RBP: Reliable Broadcast Propagation in Wireless Networks

Fred Stann, John Heidemann, Rajesh Shroff, Muhammad Zaki Murtaza
USC-ISI-TR-2005-608 *November 2005*

Institute, 4676 Admiralty Way, Marina Del Rey, CA, USA Email: fstann@usc.edu, johnh@isi.edu

Abstract

Low-powered radios, interference patterns, and multi-hop connections make most wireless networks inherently unreliable. While MAC protocols provide high reliability for unicast via mechanisms such as ARQ, higher-layer network protocols often require broadcast transmissions as well. Prior work on broadcast improvements has focused on efficiency and generally not related reliability to general routing protocols. Existing broadcast protocols often require multi-hop topology information. Instead, we present a very simple protocol, requiring only local information, and highlight the performance impact of our protocol on routing with a secondary evaluation of energy efficiency. We develop our protocol as a service between the MAC and network layer, taking information from both. Our approach is based on two principles: First, we exploit network density to achieve near-perfect end-to-end reliability by requiring moderate (50-70%) reliability when nodes have many neighbors. Second we identify areas of sparse connectivity where ***important links*** *bridge clusters of dense nodes, and guarantee connectivity over those links. Routing performance depends on wireless propagation, so we develop a new error model that considers both correlated and independent loss in broadcast traffic based on testbed experiments. We demonstrate, through controlled simulations using this model, and through complete testbed experiments, that this hybrid approach is necessary to provide near-perfect accuracy with good efficiency. In a real testbed we show 99.8% accuracy with 48% less overhead than through repeated flooding. The contributions of this paper are the introduction of our protocol* ***Reliable Broadcast Propagation****, definition of metrics that balance efficiency and reliability, and introduction of a more accurate model for broadcast error.*

1 Introduction

Numerous studies have documented a wide variance in reliable packet delivery for energy-constrained wireless environments [40, 38, 41, 31]. This physical-layer problem can be ameliorated for unicast packets via MAC layer mechanisms such as CTS/RTS/ACK, ARQ, and FEC [16, 39]. Unfortunately, control traffic overhead has prevented such techniques from being effective for PHY layer broadcasting.

There is a wide range of prior work looking at improving broadcast reliability and efficiency. Reliable broadcast schemes based on TDMA [6] and self-pruning [3], employed in high powered wireless networks do not adapt well to low power networks where dynamic connectivity and poorly convergent global knowledge are the norm. Approaches like application layer jitter, MAC layer random slot selection [36], and PHY layer capture [35] can help to lower the probability of unrecoverable broadcast collisions, but they don't guarantee broadcast propagation. Finally, very few proposed improvements for broadcasting have been tested outside of simulation. Real-world topologies rarely have uniform density. Node density in wireless networks is particularly important for broadcasting because it correlates to the number of disjoint paths between node pairs, which has a major impact on the probability of eventual packet delivery [26]. So, from a MAC and PHY layer perspective, reliable broadcasting can be difficult in some environments. Perhaps because of these challenges, widely available protocols, such as 802.11, do not provide general support for reliable broadcast.

Nonetheless, broadcasting is an integral part of popular protocols for data dissemination in wireless networks. When AODV [24] has no route to a destination node, it broadcasts Route Request messages. Similarly, ODMRP [17] broadcasts a Join Query to initially find target nodes. Tiny DB floods an SRT build request in order to construct attribute-based Semantic Routing Trees [20]. IDSQ floods an m-hop neighborhood in order to initially locate a moving target [41]. Flooding-limitation protocols like SPIN [15] and BARD [32] degrade to flooding when there is no overlap of in-network history and the current query. This paper aims to mitigate the reliance of higher-layer protocols on the reliability of broadcasting by providing reliable broadcast as a system service.

In this work we examine reliable broadcast in the context of multi-hop routing and resource discovery. The challenge of reliability is greater in low power networks, and complexities of interference are poorly understood and modeled in simulation. We therefore base the results of our work on testbed experiments from an available 20-node sensor network, augmented by simulations to systematically explore the design space. In wireless sensor networks, broadcast reliability is directly affected by radio interference, hop counts, hidden nodes, congestion, node density, and transient interference patterns [40, 34]. The limited radio power of typical sensor network motes necessitates multi-hop routes from data sources to data sinks [38].

This work was partially supported by the National Science Foundation (NSF) under grant number NeTS-NOSS-0435517, "Sensor Networks for Undersea Seismic Experimentation", and by support from Intel Corporation. Fred Stann received support from Xerox Corporation. Fred Stann and John Heidemann are with USC/Information Sciences

Reliability for broadcasting can be thought of as the probability that a broadcast "wave" will reach the most distant "tier" of nodes away from the source of the broadcast [37]. Broadcasting in wireless is most often achieve via flooding, which is the retransmission of a packet exactly once by each node in the network [23]. In this paper we present a new protocol that enhances the reliability of flood-based broadcasting in wireless networks. The goal of our protocol is to ensure that a near perfect percentage of broadcasts reach all tiers.

A general category of task that can be impacted by unreliable broadcast is "resource discovery." For routing protocols, like DSR [14] and AODV [24] this is simply target location. Resource discovery for applications is generally concerned with the location of nodes that are generating data of interest to an application sink. The reliability of resource discovery could be defined as the ratio of discovered nodes matching a query to the actual number of relevant data sources in the network. Wireless sensor network data dissemination protocols often combine resource discovery with attribute-based routing [10]. With attribute-based routing, unreliable broadcast can simultaneously affect both application and routing. For a certain class of application, reliable resource discovery is essential. Such applications exhibit a rapid decay in efficacy when even a single exploratory epoch fails to find targeted resources. Examples include interactive applications where a user is waiting for a query response [27], real-time applications like target-tracking [13, 41], and signal processing applications whose accuracy depends on a critical mass of data [18].

In this paper we present a new protocol for reliable broadcasting. Our protocol is distributed in the sense that every node makes its own decisions about retransmission without any global or hard state. Nodes examine traffic in their local neighborhood to gauge the degree of broadcast propagation. Decisions to retransmit a broadcast are parameterized by local node density. In our protocol, lower density equates to a higher likelihood of retransmission for imperfect propagation. Naturally, chain reactions that result in excessive flooding must be squelched. Timeliness is a natural side effect of our protocol because, with it, a single flood has a higher probability of achieving a sought for response. The same probability may be achieved via multiple less reliable floods, but timeliness will suffer and, as we will show, the effective overhead will be worse. Details of the protocol are presented in Sections 3 and 4. The basic methodology of our investigation is to use testbed experiments and simulations to validate a protocol based on analysis. The goal of our experiments is to unearth the cost of our scheme relative to the reliability that it provides.

The contribution of this paper, then, is threefold. First, it presents a new protocol, Reliable Broadcast Propagation (RBP), as the preferred technique for wireless network applications that rely on broadcasting for timely and accurate system response. Second, it defines a new broadcast metric that balances efficiency with reliability . Third, it introduces a new simulation error model for more accurate modeling of packet reception that is especially important for experiments related to broadcasting.

2 Related Work

Related work for this paper can be roughly divided into five categories: wired-network broadcast reliability, reliable broadcast schemes for MANETs, wireless MAC layer reliability, transport layers for sensor networks, and, to a limited extent, resource discovery algorithms.

Broadcast reliability for wired networks is concerned with upper-layer tasks such as the dissemination of routing information, real-time multicasting, and database replication. Protocols, like OSPF [21] and PGM [30], rely on the maintenance of unicast trees for the delivery of ACKs or NACKs to trigger retransmissions. Database replication techniques rely on eventual data convergence over time via repeated or multiple random transmissions [5]. Because point-to-point networks can not exploit the ubiquitous nature of wireless broadcasting, there seems to be little that can be learned from these techniques applicable to wireless networks. Nonetheless, work in reliable OSPF has demonstrated that broadcast reliability is directly affected by the number of shared edges that exist among alternate paths between node pairs [26]. Because wireless broadcasting can be modeled schematically as a set of discreet connections [34], shared-edge analysis is an effective tool for predicting the probability of eventual delivery of a broadcast packet to all nodes in a wireless network. Most wireless broadcast analysis is predicated on an assumption uniform density. We demonstrate in Section 5 that local density, in the context of shared-edge analysis, can have a profound affect on the probability that all nodes in a wireless network receive a broadcast.

