Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks

Alec Woo[†], Terence Tong[†], David Culler[†‡]
{awoo, terence, culler}@cs.berkeley.edu

[†]Computer Science Division
University of California
Berkeley, California

[‡]Intel Research: Berkeley
Intel Corporation
Berkeley, California

ABSTRACT

The dynamic and lossy nature of wireless communication poses major challenges to reliable, self-organizing multihop networks. These non-ideal characteristics are more problematic with the primitive, low-power radio transceivers found in sensor networks, and raise new issues that routing protocols must address. Link connectivity statistics should be captured dynamically through an efficient yet adaptive link estimator and routing decisions should exploit such connectivity statistics to achieve reliability. Link status and routing information must be maintained in a neighborhood table with constant space regardless of cell density. We study and evaluate link estimator, neighborhood table management, and reliable routing protocol techniques. We focus on a many-to-one, periodic data collection workload. We narrow the design space through evaluations on large-scale, high-level simulations to 50-node, in-depth empirical experiments. The most effective solution uses a simple time averaged EWMA estimator, frequency based table management, and cost-based routing.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks:**]: Network Protocols

General Terms

Algorithms, Performance, Design, Implementation

Keywords

Sensor Networks, Multihop Routing, Reliability, Link Estimation, Neighborhood Management

1. INTRODUCTION

The routing problem for sensor networks differs substantially from that of traditional ad-hoc wireless networks because sensor nets typically involve many resource constrained nodes that are densely connected by low-power radios and operate in aggregate over multiple hops to achieve some application-specific communication pattern. In the more traditional setting, we typically assume 802.11 links, abstracting away the underlying physical layer and MAC protocol, and collections of independent pair-wise connections, abstracting the application. In sensor nets we have the opportunity and the requirement to consider a larger design space, because factors from below and above potentially interact heavily with routing. For example, a basic notion like shortest path is only well-formed relative to some definition of a connectivity graph describing which nodes can communicate over a single hop. For an actual sensor network, the connectivity graph is discovered by nodes observing communication events and sharing this information. Moreover, connectivity is not a simple binary relation, but a statement of the likelihood of successful communication. A nearby node may be in communication most of the time, but not always, depending on interference, congestion, and other sources of loss. Communication may occur less reliably with nodes far away, but there are many distant nodes and a few of them are likely to have strong connectivity. Generally, many of the links are lossy and the loss rate may change dynamically with environmental factors or due to contention arising from the highly correlated behavior of the application. Thus, routing algorithms for sensor nets should take into account these underlying factors and be evaluated in concert with the lower level estimation mechanisms under realistic loads.

In this paper, we explore connectivity analysis, neighborhood management, and routing on dense sensor networks with simple, low-power radios and limited storage. To ground the application context, we consider a basic data gathering communication pattern, where a large collection of nodes route periodically sampled data over multiple hops to an individual sink. This might be one directed diffusion flow along a simple gradient or a single selection query. To ground the hardware and physical configuration factors, we focus on large sensor fields of Berkeley TinyOS motes [13] spread roughly as a uniform grid over a large, essentially unobstructed, indoor space.

Our study is conducted in four stages. Section 2 presents an empirical characterization of the links experienced on this platform. We measure the loss rates for many different pairs of nodes at a range of distances. This yields a model where both the mean link quality and variance in quality are a function of distance. For large collections of nodes, this simple model exhibits qualitatively the irregular connectivity patterns that have been observed in real deployments [8].

Section 3 investigates simple, efficient estimators for determining these link probabilities. Here, a node can estimate the quality of the link from other nodes passively by collecting statistics on packets it happens to hear, or by actively probing. Many link estimators have been proposed for various wireless channels and transport protocols, and we want to identify one that reacts quickly to changes in link rate, yet is stable with small error or bias, and requires little storage and computation. To explore this subset of the design space, we employ a synthetic loss rate generator for which the underlying probability is known. This narrows the options to a few choices that are examined on empirical traces. We find that a new exponentially weighted average of windowed averages is most attractive. This option is used for the remaining simulation and empirical evaluations.

Section 4 explores how to manage a finite, typically small, neighborhood table in which connectivity and routing information is kept. Given the observed link characteristics, we expect that in a large, dense sensor field, each node will have a few good neighbors and many low quality links to other nodes. In order to build an estimation of a link, the statistical history for the remote node must be in the neighborhood table and it must stay there long enough to obtain meaningful statistics. Connectivity varies with changes in environmental conditions and mobility, so when a packet arrives from a node that is not in the small neighborhood table, how do we determine whether to allocate a slot to it and which entry to evict to do so? This problem is related to on-line frequency determination, where the goal is to identify the most important entries in a stream using a buffer that is roughly the size of the entries of interest. We find that an insertion policy with adaptive down-sampling and an eviction policy that preserves the most frequently occurring nodes is able to retain a large fraction of the best neighbors in the table.

With these mechanisms in place, Section 5 builds a framework for cost-based distributed route formation over the neighborhood structure and investigates various forms of shortest-path and minimum transmission routing. In the case of a single data gathering flow, route formation reduces to maintaining a next-hop neighbor, or parent, forming a routing tree directed at the sink. Routing is accomplished by transmitting data along each such hop toward the sink, possibly retransmitting packets when losses occur. Underlying issues of parent selection, cost propagation, cycle avoidance, duplication elimination, and queue management are addressed in formulating the routing framework.

Section 6 evaluates a wide range of routing alternatives in terms of hop distribution, path reliability, and stability of the routes on large networks. It proceeds in three levels of fidelity. To develop intuition on the trade-offs involved, an initial analysis is performed on probabilistic connectivity graphs, where the link quality of each edge is selected from a probability distribution based on empirical characterization. Packet level simulations on 100-node networks are performed on several routing algorithms to capture additional fidelity, dynamic behavior, and contention arising from aggregate traffic. This analysis narrows the options that are selected for in-depth empirical evaluation on a 50-node network in a large indoor space. Each of numerous empirical runs takes about three hours in a building foyer, performed during times when pedestrian traffic is low.

Overall, this study provides a wealth of data on the interactions across the system layers involved in the routing problem for sensor networks, as well as presenting new, simple routing techniques. It also exhibits an empirical methodology for slicing through this very large design space, which is both application dependent and unconstrained by link-level standards. Although our evaluation is on stationary networks, our design is capable of adapting to network dynamics, mobility, or obstructions.

2. EMPIRICAL LINK CHARACTERISTICS

The starting point for development of a practical topology formation and routing algorithm is an understanding of the loss behavior of the link under various circumstances. Previous studies have indicated that radio connectivity is imperfect and non-uniform, even in ideal settings [3, 4, 8]. Rather than carry along a detailed model of the platform or the propagation physics, we seek a simple characterization of the channel that can be used in topology formation.

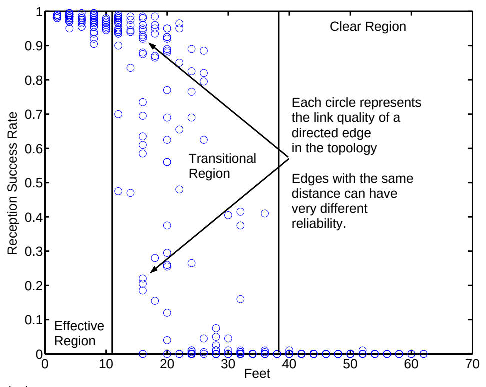
2.1 Hardware Platform

The hardware platform is the Berkeley Mica mote running TinyOS. Each sensor node consists a 4MHz Atmel microprocessor with 128kB of programmable memory and 4kB of data memory. The network device is a RF Monolithics 916MHz, amplitude shift keying (ASK) RF transceiver, capable of delivering up to 40 kbps; it emits less than a milliwatt of RF power [14]. The RF transmit power can be tuned in software, with 0 being the maximum and 100 being the minimum. A node can be configured as a base station to route over standard serial port interface by attaching an extra hardware board. The base station serves as the traffic sink. TinyOS [15] provides a programming environment and a complete network stack on this platform. Its active message layer provides a connectionless packet abstraction, with a normal packet size being about 30 bytes. A DC-balanced SECDED scheme is used for encoding each byte. A 16-bit CRC is computed over the entire packet to determine successful packet reception. A link-level acknowledgment can be sent by the receiver for each packet successfully received. A simple CSMA-based MAC is employed [22]; it adds a random delay before listening for an idle channel and backs off with a random delay over a predefined window when the channel is busy.

