

Reliable Adaptive Lightweight Multicast Protocol

Ken Tang
Scalable Network Technologies
ktang@scalable-networks.com

Katia Obraczka
UC Santa Cruz
katia@cse.ucsc.edu

Sung-Ju Lee
HP Labs
sjlee@hpl.hp.com

Mario Gerla
UCLA
gerla@cs.ucla.edu

Abstract—Typical applications of mobile ad hoc networks (MANET) require group-oriented services. Digital battlefields and disaster relief operations make data dissemination and teleconferences a key application domain. Network-supported multicast is hence critical for efficient any-to-many communications. However, very little work has been done on “reliable” transport multicast. We propose and evaluate Reliable Adaptive Lightweight Multicast (RALM). The design choices of RALM are motivated by lessons we learned from evaluating the performance of traditional wired reliable multicast transport protocols (in particular, SRM) in ad hoc networks. We argue that two components, *reliability* and *congestion control*, are essential in designing a reliable multicast transport protocol for MANETs. RALM addresses both reliability and congestion control. It achieves reliability by guaranteeing data delivery to troubled receivers in a round-robin fashion. RALM’s send-and-wait congestion control uses NACK feedback to adjust to congestion experienced by receivers. We show through simulations that RALM achieves perfect reliability while exhibiting low end-to-end delay and minimal control overhead compared against other protocols.

I. INTRODUCTION

An ad hoc network enables wireless communications without any fixed infrastructure or central administration. Each node communicates with each other through packet radios. Hence, every host acts as a packet forwarder as well as a source or a destination. Because of its ease of deployment, an ad hoc network is an attractive choice for scenarios where the fixed network infrastructure is non-existent (e.g., remote locations), unusable because it is insecure (e.g., covert military operations) or unavailable due to some catastrophic events (e.g., major earthquake). The types of scenarios targeted by MANETs make group-oriented services such as data dissemination and teleconferences a key application domain. Multicast communication is an efficient means of supporting group-oriented applications. This is especially true in mobile wireless environments where nodes are energy and bandwidth constrained. Because MANETs are particularly well suited for mission-critical applications, ad hoc network protocols must provide reliable and timely data delivery even in the presence of mobility and frequent outages.

MANETs have been the subject of extensive research. Despite the fact that reliable multicasting is vital to the success of mission critical applications, surprisingly little work has been done in this area. One of the few exceptions is the Anonymous Gossip (AG) protocol [1] that recovers from losses by having pairs of multicast members exchange information on messages they have received or lost. One potential problem with this protocol is the delay it takes for nodes to recover from

losses. Reliable multicast for wired networks on the other hand, has been a very active area of research [2]. MAC and transport level protocols for wireless cellular networks have also been proposed [3], [4], [5]. One may consider applying these schemes to MANETs. We argue that the design choices underlying wired reliable multicast transport protocols are not adequate for MANETs. Ad hoc networking protocols must handle node mobility. In addition, MANETs are extremely sensitive to network load and congestion, even more so than in wired shared-medium networks because of the hidden terminal problem. Generating additional control message overhead without performing adequate congestion control will considerably degrade the performance.

We propose the Reliable Adaptive Lightweight Multicast (RALM) transport protocol that favors reliability and congestion control over throughput. Applications that are willing to trade throughput for reliability include military covert operations and search and rescue missions. For example, an operation commander disseminating mission critical data to his troops in a covert operation is more interested in reliably delivering the commands rather than obtaining high throughput (assuming adequate throughput is obtained). In such a scenario, any data loss can be fatal to the success of the entire operation.

RALM is a reliable, rate-based, congestion controlled protocol that targets small group operation scenarios ranging from special military operations to civilian emergency rescue applications. When there is no packet loss, RALM sends packets at the specified application sending rate. Once a loss is detected, RALM recovers by initiating a modified send-and-wait procedure. Send-and-wait is performed with each multicast receiver that experiences losses, one at a time in a round-robin fashion. Once all receivers have up-to-date packets, RALM reverts to the application sending rate. In our previous work [6], we assumed that the multicast sources know the receiver information ahead of time and was able to use a window-based congestion control approach. In this paper, we do not make such an assumption and hence use a send-and-wait procedure.