Broadcasting schemes for MANETs fall into four general categories: simple flooding, area-based methods, neighborhood-knowledge-based methods, and probabilistic schemes [36]. Ho et al. make several important points about the appropriateness of flooding for MANETs [12]. Flooding is topology-independent, and the redundancy of flooding makes it inherently reliable. Also, traditional broadcasting techniques that rely on the maintenance of hard state converge too slowly for dynamic environments. Area-based and neighborhood-based methods are aimed at efficient broadcasting without loss of reliability. They require knowledge of the absolute locations of nodes or the hop counts to nodes. Area and neighborhood schemes strive

to eliminate redundancy from flooding. Nodes which can't increase the scope of a broadcast prune themselves from the set of nodes that retransmits a broadcast. Our approach is more concerned with high reliability rather than average reliability at low cost. Gossip-based ad-hoc routing is a probabilistic scheme which includes a heuristic to prevent premature gossip death [8]. The heuristic is sensitive to network density in that a decision to not forward a broadcast (due to failed probability) can be overruled when an insufficient number of neighbors have been overheard to transmit the same broadcast. The interesting aspect of many of these schemes for MANETs is the concept of making distributed decisions based on the snooping of local neighborhood traffic. The protocol that we present in the next two sections is somewhat similar to a one-hop self-pruning scheme [19] combined with the heuristic employed in Gossip.

Commonly used wireless MAC layers, like 802.11 [16] and S-MAC [39] do nothing to ensure the reliability of PHY broadcasting other than random slot selection when a busy channel goes idle. Tang and Gerla have proposed the addition of broadcast ACKs to 802.11 [33]. Numerous TDMA schemes have been proposed that provide contention-free reliable broadcast [2]. Because TDMA schemes usually require cluster heads with intimate knowledge of the nodes they support, they are not typically used in low-power networks. TRAMA is a novel adaptation of TDMA to sensor networks in which nodes use a schedule exchange protocol and adaptive election to converge on slot assignments [22]. Node schedules are piggy-backed on data packets. Z-Mac is a sensor net MAC that uses a TDMA schedule when contention is high and CSMA for low contention [29]. It requires knowledge of two-hop neighbors, and schedule exchanges. It appears that TRAMA and Z-MAC have only been simulated; studies with real-world error conditions remain to be done.

The goal for our protocol was to achieve network-wide reliability, so we also examined ideas from transport layer protocols for low-powered wireless networks. ESRT is a sensor network protocol that attempts to increase the reporting rate of a source until a requisite sampling rate is achieved at the sink (while avoiding congestion) [1]. RMST is aimed at the fragmentation and reassembly of binary objects that are larger than the network mtu [31]. ETX uses forward and backward reliabilities to select good end-to-end paths in a sensor network [4]. A common denominator of these protocols is that they are all concerned with unicast transmissions and require end-to-end book keeping. Since every node is an endpoint in a broadcast, transport layer mechanisms are not practical for our investigation. Nonetheless, our scheme may indirectly enhance transport and routing protocols that rely on lower-layer broadcasts to provide the information on which they base their decisions.

Although we decided to use resource discovery as the upper layer protocol for our investigation, it is not the primary concern of this paper. We mentioned, in Section 1, that AODV, ODMRP, TinyDB, Diffusion, IDSQ, and BARD all rely on broadcasting under certain circumstances. Even so, not all resource discovery schemes for wireless networks use broadcasting. For example, DCS/GHT [28] uses an underlying geographic routing layer to move data with common attributes to specific nodes by hashing attribute names into geographic coordinates.

3 Reliable Broadcast Algorithm

We had several goals in mind when addressing the problem of reliable broadcasting in wireless networks. We wanted an algorithm that would provide near perfect reliability in a timely manner. Unlike prior work, we avoided multihop topology/schedule exchanges. We decided that our algorithm must adapt quickly to changing network conditions without extensive control overhead. Although reliability and efficiency are covariant, we wanted to balance cost and benefit. Finally, we wanted to verify our algorithm with real-world experiments. We therefore designed for non-uniform densities, optimizing for both density and direction.

The algorithm that we evolved is very simple in that it only requires a node to know the identity of its one-hop neighbors (i.e. neighbors with whom a node has direct bi-directional communication). Accumulating a list of one-hop neighbors is a trivial task given that such neighbors are in direct communication with a node inside a relatively short window of time. Many MAC and routing protocols already accumulate this information. In contrast to the neighborhood-knowledge schemes alluded to in the last section, we are not trying to make a single broadcast more efficient. Rather, we wish to make a single broadcast more reliable, thereby reducing the frequency with which an upper-layer protocol needs to invoke broadcasting. If an unreliable broadcast propagates with $P(S) = .90$, then an application that required .99 reliability would need to broadcast twice to ensure success. Our goal is to make a single reliable broadcast more efficient than the equivalent number of unreliable broadcasts required to achieve the same level of reliability.

With our algorithm, the first time a node hears a broadcast it retransmits the packet unconditionally, as in a normal flood. As additional neighbors transmit the same packet, the node listens ands keeps track of which neighbors have propagated the broadcast. Armed with one-hop neighbor knowledge, a node can ascertain the percentage of its neighbors that are guaranteed to have seen a packet. We call a transmission by a neighbor an

implicit ACK., a term that typically refers to inbound traffic that indirectly acks outbound traffic [25]. When the number of implicit acks seen by a node falls below a predetermined threshold, a node will again retransmit the

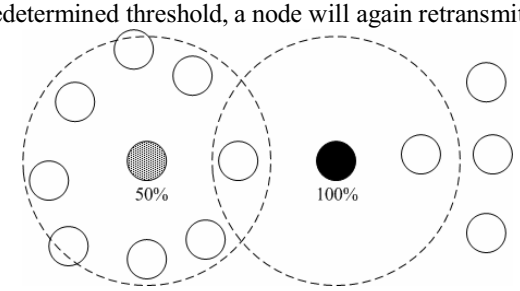


Figure 1: A sample topology with varying density and appropriate thresholds.

broadcast packet. In order to avoid the undesirable regeneration of an entire broadcast, nodes hearing a broadcast more than once from the same node will send an explicit unicast ACK to that node (and not retransmit that packet). If the number of neighbors that haven't acked a packet is less than the number of neighbors that have, the packet is unicast to the unacked neighbors. This results in the least overhead for the retransmission.

A key optimization in our algorithm is that both retransmission thresholds and the number of retries are adjusted for neighborhood density. Higher density neighborhoods have lower thresholds with fewer retries. If, for example, there are three or less neighbors, a node will make up to three attempts to propagate the message to all neighbors. For four to six neighbors the threshold is 66% and the number of retries is two. When there is a dense local neighborhood (i.e. eight or more neighbors), "successful" propagation occurs when half of the neighborhood acks and only one retry is attempted if the threshold is not met. The analysis in Section 5 and feedback from early test runs was used in selecting appropriate thresholds. Figure 1 demonstrates the general idea of variable thresholding.

We also introduced a "directional sensitivity" optimization into our algorithm. This was in response to a special case of non-uniform density that our testbed experiments revealed to us. When an upstream dense section meets a downstream sparse section, there can be a node at the edge of the dense section that has a large neighbor count, but is the sole provider of traffic to the first downstream node in the sparse section. This situation is depicted in Figure 2. The black node is the sole provider of traffic to the grey node, but it resides in a dense neighborhood. When the grey node fails to hear a broadcast from the black node, retries will rarely happen with our basic algorithm because there is a high probability that at least 50% of the black node's neighbors will have acked the broadcast. The black node is incapable of recognizing its special relationship with the gray node, but the gray node can easily do so because all of its upstream traffic will come from the black node.

