each of the three neighboring squares on each hierarchy level, a node knows the list of groups of which at least one member resides in the group. For example, a node located in square ‘442’ in Figure 1 has knowledge about the

aggregated membership information of squares ‘441’, ‘443’, ‘444’, ‘41’, ‘42’, ‘43’, ‘1’, ‘2’, and ‘3’. How this information is actually provided has been described in detail in [20].

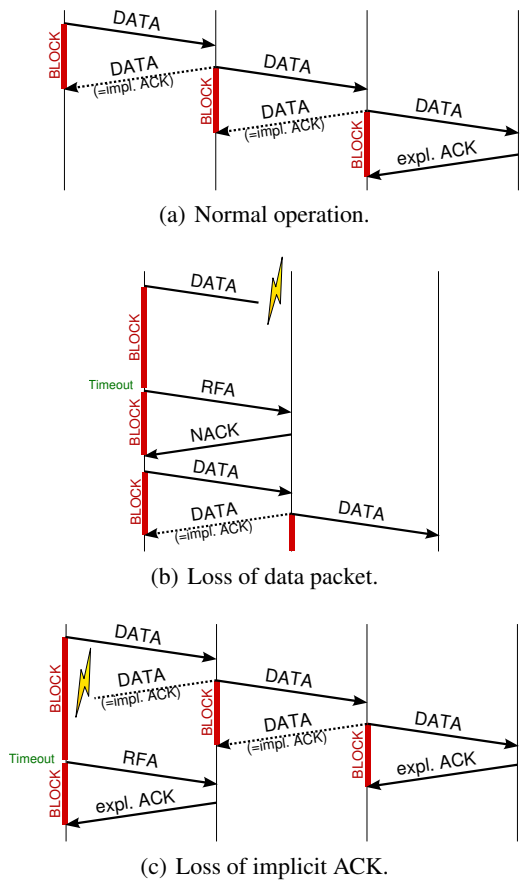
The forwarding decision is based on information about neighboring nodes. Each node maintains a table of nodes in its transmission range. This is accomplished by overhearing data messages and periodic update messages issued by the membership management service, which contain the ID and position of the sending node.

A packet that is to be forwarded includes a list of destination squares, and a group address indicating the group to which the packet is being sent. Again in the example in Figure 1, the source node in square ‘442’ would address a packet to the one shown member in ‘442’ directly (because it is located in the same level-0 square and is thus explicitly known), and to the squares ‘41’ and ‘2’ (because the membership management knows that group members reside there). Upon reception of a forwarded packet, a node checks whether it has more detailed information on the destinations. This will happen as soon as a packet enters its destination square. The respective entry is then *disaggregated* by the forwarder. For example, when the packet with destination ‘41’ in Figure 1 enters square ‘41’, the destination will be replaced by entries for squares ‘411’ and ‘414’: the forwarder, itself located within ‘41’, knows that these are the sub-squares where the group members reside.

The packet is then handed over to the forwarding algorithm, where the best-suited neighbor to forward the packet to each of these destinations is identified. This is accomplished similar to position-based unicast routing (see [8]): the source compares the geographic progress for each of the neighbors with respect to the destination and picks the neighbor with the greatest progress. After identifying the next hop for each destination, the forwarding algorithm sends a copy of the packet to each of these next hops, addressed to the respective destination field(s) that shall be reached. The forwarding uses a sequence of unicast transmissions. This increases the reliability, since there will be MAC layer acknowledgments. It comes, however, at the cost of multiple messages. Section 5 describes the adaptations we made to this transmission scheme in order to implement our congestion control algorithm.

4. IMPLICIT HOP-BY-HOP CONGESTION CONTROL

BMCC extends the implicit hop-by-hop congestion control approach to multicast. This concept has been introduced with CXCC [15], a cross-layer unicast congestion control protocol. Its key concept states that for each end-to-end connection an intermediate node may only forward a packet towards the destination after its successor along the route has forwarded the previous one. This creates a backpressure mechanism which reacts very rapidly and effectively avoids excessive packet inflow into congested network areas. It hence recombines two functions traditionally located



(a) Normal operation.

(b) Loss of data packet.

(c) Loss of implicit ACK.

Figure 2: Packet forwarding with the CXCC protocol.

on the MAC and transport layers: single-hop reliability and congestion control. The concept is termed implicit hop-by-hop congestion control because there are no explicit congestion feedback messages and no dedicated mechanisms for congestion detection, nor are any windows or rates maintained. Congestion control itself is thus not an action that is performed by the nodes of the network, but an implicit effect of the packet forwarding rules. In CXCC, there are no multihop control packets at all.

CXCC obtains the information that a successor has further forwarded a packet by overhearing. Until further forwarding by the downstream node is observed, the respective connection is “blocked”: no further packets may be transmitted. This is depicted in Figure 2(a). Overhearing then serves two purposes at once: it constitutes an implicit acknowledgment, indicating the successful reception by the downstream node, and it also allows forwarding of the next packet. The final destination acknowledges the packet explicitly, since there is no next hop.

The overheard feedback on the successful reception of a packet by the next hop node may arrive with substantial delay. Immediate feedback is not required by CXCC, the protocol is built to deal well with such asynchronous feedback. This design principle is called *soft timing*.