We start this study by evaluating how a “wired” reliable multicast protocol performs in MANETs. While we acknowledge that wired protocols were not designed for MANETs, studying the behavior of these protocols in various scenarios will give us insights into designing new protocols for MANETs. To this end, we evaluate the performance of the Scalable Reliable Multicast (SRM) protocol [7]. SRM is one

of the early, wired reliable multicast protocols and can be considered representative of reliable multicast protocol behavior, as later developed protocols use common error control mechanisms similar to SRM (e.g., negative acknowledgments, multicasting of NACKs and retransmitted data, NACK suppression and local recovery). SRM was selected as we are particularly interested in protocols that rely exclusively on error recovery to achieve reliability. Our hypothesis is that, since MANETs are extremely sensitive to offered load, protocols that fall in this category will not perform well in MANETs.

The remainder of this paper is organized as follows. The performance of SRM in MANETs and the lessons learned from that study are presented in Section II. Section III introduces the RALM protocol and evaluates its performance. Section IV concludes the paper.

II. SRM IN MANETs

We evaluate the performance of SRM in ad hoc networks using simulation. For protocol details of SRM, readers are referred to [7]. We compare SRM against UDP. UDP is chosen because it provides basic multicast support without guaranteeing reliability. Therefore, any reliable multicast algorithm should demonstrate improvements over UDP.

We use QualNet [8] for our simulation. In our experiments, 50 nodes are placed randomly in a $1500m \times 1500m$ area. Constant bit rate traffic is generated by the application, with each data payload being 512 bytes. For SRM specific parameters, we use the values specified in [7]. Packets are multicast by ODMRP (On-Demand Multicast Routing Protocol) [9] and unicast by AODV (Ad hoc On-Demand Distance Vector) [10] with IEEE 802.11 DCF as the MAC protocol. The number of multicast sources is fixed at five while the number of receivers varies from 10 to 40, depending on the scenarios. The buffer size at each node is 30 packets. The channel capacity is 2Mb/s. Propagation is modeled using a two-ray ground reflection model where free-space path loss is used for near sight and plane earth path loss is used for far sight. The maximum radio propagation range is 375 meters. When mobility is considered, the random waypoint model is used with zero pause time; thus, nodes are constantly moving. It is important to note that SRM requires multicast sources to also be the multicast receiver. This is because REPAIR REQUEST packets are multicast to the group and thus sources are required to be group members to receive and respond to REPAIR REQUEST packets. Therefore, in all experiments, multicast sources are always a subset of the receiver set.

We subject SRM to a range of network characteristics, such as load, number of receivers and mobility. Due to space limitations, we only present a subset of results. For the experiment shown here, we have five multicast sources and ten multicast receivers. We vary the “application driven” data packet inter-departure time at each source from 500 ms to 100 ms. The purpose of this study is to evaluate the impact of traffic rates on protocol reliability.

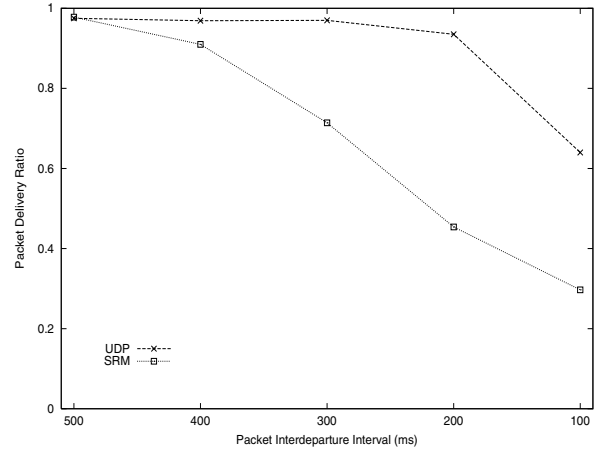


Fig. 1. Packet delivery ratio as a function of traffic rate.

A. Results and Observations

We observe from Fig. 1 that SRM experiences sharp degradation as we decrease the inter-departure time below 500 ms and drops to approximately 30% packet delivery ratio under the highest load. The poor performance of SRM stems from its attempts to recover dropped packets by injecting additional REPAIR REQUEST and REPAIR packets into the network without performing any congestion control. SRM uncouples loss recovery from normal multicast operations. When loss recovery is performed, both retransmitted and new packets are sent at the same time, thus increasing network congestion. Surprisingly, UDP obtains better data delivery than SRM even though it does not provide reliability. UDP achieves reasonable performance (above 90%) until the inter-departure time decreases to less than 200 ms. This result suggests that in MANETs, reliability cannot be accomplished by simply retransmitting lost packets and at the same time still maintaining the same sending rate. Congestion control must accompany loss recovery. Loss recovery is needed to guarantee data delivery while congestion control prevents the network from being overloaded and thus resulting in poor throughput performance.