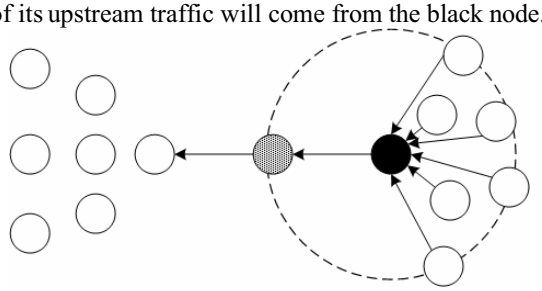


Figure 2:Special Relationship between Black and Gray Node

With our directional sensitivity optimization, nodes keep a histogram of which neighbor was the first to transmit a previously unheard broadcast. For the black node in Figure 16 there would be a uniform distribution in such a histogram. The introduction of jitter for message forwarding in dissemination protocols guarantees that no single neighbor dominates. The gray node, however, sees most upstream traffic (for a time) from a single node. Our solution includes a moving window of time in which a directional histogram is maintained. If a single neighbor has a majority of the histogram, the node sends that upstream neighbor a control message indicating that it has a special relationship to this node. Any node that gets such a message will do up to 4 retries when that downstream node does not ack. This behavior is independent of the normal rules for retries based on density. It is possible for a node to have multiple "important links." The combined algorithm is shown in Figure 3.

```
rbp_snoop_send (pkt) {
        if  (broadcast)
                set rbp_timeout
        pass through to MAC
}

rbp_snoop_rec (pkt) {
        if ( (overheard broadcast by neighbor)OR (explicit ACK) )
                mark neighbor ACKed
        pass through to Routing
}

rbp_timeout {
        if ( (percentACKed < threshold (neighborCount)  AND
                (retryCnt<maxRetry (neighborCount) )
                rebroadcast = TRUE
        else if ( (any important links)AND (retryCount < 4) )
                rebroadcast = TRUE
        else
                rebroadcast = FALSE
        if (rebroadcast){
                forward copy of bcast to MAC
                retryCnt ++
                reset rbp_timeout
        }
}
```

Figure 3: Pseudo-code for RBP Algorithm

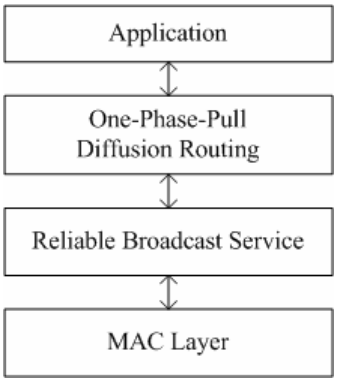


Figure 4: Diffusion Filter Architecture with RBP as a Selectable Service between the MAC and Routing Layers

4 RBP Implementation

Our primary objectives during implementation were modularity, encapsulation, and layering. We considered modification of existing MAC and routing layer protocols undesirable. Our goal was to provide reliable broadcast as a selectable service. Because of the general applicability of our algorithm, we wanted an implementation that could easily be added to existing protocol stacks. Our solution was to create a pass-through module that snoops on network traffic between the network and MAC layers.

Whenever the RBP module sees a previously unseen broadcast packet, it instantiates an object in a map indexed by a unique ID. RBP starts a short timer and keeps track of the implicit (and explicit) ACKs in the associated object. When the timer expires, the algorithm as described in the last section is used to decide whether or not to retransmist the message (beyond the initial flood). RBP keeps a duplicate of the original broadcast packet for retransmission. A cleanup timer periodically removes obsolete entries from the map. Explicit ACKs and directional sensitivity messages were implemented as a unique control messages in diffusion.

We felt that the most general open-source framework for developing and deploying sensor network software, available at the time, was EmStar [7]. EmStar provides a handy Data Link Interface that specifies a standard API for stackable sensor network modules. Numerous services have been ported to EmStar so that it is possible to deploy arbitrary combinations of PHY, MAC, and Data Dissemination layers.

Our choice for a data dissemination / routing layer was One-Phase-Pull Diffusion [9]. One-Phase-Pull employs a single flood for resource discovery that is repeated at randomly selected intervals in order to cope with changing network conditions. Sinks flood an "interest" packet that describes a desired attribute set, and sources with matching attributes then establish a unicast path back to the sink. Diffusion provides a convenient *filter* facility for the in-network monitoring and/or modification of data as it moves through a network [11]. The filter mechanism allows for the selectable modification of semantics between the routing layer and layers above or below. We encapsulated RPB in a loadable filter module in order to accomplish our dual requirements of layering and modularity. We could have alternately placed RBP into an independent EmStar module, but we had greater familiarity with the filter mechanism. Our RBP module intervenes between the MAC and routing layer (see Figure 4), and will be made available in released diffusion as the *RBP filter*, named for the algorithm.

An important requirement of our algorithm was the knowledge by each node of its one-hop neighbors. This information is conveniently available in EmStar by including certain modules (linkstats, neighbord, and blacklist) in the EmStar "stack" of network modules. These modules are routinely included when running diffusion over EmStar, in order to support the informed selection of end-to-end unicast routes.

5 Analysis

Previous work has accomplished detailed analysis of flooding reliability in wireless networks [34, 37]. Viswanath and Obraczka define flooding reliability as the total number of nodes reached by a broadcast divided by the total node count for the neighborhood [34].

$$\frac{N_T}{N_R} \downarrow \frac{1}{N_R}\left\{ P_{SN} \uparrow P_{SN}\frac{P_{bN}\leftarrow^{l} \nwarrow 1}{P_{bN}\leftarrow \nwarrow 1}\right\} \qquad (1)$$

In this equation P_{SN} is the expected number of neighbors that will successfully receive a transmission from a source node, P_{bN} is the expected number of nodes that will receive at least one secondary transmission, β is the expected increase in area coverage (around 41%), and l is the hop count of the flooding wave. In formula (1), increasing density will increase the expected number of nodes that successfully receive a broadcast packet. Most analysis of wireless flooding is simplified by an assumption of uniform density. Because our protocol is density sensitive, a more interesting analysis for our purposes deals with variable density. Rather than use formula 1 and related formulas, we instead do a "shared edge vs. disjoint path" analysis [26] that allows us to examine the problems posed by variable density. Topologies with high density near the source of a broadcast and lower density away from the source will

have misleadingly good reliability numbers using formula (1) and associated formulas.

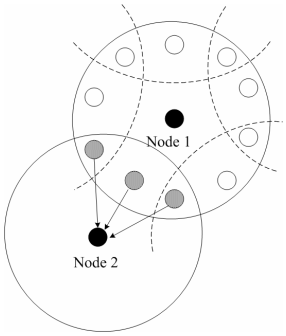


Figure 5: Broadcast Propagation Between Node 1 and Node 2 in a Uniform Density Topology with Three Common Neighbors

We begin our analysis with a uniform density topology and then demonstrate the effect of introducing variable density. Consider figure 5 in which node 1 (the source) has 10 neighbors, three of which are common to two-hop neighbor, node 2. With flooding, each of the common nodes will retransmit the broadcast packet, so there are three possible disjoint paths from the source to node 2 during the first retransmission. As the broadcast wave expands, the number of neighbors that initially transmit to a node, which has never seen the broadcast, is a fixed value (under uniform density). The probability that a node will receive the first wave of a broadcast is easily calculated:

$$P[T_n] = P[T_{n-1}] \times \left(1 - (1 - P(S))^{C}\right) \quad (2)$$

P[Tn] is the probability that nodes at Tier n will receive the broadcast wave, *P(S)* is the simple probability that a transmission succeeds, and *C* is the number of common neighbors. We now use formula 2 to calculate the probability that a broadcast wave propagates to the nodes between the dotted lines in figure 6 .

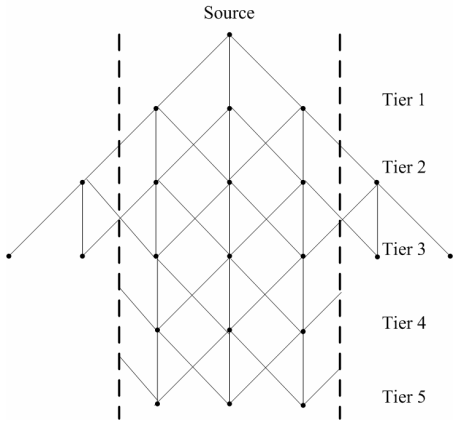


Figure 6: Broadcast Propagation Graph Divided by Time

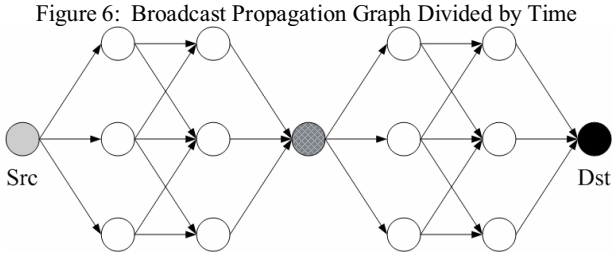


Figure 7: Topology with a Wave Front Bottleneck

If we assume that P(S) is 90%, we get the entries in Table 1 in the row labeled "Mesh" for the probabilities that the broadcast wave will reach each successive tier:

Topology	Tier 1	Tier 2	Tier 3	Tier 4	Tier 5
Mesh	.90	.899	.898	.897	.896
Bottleneck	.90	.899	.898	.808	.806

Table 1

It should be obvious that, with uniform density, the probability of a successful broadcast is primarily controlled by the success or failure of the initial transmission from the source. The likelihood of all three common nodes independently failing to receive a broadcast can be modeled as a bernoulli trial, and is extremely unlikely. For example, if the combined probability of transmit and receive failures results in a per-link success rate of 90%, the likelihood of all three common nodes failing is .001.