However, wireless connections are potentially error-prone. The reception of a transmission by the successor along the route may fail, for example due to a collision, as it is displayed in Figure 2(b). In CXCC, if a packet is not forwarded and thereby implicitly acknowledged after some time, a *Request For Acknowledgment (RFA)* packet is sent, a small control packet containing just enough information to identify the packet it refers to. Upon reception of an RFA, a node checks whether it has received the respective data packet. If not, it reacts with a *Negative Acknowledgment (NACK)*, thereby triggering a retransmission. The RFA handshake avoids unnecessary payload retransmissions.

Even if the transmission was successful, the implicit acknowledgment might not have been overheard, like in Figure 2(c). The sending node cannot distinguish this situation from the previous one, and will send an RFA. The next hop node, however, can tell the difference. It knows that the implicit ACK has been lost. If no further (re)transmissions of the packet are pending and thus it will not be possible to acknowledge the packet implicitly, this node will send an explicit acknowledgment to resolve the situation.

Finally, the forwarding may be delayed by contention or backpressure. In case of a very long such forwarding delay, an RFA might be sent for such a packet. From a congestion control perspective, the reception of an RFA may then simply be ignored. Neither is a retransmission (triggered by a NACK) necessary, nor may the packet be acknowledged through an explicit ACK: this is undesirable, since it would also release the backpressure.

CXCC allows only one untransmitted packet per flow to be queued in every intermediate node along the route. Note that this actually suffices not to waste any capacity: since the shared medium enforces a local serialization of transmissions, one packet per hop allows for enough parallelism to make full use of the available bandwidth.[1]

The central limitation of a cross-layer approach that, like CXCC, includes an adaptation of the MAC layer is that it can not be implemented on any arbitrary standard hardware. Wireless network interfaces usually do not allow for modifications of the MAC layer functionality. It is therefore constrained to environments that facilitate the required changes.

5. BACKPRESSURE MULTICAST CONGESTION CONTROL

The central difference between unicast and multicast from the perspective of packet forwarding is that for unicast each forwarder has exactly one next hop node, while with multicast there may be more than one. Essentially, each packet is forwarded along a tree of nodes, originated at the source node. Therefore, in order to apply implicit hop-by-hop congestion control to multicast traffic, we need to generalize the implicit feedback concepts appropriately to that situation. This generalization forms the core of BMCC.

5.1 Packet forwarding with local broadcasts

Implicit hop-by-hop congestion control embraces the local broadcast property of the wireless multihop medium by not using explicit feedback, but instead gathering information through overhearing. Transmitting the payload of a packet directed to multiple next hops only once is thus a natural approach, and it is followed here just like in many previous proposals.

As a first step, we introduce a corresponding, alternative way of packet forwarding in SPBM, called Broadcast SPBM (SPBM-BC). The group management and the selection of the next hops work as described in Section 3. But instead of a separate unicast transmission for each next hop, a single broadcast transmission is used for all of them. The forwarding node adds all designated next hops to the packet header, including a list of destination squares for each of them. When the packet is sent via MAC layer broadcast, all neighbors receive it and check whether they are contained in the list of designated next hops—if not, they discard the packet. This is complemented by implicit acknowledgments: if the original sender does not overhear the retransmission of a packet from all of the designated next hops, it will rebroadcast the packet after removing all next hops that already successfully acknowledged the packet.

SPBM-BC is the basis for our implementation of SPBM with Backpressure Multicast Congestion Control. In BMCC, we modify the single-hop reliability component of SPBM-BC, introducing a CXCC-like backpressure mechanism and replacing the timeout-triggered packet retransmissions by an RFA-based scheme. In our evaluation, SPBM-BC will also serve as a benchmark: it will show us the performance that can be obtained by using local broadcasts and implicit acknowledgments *without* BMCC's backpressure mechanism.

To avoid parallel medium access attempts by multiple next hop nodes of a transmission, each attempting to forward the newly received packet, a node waits for a random backoff before it transmits. Multiple next hop nodes may not all be within mutual communication range, but it is reasonable to assume that they are often within carrier sense range. In combination with carrier sensing and medium access backoff the jittering desynchronizes the answers, thus avoiding the synchronization problem. This pragmatic solution matches the soft timing principle very well, avoids complex coordination, and saves significant overhead.

5.2 Backpressure with multiple next hops

By not allowing the transmission of a subsequent packet before its predecessor has been forwarded by the next hop CXCC builds up backpressure. This guarantees that a downstream bottleneck rapidly propagates backwards along the

[1] In fact, even one packet per *collision domain* would suffice therefor, but this cannot easily be enforced with a practical protocol.

route towards the source. In BMCC, we apply the same concept, but along a tree structure. We strive for high packet delivery ratios to all receivers in this tree, i. e., towards all leaves. As a consequence, we need to adjust the source data rate to the tightest bottleneck in the forwarding tree. In other words, we need to ensure that the data inflow into any branch does not exceed the bottleneck capacity within that branch.

In this form, the scheme is in line with applications that cannot tolerate substantial packet loss, and thus require that all group members get a high fraction of the sent packets. Necessarily however, this makes it susceptible to the well-known "crying baby problem" [4]. If one group member has a particularly bad connection, its mere existence will result in a deterioration of service quality for the other group members. We will devise a way to deal with this effect later. For now we concentrate on a backpressure protocol that adjusts the source data rate to the tightest point in the multicast tree.