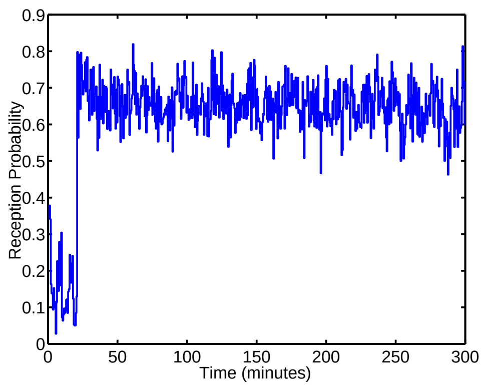
2.2 Empirical Observations

To characterize the empirical link quality on this platform, we measured loss rates between many different pairs of nodes at many different distances. To do this efficiently a sequence of sensor nodes is arranged linearly with a spacing of 2 feet. Each node is scheduled to transmit 200 packets at a given power level at 8 packets/s; at any time, there is one transmitter and the remaining nodes count the number of packets successfully received from the transmitter. Thus, for a single transmitter, we obtain numerous measurements at different distances.

Figure 1(a) shows a scatter plot of how link quality varies over distance for a collection of nodes on the ground in an open tennis court for a power level of 50. A number of other settings show analogous structure. As expected, for a given power setting there is a distance within which essentially all



(a) Reception probability of all links in a network with a line topology.



(b) After 20 minutes, the sender is moved from 15 ft to 8 ft from the receiver and remained stationary for four hours.

Figure 1: Empirical results illustrating variations in reception probability.

nodes have good connectivity. The size of this ***effective region*** increases with transmit power. There is also a point beyond which essentially all nodes have poor connectivity. However, very distant nodes occasionally do transfer packets successfully. In the ***transitional region*** between these points, the average link quality falls off smoothly, but individual pairs exhibit high variation. Some relatively close pairs have poor connectivity, while some distant pairs have excellent connectivity. A fraction of pairs have intermediate loss rates and asymmetric links are common in the transitional region; similar results have also been reported in [3].

The next question is whether link quality is stable when nodes are immobile. With a fixed source sending to a receiver at a given distance, we would like to observe how link quality changes over time. Figure 1(b) shows a situation where a transmitter sends 8 packets/s in an indoor environment for a period of 20 minutes at a distance of 15 feet and then is moved closer to the receiver where it remains stationary for four hours. We see that link quality can undergo abrupt changes. At each distance the mean link quality is relatively stable, and intermediate between the present/absent extremes. Furthermore, there is significant variation in the instantaneous link quality. For example, the link quality exhibits a mean of about 65% with about 10% swing, using a sample size of 240 packets.

If we apply this link characterization to a large field of

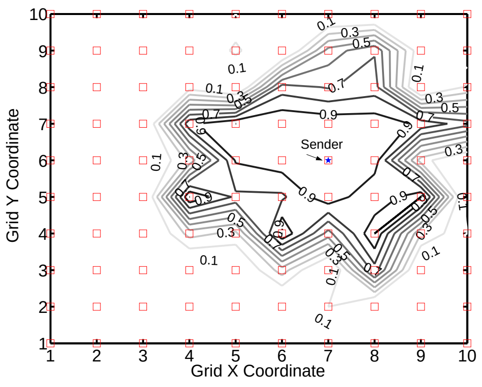


Figure 2: Cell connectivity of a node in a grid with 8-foot spacing as generated by our link quality model.

nodes, we expect a small, somewhat irregular region of nodes that share good connectivity. Some more distant nodes are expected to also have good connectivity. Many nodes over a large, very irregular region will have limited, but non-zero connectivity. Many of the intermediate nodes will have asymmetric connectivity. This is the behavior observed in deployments [8]. If all nodes transmit periodically, a node will receive packets frequently from each of its good neighbors, but it will also receive numerous packets from many more remote nodes.

These observations suggest a simple means of capturing probabilistic link behavior in simulations while abstracting away the complex sources of loss. We compute the mean and variance in Figure 1(a) to create a link quality model with respect to distance. For each directed node pair at a given distance, we associate a link probability based on the mean and variance extracted from the empirical data, assuming such variance follows a normal distribution. Each simulated packet transmission is filtered out with this probability. An instance showing how this model captures a node's connectivity cell is shown in Figure 2; it matches well with empirical observation. This model of link quality is used for all simulation studies below, allowing more of the design space to be explored while incorporating some of the most significant variations observed in practice.

3. LINK ESTIMATION

Individual nodes estimate link quality by observing packet success and loss events. Higher-level protocols use these estimations to build routing structures. We seek to find an estimator that reacts quickly to potentially large changes in link quality, yet is stable, has a small memory footprint, and is simple to compute. Reacting to changes quickly allows higher-level protocols to adapt to environmental changes and mobility. However, estimations must also be fairly stable; if they fluctuate wildly, the routing topology is unlikely to stabilize and routing problems, such as cycles and stranded nodes, will be common. The memory footprint of the estimator must be small, because we have limited storage in which to represent the neighborhood, and its computational load should be small, since only limited processing is available and it costs energy.

For sensor networks, the broadcast nature of the wireless medium allows passive estimation to be performed simply by snooping on the channel; losses can be inferred by tracking the link sequence number in the packets from each source. Estimation through snooping comes at a cost, since the node listens for packets that are not necessarily addressed to it, but in many cases the packets are received anyways. Various low-power listening mechanisms exist [21] that would enable snooping at a much lower cost. An alternative approach is to use received signal strength as an indication of link quality. However, the use of forward error correction and uncertainties involved in such measurements suggest that signal strength can be a poor indicator on link quality [4]. We focus on snooping techniques.

A subtlety in the passive estimation approach is that no loss information is detected between successful packet receptions. In particular, if a node disappears, its estimate should degrade to zero. One resolution of this issue is to assume a minimum data transmission rate for each node, as is the case in many sensor net applications. This also provides better interpolation for links with high loss rate.

Passive probing to estimate link quality for wired and power-rich wireless networks is well established. It is widely deployed over the Internet in protocols such as Internal Gateway Routing Protocol (IGRP) [12] and Enhanced IGRP (EIGRP) [1]. Link quality is measured as the percent of packets that arrived undamaged on a link. It is reported by the network interface hardware or firmware, and is calculated as a moving average. In IGRP, packet loss is not silent, since each packet comes in on a particular link, with both sender and receiver known *a priori*. Incoming packets are always detected and damaged packets are losses. For wireless networks, the channel is a broadcast medium and packets can be damaged or totally missed by the receiver. Furthermore, the link error dynamics are expected to be very different for low power wireless radios.

Many link estimation techniques exist [23], so we desired to test many of the established candidates under conditions similar to our sensor network. The resource constraints significantly limit the amount of processing and storage that can be used for estimations, so complex techniques, such as linear regression or Kalman filters, are likely to be impractical. We studied several simple and efficient candidate estimators including: exponentially weighted moving average (EWMA), moving average, time weighted moving average, and packet loss/success interval with EWMA. Each of these estimators has tuning parameters that allow them to be made either more agile or more stable. We also introduced a new estimator, window mean with EWMA (WMEWMA). WMEWMA(t, α) computes an average success rate over a time period $\frac{Packets\ Received\ in\ t}{max(Packets\ Expected\ in\ t, Packets\ Received\ in\ t)}$ and smoothens the average with an EWMA. The tuning parameters are t and α, where t is the time window represented in number of message opportunities and $\alpha \in [0, 1]$ controls the history of the estimator. In each case, a minimum message rate is assumed and a periodic timer event is provided so the estimator can infer losses prior to the next packet reception.

To study this relatively large design space systematically, we created a simple synthetic model that generates link loss characteristics similar to our empirical traces, i.e., a Bernoulli loss process between discrete changes to the mean loss rate. In addition to efficient simulations, the estimator error is well defined. The leading candidates were also tested on empirical traces, using the simple EWMA estimator as a basis for comparison. The details of the link estimator study can be found in [23].

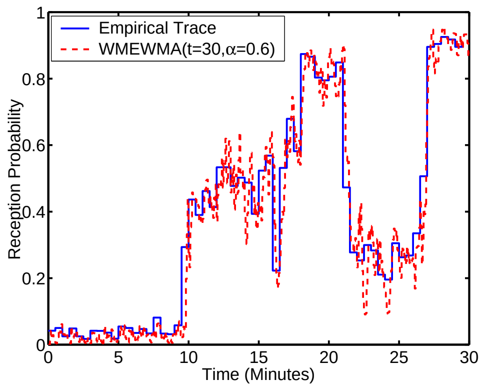


Figure 3: WMEWMA($t = 30, \alpha = 0.6$) with stable setting using empirical traces.

We tuned each estimator for two settings: stable and agile. For stable estimators, we aim to minimize the settling time, which is the length of time for the estimator to converge within ±10% of the actual value and remain within this error bound. For agile estimators, we aim to minimize the mean squared error while meeting a crossing time, i.e., requiring that the estimators reach ±10% of the actual value within 40 packet opportunities. With such tuned estimators, we compare settling time, crossing time, mean square error, and coefficient of variance, as well as memory resources and computational requirement.