We also learned from this study that error recovery must not incur too much overhead. Multicasting retransmission requests and lost data may not be the best approach; it introduces significant overhead, and multicast loss correlation in MANETs is not high enough to warrant the extra overhead, especially when nodes are mobile. Another drawback of multicasting retransmission requests is the additional overhead needed by the underlying multicast routing protocol. This is because each transport layer receiver must become a routing layer source and therefore establish its own multicast source tree or mesh. What is worse is the fact that if the retransmission requests do not occur frequently, most multicast routing protocol will time out the multicast structure and thus the source tree or mesh will have to be rediscovered, which is an expensive operation.

III. RELIABLE ADAPTIVE LIGHTWEIGHT MULTICAST

A. Protocol Operation

Each multicast source maintains a *Receiver List*. When a source receives a NACK from a node that is not in the *Receiver List*, the node is added to the list. Nodes are removed from the list when the source receives an ACK from them. In addition, the source keeps track of the end-to-end latency between itself and each receiver that sent NACKs through a timestamp.

The source initially multicasts data packets at the rate specified by the application and continues to do so until a NACK is received. Upon reception of a NACK, the source adds the NACK sender to the *Receiver List* and enters the loss recovery phase. The source initiates the loss recovery by selecting a receiver from the *Receiver List* to reliably transmit to. We call this node a *feedback receiver*. The source multicasts a new packet or retransmits the lost packet requested by the feedback receiver. The packet header includes information instructing the feedback receiver to reply via “unicast” with an ACK indicating that all packets are successfully received or a NACK with the packet sequence number informing that the source needs to retransmit the specified packet. All other receivers process the packet without replying to the source.

The goal of unicasting ACKs/NACKs instead of multicasting is to reduce control overhead and thus congestion and contention. Multicasting ACKs/NACKs would incur the overhead of the underlying routing protocol to flood the entire network with query packets to discover the members. There are a few protocols that do not flood to achieve multicast. Such protocols however, rely on unicast routing protocols that require proactive maintenance and therefore would incur additional overhead. Although unicasting NACKs also results in a flood search to locate the node, the overhead is less since only the destination replies. Unicasting NACKs also speeds up the loss recovery phase compared with multicasting since multicasting often relies on suppression techniques, which delays loss recovery.

When there are lost packets, the feedback receiver sends a NACK to the source to request the lost packets one at a time until it has all up-to-date packets (i.e., send-and-wait). The design philosophy behind retransmitting one packet at a time is to slow down the transmission of the source when congestion is detected. Note that because the source multicasts retransmitted packets, all group members will receive them. Since only the feedback receiver replies to the source, the ACK/NACK implosion problem is avoided. Moreover, to reduce the response overhead, NACK transmissions are rate limited to no more than once every κ seconds (κ value is five in our simulations).

By allowing only the feedback receiver to respond to the source, we avoid the inaccuracy of feedback suppression timers. This is particularly advantageous in a mobile environment where the majority of suppression techniques will fail since they often rely on distances and end-to-end delay measurements. Such measurements are often in flux when mobility exists and are ineffective in suppressing feedback.

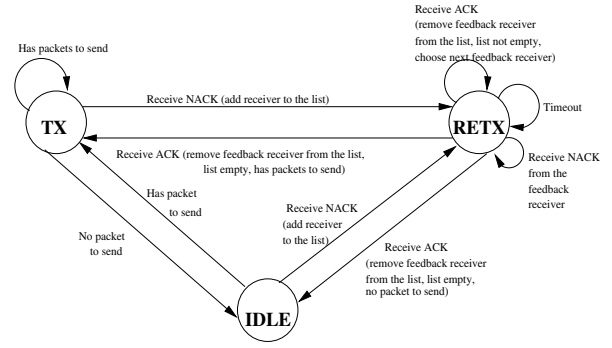


Fig. 2. RALM state transition diagram at the source.

Duplicate packets are discarded by the receivers. Note that retransmitted packets are *multicast* to the group since other receivers may also be missing these packets. The expectation is that as the source “visits” other receivers, those receivers would already have recovered the lost packets previously retransmitted by the source and thus no further retransmissions are required. Once the feedback receiver obtains all the packets, it unicasts an ACK to the source. Upon reception of the ACK, the source removes the node from the *Receiver List*, chooses a new feedback receiver in a round robin fashion, and repeats this process until the *Receiver List* is empty. Feedback receiver selection is not restricted to the round robin approach, although we do so here for simplicity. Other alternatives include selecting the feedback receiver based on common loss packets to reduce the number of retransmissions and oldest packet lost to limit latency.