BMCC achieves the desired congestion controlling behavior by generalizing the CXCC backpressure rule in the following way: the next packet may only be forwarded if *all* the next hop nodes for that packet have forwarded the previous one. Similar to the backpressure building up backwards along the route with CXCC, this rule in BMCC results in backpressure along the tree. Thereby, packets that are not able to traverse the network will not be allowed to leave the source node. This implicitly regulates the source data rate, and it keeps the queues in the intermediate nodes extremely short. The backpressure is applied per-flow, defined by source node and destination group. Consequently, each forwarder can queue at most one untransmitted packet of a given flow. Intermediate nodes that are traversed by multiple flows may hold more than one untransmitted packet in total; however, due to the constrained capacity of the shared wireless medium, the number of parallel flows will be rather limited.

The source node can also communicate the backpressure to the application. This allows to adapt the packet generation to the medium situation, for example by adjusting the bit rate dynamically.

Since transmissions in BMCC are directed towards a set of next hop nodes, the situation is significantly more complex than in the single next hop case of unicast forwarding. Each single next hop node may have received the transmission correctly or not. If the packet has been received correctly, each of the next hops may already have forwarded it again or might still hold it back due to backpressure. Finally, for each successor having forwarded the packet, the implicit acknowledgment may have been overheard or not. The central challenge in BMCC is to deal with this additional complexity efficiently while adhering to the principles of implicit feedback and soft timing, and avoiding unnecessary control traffic.

To tackle this challenge, a forwarding node in BMCC keeps track of the list of next hop nodes from which an acknowledgment is still missing. After transmitting a packet addressed to a set of one or more next hops, this list is initialized to contain all these next hops. If an implicit (or explicit) acknowledgment from one of them is detected, the respective node is removed from the list. The transmission of the subsequent packet is allowed once the last entry has been removed from the list.

If acknowledgments are missing for a too long time, a generalization of CXCC's RFAs is used. Analogous to data packets, RFAs in BMCC are directed to a whole set of next hop nodes: they address all the next hop nodes from which an acknowledgment is still missing. All thereby challenged forwarders can decide individually whether they should react with an explicit ACK or NACK.

A number of optimizations is possible to exploit the information contained in these handshakes most effectively. Since a single next hop node not having received the data packet already necessitates a retransmission, it is not mandatory to wait for feedback from all nodes if a NACK is received. In this case, an immediate retransmission of the data packet is triggered, addressed to the nodes from which acknowledgments are missing. Ideally, this makes the transmission of further NACKs by other next hop nodes unnecessary. Furthermore, such a retransmission may also fulfill the purpose of an RFA for nodes that had already received and forwarded the packet. If their forwarding has not been overheard by their predecessor, they will be addressees of the retransmission. They can easily detect this situation and repeat the lost feedback through an explicit ACK.

Like for the packet transmissions themselves, a possible synchronization of the answers to an RFA needs to be considered. If multiple addressees all access the medium immediately after receiving the RFA this will cause severe collisions. For this reason, such reactions by forwarders are, just like forwarded data packets, sent with substantial jitter.

The design of BMCC is of course a tradeoff. The protocol needs to keep track of the receivers from which no acknowledgment has yet arrived, construct RFA packets, etc.; this requires small, but not totally negligible storage and computational effort. But BMCC is tailored to wireless multihop networks, and there the tradeoff between computation power and communication bandwidth is very different from the situation in, for instance, high-speed Internet routers. We consider the—still limited—additional effort in the intermediate nodes appropriate, because it helps to use the scarce MANET bandwidth more efficiently.

5.3 Dealing with unavailable next hops

In order to perform effective congestion control, backpressure should be maintained as long as the downstream nodes are not able to forward the previous packet. It must, however, be avoided to wait indefinitely for an implicit acknowledgment from a downstream node which is no longer reachable. Such a node will obviously not react to RFAs. But since this also applies to a node keeping a packet back

due to backpressure, a lightweight mechanism is needed which helps to distinguish these two cases.

A basic solution to this problem is already provided by SPBM: if no more update beacons from a neighbor are received over some time, it is considered unavailable. But due to the relatively low beaconing frequency, this reacts rather slowly. In BMCC, we speed up the detection of no longer available next hops by using *keepalive (KAL)* packets. A KAL is a small control packet sent if an RFA is received for a packet which has arrived, but is currently held back due to backpressure. It may also be sent when a new packet is received from the previous hop, while an acknowledgment for the preceding one has not yet been received. It indicates that its sender is reachable, but it does not release the backpressure. With this extension, the link to a next hop node may be considered broken if the number of consecutive unanswered RFAs exceeds some threshold.

While at a first glance the additional feedback messages seem to increase the protocol overhead in a situation in which medium bandwidth is particularly scarce, they can in fact help to *reduce* the total amount of control traffic. The reception of a KAL indicates that backpressure definitely exists. Consequently, RFAs are then sent less aggressively, resulting in a lower overall network load.

5.4 Handling inhomogeneous receivers: backpressure pruning

One earlier mentioned issue still deserves attention: BMCC will adjust the data rate to the tightest bottleneck in the multicast tree, i. e., according to the slowest receiver. While this is necessary in order to achieve high delivery ratios at all receivers—and might thus well be desirable in certain usage scenarios—, it is susceptible to the "crying baby problem". If there is one group member with a particularly bad reachability, this will thwart a higher data rate to all other receivers. For applications that can tolerate partial delivery of the transmitted data to weaker receivers (using, e. g., an encoding that can still deliver a lower quality of service under packet loss), it is thus of interest to see whether a variant of BMCC can be built that exhibits a different behavior in this regard: is it possible to modify the algorithm to adjust the inflow into each branch of the multicast tree to the highest rate sustainable by at least one receiver in that branch, thus maximizing the throughput to each individual receiver? Depending on the application, the original version or such a variant may be favorable.