Based on this study, we found that WMEWMA performs best overall. The storage requirement is constant for all tunings. The link quality estimation at both extremes (0% and 100%) is accurate and has a small settling time. The worst case is when link quality is close to 50%. Figure 3 shows how WMEWMA performs on an empirical trace with a stable setting tuned in simulation. It is clear that WMEWMA tracks the empirical trace fairly well. The degree of overshoot and undershoot is higher than the tuning objective of 10% in simulation. This is expected since empirical traces have larger variances than our simulated traces.

We found that all of the agile estimators yielded poor estimations with large mean square error and variances. Therefore, agile estimations should only be used to detect significant changes of link quality, such as node disappearance.

For stable estimators, we found that decreasing the settling time increases the degree of overshoot and undershoot. A ±10% noise margin already requires a settling time on the order of 100 packets. This suggests a limit on how rapidly routes can adapt to link quality changes, especially when the message rate is low. Nonetheless, the actual traffic rate is greater than the data generation rate of a node due to forwarded traffic, especially near the sink.

Based on this evaluation, we can narrow the design space and use a stable WMEWMA link estimator underneath the routing protocols studied below.

4. NEIGHBORHOOD TABLE MANAGEMENT

A node performs neighbor discovery by recording information about nodes from which it receives packets, either as a result of passive monitoring or active probing. Link estimation is used to determine which nodes should be considered neighbors in the distributed connectivity graph. However, in a dense network, a node may receive packets from many more nodes than it can represent in its neighbor table. How does a node determine, over time, in which nodes it should invest its limited neighbor table resources to maintain link statistics? The problem is that if a node is not in the table, there is no place to record the link information, so there is no way for it build up its link quality and become a neighbor. Controlling the transmission power to adjust cell density is often an application-specific or deployment parameter since it affects overall network hop count, latency, channel utilization, network lifetime, and quality of links. For example, [20] adjusts the transmit power to control the topology and minimize energy required to transport data.

Neighborhood management essentially has three components: insertion, eviction, and reinforcement. For each incoming packet upon which neighbor analysis is performed, the source is considered for insertion or reinforcement. If the source is represented in the table, a reinforcement operation may be performed to keep it there. If the source is not present and the table is full, the node must decide whether to discard information associated with the source or evict another node from the table. We seek to develop a neighborhood management algorithm that will keep a sufficient number of good neighbors in the table regardless of cell density. Ultimately, the goodness criteria should reflect which nodes are most useful for routing. For example, we would want to discard nodes with low-quality links, as they are poor routing neighbors. With a link quality distribution as indicated by Figure 1(a), a node in a field of sensors will hear from many more weakly connected, distant nodes, than from well-connected ones. However, a node should hear from the well-connected nodes more frequently, since few of their packets are lost. The management algorithm should prevent the table from being polluted by many low utility neighbors and allow new valuable neighbors to enter.

This problem has aspects in common with cache management and with statistical estimation techniques in databases. There is a growing body of work in gathering statistics on Internet packet streams using memory much smaller than the number of distinct classes of packets. Heuristics are used in [7] to identify a set of most frequently occurring packet classes. Two algorithms are presented in [10] to identify all values with frequency of occurrence exceeding a user specified threshold. A sliding window approach is used in [5] that can be generalized to estimate statistical information of a data stream. Finally, [6] showed a simple FREQUENCY algorithm that estimates frequency of Internet packet streams with limited space. We first consider table management that seeks to retain a stable subset of high-quality links and then specialize this approach for routing.

4.1 Neighborhood Management Policy

We focus on passive neighborhood discovery, where nodes snoop on periodic data messages. Each table entry contains link estimation and routing data. When a node is evicted from the table, its link estimation is lost. Insertions are performed if the table is not full, while evictions are performed only if the table is full.

4.1.1 Insertion Policy

Upon hearing from a non-resident source, we must determine whether to insert it. No historical information can be used, since there is no table entry allocated. In some cases it might be possible to use geographic information or signal strength associated with the packet, but geographic data is often absent and does not account for obstructions and signal strength is highly variable, so we look for a simple statistical method. The insertion policy should avoid overrunning the neighbor table with a high rate of insertion so a stable set of neighbors can be established. Probabilistic down-sampling is the usual technique for controlling insertion rate.

A down-sampling scheme with a fixed probability is simple, but setting a right value for every node is difficult since the number of potential neighbors can vary significantly. We employ an adaptive down-sampling scheme that sets the probability of insertion to be the ratio of the neighbor table size, T, to the number of distinct neighbors, N. We assume that it is possible to only consider periodic messages, such as beacons, for insertion, so all N neighbors generate a roughly equal discovery rate. For data collection, we may consider originating packets and ignore forwarding packets. On average, at most T entries can be inserted into the table for every N messages received, giving all nodes a chance to get established. To estimate N, there exists much prior work in the database literature to estimate the number of distinct values over a continuous stream [9]. However, in the case when periodic beacons are present, we simply count the average number of beacons received between beacons from a particular node.

The insertion policy should reinforce good neighbors that are in the table. We use the sticky policy described in [10]. At insertion time, if the node is already in the table, the adaptive down-sampling mechanism is bypassed and an action is performed to reinforce this node. The specific action depends on the eviction policy. Finally, the insertion policy can also take into account other goodness metrics in making its insertion decision, such preferring potential parents and children to siblings in the routing graph.

4.1.2 Eviction and Reinforcement Policy

We consider several candidate eviction policies. The simplest approach is merely round robin through the table. No reinforcement is performed. Drawing on cache management techniques, we may also consider FIFO, Least-Recently Heard (LRH), or CLOCK algorithm approximations to LRU. For FIFO, eviction is based on order of entry, so no reinforcement is performed. For LRH, the resident entry is made most recently heard. For the CLOCK algorithm, reinforcement sets the reference bit to 1. On eviction, the table is scanned, clearing reference bits, till an unreferenced entry is found. A simple policy found in estimating the most frequent values over a stream using limited space is the FREQUENCY algorithm[6]. It keeps a frequency count for each entry in the table. On insertion, a node is reinforced by incrementing its count. A new node will be inserted in the table if there is an entry with a count of zero; otherwise, the count of all entries is decremented by one and the new candidate is dropped.

4.2 Evaluation

The management policy should retain as many good neighbors in the table as possible. To evaluate the different policies, we first measure the ***yield***, i.e., the fraction of good neighbors that are found in the table more than 75% of the time. A good neighbor is defined to be a node with link quality greater than 75%.

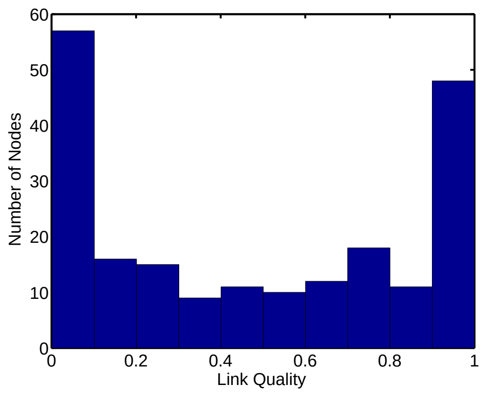


Figure 4: A center node in a 80x80 grid with 4 feet spacing using our empirical link model can have up to 207 neighbors with very different link quality distribution.

The nodes are placed uniformly as a grid. We use the probabilistic link model for connectivity derived from Figure 1(a) and simulate each node transmitting 100 packets with no routing. Using 80x80 grid with 4 feet spacing, so that the effective region covers nodes within 3 grid points in either direction, we consider a typical node near the center in a dense network. It has 207 potential neighbors, i.e., nodes from which it hears at least one packet. Figure 4 shows the link quality distribution of this node's cell. As expected, many nodes fall in the transitional distance with unreliable links. About 30% of the nodes have link quality greater than 75%. Repeating the study for different grid spacings shows that this ratio remains roughly constant as the number of potential neighbors ranges from 20 to 200.

To compare table management policies, we fix the table size and measure the yield as the node density is increased. Figure 5 shows how the different policies perform at different densities with a table size of 40 entries. As expected, all policies perform well when the table holds all the potential neighbors. When the number of good neighbors exceeds the table size, i.e., when the potential neighbors is three times the table size, the cache based policies are unable to hold onto a subset of good neighbors. FREQUENCY retains 20 neighbors, or 50% yield, even at high densities. Further experiments varying the table size show that this policy maintains over 30% yield for table sizes ranging from half to three times the number of good neighbors.

We conclude that FREQUENCY is very effective in maintaining a subset of good neighbors over a fixed-size table, even for densities much greater than the table size. For example, with a table of 32 entries, this policy yields at least 10 good neighbors at all measured densities.