Suppose we now introduce a bottleneck in our topology such that there is a non-uniform density and poorly connected portions of the network. This situation is depicted in Figure 7 where there is a single node connecting two denser portions of the network. This could occur for a number of reasons such as: node placement, dying nodes, or interference patterns. All paths leading from the source to the destination now share a link. A shared link has a significant impact on end-to-end broadcast reliability as shown in the Table 1 row labeled "Bottleneck." Note that Tier 4 now has a probability of 80.8%, and the probability that the broadcast actually reaches the destination would is reduced by 9%. When multiple disjoint paths meet at a shared edge, the effect on end-to-end broadcast propagation is similar to that caused by the initial broadcast from the source. A broadcast protocol that bolsters reliability only for initial transmissions from a source will not work in sparse topologies. Because shared edges are more common in areas of low density,

sensitivity to density is a key enabler of our algorithm, as will be shown in the experiment results that follow.

6 RBP Initial Simulation Results

Having conceived of an algorithm that adapts to poorly connected topologies like that of Figure 7, we wanted to quickly evaluate the feasibility of our scheme. We therefore ran a series of preliminary simulation experiments based on our analysis.

6.1 Methodology

The simulation environment we employed was EmSim [7], which is available with EmStar. It allows for actual network code to run unaltered while providing simulated radios, channels, and error models. For a MAC layer protocol we used the generic MAC delivered with EmSim. It provides multiple access with collision avoidance. The EmSim MAC does not do retries for broadcasting, as is typical in most wireless protocols. Software loaded above the MAC layer in each experiment included the diffusion filter core, the one-phase-pull routing filter, the RBP filter (when using reliable broadcast), and simple source and sink apps. We configured One-phase-pull (OPP) [9] to initiate a resource discovery broadcast every 60 seconds. The broadcast send rate was not accelerated to yield more samples per unit of time because we did not want broadcasts to overlap one another. Our current work doesn't take into account loss due to congestion induced by overlapping broadcasts. We consider congestion and its relation to send rate an orthogonal topic at this point in time. With One-phase-pull diffusion, once a broadcast reaches a node with sensor data matching the attributes in the broadcast packet, the node becomes a data source and commences to send data back to the sink which initiated the broadcast. We programmed sources to not transmit actual sensor data, because such transmissions are unicast and don't concern us in this paper. Every data point on a graph represents the average of 10 runs, and 95% confidence intervals are shown. Each run was one-hour long allowing for 60 broadcasts. We collected per-node statistics by instrumenting our code to log relevant events and byte counts. We processed our logs via scripts to produce statistical results.

6.2 Simulation Error Model

Because our protocol is intended to handle a reliability problem endemic to low power wireless networks, we wanted an error model for our simulations that approximates what happens in the real world. The EmSim error model uses transmission power, distance, and a normal distribution of receive failures to calculate the probability of packet loss between a pair of nodes. This calculation is done independently for each combination of transmitter and potential receiver (i.e. every other node in the simulation). We noted that the error model did not have any correlated receiver loss.

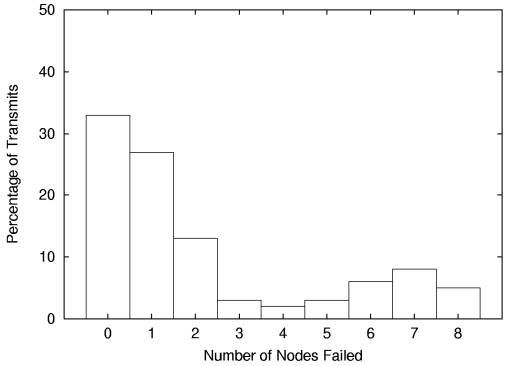


Figure 8: Observed Distribution of Node Failures for Eight Real-World Neighbors

We wondered, therefore, to what degree real-world packet losses were spatially correlated. To that end we profiled simple loss between a real-world transmitter and a set of receivers. We placed eight Stargate nodes in a circular pattern around a ninth node which transmitted a 68-byte packet once per second for an hour. Nodes were approximately 12 feet from the transmitter and transmit power was set to -4 dBm. Packets were numbered so that we could monitor how many receivers got each transmission. The histogram in Figure 8 shows the distribution of how many receivers failed individual transmissions (as a percentage of 3600 attempts). There are several interesting aspects to this bimodal distribution. First, the pair-wise independence of receive errors is evident on the left side of the histogram which decays exponentially. Second, the non-trivial peak on the right shows that correlated transmission failures are present even when there are no line-of-sight obstacles.

Influenced by this finding, we added a normally distributed correlated failure probability to the EmSim packet loss model. When a packet is failed by our addition, it is failed for every receiver in the simulation. If the packet "succeeds" our modification, it is subjected to the normal pair-wise calculations. The bimodal distribution shown in Figure 9 is the measured result of our combined error model. We feel this is a reasonable approximation of the histogram in Figure 8.

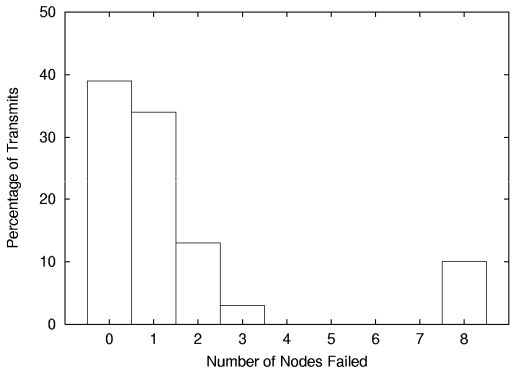


Figure 9: Error Model Employed in Simulations

6.3 Metrics

We express experimental results in terms of: reliability, bytes-per-flood, and a reliability cost metric (RCM).

As we noted in the Analysis Section, when reliability is defined as the percentage of nodes that receive a broadcast, topologies with high density near the source of a broadcast and lower density elsewhere may have misleadingly good numbers. We instead define reliability as the percentage of broadcasts that traverse the network diameter (reaching the outermost "tier") divided by the total number of broadcasts initiated in an experiment. In our initial measurements, using either definition of reliability did not significantly affect the reliability of RBP. Simple flooding, however, often had significantly lower network-diameter reliability than node-percentage reliability.

In all of our experiments we define an "event" as the initiation of a broadcast by a sink node. Bytes-per-flood is the sum total of byte transmissions in a network triggered by a single event. Without RBP, it equates to the number of bytes contained in the broadcast packet times the number of nodes that transmitted it. The number of nodes may be less than the total node count if the flood fails to reach all nodes. RBP adds the cost of retransmissions and explicit ACKS.

Bytes-per-flood, by itself, is a poor metric to use for comparing protocols because it doesn't take reliability into account. If several floods are required to guarantee a certain level of reliability, then the true cost of reliability is the combined cost of the floods. We needed a metric that reflects the true cost of achieving a reliable broadcast. To that end we created the Reliability Cost Metric (RCM). To calculate the RCM we first solve for the number of floods required to achieve near-perfect reliability (.99), given the propagation reliability.

$$.99 \uparrow 1 \nwarrow P\ S^{F} \qquad (3)$$

We then multiply the number of floods required to achieve near-perfect reliability by the bytes-per-flood and normalize by the cost of a "perfect" flood. A perfect flood achieves 100% reliability with each node transmitting the broadcast packet exactly once.

$$RCM \uparrow F * BytesPerFlood / Nodes * PktSize \qquad (4)$$

As an example of RCM, suppose there are 20 nodes, an 80 byte broadcast packet, propagation reliability of 85% (F is 2.4), and a measured bytes-per-flood of 1200; RCM will be 1.8. Although fractional broadcasts are not possible in the real world, we allow continuous values of F. RCM is a metric for fine-grained normalized comparison. We don't want broadcasts with RCMs of 1.1 and 1.9 to both quantize to 2.0..