However, due to the shared broadcast medium the rates to the receivers cannot be individually and independently maximized. For clarification, let us consider two simple examples in a scenario like in Figure 3. There is one sender and two receivers. While receiver 1 is directly reachable from the source, receiver 2 is further away. In the first example, there is no additional traffic in the network. Transmissions from the source to receiver 1 are affected by transmissions made by at least the first two forwarders towards

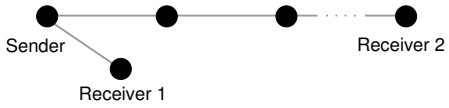


Figure 3: Simple scenario with unequal receivers.

receiver 2, because of the shared medium and carrier sensing at the source node. BMCC aims at high packet delivery ratios to all receivers. The backpressure rule as presented above will achieve the following: it will always allow to forward a packet towards receiver 2 before the next packet enters the network—even though this reduces the throughput to receiver 1. When striving for fairness between multiple receivers and high packet delivery ratios this is generally the desired behavior. Ideally, receiver 1 should not receive data at a higher rate, if this comes at the cost of receiver 2's rate.[2]

As a second example, let us consider a situation in which the medium around receiver 2 is severely congested. Then, backpressure towards the source will build up, and forwarding of packets along the route to receiver 2 may be substantially delayed—this is inevitable if it is not possible to forward packets towards receiver 2 at a higher rate. But this behavior can result in substantial *underutilization* of the medium around the source and receiver 1. Depending on the application, it may be desirable to use such otherwise unused medium bandwidth for the forwarding of additional packets to better reachable receivers. Nonetheless only those packets should enter the network for which the bandwidth towards at least one receiver suffices.

We will now present a modification of the backpressure rule of BMCC that is able to yield just these effects. We call it BMCC with *backpressure pruning* (BMCC-BP). This backpressure pruning mechanism allows for branches to be cut off if backpressure exists in them. It makes use of the keepalive packets introduced above for the purpose of improved unavailable forwarder detection. Recall the conditions under which a KAL packet is sent: it occurs in backpressure situations, when the forwarding of packets is delayed. Thus, the reception of a KAL from one next hop node indicates that the respective subtree is currently a bottleneck.

In standard BMCC, a node must wait for all next hop nodes to acknowledge the packet (neglecting, for simplicity of discussion, possible unavailable next hop nodes). BMCC-BP replaces this with a slightly more complex rule set, as follows. A node may stop further attempts to deliver a packet to all next hop nodes if

[2]This is actually related to the notion of max-min-fairness, which states that a resource allocation is max-min-fair if increasing the share of any component is only possible at the cost of decreasing the share of an already lower component (for an in-depth discussion in the networking context see, e. g., [13]). We do not claim that the variant of BMCC to be introduced now will guarantee max-min-fair bandwidth allocations—due to the complexity and stochastic nature of a wireless multihop environment such a guarantee is hardly possible. But we aim for a heuristic that follows this general idea.

1. at least one next hop node has acknowledged the packet,

2. a KAL has been received from all other next hops, and

3. a subsequent packet is already available for forwarding.

At the source node, the latter criterion is fulfilled if the application has already generated a subsequent packet which is waiting in the queue. In intermediate nodes, it holds as soon as a follow-up packet has been received from the upstream node. This may happen when the upstream node, in turn, has received at least one implicit or explicit ACK and KALs from all *its* remaining next hops.

The first backpressure pruning criterion guarantees that each packet will eventually arrive at at least one receiver: if one next hop node has acknowledged the packet, this implies that it has been forwarded into at least one branch. Packets will thus still not enter the network at a rate higher than what can be sustained by the "best" group members. The second criterion antagonizes the "stealing" of bandwidth from other branches, by providing each next hop with a chance to access the medium and thus at least with an opportunity to forward a packet.

Backpressure pruning may result in situations where a node receives a follow-up packet before it has attempted to forward the previous one. In this case, it should drop the previously known packet (for which it had sent a KAL), and instead enqueue the newly received one.

Summarizing so far, BMCC, as originally introduced, is designed to result in an adjustment of the source data rate to the tightest bottleneck in the network. This ensures high delivery ratios whenever possible. BMCC-BP is in some sense complementary: it is built to deliver the maximum individually sustainable rate to each receiver, as long as this does not come at the cost of other branches of the tree. Both adjust the rates of the source and of intermediate nodes without explicit rate feedback and without multihop control packets, by using implicit backpressure.

6. EVALUATION

For the evaluation we implemented SPBM with BMCC in the network simulator ns-2.30 [10]. As a comparison, we used the plain (unicast) version of SPBM as described in Section 3, the broadcast version from Section 5.1, and an implementation of ODMRP [7] that was originally obtained from [11], ported to ns-2.30, and optimized as described in [20]. SPBM and ODMRP are simulated with an IEEE 802.11 MAC, BMCC inherits some minor modifications to 802.11's upper layer interface from CXCC [15].

For the results presented below, we simulated a network area of $1500 \times 1500\,\mathrm{m}^2$ with a total of 200 nodes. Each data point is an average of ten simulation runs in different scenarios, the error bars show 95 % confidence intervals. One multicast group was defined, with two senders and ten receivers; thus, two independent multicast trees were used in parallel. Other combinations of sender and receiver counts yield similar results. We consider scenarios with and without node mobility. The source applications generate data packets with 64 bytes of payload at an increasing rate between 1 and 50 packets per second, or the highest frequency at which packets are able to leave the source node, whichever is lower. While BMCC can provide fine-grained feedback to the application about when packets may be sent, the other protocols used here are not able to generate such feedback. By accounting only for packets that are able to leave the source, we thus avoid distortions of the results for the other protocols and keep the comparison fair.