Besides frequency count, the neighborhood management policy can also take into account other goodness metrics in selecting its neighbors such as routing cost, lifetime of a neighbor, scheduling issues, or aggregation opportunity. In this study, we take the basic goodness criteria and focus mostly on the frequency metric. In the evaluation of routing protocols in Section 6, we augment the route table management policy further by taking into account routing cost as the goodness metric. Basically, we avoid maintaining sibling nodes (nodes with roughly the same routing cost) since they are likely to find their own routing paths.

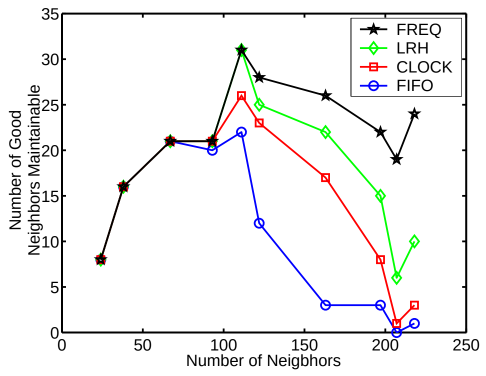


Figure 5: Number of good neighbors maintainable at different densities with a table size of 40 entries.

5. ROUTING PROTOCOL

The primary focus of this section is to explore the design issues for routing protocols that exploit connectivity determined by lower level link estimations to form a routing topology. Many wireless routing protocols assume connectivity by hearing certain packets[2, 16, 17, 18, 19]. For example, ROUTE_REQUEST and ROUTE_REPLY messages are often building blocks for creating a routing structure. However, Figure 1(a) suggest that long unreliable links are likely to influence these protocols and result in building paths with poor end-to-end reliability. On hardware similar to ours, a flood-based routing protocol, where the source of the first message heard in each time epoch is chosen as parent, has been shown at scale to form convoluted trees with many long and unreliable links [8]. It is essential that the quality of a link be established before it is selected for routing and that this selection is not undermined by contention. We consider connectivity based on link estimation. Various protocols utilize this information to formulate the specific cost metrics for which they optimize.

Our study focuses on the data-collection (many-to-one) routing scenario, as it is the most basic communication pattern for sensor networks and brings forward the issues that need to be considered in any pattern. We first provide an overview of our proposed routing protocol framework. We then discuss several underlying design choices and propose a set of cost metrics for routing.

5.1 Protocol Implementation Framework

Figure 6 captures the high level interactions of all the components implementing our routing protocols. Each node maintains estimates of inbound (reception) link quality. Routing should be based on outbound (transmission) link, so this information needs to be propagated back to the neighbors. The core component is the neighbor table which contains status and routing entries for neighbors; its fields include MAC address, routing cost, parent address, child flag, reception (inbound) link quality, send (outbound) link quality, and link estimator data structures. Below the routing layer, all packets on the channel are snooped by the estimator, with insertions controlled by the neighbor table manager.

The routing protocols are distributed distance-vector based approaches implemented by the parent selection component. Parent selection is run periodically to identify one of the neighbors for routing; it may also broadcast (locally) a route message. The route messages include parent address, esti-

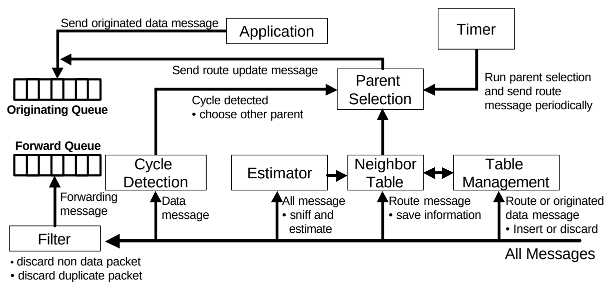


Figure 6: Message flow chart to illustrate the different components for routing.

mated routing cost to the sink, and a list of reception link estimations of neighbors. When a route message is received from a node that is resident in the neighbor table, the corresponding entry is updated. Otherwise, the neighbor table manager decides whether to insert the node or drop the update. Data packets originating from the node, i.e., outputs of local sensor processing, are queued for sending with the parent as the destination. Incoming data packets are selectively forwarded through the forwarding queue with the current parent as destination address. The corresponding neighbor table entry is flagged as a child to avoid cycles in parent selection. Duplicate forwarding packets are eliminated. When cycles are detected on forwarding packets, parent selection is triggered with the current parent demoted to break the cycle.

5.2 Underlying Issues

Parent Selection: Many distance-vector based algorithms can be implemented in this framework, using different cost metrics to guide routing. The cost of a node is an abstract measure of distance; it may be number of hops, expected number of transmissions, or some other estimate of energy required to reach the sink. When scheduled to run, the routing algorithm accesses the neighbor table and extracts a set of potential parents. A neighbor is selected as a potential parent only if its cost is less than the current cost of the node. A node may switch to a new parent if one is sufficiently smaller in cost by some margin than the current parent. It may also switch to a new parent if the link quality to the current parent drops below some threshold, if the sink is unreachable through the current parent, or if a cycle is detected.

When connectivity to the current parent worsens, its link estimation will automatically degrade over time, allowing the selection of a new parent. This is in contrast to traditional link detection technique found in [17, 18, 19], which counts the number of transmission failures, and is better suited to handle semi-lossy links. If connectivity to the current parent is lost and no potential parents are available, the node declares it to have no parent, disjoins from the tree, and sets its routing cost to infinity.

Rate of Parent Change: Regardless of the routing algorithm, routes can be changed whenever the parent selection algorithm is scheduled to run. For fast adaptation, it is tempting to schedule the parent selection component to evaluate new routes for every route update received from neighboring nodes. However, a domino effect of route changes is likely to be triggered across the entire network, especially when routing costs are very sensitive. To achieve a stable topology, routes are evaluated on a periodic basis, rather than upon receiving a route update, except when a cycle is detected.

Packet Snooping: Given the wireless network is a broadcast medium, a lot of information can be extracted by snooping. Link estimation is one example. At the routing level, since each node is a router, snooping on forwarding packets allows a node to learn about all its children, which is useful to prevent cycle formation. Furthermore, snooping on a neighboring node's messages is a quick way to learn about its parent, which decreases the chance of stale information causing a direct two-hop cycle. The same technique can also be used to prune children quickly in the case of a network partition. When a node with an unreachable route receives a forwarding message from its child, it will NACK by forwarding the child's message with a 'NO_ROUTE' address. All neighboring nodes, including its children, snooping on this packet can quickly learn about an unreachable route. In fact, this naturally provides feedback deep down into the tree, in effect solving the counting-to-infinity problem.

Cycles: For many-to-one routing over relatively stationary sensor networks, we believe that it is better to use simple mechanisms to mostly avoid loop formation and to break cycles when they are detected, rather than to employ heavy weight protocols with inter-nodal coordination. DSDV [18] provides an attractive approach to avoid cycles for mobile networks, but it requires sequence number propagation and sequence number settling time tuning, which may differ in each deployment.

We rely on techniques similar to poison-reverse or split-horizon [11]. By monitoring forwarding traffic and snooping on the parent address in each neighbor's messages, neighboring child nodes can be identified and will not be considered as potential parents. We only need to maintain this information for nodes in the neighbor table. Route invalidation when a node becomes disjoint from a tree and tree pruning by 'NACKing' children's traffic are used to alleviate stale information, which leads to cycles.

With these simple mechanisms, cycles may potentially occur and must be detected. Since each node is a router and a data source, cycles can be detected quickly when a node in a loop originates a packet and sees it returning. This mechanism works as long as queue management policy avoids letting forwarding traffic suppress originated traffic. (Otherwise, packets may get stuck in a loop in the middle of a route without detection.) This level of fairness is an appropriate policy in any case. Once a cycle is detected, discarding the parent by choosing a new one or becoming disjoint from the tree will break it. Alternatively, a Time-To-Live field can be added, but we did not use it in our evaluation.

Duplicate Packet Elimination: Duplicate packets can be created upon retransmission when the ACK is lost. Without duplicate packet elimination, these will be forwarded, possibly causing more retransmissions and more contention, plus they waste energy. To avoid duplicate packets, the routing layer at the originating node appends the sender ID and an originating sequence number in the routing header. To suppress forwarding duplicate packets, each parent retains the most recent originator ID and originating sequence number in child entries in the neighbor table. This approach relies on in-order packet delivery during retransmission and assumes that the neighbor table is able to track children. Alternatively a recent originator cache could be employed[16].

Queue Management: Nodes high in the tree forward many more messages than they originate. Care must be taken to ensure that forwarding messages do not entirely dominate the transmission queue, since it would prevent the node from originating data and undermine cycle detection. We separate the forwarding and originating messages into two queues so that upstream bandwidth is allocated according to a fair sharing policy. The policy that we implemented is very simple. With the assumption that originating data rate is low compare to that of forwarding messages, we give priority to originating traffic. For data collection it is possible to estimate the ratio of forwarding to originating packets by counting the descendents of each parent, but a general treatment of fair queuing is beyond the focus of this study.