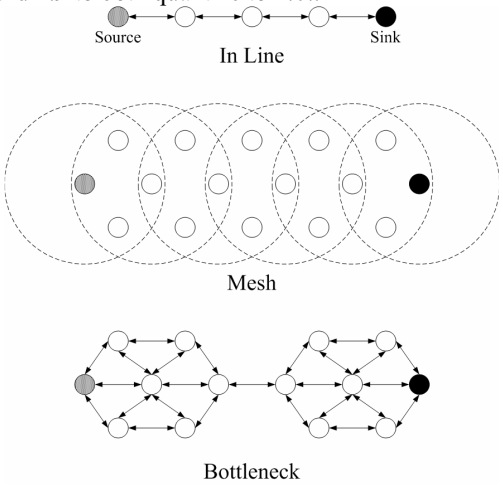


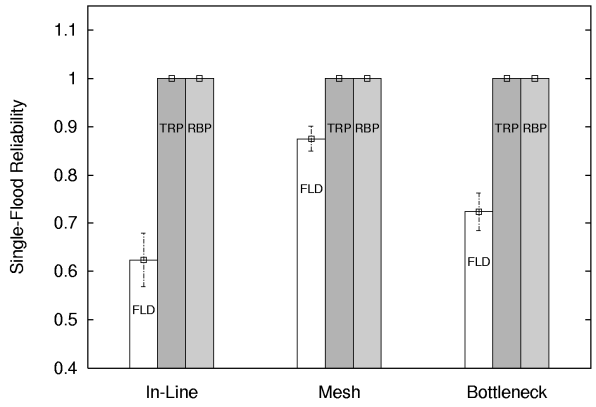
Figure 10: Three topologies used in Simulation Experiment

6.3 Effect of topology on broadcast propagation

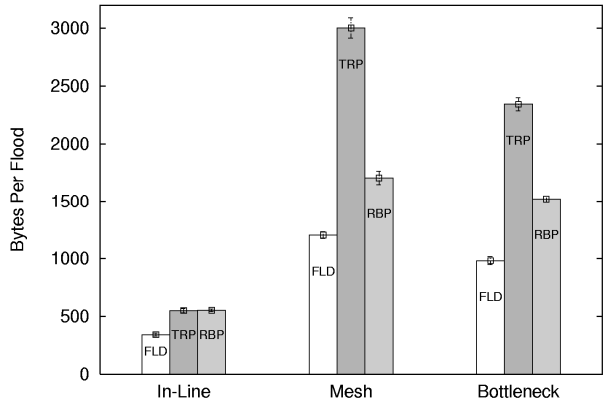
For these initial simulation experiments, we had two goals in mind: basic proof of concept, and controlled exploration of the effect of density in terms of disjoint paths and shared edges. We created three different topologies with a varying degree of shared edges. The three topologies are shown in Figure 8: inline, mesh, and bottleneck. The inline topology consists entirely of shared edges, the mesh topology has no shared edges, and the bottleneck topology introduces a single shared edge halfway between source and sink nodes.

Our hypothesis was that simple flooding performance relative to RBP would deteriorate with an increase in the number of shared edges. We expected that broadcasts without RBP over the inline topology would display the exponential decay in reliability predicted by formula 2 (in the Analysis Section). The mesh topology should demonstrate the inherent reliability of broadcasting in uniform topologies with multiple neighbors, so the difference between RBP and flooding should be the least. Between these two extremes, the bottleneck topology should demonstrate how a single shared edge can negatively affect reliability.

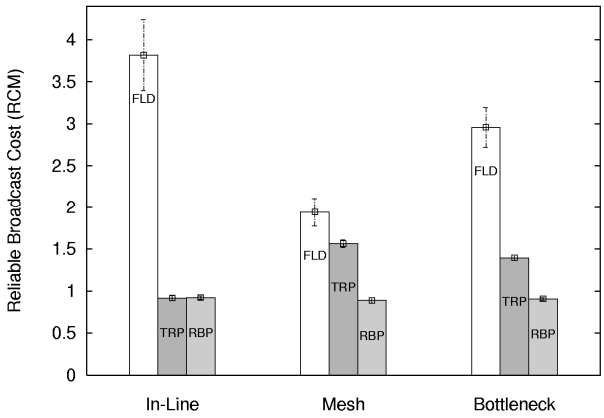
Another important research question was 'how does RBP compare to a protocol that guarantees reliability regardless of density.' In other words, if a version of RBP treated all densities in a uniform fashion, could we achieve the same normalized cost (RCM) as our density-sensitive algorithm? In order to answer that question we created the Total Reliability Protocol (TRP), a version of RBP that does up to the maximum number of retries whenever less than .99 of neighbors have implicitly or explicitly acked a flood. With our error model it is trivial to prove that TRP should easily achieve near perfect reliability over the three topologies in Figure 10.



A.



B.



C.

Figure 11 Reliability, Bytes-per-flood, and Reliable Broadcast Cost (RCM) for FLD, TRP, and RBP over Three Simulation Topologies

Figure 11 plots our three metrics for each protocol over the three simulation topologies. Simple flooding is designated as FLD; our algorithm and the total reliability protocol are marked with their acronyms.

The reliability results, summarized in Figure 11A, are in line with our expectations. Reliability in a purely in-line topology, without RBP, decays exponentially with hop count ($.88^h$). Pure flooding (FLD) reliability was 62% for the in-line topology, and both RBP and TRP had perfect results (1.0). The mesh topology run with FLD had a relatively good reliability of 88% (in line with our analysis), and again RBP and TRP were perfect. The bottleneck topology resulted in a non-RBP reliability of 72%, which is only slightly worse than the result predicted by formula 2 of the Analysis Section (77%).

As shown in Figure 11B, the transmission cost for a single flood is always higher for RBP and significantly higher for TRP. This is due to retransmission and control byte costs, which are much higher for TRP as it tries to achieve guaranteed reliability regardless of density. Notice that when multiple disjoint paths provide inherent reliability the difference in bytes per flood between RBP and OPP is less (as a percentage). Also, considering the graphs in 11A&B, RBP is providing the same reliability as TRP at a much lower cost.

The most important metric in our estimation is RCM. Looking only at graphs 11A and 11B, it is difficult to gauge the relative cost of providing reliability with each protocol. The normalized cost (RCM) shown in figure 13 demonstrates the significantly lower cost of achieving high reliability with RBP than simple flooding. For example, in the bottleneck topology the exponent in our RCM formula for FLD is 3.5. In other words it would take 3.5 floods (unquantized) to achieve 99% reliability. This number of floods multiplied by the

bytes-per-flood and normalized by a perfect flood results in an RCM of 3.8 which is 3.1 times worse than RBP. A very interesting result in figure 13 is how the cost of TRP changes relative to RBP in different topologies. The in-line topology RCM is nearly identical for both TRP and RBP. This is because RBP "degrades" into TRP in low density networks. Every neighbor is a "special" neighbor for RBP in the in-line topology. The Mesh topology provides the greatest density, and the greatest difference in RCM between TRP and RBP, with TRP 76% worse. TRP is unnecessarily attempting to achieve perfect reliability in dense neighborhoods. The bottleneck topology is between the in-line and mesh topologies with TRP 54% worse than RBP.

This experiment had the primary result that we had predicted. The cost of doing multiple unreliable floods in order to guarantee reliability is actually higher than a single reliable flood using RPB, and RBP is cheaper than a protocol which does not exploit the inherent reliability of dense neighborhoods. With these results in hand, we felt that we now needed to apply our protocol to a real-world environment with a less predicable problem space.

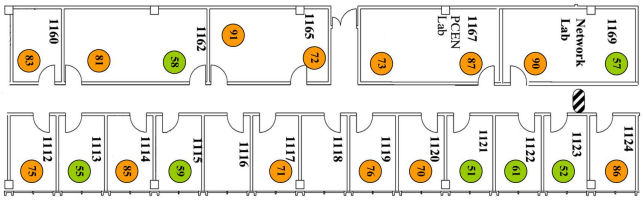


Figure 12: Testbed Topology at USC/ISI

7 RBP Testbed Results

Having validated some basic assumptions about our algorithm in simulation, we wanted to quickly move the protocol onto a testbed and debug it in the real-world. Many papers do extensive simulations prior to real-world testing. Our experience with wireless protocols suggests that significant alterations of a protocol are often needed once it is deployed in a real-world environment. We decided to do our more complex simulations after debugging the protocol on our testbed.