6.1 Delivery ratio and throughput

In Figure 4, the packet delivery ratio achieved by the different protocols is shown, in a static setting without mobility. A value of one means that all packets that had left the source nodes arrived successfully at each receiver. Adjusting the source rate in order to allow for high packet delivery ratios was a main design goal of BMCC. The results show that it has been achieved. At packet generation rates below 20 packets per second the delivery ratios of unicast SPBM and ODMRP are very similar, with slight advantages for SPBM. For higher rates, ODMRP delivers a greater fraction of the packets. However, the fact that the delivery ratio is continuously decreasing for all three shows that more and more packets are able to leave the sources, but then do not make it to the receivers. Interestingly, the broadcast version of SPBM with implicit acknowledgments does not reach the performance of unicast SPBM—although it theoretically needs fewer packet transmissions. If the forwarding of a packet in one of the next hops is delayed, no implicit acknowledgment will arrive, which causes SPBM-BC to unnecessarily retransmit the packet. This wastes substantial medium bandwidth. Obviously, it is not enough to use implicit acknowledgments on the network layer. It is the backpressure mechanism in BMCC that turns the balance. It is able to outperform all others and reach delivery ratios very close to 100 % at all sending rates. This shows that the congestion control mechanism is successful in regulating the source rate. The sending nodes only put on air as many packets as the network is able to handle. Thus, the protocol is able to retain high delivery ratios even for high packet generation rates.

In Figure 4, there is a small difference between the results with standard BMCC and BMCC with backpressure pruning. This difference will likewise be small in all our other random topology simulations. As we will soon demonstrate, the reason is not that the protocols do generally behave very similar or even identically. This effect is rather caused by the relative homogeneity of the receivers and therefore the load distribution in these networks. There seems to be virtually no free medium capacity that could be used to deliver

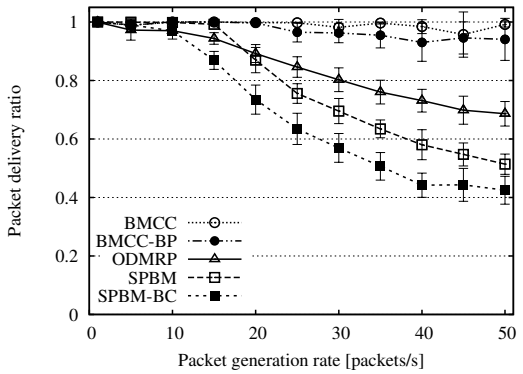


Figure 4: Packet delivery ratio with increasing packet generation rate.

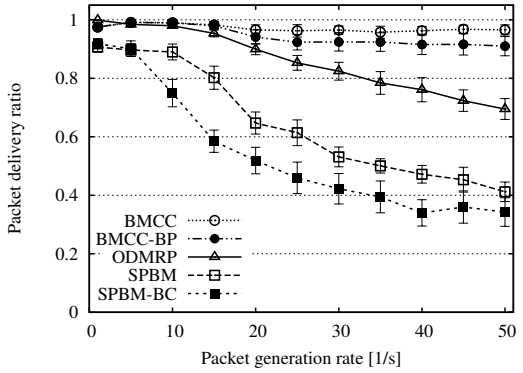


Figure 5: Packet delivery ratio in mobile scenarios.

data faster to certain receivers, without at the same time negatively affecting others.

Figure 5 shows how the packet delivery ratio develops in the presence of mobility. In these simulations, the nodes move according to the random waypoint mobility model, with a maximum speed of 5 meters per second, a minimum speed of 0.5 meters per second, and no pause times. Initial node positions and speeds are sampled according to the mobility model's stationary distribution [9]. Data is again generated by the source applications at varying rates. It can be seen that the relative performance of the protocols remains largely unchanged, all deal reasonably well with mobility—only losing a few packets. Rapid topology changes cause inconsistencies in SPBM's routing tables, and thus also affect SPBM with BMCC.

A very high packet delivery ratio could of course be achieved relatively easily if the total number of packets in the network is kept at a low level. Figure 4 does only show that almost all out of a so far unknown number of packets leaving the source do arrive with BMCC. We therefore have to consider these results in conjunction with the obtained data rate. Figure 6 presents the average data rate received by the group members. For packet generation rates of up to

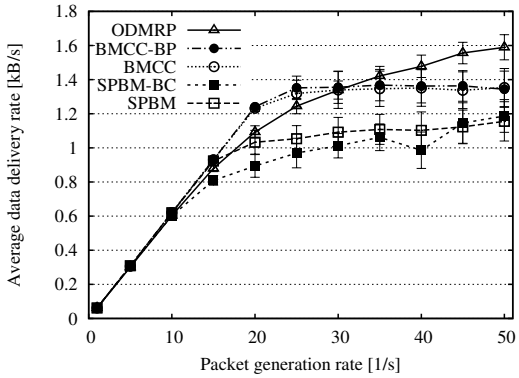


Figure 6: Receiver data rate.

