Coordinated Interaction Using Reliable Broadcast in Mobile Wireless Networks

Theodore L. Willke[1,2] and Nicholas F. Maxemchuk[1]

[1] Columbia University, Dept. of Electrical Engineering
1312 S.W. Mudd
500 West 120th Street
New York, NY 10027
tlw24@columbia.edu, nick@ee.columbia.edu
[2] Intel Corporation, Enterprise Platforms Group
2800 Center Drive, M/S DP3-307
DuPont, WA 98327
theodore.l.willke@intel.com

Abstract. We consider the challenges of supporting collaborative decision making and applications requiring coordinated interaction in mobile ad hoc networks. This environment makes group coordination difficult due to relatively high packet loss and the presence of a continuously evolving network topology that leads to a changing set of participants. We describe and evaluate an application-level mobile reliable broadcast protocol (M-RBP) that provides applications with network characteristics well-suited to supporting coordinated interaction, such as reliable broadcast with data consistency and global ordering. The protocol is time-, rather than event-, driven, providing it with unique, deterministic operational characteristics that can be verified in a relatively straightforward manner.

1 Introduction

Wireless mobile computing and sensor networks may consist of nodes that desire to collaborate to find an interactive solution to a problem. These nodes can combine information from local sources, such as node location and other measurements, with critical information shared by their peers. By contributing this information to a global view of the problem space, the nodes are enabled to act coherently and achieve common goals.

One example of a coordinated interaction problem involves a lane of vehicles traveling in a group on a highway. The leading automobile may be faced with a decision to either brake aggressively or veer to avoid a serious accident. Its on-board computer may determine that hard braking is the lowest risk solution, as long as the vehicles close behind brake as well. If the on-board computer can broadcast this intention to the other vehicles and confirm their reception and compliance with the message by the critical deadline, the automobiles can rapidly decelerate and avoid an accident. To do this requires a communication protocol that supports reliable broadcast with consistent message delivery and confirmation.

Reliable broadcast and multicast for mobile ad hoc networks (MANETs) has been studied intensively in recent years [1], [2], [3], [4], [5], [6]. Many of the protocols developed drastically improve the reliability of packet delivery, but do not provide a framework for data consistency (commitment and ordering). Token ring-based protocols pursued by other researchers, such as the work described in [7], [8], may ultimately support this framework; however they currently lack reliable message delivery mechanisms.

The Mobile Reliable Broadcast Protocol (M-RBP) was developed to support coordinated interaction and collaborative decision making in mobile ad hoc networks. It was initially described in [9]. The protocol guarantees that nodes able to make forward commitment progress will:

- Put messages in the same message order.
- Commit the same set of messages. With this and the previous attribute, each copy of the distributed program can reach the same conclusion.
- Be able to verify that a specific set of collaborators have received a message. By guaranteeing this, the protocol can enable coordinated operations.

In this paper, we provide precise operational guarantees and initial simulation results for M-RBP. In particular, we show that the protocol operates efficiently in the presence of arbitrary losses for a reasonably-sized group of collaborators located in a local geographical region. We also determine the network topologies for which the addition of an underlying routing or flooding protocol would be most beneficial.

The remainder of this paper is organized as follows. The problem of coordinated interaction is described in Section 2. Section 3 reviews the operational assumptions. Section 4 describes the protocol's operation. Section 5 describes the simulation methodology, and Section 6 presents the results with analysis. Section 7 concludes the paper.

2 Coordinated Interaction

In a replicated database, identical copies of a transaction commitment record are kept at more than one location, and more than one entity may submit and coordinate transactions [10], [11], [12]. We can draw an analogy between distributed database replication and coordinated interaction by noting that the requirement for maintaining a replicated, in order, transaction history is similar to the requirement of sharing a global view of the problem space in interaction problems. We desire to transform this sort of algorithm into one that presumes nothing about the stability of network resources or connections, and that is useful for general coordinated interaction applications. This distributed algorithm may carry out globally-optimized interactions by enabling entities to reliably distribute local information (e.g., sensor inputs or GPS location) to subsets of peers and gather pertinent feedback.

For nodes in a group to form a cohesive view of the problem space and coordinate their actions, they must be able to confirm that specific messages were received by specific peers, and that these peers are capable and willing to act on the supplied message content. In some cases the requester may not want an action to be carried out unless a specific set of nodes receive the message and respond appropriately. M-RBP provides this capability by using a process similar to a two-phase database commitment process.