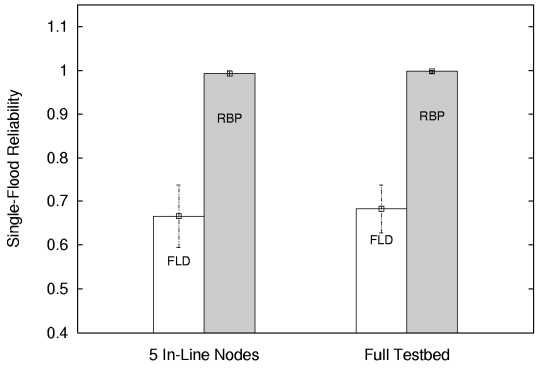
The testbed that we used consisted of 20 *Stayton and Stargate* nodes. These are small form factor systems running Linux over a 32-bit Intel processor, with 64M of SDRAM, and 32M of flash memory. They are fitted with a multi- channel radio capable of 38.4 Kbaud. Radio range can be modified to enable multi-hop experiments. The actual layout of our testbed is shown in Figure 12. Nodes are placed in private offices and labs separated by walls. Connectedness was nonuniform with pockets of good connectivity separated by areas of low density. Power was intentionally set low to enable multi-hop.

Our traffic model for the testbed was the same one used in simulation. A broadcast was initiated by the sink node once per minute, and each test run lasted an hour. Each data point represents the average of 10 test runs.

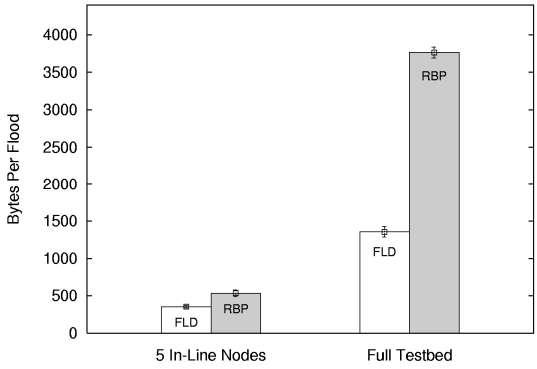
7.1 Five In-line Nodes

In our first testbed experiment we enabled 5 nodes from the testbed such that each node had one and only one upstream and downstream neighbor. We thereby created one of the topologies that we presented in both the Analysis and Simple Simulation Sections. This allowed us to do a limited comparison of our simulation model to the real world and also gauge the average error rate on single links in the testbed. Looking at the left side of figure 15 we see that FLD over four hops had an average end-to-end loss rate of 66%. This is very close to loss rate we saw in simulation for 5 in-line nodes (62%). The average per-link reliability for the four testbed links was 89%.

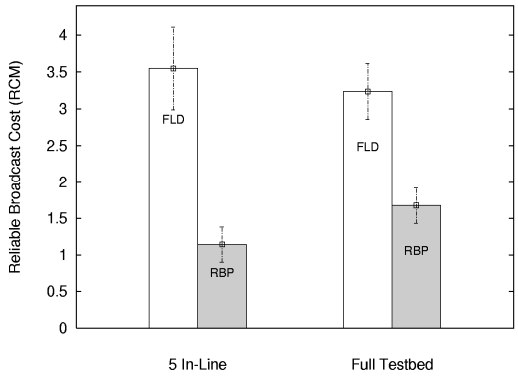
If we compare the RCMs for 5 in-line testbed nodes in Figure 17 to the RCMs for in-line simulation in Figure 13, we see that FLD numbers are very close (3.55 vs. 3.82). We concluded that we had a good correspondence



A.



B.



C.

Figure 13: Single-Flood reliability, Bytes Per Flood, and Reliable Broadcast Cost (RCM) for two topologies in the USC/ISI testbed.

between simulation and testbed for a simple topology. Note that TRP is not represented in our testbed graphs, due to time constraints on testbed utilization. Having validated a reasonable degree of congruence between environments, we felt that we could complete this work in the future.

7.2 Full Testbed Experiment

The primary experiment that we performed on the testbed was to place a source and sink node at opposite ends of the testbed, ensuring that the maximum possible network diameter would be traveled by each broadcast. Initial results from this experiment were quite disappointing. RBP was only slightly more reliable for a single flood than simple flooding, and the RCM was worse for RBP because the higher per flood cost did not yield significantly better reliability. Log analysis revealed a flaw in our initial algorithm, which required the directional sensitivity optimization outlined in section 3. "Important links" have the same affect on a topology that a shared edge has. The problem with important links is that they are not recognized by the sending side of the link. Important links originate in dense neighborhoods and terminate in sparse ones. This situation requires that recognition of the importance of the link be made in the sparse neighborhood and relayed back to the dense neighborhood.

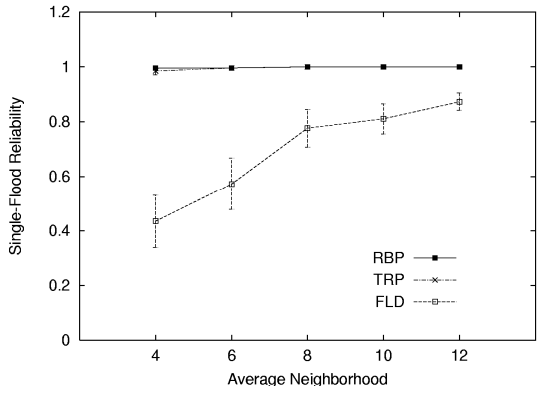
The result of adding directional sensitivity to our protocol is reflected in the excellent results shown in Figure 13 A,B,&C. Single flood reliability averaged 99.8% for RBP vs. 68.2% for simple FLD. Although the bytes-per-flood is 2.7 times higher for RBP, the RCM cost for RBP is 48% less than FLD. Our experience with testbed led to two conclusions. First, testbed experiments can bring to the surface problems that may not be thought of a priori. Second, directional sensitivity can be as important as density sensitivity in order for RBP to achieve reliability at a reasonable cost.

8 RBP Further Simulation Results

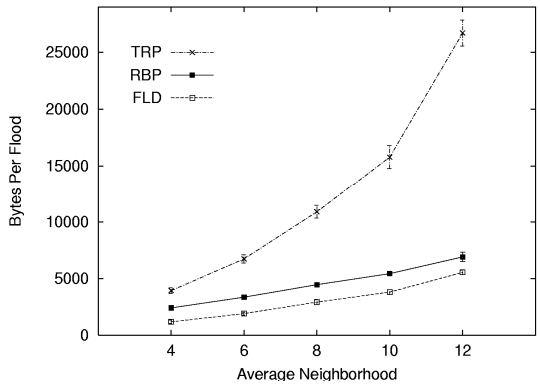
Having shown the need for density and directional sensitivity, we now turn to a systematic exploration of the design space. Simulation allows for the exploration of a multi-dimensional problem space in a manner that is impractical with an actual testbed.

8.1 Methodology

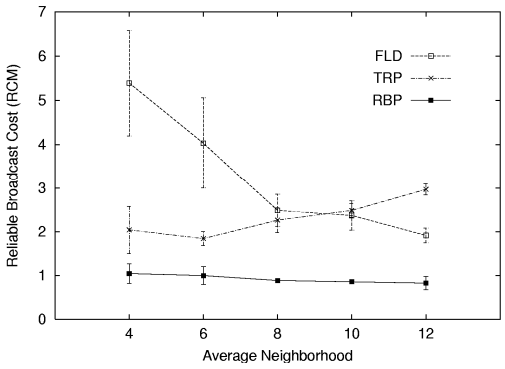
For these experiments we used random node placements. Every data point presented in this section represents an average over ten randomly generated topologies. For random node placement we employed an existing topology-generator program [9] capable of creating EmSim topologies of programmable size and node count. The generator has useful options such as approximate corner placements of sources and/or sinks in order to maximize the distance (hop count) between data producers and consumers. The generator tests topologies for connectedness given the radio range. If sources and sinks cannot communicate with each other, the generator scraps the topology and generates another. In order to increase the number of tiers traversed by a broadcast we



A



B



C

Figure 14: Single-flood Reliability Reliable Bytes Per Flood Broadcast Cost (RCM) with Varying Density as an Independent Variable

employed a 90 by 30 meter area and placed the source and sink nodes at diagonally opposite ends. Average local neighborhood size was calculated as (Nπ*R2)/A, where N is the total node count, R is the maximum radio transmission range (12.5 meters), and A is the total simulation area. The packet error rate was achieved by the error model that we evolved in section 6.1 (see figure 9). Our goal in these experiments was to explore the role of density and packet-loss rates on protocol performance.