When the *Receiver List* is empty, the source reverts to the application sending rate. If, however, the source does not receive a NACK or ACK from the feedback receiver after the timeout (determined by the measured end-to-end delay from the source to the feedback receiver via NACKs), the source backs off and tries again up to a maximum number of times. If the source still does not hear from the feedback receiver after the maximum number of retransmissions, it removes the node from the *Receiver List* and moves on to the next node in the list. The removed receiver may later re-synchronize with the source with the normal NACK mechanism.

The round-robin send-and-wait does not require retransmissions of the same lost packets multiple times to each receiver. In the best-case scenario, lost packets are retransmitted only once by the source since retransmissions are multicast. For instance, if a set of receivers lost the same packet, it is retransmitted only once assuming the retransmitted packet is received by all the receivers. In the worst-case scenario, each receiver experiences different packet losses. In this case, all lost packets must be retransmitted to each receiver. Remember also that when loss recovery mode is entered, the source first sends a new packet instead of retransmitting the lost packet if there are any new packets in its buffer.

Fig. 2 depicts RALM state transition graph. Note that a node can unicast a NACK to the source also when no feedback receiver is specified in the packet (i.e., indicating that the

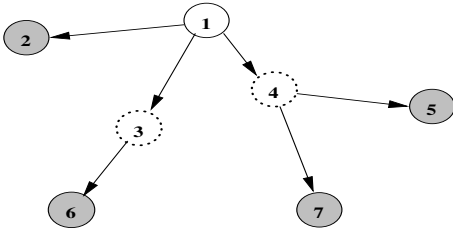


Fig. 3. RALM in operation.

source is not in loss recovery mode).

Fig. 3 shows the example of RALM operation. Node 1 is the source and nodes 2, 5, 6 and 7 are the multicast receivers. Node 1 begins by multicasting packets at the application sending rate. Assume that after some duration, nodes 6 and 7 detect packet losses and each transmits a NACK to node 1. Upon receiving NACKs, node 1 adds nodes 6 and 7 to its *Receiver List*. Node 1 then selects a node from the list, say node 6 as the feedback receiver, and multicasts a new packet (if any) or retransmits the requested lost packet. This data packet will instruct node 6 to respond to the source. All other receivers, including node 7, are prohibited from sending any feedback, thus avoiding the feedback implosion. Node 1 waits for either node 6 to reply or a timeout to occur. If a NACK is received from node 6, node 1 continues to multicast the lost packet indicated in the NACK. If a timeout occurs, the packet is retransmitted after a backoff. When an ACK is received, node 1 removes node 6 from the *Receiver List* and chooses node 7 as the next feedback receiver to perform loss recovery. It is important to note that retransmitted packets are multicast. If node 7 missed the same packets as node 6 and node 7 received them when they were retransmitted, node 7 does not need to request retransmission of those packets. Upon receipt of an ACK from node 7, node 1 removes node 7 from the list. Since the *Receiver List* is now empty, node 1 reverts to the original application sending rate.

B. Results and Analysis

We compare the performance of RALM with that of SRM and UDP. Simulation configurations and parameters are the same as described in Section II. The performance metrics we use are packet delivery ratio, control overhead, and end-to-end delay. Packet delivery ratio measures the effectiveness and reliability of a protocol. Control overhead is defined as the ratio between the total number of data and control packets transmitted by the routing and transport layer protocols and the number of data packets received by the multicast receivers. It is used to assess protocol efficiency. End-to-end delay measures data packet latency from the source to the destination and evaluates the protocol's timeliness. Due to space limitations, we show only the subset of results that are of interest.

1) *Traffic Rate*: We start by examining the performance of RALM under varying traffic rate. We point out that the application sending rate under RALM is the rate RALM uses when it is not in loss recovery mode. Under loss recovery/congestion

TABLE I
PERCENTAGE OF TIME SPENT IN LOSS RECOVERY MODE BY RALM.

Interval (ms)	500	400	300	200	100
Percentage	0.2	0.5	0.7	4	12

control mode, RALM abandons the application sending rate in favor of a send-and-wait approach.