In the simplest two-phase commitment process, shown in Figure 1a, a transaction is submitted to subordinates (each in initial state *q*) by a coordinating site, as described in [10]. To complete the first phase, the subordinates receiving, and able to commit, the transaction respond with a "*YES*" vote, move to wait state *w*, and the coordinator collects these votes. If one or more nodes respond "*NO*", or fail to respond the transaction will be aborted (state *a*). In phase two, the coordinator tabulates the votes, and, based on the quorum rule selected, will either issue a command to commit (state *c*) the message or abort it.

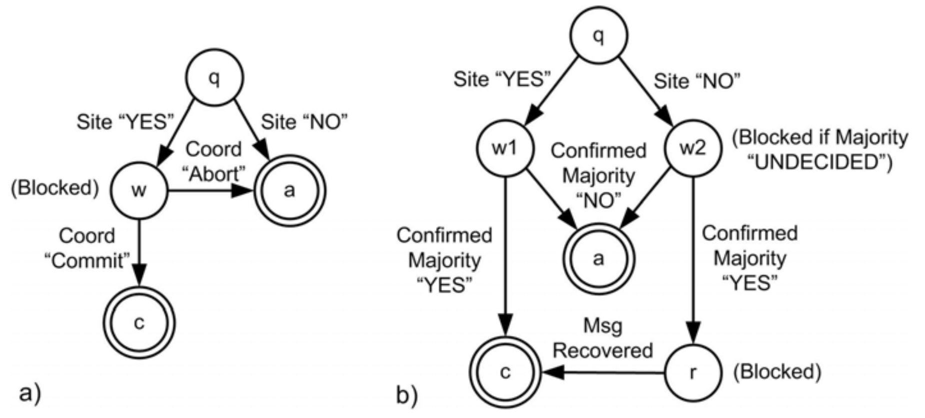

Fig. 1. a) Basic two-phase commit process (blocking) used in distributed databases, and b) M-RBP's two-phase commit process

The voting process accomplishes two things: 1) It ensures that the coordinator is made aware of potential divergences in database replications, and 2) in partitionable networks, it ensures that only one partition is committing messages at any given instant so that only one unique transaction history exists.

M-RBP takes advantage of the voting process in a modified form of the two-phase commit process that is shown in Figure 1b. In the first phase (state *q*), an elected node acknowledges the source message. Then, after a recovery period, each node individually votes "*YES*" or "*NO*" to commit the message. Using peer-to-peer communication, the nodes attempt to recover as many peer votes as possible to confirm a majority "*YES*" or "*NO*" vote. A node cannot leave the wait states, *w*1 and *w*2, until it determines a majority. Usually the decision leads to commit state, *c*, or abort state, *a*. In rare cases, the majority may vote to commit a message, whereas the individual failed to recover the message by a deadline. In this case, the node remains blocked in state *r* until it recovers the message from a peer.

The M-RBP approach to the two-phase commit process permits it to proceed in a time-driven manner, with no coordinator-slave interaction. Also, the commit quorum is a simple majority and the majority is determined by each node in a peer-to-peer manner. Nodes that block are responsible for unblocking themselves, and mechanisms are built in to the protocol to permit the group to know when a peer is blocked.

3 Operational Assumptions

Coordinated interaction is difficult in infrastructure-based IP networks, and even harder in MANETs because: 1) the network topology may continuously evolve leading, in turn, to a changing group of collaborators and 2) the wireless broadcast medium is relatively unreliable due to transmission limitations, noise, and contention. We focus in this paper on applications that involve a set of mobile collaborators residing in a geographically-localized MANET, limiting the size of the region to several transmission distances, as shown in Figure 2.