8.1 Density Effects on Overhead

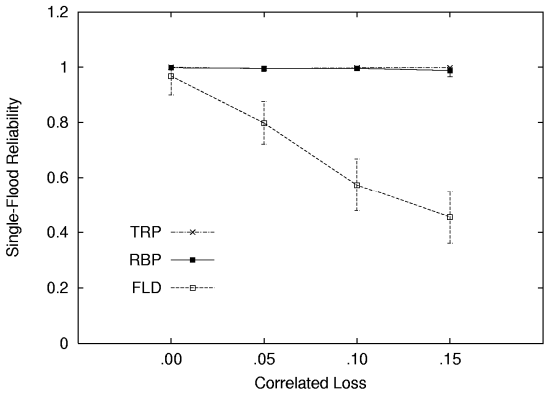
Because density plays a major role in our analysis, we felt that an obvious investigation was to make density the independent variable in an experiment. We therefore used simulation to vary node density (average neighborhood size) while holding the packet error rate constant. The primary method our algorithm uses to achieve efficiency is by adapting to local density. TRP is really just RBP without density sensitivity, and initial simulations seemed to indicate that RBP achieved the same reliability at much lower cost. Sensitivity to density allows us to economize in dense neighborhoods where the existence of multiple disjoint paths can facilitate the propagation of a broadcast. So a key question is 'does increasing density eventually obviate the need for a protocol in order to achieve reliable broadcasting at a low normalized cost.'

Our expectation for this experiment was that increasing density would increase the reliability of simple flooding (FLD) and thus minimize the difference in RCM between RBP and FLD. We also expected TRP to deliver the same reliability as RBP, but at a much higher price. Results for single-flood reliability appear in Figure 14A. As average density increased from 4 to 12 neighbors, the positive shift in reliability for flooding is quite dramatic. Reliability increased from 43% with 4 neighbors to 87% with 12 neighbors.

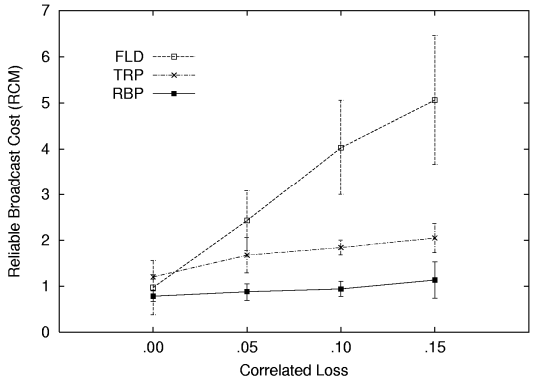
Looking at Figure 14B, the difference in bytes-per-flood between RBP and FLD remains relatively constant, but TRP increases non-linearly. Because RBP and TRP achieve nearly identical reliability at vastly difference cost, the RCM graph for this experment (Figure 14C) yields one of our most interesting results. With 10 neighbors or more, the RCM cost is higher with TRP than flooding. In other words, with high density, multiple floods to achieve 99% percent reliabiliity is cheaper than a single TRP flood. Because the reliability is converging in Figure 14A, we expect that guaranteed high density lessens the need for RBP.

8.2 How does Error Rate affect the Performance of RBP vs. non-RBP?

We saw that high density reduces the performance and advantage of RBP. Will decreasing the error rate have a similar affect? Complicating this research question was the fact that our error model has two components: correlated errors and pair-wise errors. We decided to vary each of these components independently, while holding the other constant. Density was also held constant at 6 neighbors. Our expectation was that reducing correlated errors would, like increasing density, result in a reduced performance advantage of RBP over flooding. We did not, however, expect the same results from varying the pair-wise error rate. Our error-model



A



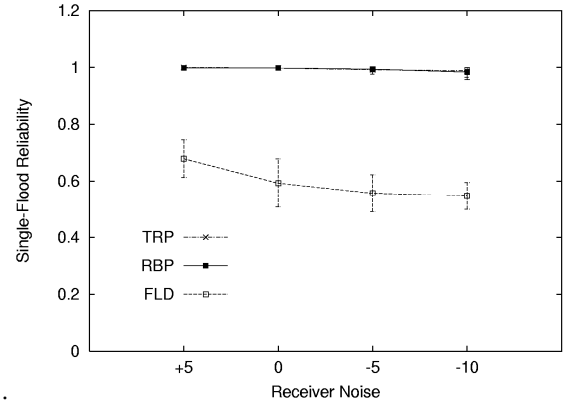
B

Figure 15: Single-flood Reliability and Reliable Broadcast Cost when Varying Correlated Loss

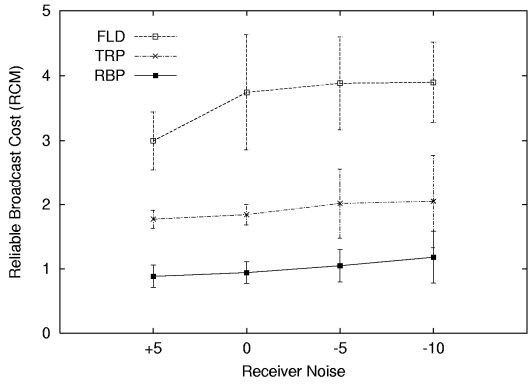
analysis (6.1) made the point that the likelihood of multiple pair-wise errors decays exponentially. The pair-wise error rate would need to get unrealistically high before a high percentage of disjoint paths between broadcast tiers failed to forward a flood.

The result of varying the correlated loss is shown in Figure 15. The results are in line with our expectations. Similar to increasing density, decreasing the correlated error rate causes convergence between RBP and simple flooding in terms of both reliability and normalized cost (RCM). With correlated loss set to zero, the single flood reliability of FLD is 97% and RBP is 99.8%. At the other extreme, with correlated loss at .15 the single flood reliability of OPP is 46% and RBP is 99%. The RCM numbers, shown in Figure 22 follow suit. In effect, if there are no correlated errors (an unrealistic expectation in wireless networks) and uniform moderate density then simple flooding is reliable.

Varying the pair-wise errors was accomplished by shifting the center of the normal distribution for receiver noise in .05 increments while holding transmit errors constant at the value used in our standard error model.



A



B

Figure 16: Single-Flood Reliability & Reliable Broadcast Cost (RCM) when Varying Receiver Loss

The receiver noise distribution, when centered on zero, gives the loss rates shown in the left side of the bimodal error model of Figure 9 in Section 6.1.

The results of this experiment, shown in Figure 16, show much less impact on simple FLD than when varying transmit failure rates. This is in line with our analysis and the distribution of our error model. Although one, two, or three single link failures are far more likely in our error model than a correlated error, the likelihood of a large percentage of neighbors failing during pair-wise calculations is very small.

Broadcasting is tolerant of individual link failures when network density provides multiple disjoint paths for the continued propagation of the broadcast wave. Pair-wise errors have a much larger influence in sparse neighborhoods. When a node has a single upstream or downstream neighbor, the effect of pair-wise errors is the exactly same as for correlated errors. Because RBP does up to 4 retries in sparse neighborhoods, it handles pair-wise errors in low density in the same manner as it handles transmit errors.

9 Future Work

The primary issue not addressed in this paper issue is that of mobility. We believe that mobility may affect our protocol in two specific ways. First, mobility increases the variability of topology over time; therefore, the identification of one-hop neighbors must occur at a rate that adapts to this variability, or updates must occur "as needed." We feel that our algorithm will adapt more easily to this problem than schemes which require large amounts of converged information in order to be effective. Second, the identification of special one-hop neighbors, outlined in Section 7 must utilize a small window of samples that is updated frequently. Other future work may include the encapsulation of our algorithm in wrappers other than a diffusion filter. Numerous standard frameworks exist for the easy stacking of network modules. Because our algorithm is not specific to sensor networks or directed diffusion, it should be easily ported to other protocol stacks.

10 Conclusions

In this paper we have explored how broadcast reliability interacts with routing protocols in wireless networks. While there is a great deal of prior work in the area of reliable broadcasting, most of it focuses on efficient flooding, in simulated topologies, often with multi-hop topological information. We instead focus on end-to-end reliability of higher-layer protocols and the study of topologies with variable density as we found in

our testbed. In addition, we develop a very simple mechanism using only local density information, sometimes augmented by indications from immediate neighbors of important links. We show that this combination is both effective at obtaining near-perfect reliability, and much more efficient than either repeated flooding or guaranteed transmissions. We also derived a cost metric for broadcasting that takes into account reliability. Most importantly, we demonstrated that the cost of achieving 99% reliability for broadcast propagation in a real testbed is on average 48% lower with RBP than simply increasing the flood rate. To evaluate our algorithm we developed a new wireless transmission error model, based on controlled experiments, that considers both pairwise and correlated packet loss. Our key conclusions are shown by experiments in a 20-node wireless testbed. To fully explore the parameter space we augment these results with simulations.