Relationship to Link Estimation: Link estimation and routing are not entirely independent. Link failure detection based on fixed number of consecutive transmission failures can be ineffective over semi-lossy links. An estimation of the link quality yields a much better judgment of link failure. With bi-directional link estimations, routing over asymmetric links can be avoided. The stability and agility of link estimation can directly affect the stability of the routes and the rate of route adaptation, especially when the estimations are combined to form a distance metric describing a path. Therefore, the final tuning of the link estimator must be done while observing its effect on routing performance.

5.3 Routing Cost Metrics

The traditional cost metric for distance-vector routing is hop count. In power-rich networks with highly reliable links, retransmissions are infrequent and hop count adequately captures the underlying cost of packet delivery to the destination. However, with lossy links, as found in many sensor networks, link-level retransmission is critical for reliable transport, as each hop may require one or more retransmissions to compensate for the lossy channel. If link quality is not considered in route selection, the real cost of packet delivery can be much larger than the hop count. Nevertheless, shortest path routing can still be useful in unreliable networks, given that poor links are filtered out from route selections. A simple technique is to apply shortest path routing only to links that have estimated link quality above a predetermined threshold. This has an effect of increasing the depth of the network, since reliable links are likely to be shorter. However, cell density and physical deployment may result in a connectivity graph where the set of above-threshold links fail to connect the network.

With links of varying quality, a longer path with fewer retransmissions may be better than a shorter path with many retransmissions. An alternative approach is to use the expected number of transmissions along the path as the cost metric for routing. That is, the best path is the one that minimizes the total number of transmissions (including retransmissions) in delivering a packet over potentially multiple hops to the destination. We call this the Minimum Transmission (MT) metric. In considering the expected number of transmissions of a link, it is important to determine link quality for both directions since losing an acknowledgment would also trigger a useless retransmission. For each link the MT cost is estimated by the product $\frac{1}{link\ quality_{forward}}\ x\ \frac{1}{link\ quality_{backward}}$, which is also proposed in [4]. The distance-vector algorithm computes overall cost of a path in the same manner as hop count with these weighted hops. In addition to optimizing for something closer to the true cost, MT eliminates the need for predetermined link thresholds. However, the stability of MT routing is potentially an issue, since the MT metric utilizes link estimations in a non-linear fashion. Thus, for MT a substantial noise margin should be used in parent selection to enhance stability.

Another cost metric is path reliability, which is a product of link qualities along the path. It is used in [24] to optimize end-to-end success rate without link retransmission. It has a tendency to yield long paths. We do not study this metric as our protocols take advantage of link retransmissions.

6. EVALUATION

Having established the framework for concrete implementations of a variety of routing protocols and the underlying building blocks, this section seeks to compare and evaluate a suite of distance-vector routing protocols in the context of data collection over a large field of networked sensors. We proceed through three levels of evaluation. The ideal behavior of these protocols, with perfect link estimation and no traffic effects, is assessed on large (400 node) networks using a simple analysis of network graphs with link qualities obtained from our probabilistic link characterization. The dynamics of the estimations and the protocols is then captured in abstract terms using a packet-level simulator. A wide range of protocols is investigated on 100-node networks under simulation. This narrows the set of choices and sheds light on key factors. The best protocols are then tested on real networks of 50 nodes in greater detail.

6.1 Protocols

The set of routing protocols under evaluation include:

I **Shortest Path (SP and SP(t))** protocols are the conventional distance-vector approach where each node picks a minimum hop-count neighbor and sets its hop-count to one greater than its parent. For SP a node is a neighbor if a packet is received from it. For SP(t) a node is a neighbor if its link quality exceeds threshold t. Thus, shortest path routing is performed within a sub-graph of high quality links. Based on Figure 1(a), we consider two values for t. With $t = 70\%$, we consider only links in the effective region, while leaving a significant noise margin for the estimators. With $t = 40\%$, we allow most of the good links in the transitional region, resulting in larger, less regular cells.

II **Minimum Transmission (MT)** uses the protocol described in Section 5 with the expected number of transmissions as its cost metric. Since the cost is built upon estimations, to enhance stability, a new path is used if the new cost is lesser by a noise margin.

III **Broadcast** is a naive flooding based protocol. The root periodically floods the network; a node chooses a parent based on the source address of the first flooding message that it receives in each epoch. The broadcast protocol essentially captures reverse path routing as found in protocols such as DSR [17].

IV **Destination Sequenced Distance Vector (DSDV)** uses destination based sequence numbers to avoid cycle formation [18]. We adopt DSDV into our framework and preserve the essence of the protocol; parent is chosen based on the 'freshest' sequence number from the root while maintaining a minimum hop count when possible. Similar to SP, DSDV ignores link quality and considers all nodes it hears as neighbors. Since our DSDV protocol evaluates routes on a periodic basis, except when a node is disjoint from the network, we naturally support sequence number settling for DSDV. A fixed number of consecutive packet losses to the next hop is used for link failure detection. When link failure is detected, a node is disjoint from the network and declares route unreachable to its neighbor over periodic route messages and originated traffic.

6.2 Evaluation Metrics

We define four important metrics for evaluating the performance of these protocols.

I **Hop Distribution** measures routing depth of nodes throughout the network, which reflects both end-to-end latency and energy usage.

II **Path Reliability** is the product of link quality along the path from each node in the network. It approximates the end-to-end reliability of a routing path in the absence of retransmission.

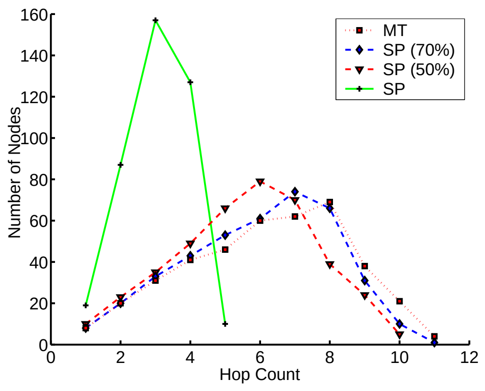
III **End-to-End Success Rate** is the number of packets received at the sink for a node divided by the number originated. A maximum number of link retransmissions is performed at each hop. Losing packets before reaching the sink not only wastes energy and network resources, but also degrades the quality of application. Another subtle issue is fairness. Nodes far away from the sink are likely to have a lower end-to-end success rate than nodes that are close. The breakdown of success rate by hop or distance should show this behavior.

IV **Stability** measures the total number of route changes in the network over a period of time, which indicates the stability of the routing topology. A stable topology should make higher-level operations, such as scheduling or aggregation, easier to design and implement.

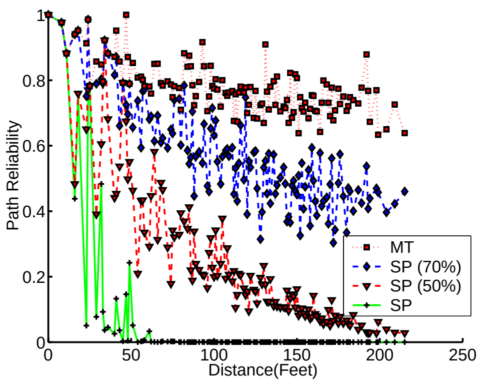
6.3 Network Graph Analysis

The first method of evaluation is graph analysis. Given a static, connectivity graph with probabilistic link qualities of all edges derived from inter-node distance, we compute optimal trees for each routing algorithm based on different cost metrics. We only consider SP, SP(t), and MT since only the cost metrics, rather than the means of determining them, matter at this level of analysis. Without packet level dynamics, only hop distribution and path reliability are meaningful. Nonetheless, this high level analysis enables us to explore large scale networks; given perfect information, it also establishes optimistic bounds on routing costs.

We analyze a network of 400 nodes, organized as a 20x20 grid with 8 feet spacing. The sink node is placed at the corner to maximize network depth. Connectivity information is derived from data shown in Figure 1(a). Without the



(a) Hop Distribution



(b) Path Reliability

Figure 7: Graph analysis results for a 400 node network with 8 feet grid size.

$\pm 10\%$ estimation error due to perfect information, we study $SP(50\%)$ instead of 40%.

Figure 7(a) shows the hop-count distribution for four protocols. SP yields a very shallow network, while the rest yield deeper networks with wider hop distribution. With most nodes being 2 and 3 hops away from the root in a network of 160-foot extent, many of the links must cover 40 to 50 feet. This suggests they are at the border of the transitional and clear region of Figure 1(a) and have very low quality.