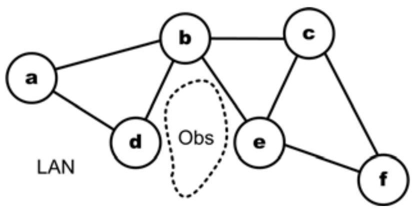

Fig. 2. A set of mobile collaborators in a small, localized, multi-hop network

M-RBP operates with the following assumptions about the nodes and the network:

- Communication links may fail or recover at any time. M-RBP was designed for the application layer and makes no assumption about the presence, or lack of, an underlying unreliable routing or flooding protocol. In its current form, M-RBP is designed to perform adequately if all control and data messages are simply broadcast to one-hop neighbors.
- The network is partitionable. One or more partitions may form temporarily or permanently. A primary partition is one in which a majority of group members reside.
- Nodes may fail and, when they do, they stop transmitting altogether. That is, Byzantine failures are not permitted.
- Messages may be received, not received, or received with CRC errors. If received with errors, the packets are dropped without notification to M-RBP.

4 Protocol Operation

An early description of M-RBP can be found in [9]. The abbreviated description of M-RBP that follows emphasizes the means by which it ensures global message ordering and consistent message commitment (or abort), as well as group membership services for the participating nodes.

The basic framework of M-RBP consists of a token ring of receivers, some of which may also be message sources. Figure 3 is a diagram of sources and receivers in the token ring, adapted from [13]. Transmitted source messages are identified by source number, *s*, and source sequence number, M_s. The *n* receivers take turns as the token site, passing an implicit, time-based token every Δ_T seconds in an order defined by a shared data structure called the *Token Passing List*. As each token site relinquishes the token, it transmits an *ACK* with a globally-unique acknowledgement sequence number. The *ACK* references the messages received during the token interval and assigns them a relative order.

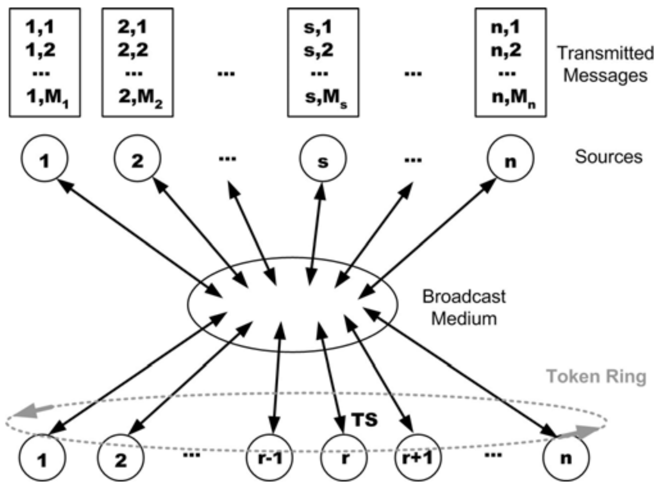

Fig. 3. Message sources transmit messages into the unreliable broadcast medium and receivers in a time-driven token ring take turns acknowledging and sequencing these messages

Beginning a short time, Δ_R, after the <u>scheduled</u> *ACK* transmission at $m{\cdot}\Delta_T$, where *m* is a positive integer, all receivers that did not receive the initial broadcast of the *ACK* request its retransmission using an *ACK Retry*. Once the *ACK* is recovered, any missing source messages are requested using a *NACK*. Retry and retransmission implosion is suppressed using mechanisms described in [9].

Starting at time $m{\cdot}\Delta_T + k{\cdot}\Delta_R$, after *k* potential retry rounds, <u>scheduled</u> *ACK*s contain information on earlier *ACK*s and source messages that could not be recovered. This information on missing *ACK*s and source messages is used as a vote to determine what *ACK*s to use for sequencing and what messages to commit.

4.1 Group Membership Service

Any node that wants to join the group requests timing and state information from a current group member and then transmits a source message including a join request. When, and if, the source message is committed, the new member is added to each node's copy of the *Token Passing List*.

To leave the group, a node may transmit a source message including a drop request. When, and if, the source message is committed, the member is removed from the *Token Passing List* and must cease transmitting M-RBP messages.