10 to 15 packets per second, all examined protocols are able to deliver all the data produced by the applications. Since each data packet carries 64 bytes of payload, the resulting optimal data rate is 640 Byte/s at 10 packets per second and sender. The simple broadcast version of SPBM breaks first. Starting from 10 packets per second, its goodput increases much less than the data generation rate. Plain SPBM and BMCC show similar trends at different levels: the goodput grows up to a certain saturation and stays at the same level for all higher packet generation rates. While the unicast version of SPBM delivers on average around 1.1 kB/s, BMCC regulates the source rates to a level that yields 1.35 kB/s of goodput. ODMRP delivers higher data rates starting from 33 packets per second. This, however, comes at a high cost: not only does ODMRP, as seen before, lose at least 20 % of the packets. As we will soon see, it also allocates resources unfairly, preferring near-by receivers, burdens the network with a heavy traffic load, and suffers from high end-to-end delays.

Figure 7 shows the average receiver data for mobile scenarios. Again, BMCC achieves a perfectly shaped throughput curve. As can be seen, ODMRP is nearly unaffected by mobility. The variants of SPBM, including BMCC, achieve lower rates in the presence of mobility. Starting from the point where ODMRP achieves higher data rates, it also—as described above—exhibits decreasing packet delivery ratios. In [20], it has already been shown that SPBM suffers from mobility, because of its group management. Nevertheless, with BMCC it is able to keep up high delivery ratios.

6.2 Fairness between senders

The previous evaluation raises the question why BMCC does not achieve the somewhat higher data rates obtained with ODMRP, if the network is seemingly able to support them. The key to understand this property lies in the vastly different effort that is required to deliver a packet to different receivers, depending on their distance from the source. It is much more resource intensive to bring a packet to a far away receiver than to one close by. A high data rate might simply be obtained by preferring transmissions over shorter

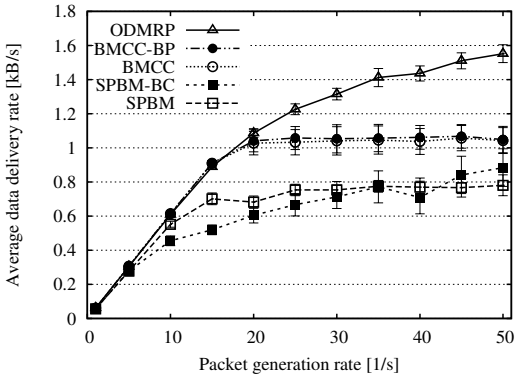


Figure 7: Receiver data rate in mobile scenarios.

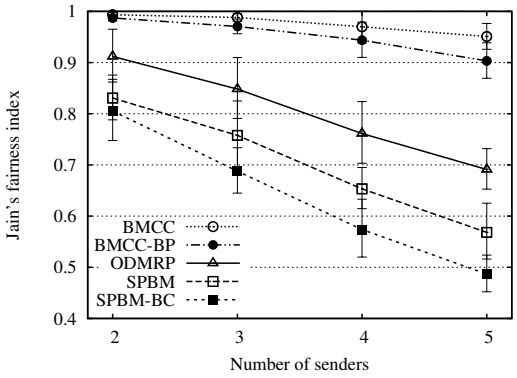


Figure 8: Jain's fairness index for packet distribution over senders.

distances. This issue is closely related to the fairness between senders: do receivers preferably receive packets from closer source nodes?

In order to analyze this aspect, we look at the distribution of packet sources amongst the packets arriving at the receivers, with an increasing number of senders in a multicast group. To quantify the fairness of this distribution, we use Jain's fairness index as introduced in [5]. This index establishes a measure for the fairness of resource allocation in a multi-user system. It yields a value between zero and one, where one means perfect fairness and zero is approached if one out of more and more participants is assigned all resources. The fairness index is defined as

$$\frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i^2},$$

where x_i is the resource share assigned to the i-th participant.

Here, we apply Jain's fairness index to the packet counts received from each source, giving us a fairness value for each receiver. I. e., in our case x_i is the number of packets received from the i-th source. In Figure 8, the average of the resulting index values for all receivers is shown, for an increasing number of senders. ODMRP achieves a better fairness than plain unicast SPBM. Broadcast SPBM does not meet the performance of the unicast version in this metric either. But BMCC again clearly outperforms ODMRP. Here lies the reason why BMCC does not allow for higher data rates: these seem possible only at the cost of an increased unfairness.

BMCC-BP is built to spend otherwise unused medium bandwidth for the delivery of additional packets to well-reachable receivers. Thus, the slightly lower Jain fairness in terms of achieved throughput is in accordance with the protocol's design goals.

6.3 Delay and protocol overhead

Another central metric of protocols is the overhead incurred by their use. This is not only related to the efficiency of the medium use by the protocol, but also to the energy consumption caused by the transmissions. Figure 9

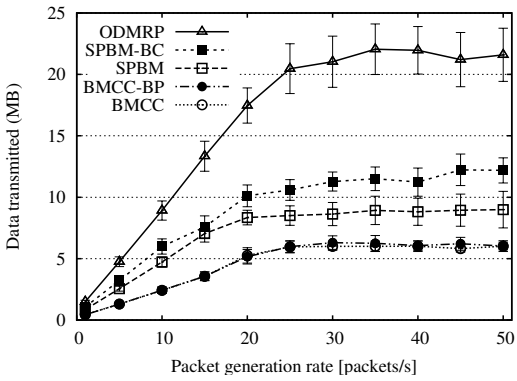


Figure 9: Data transmitted on physical layer.

shows the average total amount of data that has been transmitted on the physical layer during one simulation run; it includes data packet retransmissions as well as all control traffic. ODMRP has the highest resource requirements. The reasons lie within the structure of the protocol. ODMRP floods data packets through the whole network on a regular basis, and it uses redundant paths in a mesh structure, both of which result in a higher number of transmissions. The broadcast and unicast version of SPBM produce similar amounts of data on the physical layer. If local broadcasts are used with SPBM without employing BMCC's backpressure mechanism, it needs even somewhat more bandwidth, instead of saving it. This results from a high number of retransmissions performed by this approach. The backpressure mechanism of BMCC, because of its effective ways to avoid unnecessary retransmissions, is once again able to turn this into the opposite, avoiding unnecessary control traffic and retransmissions.