Figure 7(b) shows the path reliability for these protocols. Indeed, reliability for SP drops below 5% for nodes of distance greater than 50 feet. Protocols that utilize link quality estimates yield much higher path reliability by taking more, higher quality hops. SP(70%) only considers links in the effective region. The lowest path reliability for two and three hop in SP(70%) is $(0.7)^2 = 49\%$ and $(0.7)^3 = 34.3\%$. SP(50%) takes advantage of links in the transitional region for fewer, longer hops, but reliability is hindered as a result. MT takes reliability into account and performs best without the need to set a threshold. This higher path reliability comes with a tradeoff of a slightly higher hop count for MT, as shown in Figure 7(a).

6.4 Packet Level Simulations

We turn to packet level simulations to further understand the dynamic behavior of the routing protocols and their interactions with link estimation and neighbor table management. We built a custom discrete time, event-driven network simulator in MATLAB and implemented the current network stack found in the TinyOS sensor node operating system. The low level details of retransmissions and media access are captured by the simulator, while the radio connectivity model is based on Figure 1(a).

To capture the effect of collisions, we performed an empirical study similar to that in Section 2 using three nodes at a time (a sender, a receiver, and a collider that also transmits) at various distances. In general, noticeable interference is observed only if a collider is within the transitional region of the receiver. Almost no reception is possible if the collider is within the effective region of the receiver. We use the following model to approximate this behavior in simulation. Assume $p_{i,j}$ is the probability of successful reception for node i to receive j's message. Let node b be the receiver and node a be the sender. The probability for b to receive a's message given there are k colliders equals $p_{b,a} * \prod_{i \in k} 1 - p_{b,i}$.

We use this simulator to analyze a 100-node network, placed as a 10x10 grid with 8 foot spacing, with the sink node located at the lower left corner of the grid. The simulation time for each experiment is 2000 seconds. Each node offers a load of periodic traffic at 10s/data packet and 20s/route packet. For protocols that utilize link estimations, WMEWMA is used with stable settings. We simulated all the protocols, except SP, since graph analysis has shown its poor performance confirming our experience in practice. For MT, we additionally consider the effect of using the FREQUENCY algorithm to manage a neighbor table of only twenty entries. We call this case MTTM. Other cases use a table large enough to hold all neighbors.

Figure 8 shows the resulting hop distributions. These agree with graph analysis fairly well considering that this network has half the physical extent in each dimension of that used in graph analysis. SP(40%), Broadcast, and DSDV all have a tight distribution, but wider than SP in graph analysis. SP(70%) and MT yield wider spreads in hop distribution and generally take more hops. For DSDV, about 15% of the nodes have no routes or infinite hops at the end of the simulation; these nodes have become disjoint from the network as a result of link failures or unreachable routes. Without link quality information, long, unreliable links are likely to be selected for routing and these are likely to experience link failures, causing nodes and their children to become disjoint from the network.

Figure 9 shows the average actual path reliability obtained by accumulating the link qualities of each packet that moves through the network. The top graph includes the protocols that utilize only high quality links in route formation. These yield relatively high path reliability even at 100 feet. The differences are much smaller than those under graph analysis. This appears because variations in link estimation eliminate some of the long, unstable links in SP(70%).

Although SP(40%) exploits link estimates in determining the next hop, a higher tolerance of lossy link yields poor performance similar to DSDV and Broadcast, as indicated in the lower graph. Protocols having similar hop distributions yield similar path reliability over distance; a higher majority hop count yields higher path reliability over distance.

Figure 10 shows the stability over time of the routing structures due to stochastic variations in packet loss and the associated estimation error. Broadcast and DSDV are highly unstable. Broadcast is unstable because its parent selection mechanism is opportunistic. DSDV suffers because poor links trigger link failure detection, which cause nodes to join and disjoin from a tree. The other protocols yield stable routing trees. MTTM is the most stable because the number of alternative parents in the neighbor table is reduced, while still presenting some good parents.

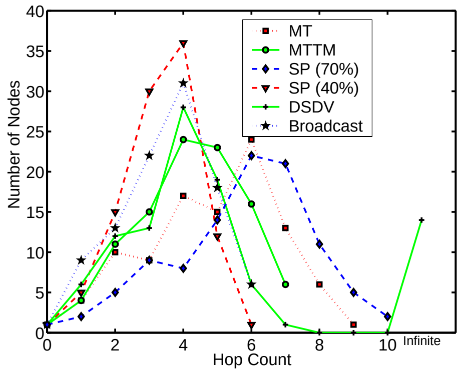


Figure 8: Hop distribution from simulations.

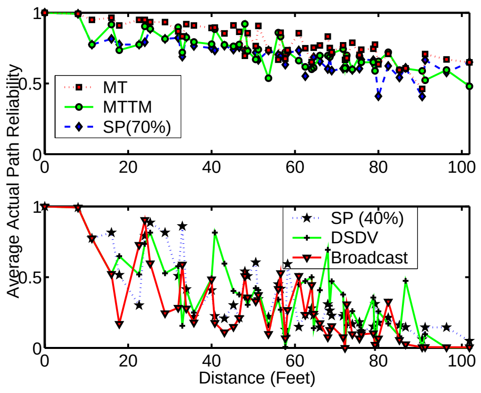


Figure 9: Path reliability over distance from simulations.

Figure 11 shows that given a maximum of two link layer retransmissions, end-to-end success rate is close to 90% for protocols that utilize high quality links. SP(40%) suffers non-negligible packet loss. DSDV suffers from nodes joining and disjoining from the network, while Broadcast performs very poorly even with retransmissions.

In all of the simulation runs, no cycles occurred. Furthermore, MTTM yields no significant difference in overall performance; it maintains an adequate number of good choices for route formation to succeed.

6.5 Empirical Study of a Sensor Field

Our simulation results allow us to further narrow down our evaluation space and focus on SP(40%), SP(70%), and MT in realistic settings. We implement these protocols and the WMEWMA link estimator on the TinyOS platform. Our first realistic test-bed is a 50-node network placed as a 5x10 grid with 8 foot spacing in the the foyer of the Hearst Mining building on the UC Berkeley campus. The nodes are placed on cups 3 inches above the ground, since ground reflection can significantly hinder the range of these radios. The sink node is placed in the middle of the short edge of the grid to avoid the potential interference from the metal building supports at the corners of the grid. It is attached to a laptop computer over a serial port interface for data collec-

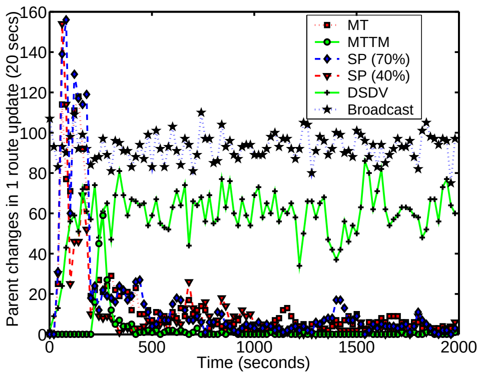


Figure 10: Stability from simulations.

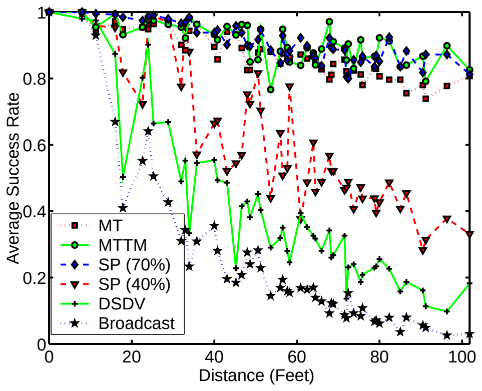


Figure 11: End-to-End success rate over distance from simulations.

tion. A typical run lasts about three hours and is performed at night when pedestrian traffic is low.

We found that to set the radio transmission power levels appropriately and to understand behavior of the protocols, we had to repeat the connectivity vs. distance study of Figure 1(a) in this indoor setting. We deployed a 10-node line topology network diagonally across the foyer with 8 foot inter-node spacing. To have several hops while preserving good neighbor connectivity, we wanted to find the lowest power setting such that the effective region would cover the grid spacing. Figure 12 shows the reliability scatter plot for a low transmit power setting. The fall off is more complex, presumably due to various multipath effects, even though the space is quite open. At 8 feet, most of the links are above 90%. It is apparent that a significant number of reliable, long links exists, with a few of them covering more than half of the network extent.

We performed the data collection experiments with this power setting for SP(70%), SP(40%), and MT. The maximum number of link retransmissions is two. The link estimator setting for WMEWMA is ($t = 30, \alpha = 0.5$). We used a neighbor table size of 30 in all our 50-node experiments. The traffic load is 30s/data packet and 60s/route packet per node. To expedite the warm up phase of the estimator, the route update rate is 10s/route packet for the first 10 minutes. We also explore effect of tripling the data rate and route update rate on MT, without any rate control, to deliberately drive the network into congestion.