A more difficult problem is coping with receivers that fail, or are isolated, unexpectedly. To keep the timed token ring operating, the protocol requires a scheme to detect when this happens and reassign the token slots. The method chosen is to use unrecoverable *ACK*s as an indication of node failure. At a deadline peers vote, via a field in their own scheduled *ACK*s, on which *ACK*s are missing. Each member of the group uses the votes that it recovered and the following agreement function, *F*, to decide whether to remove the node in question from its copy of the *Token Passing List*:

$$F = \begin{cases} \mathrm{Y} & \text{if } \sum \text{"Yes"} \geq \mathrm{C(t)} \\ \mathrm{N} & \text{if } \sum \text{"No"} > \mathrm{C(t)} \\ \mathrm{U} & \textit{o.w.} \end{cases}, \quad (1)$$

where *C*(*t*) is one-half the number of entries on the *Token Passing List* (rounded up) at the time, *t*, that the vote ends. This agreement function returns a *Yes* (*Y*) or *No* (*N*) drop decision if a majority of the nodes voting return "*Yes*" or "*No*" votes. If a majority is not recovered, the vote remains *Undecided* (*U*). A node with an undecided vote remains blocked until it recovers the vote outcome from a peer. The node must then re-join the token ring.

The voting and agreement timeline is summarized in Figure 4.

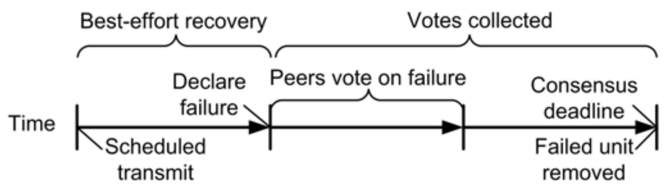

Fig. 4. Timeline for best-effort *ACK* recovery, peer voting, and peer consensus to remove a node that failed to transmit its scheduled *ACK*

4.2 Global Ordering and Consistent Commitment

The process used to build a consensus on nodes that have unexpectedly failed is also used to ensure global message ordering and consistent message commitment by each member of the group. A consensus process is required because the unreliable wireless network and best-effort packet recovery processes can result in each node recovering a different subset of the available source message and acknowledgement information.

If nodes vote and reach agreement on both the *ACK*s to use for message ordering and the source messages to commit, all non-blocked nodes can achieve a consistent global view of the message history. Furthermore, since only non-blocked nodes continue to participate in the token ring and transmit *ACKs* as the token site in the token round following a message commitment, all peers, including the message source, can verify the set of non-blocked nodes that received a message. The process timeline is summarized in Figure 5.

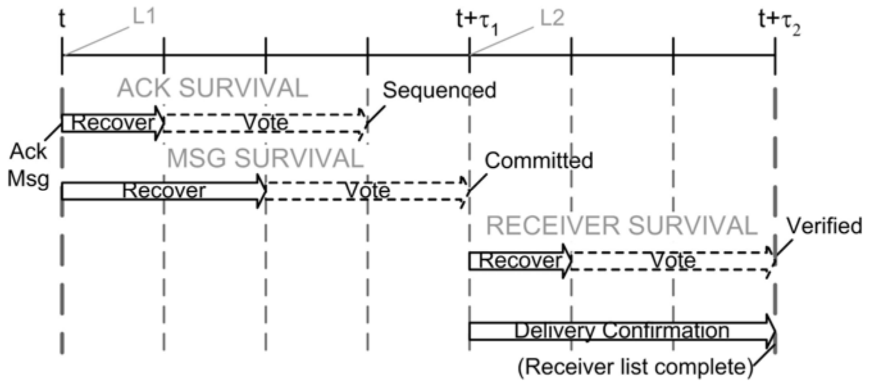

Fig. 5. *ACK* and source message commitment and receiver verification timeline

The first phase in the process is *L1* commit, or message injection. Sources retransmit messages until they receive an *ACK*. At time *t*, the message source, as well as other nodes in broadcast range, receives the *ACK* that references the message. Subsequently, the entire group attempts to recover and reach majority agreement on whether or not to use the *ACK* for message sequencing.

The second phase of message commitment begins with active message recovery, which starts when *ACK* recovery ends, and ends with *L2* commit, the point at which the message may be sent to the application. As with *ACK*s, a majority must recover the message and vote "*YES*" in order to use it. If this happens, the message is committed. This policy ensures that: 1) decisions are made only by a primary partition, and 2) there is a high probability that all peers in the primary partition will successfully commit the message. To maintain consistency, nodes cannot reverse a decision to commit or abort a message after reaching one.

Because the protocol is time-driven, all nodes that can reach a decision to commit will do so at a deterministic time, $t + \tau_1$, following initial message injection into the network. Nodes that are not in the primary partition, or that experience transient communication failures for an amount of time, are blocked from committing or aborting messages, and these messages remain undecided. Nodes that re-join the primary partition and retrieve vote outcome information may then disposition undecided messages.