Last but not least let us have a look at the end-to-end delay. Figure 10 depicts the average end-to-end delay of all delivered data packets, from the time the packet leaves the source node until it arrives at the receiver. Again, up to a packet generation rate of 15 packets per second in each source, all protocols deliver the packets at sufficiently low delays. For

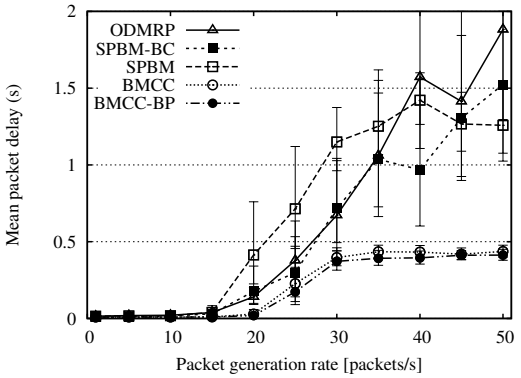


Figure 10: End-to-end delay.

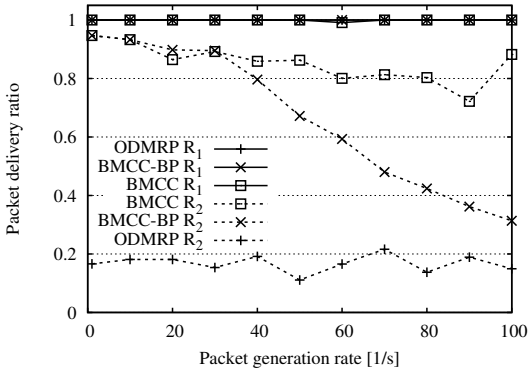


Figure 11: Packet delivery ratio for receivers R_1 and R_2 in a simple static scenario with congestion at R_2.

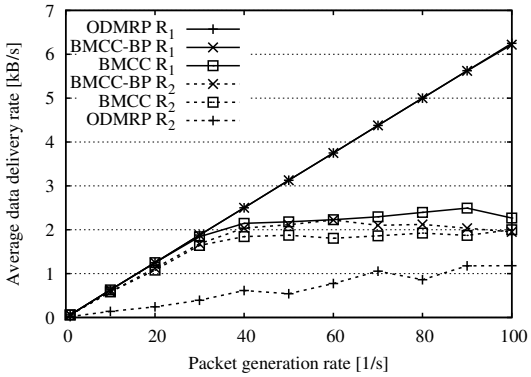


Figure 12: Data rate of receivers R_1 and R_2 in a simple static scenario with congestion at R_2.

higher data generation rates, only BMCC is able to maintain short packet latencies. The other protocols delay the packets for up to two to four seconds, which is definitely unacceptable. This problem stems from long queues building up in the intermediate nodes, a problem being avoided in BMCC by the very design of the protocol, which implies very short queues.

6.4 Backpressure pruning

So far, it seemed that backpressure pruning does not have any noteworthy effect. This is, however, not true. The impression is a result of the relative homogeneity of the so far considered settings. To precisely analyze the behavior of BMCC-BP in a scenario in which a difference must clearly appear, we use a simple static topology similar to the one depicted in Figure 3. Based on an equidistant chain topology, the nodes are set up such that the source is a direct neighbor to one receiver, R_1, while a second one, R_2 is seven hops away. An additional interfering data stream transmits packets continuously in the neighborhood of this second receiver. The source node again generates data packets at an increasing rate.

We analyze the packet delivery ratio as well as the data rate for both receivers separately. This allows for a detailed analysis of the operation of BMCC-BP. The respective results are depicted in Figures 11 and 12. There are six curves, describing the results with ODMRP and the two BMCC variants. For improved readability of the charts we leave out the results with SPBM. Not surprisingly, all protocols are able to transmit packets with a high delivery ratio to the first receiver. BMCC, aiming at the maximum possible fairness, notices the congested area via its implicit backpressure mechanism and thus maintains a high delivery ratio also towards the second receiver—which is only possible at a limited rate for *both* receivers. ODMRP, without a mechanism to deal with such a congestion situation, results in a high number of lost packets on the path to the second receiver; the first receiver gets packets at a high data rate, while the second receiver is mostly cut off. The backpressure pruning mechanism in BMCC-BP handles the congestion situation correctly. It reduces the data rate to the second receiver (and thus the packet delivery ratio) without affecting the ability of the non-congested destination to receive more packets.

7. CONCLUSIONS

In this paper, we have proposed a novel way to effectively control congestion of multicast traffic in wireless multihop networks. Our scheme is based on implicit feedback, establishing multihop backpressure through simple packet forwarding rules. It solves single hop reliability and implicit multihop backpressure congestion control conjointly, thereby avoiding many unnecessary control messages and packet retransmissions.