Figure 13(a) shows the hop distribution for SP(40%) and

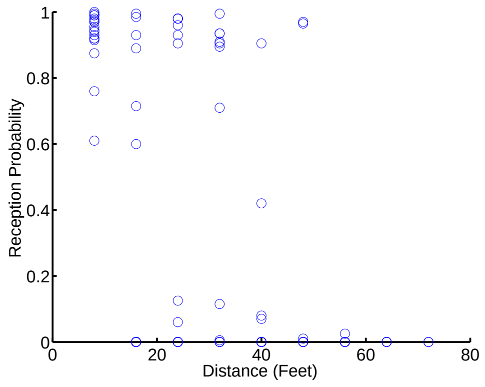


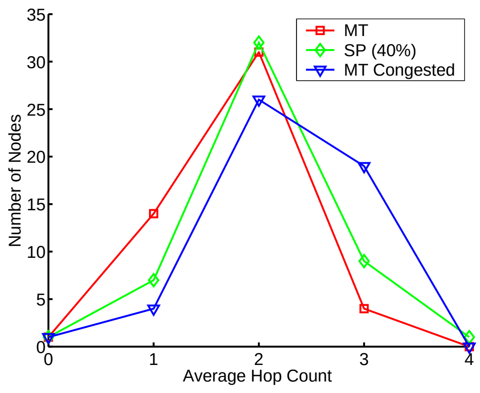
Figure 12: Indoor reception probability of all links of a network in a line topology at low transmit power setting (70) in the foyer.

MT. SP(70%) is not shown because it failed to construct a viable routing tree. The distributions for SP(40%) and MT are quite similar and both surprisingly shallow, given that the transmission strength was set to just cover the grid spacing. Also MT is the shallower of the two, unlike in simulation. To see why this occurs, a contour plot of hop-count over the grid is shown in Figure 13(b). The sink node is located at (1,3). Three nodes in column 9 are at one hop, even though nodes in column 6 are at three hops. These are long, stable links with good connectivity. Similarly, nodes in column 4 are at 1 hop, while nodes in column 3 are at 2 hops. This contour plot represents an aggregation of evolving routing tree over the run of the experiment, so the nodes in column 9 are usually at the first level of the tree and the nodes at (3,6) and (3,7) are generally deep in the tree, but their parents may be neighbors in any direction.

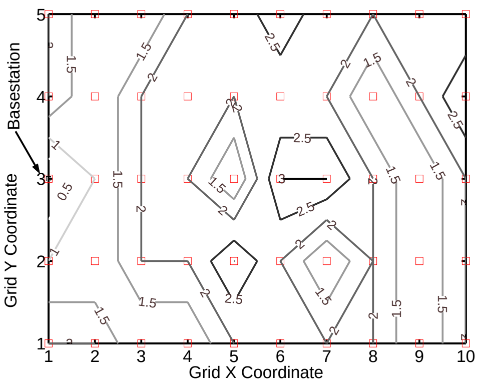
For the congested case, we see a reduction in reliability of links in the upstream direction, causing more nodes to take more hops. The curve for MT Congested shifts to the right.

To see why SP(70%) fails to form a routing tree, even though Figure 12 suggests that average link quality to neighboring nodes should be around 90%, we had each node include link estimates to and from the parents under MT in its data packets. These average estimations are shown in Figure 14. (This information is not available for SP(70%) because few nodes could deliver information.) Similar data is also observed for SP(40%). They vary right around 70%, for all nodes whose next hop is not the sink node. (For nodes connected to the sink, the upward link is much less reliable than the downward link.) When interference from other traffic is present, the average link quality decreases. The threshold in SP(70%) is no longer sufficient to maintain a connected subgraph.

SP(40%) experiences a similar thresholding problem at high data rates that cause the network to become congested. We observed that congestion prunes the tree built by SP(40%) due to the link quality dropping below the threshold. Packets are not forwarded, reducing the contention; SP(40%) rebuilds the network as congestion goes away and this cycle repeats. MT avoids these problems by picking the best available paths, without an arbitrary threshold. Notice also that by tracking the link quality, routing protocols can successfully avoid routing over asymmetrical links. The rest of the study only focuses on SP(40%), MT, and MT Congested.



(a) Hop Distribution



(b) Average Hop over Distance Contour Plot for MT at power 70

Figure 13: Hop count analysis for indoor 50-node field.

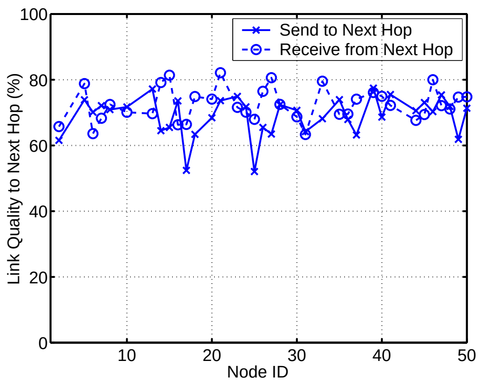


Figure 14: Non-sink node next hop link quality for MT in the foyer.

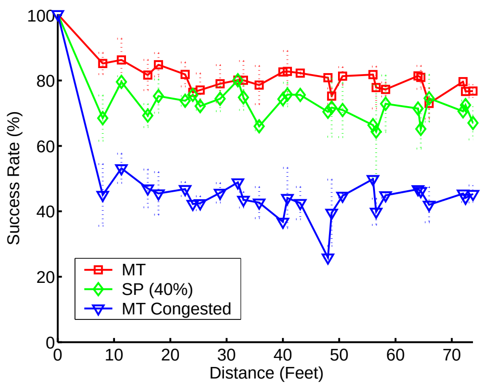


Figure 15: End-to-end success rate over distance in the foyer.

Figure 15 shows the end-to-end success rate versus distance of MT and SP(40%). MT delivers roughly 80% of the originated data consistently throughout the sensor field. This indicates that each of the underlying components of the protocol, including link estimation, parent management and queue management are working together effectively. The success rate of SP(40%) is lower, but is still much more robust than the simulations would suggest. Even though this protocol considers links that are estimated at 40%, it appears that many of the links it chooses are in fact of much higher quality.

To further test the robustness of MT, we examine its behavior under a high enough load to cause substantial congestion in the network. At 3 times the data origination and route update rate, the success rate drops to roughly 50%. The network is delivering 1.4 times the absolute data rate to the sink. Even under congestion, the success rate is only slightly impacted by distance and network depth.

The average number of link retransmissions per packet delivered to the base station is about 1 along the entire path for MT, SP(40%), and MT Congested. With the average next hop link quality of 70%, we would expect a higher data success rate. To probe this issue further we extracted the link quality for nodes sending to the base station throughout the run. The quality in sending to the base station is only 50% and drops to 40% for the congested case. This can be seen in Figure 14; nodes 16 and 25 are depth 1 most of the time and exhibit a low parent link quality. We have previously observed on this platform that traffic over the serial port reduces quality of RF reception at the base station.

To see the effect on nodes deeper in the network, we examine a three-hop node and look at the difference between its estimated number of transmissions to reach the base station and the number of transmissions that actually occurred for packets that arrive at the base station. With the estimated number of transmissions being six, one retransmission is expected on average on each hop of a three-hop path. However, the packets that reached the base station only experience one retransmission along the entire path. This suggests that two retransmissions per hop are ineffective in moving the packet along the path. With a maximum of three retransmissions per hop, the end-to-end success rate is greater than 90%.

Our data also shows that the cost of the path, which is composed of three different link estimations over three hops, is very stable. The fluctuations are ± 1, which suggests they are changes in hop count. This data supports that our WMEWMA estimator performs well on real networks.

Figure 16 shows the stability of routing tree with MT and MT Congested. For MT after an early formation stage, the network is fairly stable. However, we see a substantial change about every 1,000 seconds. The stability of SP(40%), not shown, is similar to MT. MT Congested exhibits much greater instability. It does operate at three times the data rate and route update rate, so the time-scale is effectively compressed, but the underlying cause of the volatility is the fluctuations of link quality due to contention. We return

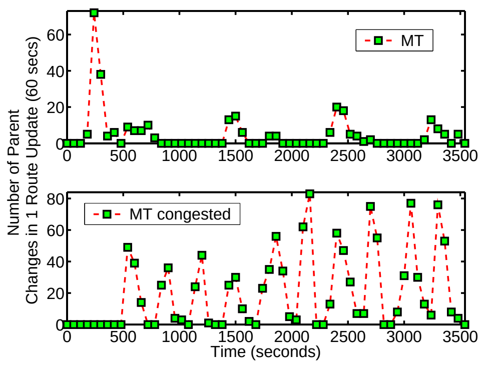


Figure 16: Stability of the entire network in the foyer.

to this issue below in analyzing more detailed results from another test-bed.