By time $t + \tau_2$, the non-blocked members of the primary partition are able to verify the set of peers that committed the message. Sources may retransmit messages that are aborted at *L2*, or that do not reach the intended receivers.

Since a lossy broadcast network provides no FIFO guarantee for message delivery order, all nodes use the ordered set of acknowledgements to globally sequence messages. Global message ordering requires two steps. First, relative message order is assigned in each (bulk) *ACK*. Second, the *ACK*s, and their referenced messages, are ordered by increasing *ACK* sequence number, with the dropped *ACK*s and messages removed from sequencing. The earliest instance of a message reference is used; any duplicate references are ignored.

5 Simulation Methodology

Since M-RBP is the first reliable broadcast protocol for mobile wireless networks to provide global data ordering and consistency guarantees, it is impossible to quantitatively compare its performance with existing methods. However, because it is a time-, rather than event-, driven protocol, many of the simulation results are easily analyzed.

The performance characteristics of M-RBP in a MANET were evaluated using the QualNet simulator [14]. M-RBP was implemented as an application that receives data from a traffic generator application and passes control or data messages to/from other network stack layers. The supplied library models for UDP, IPv4 and the 802.11 DCF MAC [15] were also used.

The constant bit rate traffic consisted of source messages with 512 byte data payloads. The message interdeparture interval varied by experiment. The channel capacity was fixed at 2 Mb/s. The propagation range was 375 meters and a two-ray propagation model was used. The channel characteristics match those used extensively in the literature on mobile ad hoc networking.

22 nodes were randomly placed in a field size that varied by experiment. All nodes were instantiated with a copy of M-RBP and the network stack. A subset of the nodes, determined in each experiment, were chosen to be message sources and implemented copies of the traffic generator. A random waypoint mobility model with zero pause time was used and the mobility speed was varied by experiment.

6 Results and Analysis

The performance metrics we used are packet delivery ratio, additional messages per source message committed, total number of packets transmitted during the simulation, and commitment delay. Packet delivery ratio is the portion of *L1* committed messages that achieve *L2* commitment. The additional messages per source message committed is the average number of data and control messages (i.e., source message retransmissions, *NACK*s, scheduled *ACK*s, *ACK* retransmissions, and *ACK* Retries) transmitted per *L2* commit by the group. The total number of packets transmitted during the simulation is a measure of total network load. The commitment delay is the time for a *L1* committed message to reach *L2* commit, where the information can be passed to the application.

In all scenarios tested, the packet delivery ratio was 100%. Additionally, data consistency and ordering was checked across the set of receivers in the group and found to be correct in all scenarios tested.

Traffic Rate - Since M-RBP uses bulk acknowledgements and a fixed acknowledgement rate, we expect the protocol to become more efficient as traffic rate increases. This is illustrated in Figure 6, where the traffic rate is expressed as the average number of source messages transmitted per token passing interval (4 sources). High and low node density cases were simulated. In the high density case, all nodes in the group are within one hop of each other, whereas, in the low density case, nodes can reach ~20% of the remainder of the group in one hop. The high density results are very close to the lower theoretical bound, which is comprised of the average number of scheduled ACK transmissions per source message commitment.

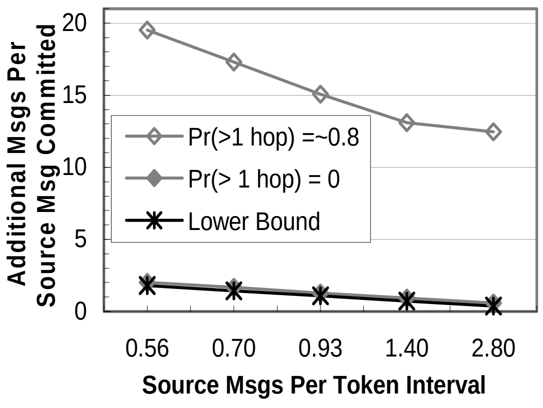

Fig. 6. Protocol efficiency as a function of traffic rate

Commitment Delay – Applications desire a low commitment delay to enable fast program reaction time. M-RBP's commitment delay is a deterministic function of the token passing rate, group size, and best-effort recovery policies. The protocol packet overhead is also a function of these parameters and the offered source message load.