Two flavors meet different demands: standard BMCC adapts to the slowest receiver and thus ensures that all group members receive each packet at a high probability, while BMCC-BP supplies all receivers with as many packets as they are individually able to receive, accepting packet losses in the multicast tree if necessary.

Whereas the approach inherits the limitation of not being easily implemented on arbitrary standard hardware, simulations show that it exhibits superior performance. A concrete

implementation of the approach in combination with the geographic multicast routing protocol SPBM demonstrates the effectiveness of the source rate adaptation. Our scheme yields competitive throughput while maintaining very high packet delivery ratios for all receivers, combining these traits with very low end-to-end packet delays thanks to extremely short queues. It achieves all this at a low protocol overhead.

Acknowledgments

This research was funded by the German Research Foundation (DFG) in priority program 1140.

8. REFERENCES

[1] P. Baumung. Stable, congestion-controlled application-layer multicasting in pedestrian ad-hoc networks. In *WoWMoM '05: Proceedings of the 6th IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks*, pages 57–64, June 2005.

[2] P. Baumung, M. Zitterbart, and K. Kutzner. Improving delivery ratios for application layer multicast in mobile ad-hoc networks. In *ASWN '04: Proceedings of the 4th Workshop on Applications and Services in Wireless Networks*, pages 132–141, Aug. 2004.

[3] H. Gossain, N. Nandiraju, K. Anand, and D. P. Agrawal. Supporting MAC layer multicast in IEEE 802.11 based MANETs: Issues and solutions. In *LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, pages 172–179, Nov. 2004.

[4] H. W. Holbrook, S. K. Singhal, and D. R. Cheriton. Log-based receiver-reliable multicast for distributed interactive simulation. In *SIGCOMM '95: Proceedings of the 1995 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 328–341, Aug. 1995.

[5] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe. A quantitative measure and discrimination for resource allocation in shared computer systems. Technical Report DEC-TR-301, Digital Equipment Corporation, Eastern Research Lab, Sept. 1984.

[6] S. Jain and S. R. Das. MAC layer multicast in wireless multihop networks. In *COMSWARE '06: Proceedings of the 1st International Conference on Communication System Software and Middleware*, Jan. 2006.

[7] S.-J. Lee, M. Gerla, and C.-C. Chiang. On-demand multicast routing protocol. In *WCNC '99: Proceedings of the IEEE Wireless Communications and Networking Conference*, volume 3, pages 1298–1302, Sept. 1999.

[8] M. Mauve, J. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad-hoc networks. *IEEE Network*, 15(6):30–39, Nov. 2001.

[9] W. Navidi and T. Camp. Stationary distributions for the random waypoint mobility model. *IEEE Transactions on Mobile Computing*, 3(1):99–108, Jan. 2004.

[10] The ns-2 network simulator. `http://nsnam.isi.edu/nsnam`. [Online; last accessed 2007-07-06].

[11] Wireless multicast extensions for ns-2.1b8. `http://www.monarch.cs.rice.edu/multicast_extensions.html`. [Online; last accessed 2007-07-06].

[12] J. Peng and B. Sikdar. A multicast congestion control scheme for mobile ad-hoc networks. In *GLOBECOM '03: Proceedings of the IEEE Global Telecommunications Conference*, volume 5, pages 2860–2864, Dec. 2003.

[13] B. Radunović and J.-Y. Le Boudec. A unified framework for max-min and min-max fairness with applications. In *Allerton '02: Proceedings of the 40th Annual Allerton Conference*, Oct. 2002.

[14] V. Rajendran, K. Obraczka, Y. Yi, S.-J. Lee, K. Tang, and M. Gerla. Combining source- and localized recovery to achieve reliable multicast in multi-hop ad hoc networks. In *NETWORKING '04: Proceedings of the 3rd International IFIP-TC6 Networking Conference*, pages 112–124, May 2004.

[15] B. Scheuermann, C. Lochert, and M. Mauve. Implicit hop-by-hop congestion control in wireless multihop networks. *Elsevier Ad Hoc Networks*. To appear, available online. DOI 10.1016/j.adhoc.2007.01.001.

[16] K. Tang and M. Gerla. Congestion control multicast in wireless ad hoc networks. *Elsevier Computer Communications*, 26(3):278–288, Feb. 2003.

[17] K. Tang, K. Obraczka, S.-J. Lee, and M. Gerla. Congestion controlled adaptive lightweight multicast in wireless mobile ad hoc networks. In *ISCC '02: Proceedings of the 7th IEEE International Symposium on Computers and Communication*, pages 967–972, July 2002.

[18] K. Tang, K. Obraczka, S.-J. Lee, and M. Gerla. A reliable, congestion-controlled multicast transport protocol in multimedia multi-hop networks. In *WPMC '02: Proceedings of the 5th International Symposium on Wireless Personal Multimedia Communications*, pages 252–256, Oct. 2002.

[19] K. Tang, K. Obraczka, S.-J. Lee, and M. Gerla. Reliable adaptive lightweight multicast protocol. In *ICC '03: Proceedings of the IEEE International Conference on Communications*, pages 1054–1058, May 2003.

[20] M. Transier, H. Füßler, J. Widmer, M. Mauve, and W. Effelsberg. A hierarchical approach to position-based multicast for mobile ad-hoc networks. *Springer Wireless Networks*, 13(4):447–460, Aug. 2007.