In all of our experiments, no cycles were detected, suggesting that simple cycle avoidance mechanisms are sufficient for relatively immobile networks. The policy on multiplexing between originating traffic and forwarding traffic appears to be a smaller factor in these experiments, as the forwarding queues in all the nodes are almost empty. Furthermore, even at this low power setting, the number of potential *neighbors* is quite large. For example, the base station ended up recording twenty six neighbors, which is half the network. This reinforces the need for neighborhood management.

6.6 Irregular Indoor Network

We repeated a similar set of experiments with 30 nodes scattered around an indoor office space of 1,000 ft^2. We did not perform any *a priori* analysis of connectivity and distance relationship in this environment. We simply placed nodes on handy spots and set the transmit power to maximum. Although this is a smaller scale network, the office test-bed provides a back channel that allows us to periodically archive information within each node.

In this setting, SP(70%) also failed to form a routing tree. Figure 17 shows the end-to-end success rate of the algorithms. MT can achieve 90% success rate over six hops, with an average of 1 retransmission. SP(40%) performed the same, with an average of 1.3 retransmissions. The actual results are also not too different between the protocols. MT Congested has a sharp drop on end-to-end success rate and has almost 0% from a 6 hop node.

To understand the dynamics due to congestion, we examine how link estimations over the same pair of nodes behave differently under different channel utilization in Figure 18. The estimation under congested traffic is unstable due to interference from collisions and hidden node problems. Periods of contention lower the estimation quite significantly. This creates global instability on the routing tree as it attempts to adapt; such behavior is captured in Figure 19. This is consistent with the congested behavior that we observed in the open foyer experiments.

7. CONCLUSIONS

This study has shown that link quality estimation and neighborhood management are essential to and tightly coupled with reliable routing in sensor networks. WMEWMA is a simple, memory efficient link estimator that reacts quickly,

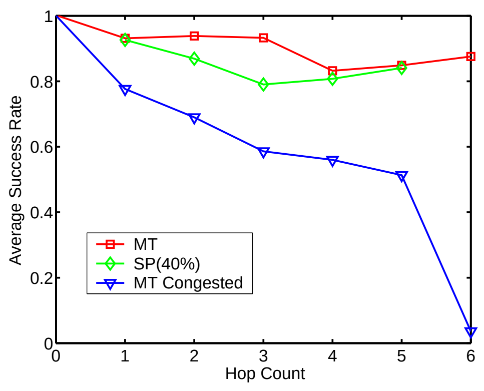


Figure 17: End-to-end success rate versus hop in an office environment.

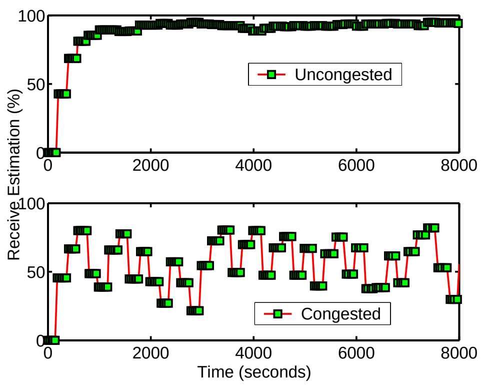


Figure 18: Link estimation of a node to its neighbor over time in an office environment.

yet is stable enough for path characterization in cost-based routing. The FREQUENCY algorithm performs well in maintaining a subset of good neighbors in a constrained neighbor table regardless of cell density. Minimum expected transmissions is an effective metric for cost-based routing, it does not require a predefined link quality threshold and is robust under varying connectivity characteristics. The combination of these techniques has been shown empirically to yield high end-to-end success rate in sizable networks on a resource-constrained platform. This study focuses on a data collection workload using the Mica platform, but the techniques and the methodology are relevant to many other communication patterns and to other platforms.

Our empirical study crosses several system layers and illustrates interactions among global network structure, high-level protocols, and the underlying low-level issues. The empirical data should shed light on application deployment by illustrating the tradeoffs among transmit power, inter-nodal spacing for data sampling, average and maximum network hop count, and overall network load. We deliberately put the network into congestion in our study, but effective methods, such as [22], exists to alleviate such issues in practice. Although new generations of radios will have different connectivity characteristics from the one that we sampled, the observed three-region structure is expected to persist; link estimation, neighborhood table management, and reliability-based cost metrics are likely to remain as core underlying issues for reliable routing in sensor networks.

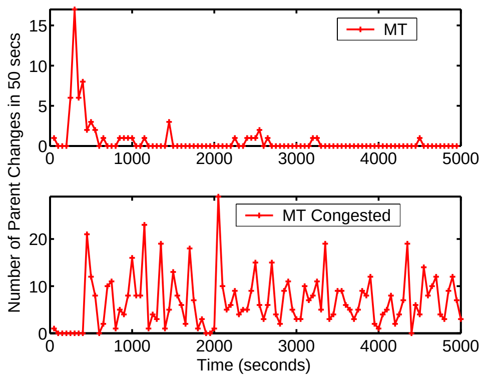


Figure 19: Stability for MT in an office environment.

Acknowledgments

This work was supported, in part, by the Defense Department Advanced Research Projects Agency (grants F33615-01-C-1895 and N6601-99-28913), the National Science Foundation under Grant No. 0122599, California MICRO program, and Intel Corporation. Research infrastructure was provided by the National Science Foundation (grant EIA-9802069).

8. REFERENCES

[1] B. Albrightson, J. Garcia-Luna-Aceves, and J. Boyle. EIGRP - a fast routing protocol based on distance vectors. In *Proceedings of Networld/Interop*, May 1994.

[2] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *International Conference on Mobile Computing and Networking (MobiCom 2001)*, pages 85–96, Rome, Italy, July 2001.

[3] Y. Choi, M. G. Gouda, M. C. Kim, and A. Arora. The mote connectivity protocol. Technical Report TR-03-08, Department of Computer Sciences, The University of Texas at Austin, 2003.

[4] D. D. Couto, D. Aguayo, B. Chambers, and R. Morris. Performance of multihop wireless. First Workshop on Hot Topics in Networks (HotNets-I), October 2002.

[5] M. Datar, A. Gionis, P. Indyk, and R. Motwani. Maintaining stream statistics over sliding windows. In *13th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2002.

[6] E. D. Demaine, A. Lopez-Ortiz, and J.I.Munro. Frequency estimation of internet packet streams with limited space. In *Proceedings of the 10th Annual European Symposium on Algorithms ESA 2002*, pages 348–360, September 2002.

[7] C. Estan and G.Varghese. New directions in traffic measurement and accounting. In *ACM SIGCOMM Internet Measurement Workshop*, 2001.

[8] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. Complex behavior at scale: An experimental study of low-power wireless sensor networks. In *Technical Report UCLA/CSD-TR 02-0013*, February 2002.

[9] P. G. Gibbons and Y.Matias. New sampling- based summary statistics for improving approximate query answers. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 311–342, June 1998.

[10] G.Manku and R.Motwani. Approximate frequency counts over data streams. In *Proceedings of the 28th International Conference on Very Large Data Bases*, August 2002.

[11] C. Hedrick. Routing information protocol. In *RFC 1058*, June 1988.

[12] C. Hedrick. An introduction to IGRP. In *Rutgers - The State University of New Jersey Technical Publication, Laboratory for Computer Science*, August 1991.

[13] J. Hill and D. Culler. Mica: a wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, nov/dec 2002.

[14] J. Hill and D. Culler. A wireless-embedded architecture for system level optimization. In *UC Berkeley Technical Report*, 2002.

[15] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *the Ninth international Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000)*, pages 93–104, November 2000.

[16] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the International Conference on Mobile Computing and Networking Mobicom*, pages 56–67, August 2000.

[17] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[18] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *Proceedings of the ACM SIGCOMM*, pages 234–244, August 1994.

[19] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance-vector (aodv) routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[20] R. Ramanathan and R. Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *IEEE Infocom*, March 2000.

[21] E. Shih, P. Bahl, and M. J. Sinclair. Wake on wireless:: an event driven energy saving strategy for battery operated devices. In *Proceedings of the eighth annual international conference on Mobile computing and networking*, pages 160–171. ACM Press, 2002.

[22] A. Woo and D. Culler. A transmission control scheme for media access in sensor networks. In *International Conference on Mobile Computing and Networking (MobiCom 2001)*, page 221, Rome, Italy, July 2001.

[23] A. Woo and D. Culler. Evaluation of efficient link reliability estimators for low-power wireless networks. Technical Report UCB//CSD-03-1270, U.C. Berkeley Computer Science Division, September 2003.

[24] M. D. Yarvis, W. S. Conner, L. Krishnamurthy, A. Mainwaring, J. Chhabra, and B. Elliott. Real-world experiences with an interactive ad hoc sensor network. In *International Conference on Parallel Processing Workshops*, August 2002.