To determine the relationship between protocol overhead and commitment delay, we ran a series of simulations in which the source message load was kept constant while the commit delay was varied. A relative commitment delay of 1.0 is defined as the delay that results for a group of 22 nodes, a 30 ms token passing interval, a maximum retry count of 15, and a retry period of 24 ms. Four CBR sources injected packets every 500 ms into a network of nodes moving at 30 m/s in a 750 m x 750 m field.

The results are shown in Figure 7 for a 40 second simulation. Note that the number of packets committed in each run is approximately the same. The number of total packets transmitted reduces exponentially as the commitment delay is relaxed, and is strongly dependent on control packet load (i.e., *NACK*s, *ACK*s, and *ACK Retries*).

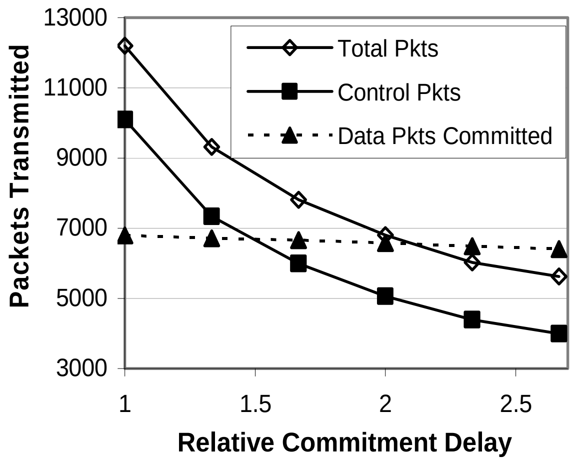

Fig. 7. Total number of data and control packets transmitted during a simulation run, as a function of the relative commitment delay

Field Size – To study the effect of hidden nodes, or multi-hop networking, on protocol overhead, the field size was varied for a group of constant size. Changing the field size effectively alters the probability, *Pr*(> 1 *hop*), that any given node is greater than one hop from the original message or *ACK* source. For *Pr*(> 1 *hop*) = 0, the entire group is a clique. Traffic load matched that used in the commit delay experiment.

The results are shown in Figure 8. The protocol is reasonably efficient as long as at least ~40% of the nodes are within one hop of each other. This suggests that the protocol may benefit from an underlying routing or flooding algorithm in networks with larger hop radii, but that the protocol operates efficiently in small networks.

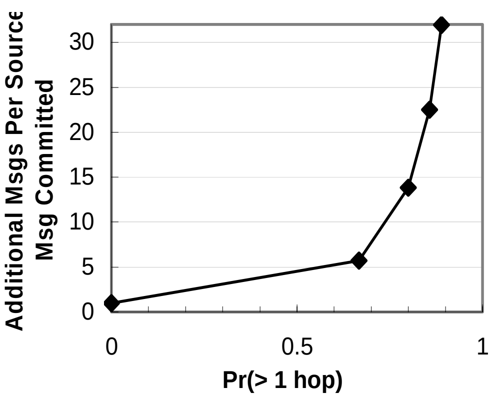

Fig. 8. The additional packet overhead required, as a function of the probability that any given group member is greater than one hop from the original message or *ACK* source

Mobility Rate – To study the effects of node mobility rate on protocol efficiency, simulations were run at a field size of 750 m × 750 m, with the same traffic load as the commit delay experiment. Node speed was varied from 0 m/s to 60 m/s, in 15 m/s steps. Because M-RBP does not rely on topology-related state information, its performance does not degrade as mobility rate is increased. In fact, as shown in Figure 9, the protocol performs better over a wide range of terrestrial speeds when compared to the static scenario. The increased efficiency is likely due to opportunistic information spreading and increased randomness in retry and retransmission timing.