We see from Fig. 4(a) that RALM achieves 100% packet delivery ratio for all traffic rates. As the application rate increases and thus more packet losses occur due to congestion, RALM slows down the transmission rate by entering loss recovery/congestion control mode with the send-and-wait approach. Table I shows the percentage of simulation time that is used for loss recovery by RALM. At a rate of 2 packets per second, RALM spends 0.2% of the simulation time in loss recovery. At such a low traffic rate, packet losses are infrequent. As the traffic rate increases, we see that the percentage of time RALM spends in loss recovery increases and up to 12% for the rate of 10 packets per second.

Fig. 4(b) shows that RALM is comparable to UDP and is superior to SRM in control overhead. The latency of RALM matches with UDP for the low and medium data rates and improves upon UDP under high rates, as shown in Fig. 4(c). Fig. 4(d) reveals that RALM approaches only 2,000 packets sent while for SRM and UDP, the number of packets sent directly relates to the application sending rate. Here, we witness RALM's congestion control mechanism at work; the higher the application sending rate, the more RALM adjusts.

2) *Mobility*: We turn our attention to the mobility experiment. Fig. 5(a) shows that RALM still obtains a packet delivery ratio of 100% in all of the mobility speed we examined. RALM outperforms both SRM and UDP. In all other metrics of interests, RALM is comparable to that of UDP (not shown due to space constraints).

3) *Sensitivity to Random Errors*: In a wireless network, random errors are common (e.g., external radio interference, jamming, etc). These errors cause packet losses that are not related to congestion. RALM still recovers from such losses. Nevertheless, as RALM is based on a TCP-like scheme that interprets losses as an indication of congestion, it is important to investigate the impact of random errors on RALM performance. Experiments on mixed wired/wireless networks [3] have shown performance degradation with packet loss rates as low as 1%, because of large operating TCP window. Packet losses cause drastic window reduction and performance degradation. On the other hand, the operational window in RALM is $W = 1$ (send-and-wait). Random losses have little effect on its performance as confirmed in Fig. 5(b).

We note that a 4% random loss rate (at $1.0e-5$ since data size is 512 bytes) has negligible impact on delay. Even at 40% packet loss rate, the delay remains within a reasonable range, showing the protocol robustness to random errors and losses.

4) *RALM vs. Multiple Unicast TCP Sessions*: Our simulation results indicate that RALM attains reliability while maintaining control overhead that is comparable to UDP. In

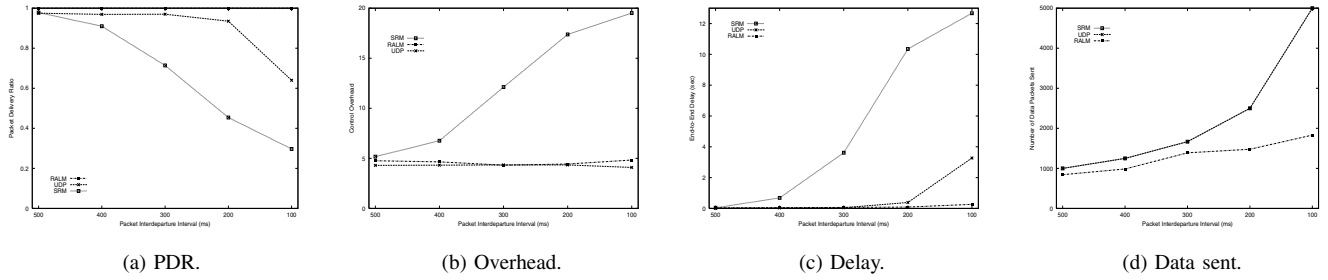


Fig. 4. Traffic rate experiments.