11 References

[1] O. B. Akan, and I. F. Akyildiz. Event-To-Sink Reliability. In *Proceedings of ACM Mobi-Hoc 2003*, pages 177-188, June 2003

[2] I. Cidon and M. Sidi. Distributed Assignment Algorithms for Multihop Packet Radio Networks. *IEEE Transactions on Computing*, 38:1353-1361 Oct.1989.

[3] F. Dai, and J. Wu. Performance Analysis of Broadcasting Protocols in Ad Hoc Networks Based on Self-Pruning. IEEE Transactions on Parallel and Distributed Systems, 15(11):1027-1040, November 2004

[4] D. De Couto, D. Aguayo, J. Bicket, and R. Morris. A High-Throughput Path Metric for Multi-Hop Wireless Routing. In *Proceedings of the 9th ACM MobiCom*. San Diego, CA, Sept 2003.

[5] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic Algorithms for Replicated Database Maintenance. In *Proc. ACM Symposium on Principles of Distributed Computing*, pages 1-12, 1987.

[6] A. Ephremides and T.V. Truong. Scheduling Broadcasts in Multihop Radio Networks. *IEEE Transactions on Communications*, 38(4):456-60, April 1990.

[7] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, and D. Estrin. Em*: a Software Environment for Developing and Deploying Wireless Sensor Networks. *In Proceedings of the 2004 USINEX Technical Conference*. 2004

[8] Z. Haas, J. Halpern, L. Li. Gossip-Based Ad Hoc Routing. In *Proceedings of the IEEE Infocom* Pages 1707-1716. New York, NY, June 2002.

[9] John Heidemann, Fabio Silva, and Deborah Estrin. Matching Data Dissemination Algorithms to Application Requirements. In Proceedings of the ACM SenSys Conference, pp. 218-229. Los Angeles, California, USA, ACM. November, 2003.

[10] John Heidemann, Fabio Silva, Charlemek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, and Deepak Ganesan. Building efficient wireless sensor networks with low level naming. In *Proceedings of the Symposium on Operating System Principles*, pages 146-159, Chateau Lake Louise, Banff, Alberta, Canada, October 2001.

[11] John Heidemann, Fabio Silva, Yan Yu, Deborah Estrin, and Padma Haldar. Diffusion Filters as a Flexible Architecture for Event Notification in Wireless Sensor Networks. USC/ISI Technical Report 2002-556

[12] C. Ho, K. Obraczka, G. Tsudik, and K. Viswanath. Flooding for Reliable Multicast in Multi-hop and Ad-hoc Networks. In *Proceedings of the Intl. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM)*, pages 64-71, 1999.

[13] Q. Huang, C. Lu, and G. Roman, Spatiotemporal Multicast in Sensor Networks. In Proceedings of the ACM SenSys Conference, pp. 218-229. Los Angeles, California, USA, ACM. November, 2003.

[14] D. Johnson, and D. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. in Mobile Computing, pages 153-181. Kluwer Academic, 1996.

[15] J. Kulik, W Rabiner, and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, pages 174-185, Seattle, Washington, USA 1999

[16] LAN MAN Standards Committee of the IEEE Computer Society, Wireless LAN medium access control (MAC) and physical layer (PHY) specification. IEEE Std. 802.11, IEEE 1997

[17] S.-J. Lee, W. Su, and M. Gerla. On-demand multicast routing protocol in multihop wireless mobile networks. *ACM/Kluwer MONET*, 7(6):441–453, Dec. 2002.

[18] D. Li, K. Wong, Y.H. Hu, and A. Sayeed. Detection, Classification and Tracking of Targets in Distributed Sensor Networks. IEEE Signal Processing Magazine, vol. 19, no. 2, March 2002.

[19] H. Lim, and C. Kim. Multicast Tree Construction and Flooding in Wireless Ad Hoc Networks. In *Proceedings of the International Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems* (MSWIM), 2000.

[20] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. The Design of an Acquisitional Query Processor for Sensor Networks. In *Proceedings of the ACM SIGMOD*, 2003.

[21] J. Moy. OSPF Version 2, RFC2328, July 1997

[22]V. R. Obraczka, J. Garcia-Luna-Aceves. Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks. In *Proc. First International Conference on Embedded Networked Sensor System (SenSys 01),* , pages 181-192. Los Angeles, 2003.

[23] K. Obraczka & K. Viswanath, *Flooding for Reliable Multicast in Mutlti-Hop Ad Hoc Networks*, PARSEC Workshop'99

[24] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 90–100, New Orleans, LA, Feb. 1999.

[25] Larry Peterson and Bruce Davie. Computer Networks A Systems Approach, Morgan Kaufman Inc., 2003. ISBN 1-55860-832-X.

[26] J.Pu, E.Manning, G.Shoja, Routing Reliability Analysis of Partially Disjoint Paths, In *Proceedings of PACRIM'01*, Victoria, 2001.

[27] Abhishek Rajgarhia, Fred Stann, and John Heidemann. Privacy Sensitive Monitoring With a Mix of IR Sensors and Cameras. In *Proceedings of the Second International Workshop on Sensor and Actor Network Protocols and Applications*, pages 21-29, Boston, August 2004

[28] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. Data-Centric Storage in Sensornets with GHT, A Geographic Hash Table. In Mobile Networks and Applications (MONET), pages 427-442 Kluwer, 2003.

[29] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: a Hybrid MAC for Wireless Sensor Networks," Technical Report, Department of Computer Science, North Carolina State University, April 2005. http://www.csc.ncsu.edu/faculty/rhee/export/zmac

[30] Tony Speakman, Dino Farinacci, Steven Lin, Alex Tweedly, Nidhi Bhaskar, Richard Edmonstone, Rajitha Sumanasekera, and Lorenzo Vicisano. PGM Reliable Transport Protocol. *Internet-Draft (describing protocols used by Cisco and Whitebarn)* Feb 2001.

[31] Fred Stann and John Heidemann. RMST: Reliable Data Transport in Sensor Networks. In *Proceedings of the First IEEE Intl. Workshop on Sensor Network Protocols and Applications*, pages 102-112, Alaska, May 2003.

[32] Fred Stann, and John Heidemann. BARD: Bayesian-Assisted Resource Discovery In Sensor Networks. In Proceedings of the 24th IEEE INFOCOM Conference. Miami, Florida, USA. March, 2005.

[33] K. Tang, and M. Gerla. MAC Layer Broadcast Support in 802.11 Networks. In *Proceedings of IEEE MILCOM 2001*, pages 544-548 Oct. 2001

[34] Kumar Viswanath, and Katia Obraczka. Modeling the Performance of Flooding in MultiHop Ad Hoc Networks, *Symposium on Performance Evaluation of Computer and Telecommunication Systems* (SPECTS'04)

[35] K. Whitehouse, A. Woo, F. Jiang, J. Polastre, and D. Cutler. Exploiting the Capture Effect for Collision Detection and Recovery. *The Second IEEE Workshop on Embedded Networked Sensors (EmNetS-II)*. Sydney, Australia. May 30-31, 2005.

[36] B. Williams, and T. Camp. Comparason of Broadcasting Techniques for Mobile Ad Hoc Networks. In *Proceedings of MOBIHOC'02,* Lausanne, Switzerland, 2002.

[37] B. Williams, D Mehta, T. Camp, and W. Navidi. Predictive Modeling of Network Wide Broadcasting Protocols for Mobile Ad Hoc Networks. Technical Report MCS-03-05, The Colorado School of Mines, April 2003.

[38] A. Woo, T. Tong, and D. Culler. Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks. In *Proc. First International Conference on Embedded Networked Sensor System (SenSys 01),*, pages 14-27. Los Angeles, 2003.

[39] Wei Ye, John Heidemann, and Deborah Estrin. An Energy-efficient Mac Protocol for Wireless Sensor Networks. In *Proc. IEEE Infocom*, pages 1567-1576. New York, 2002.

[40] J. Zhao, and R. Govindan. Understanding Packet Delivery Performance in Dense Wireless Sensor Networks. In *Proc. First International Conference on Embedded Networked Sensor System (SenSys 01),*, pages 1-13. Los Angeles, 2003.

[41] F. Zhao, J. Shin, and J. Reich. Information-Driven Dynamic Sensor Collaboration for Tracking Applications. In *IEEE Signal Processing Magazine*, 19(2):61-72, March 2002