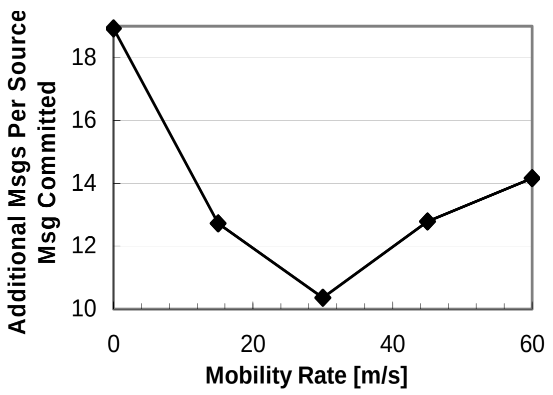

Fig. 9. Protocol packet overhead as a function of node mobility rate using a random waypoint model and zero pause time

7 Concluding Remarks

We have presented the problem of coordinated interaction amongst mobile nodes in a wireless ad hoc network and described a mobile reliable broadcast protocol, M-RBP, that can provide deterministic reliability guarantees, data consistency, and ordering, to support this challenging class of application. The functionality provided by M-RBP resembles that provided by a distributed database with data replication, and is what sets it apart from other highly-reliable broadcast and multicast protocols developed for MANETs.

We have provided a detailed overview of M-RBP's operation and have investigated its characteristics using the QualNet simulator. At moderate levels of network loading, M-RBP operates with low control overhead, due its use of bulk acknowledgements, *NACK*s, and *ACK Retries*. Its time-driven nature supports deterministic commitment delays for non-blocked nodes, but there is a price to be paid for lower commitment delays, in terms of packet overhead.

M-RBP works efficiently over wide range of mobility rates pertinent to terrestrial applications. It was shown to operate efficiently without an underlying routing or flooding protocol, even when most of the group is not in direct communication with one another. In larger multi-hop networks, M-RBP should maintain its efficiency with the addition of an unreliable, state-based message forwarding layer.

References

1. Chandra, R., Ramasubramanian, V., Birman, K., Anonymous Gossip: Improving Multicast Reliability in Mobile Ad-Hoc Networks, Proc. IEEE ICDCS, (2001) 275-283
2. Luo, J., Eugster, P. Th., Hubaux, J.-P. , Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks, IEEE Proc. INFOCOM, (2003) 2229-2239
3. Tang, K., Gerla, M., MAC Reliable Broadcast in Ad Hoc Networks, IEEE MILCOM, (2001) 1008-1012
4. Tang, K., Obraczka, K., Lee, S.-J., Gerla, M., Reliable Adaptive Lightweight Multicast Protocol, Proc. IEEE Intl. Conf. on Comm., vol. 2, (2003) 1054-1058
5. Gopalsamy, T., Singhal, M., Panda, D., Sadayappan, P., A Reliable Multicast Algorithm for Mobile Ad Hoc Networks, Proc. ICDCS, (2002) 563-570
6. Liao, W., Jiang, M.-Y., Family ACK Tree (FAT): Supporting Reliable Multicast in Mobile Ad Hoc Networks, IEEE Trans. Veh. Tech., vol. 52, no. 6, (2003) 1675-1685
7. Malpani, N., Chen, Y., Vaidya, N.H., Welch, J.L., Distributed Token Circulation on Mobile Ad Hoc Networks, to appear in IEEE Trans. Mobile Computing, (2004)
8. Lee, D., Attias, R., Sengupta, R., Tripakas, S., A Wireless Token Ring Protocol for Ad-Hoc Networks, Proc. IEEE Aerospace Conf., (2002)
9. Willke, T., Maxemchuk, N., Reliable Collaborative Decision Making in Mobile Ad Hoc Networks, Proc. 7th IFIP/IEEE Intl. Conf. MMNS, (2004) 88-101
10. Keidar, I., Dolev, D., Increasing the Resilience of Atomic Commit, at No Additional Cost, Proc. 14th ACM Sym. Principles Database Sys., (1995) 245-254
11. Amir, Y., Danilov, C., Miskin-Amir, M., Stanton, J., Tutu, C., On the Performance of Wide-Area Synchronous Database Replication, Technical Report CNDS-2002-4, Johns Hopkins University (2002)
12. Jajodia, S., Mutchler, D., Dynamic Voting Algorithms for Maintaining the Consistency of a Replicated Database, ACM Trans. Database Sys., vol. 15, no. 2, (1990) 230-280
13. Maxemchuk, N., Reliable Multicast with Delay Guarantees, IEEE Comm. Mag., vol. 40, no. 9, (2002) 96-102
14. QualNet User's Manual, version 3.6, Scalable Network Technologies, Inc., (2003)
15. IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, P802.11, (1999)