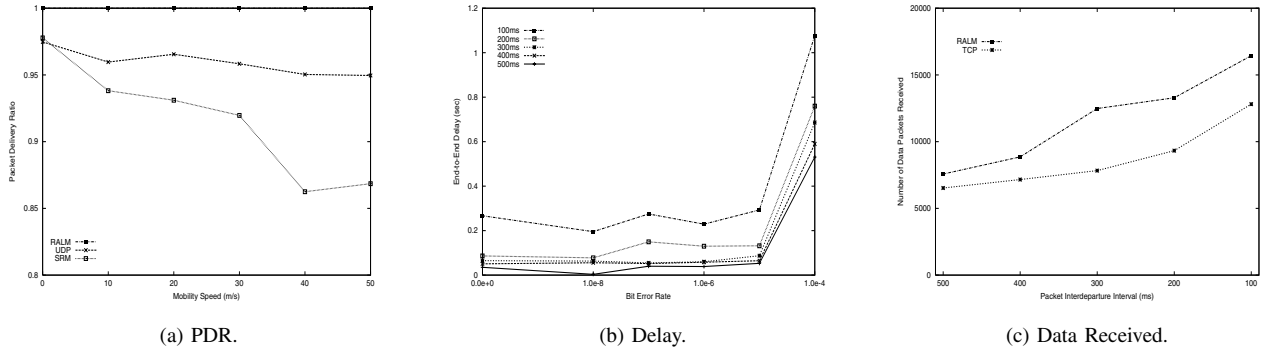


Fig. 5. More simulation results.

order to consider RALM as a viable reliable multicast transport protocol, it must also show superiority when compared with multiple TCP unicast streams. We revisit the traffic rate experiments and configure individual TCP unicast sessions to duplicate the behavior of multicast. Since there are five sources and ten receivers, each source will run a TCP unicast session with each of the nine other receivers. RALM and multiple TCP unicast sessions are run separately.

We see from Fig. 5(c) that RALM achieves better throughput than TCP. RALM receives approximately 24% more data packets than multiple unicast TCP. TCP guarantees reliable delivery, so the packet delivery ratio is not shown. Note that the number of multicast receivers in this experiment is ten. We expect RALM to further outperform TCP as the number of receivers increases.

IV. CONCLUDING REMARKS

We proposed Reliable Adaptive Lightweight Multicast (RALM) protocol for ad hoc networks. The design principles behind RALM were motivated by our experience evaluating the performance of SRM in MANET environments. We learned that in MANETs, reliable delivery cannot be achieved by error control alone; congestion control is fundamental for reliability. RALM is a transport protocol that provides both error recovery and congestion control. It achieves reliability by guaranteeing data delivery to one multicast member at a time in a round-robin fashion. RALM uses send-and-wait approach for congestion control. This not only ensures that

nodes receive the lost packets, but also that the sources are self-clocked (*a la* TCP) not to send packets faster than they receive ACKs when congestion is experienced. Simulation results showed that RALM performs well in diverse ad hoc network scenarios. It achieved perfect reliability while yielding low control overhead and latency compared with UDP and SRM.

REFERENCES

- [1] R. Chandra, V. Ramasubramanian, and K. Birman, "Anonymous gossip: Improving multicast reliability in mobile ad-hoc networks," in *Proceedings of the IEEE ICDCS*, Apr. 2001, pp. 275–283.
- [2] K. Obraczka, "Multicast transport mechanisms: A survey and taxonomy," *IEEE Commun. Mag.*, vol. 36, no. 1, pp. 94–102, Jan. 1998.
- [3] H. Balakrishnan, S. Seshan, and R. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *ACM/Baltzer Wireless Networks*, vol. 1, no. 4, pp. 469–481, Dec. 1995.
- [4] K. Brown and S. Singh, "Relm: Reliable multicast for mobile networks," *Computer Communications*, vol. 21, no. 16, pp. 1379–1400, Oct. 1998.
- [5] L. Rizzo and L. Vicisano, "RMDP: an FEC-based reliable multicast protocol for wireless environments," *ACM Mobile Computing and Communications Review*, vol. 2, no. 2, pp. 23–31, Apr. 1998.
- [6] K. Tang, K. Obraczka, S.-J. Lee, and M. Gerla, "A reliable, congestion-controlled multicast transport protocol in multimedia multi-hop networks," in *Proceedings of the IEEE WPMC*, Oct. 2002.
- [7] S. Floyd, V. Jacobson, S. McCanne, C. Liu, and L. Zhang, "A reliable multicast framework for lightweight sessions and application-level framing," *IEEE/ACM Trans. Networking*, vol. 5, no. 6, pp. 784–803, Dec. 1997.
- [8] QualNet. [Online]. Available: <http://www.scalable-networks.com>
- [9] S.-J. Lee, W. Su, and M. Gerla, "On-demand multicast routing protocol in multihop wireless mobile networks," *ACM/Kluwer Mobile Networks and Applications*, vol. 7, no. 6, pp. 441–453, Dec. 2002.
- [10] C. Perkins and E. Royer, "Ad-hoc on demand distance vector routing," in *Proceedings of the IEEE WMCSA*, Feb. 1999, pp. 90–100.