Link Estimation and Routing in Sensor Network Backbones: Beacon-based or Data-driven?

Hongwei Zhang, *Member, IEEE*, Anish Arora, *Senior Member, IEEE*, and Prasun Sinha, *Member, IEEE*

***Abstract*— In the context of IEEE 802.11b network testbeds, we examine the differences between unicast and broadcast link properties, and we show the inherent difficulties in precisely estimating unicast link properties via those of broadcast beacons even if we make the length and transmission rate of beacons be the same as those of data packets. To circumvent the difficulties in link estimation, we propose to estimate unicast link properties directly via data traffic itself without using periodic beacons. To this end, we design a data-driven routing protocol *Learn on the Fly* (LOF). LOF chooses routes based on ETX/ETT-type metrics, but the metrics are estimated via MAC feedback for unicast data transmission instead of broadcast beacons. Using a realistic sensor network traffic trace and an 802.11b testbed of ∼195 Stargates, we experimentally compare the performance of LOF with that of beacon-based protocols, represented by the geography-unaware ETX and the geography-based PRD. We find that LOF reduces end-to-end MAC latency by a factor of 3, enhances energy efficiency by a factor up to 2.37, and improves network throughput by a factor up to 7.78, which demonstrate the feasibility and the potential benefits of data-driven link estimation and routing.**

***Index Terms*— wireless sensor network, data-driven link estimation, MAC latency, IEEE 802.11b, real time, energy, reliability**

I. Introduction

Wireless sensor networks are envisioned to be of large scale, comprising thousands to millions of nodes. To guarantee real-time and reliable end-to-end packet delivery in such networks, they usually require a high-bandwidth network backbone to process and relay data generated by the low-end sensor nodes such as motes [3]. This architecture has been demonstrated in the sensor network field experiment ExScal [6], where 203 Stargates and 985 XSM motes were deployed in an area of 1260 meters by 288 meters. Each Stargate is equipped with a 802.11b radio, and the 203 Stargates form the backbone network of ExScal to support reliable and real-time communication among the motes for target detection, classification, and tracking. Similar 802.11 based sensor networks (or network backbones) have also been explored in other projects such as MASE [1] and CodeBlue [2]. In this paper, we study how to perform routing in such 802.11 based wireless sensor network backbones.

As the quality of wireless links, for instance, packet delivery rate, varies both temporally and spatially in a complex manner [7], [27], [39], estimating link quality is an important aspect of routing in wireless networks. To this end, a commonly used approach is letting peers exchange broadcast beacons periodically, and the measured quality of broadcast acts as the basis of link estimation [13], [14], [15], [33], [35]. Nonetheless, beacon-based link estimation has several drawbacks:

- Firstly, link quality for broadcast beacons differs significantly from that for unicast data, because broadcast beacons and unicast data differ in packet size, transmission rate, and coordination method at the media-access-control (MAC) layer [12], [31]. Therefore, we have to estimate unicast link quality based on that of broadcast.
- It is, however, difficult to precisely estimate unicast link quality via that of broadcast, because temporal correlations of link quality assume complex patterns [34] and are hard to model. As a result, existing routing protocols do not consider temporal link properties in beacon-based estimation [13], [35]. Thus the link quality estimated using periodic beacon exchange may not accurately apply for unicast data, which can negatively impact the performance of routing protocols.
- Even if we could precisely estimate unicast link quality based on that of broadcast, beacon-based link estimation may not reflect in-situ network condition either. For instance, a typical application of wireless sensor networks is to monitor an environment (be it an agricultural field or a classified area) for events of interest to the users. Usually, the events are rare. Yet when an event occurs, a large burst of data packets is often generated that needs to be routed reliably and in real-time to a base station [37]. In this context, even if there were no discrepancy between the actual and the estimated link quality using periodic beacon exchange, the estimates still tend to reflect link quality in the absence, rather than in the presence, of bursty data traffic. This is because: Firstly, link quality changes significantly when traffic pattern changes (as we will show in Section II-B.2); Secondly, link quality estimation takes time to converge, yet different bursts of data traffic are well separated in time, and each burst lasts only for a short period.

Beacon-based link estimation is not only limited in reflecting the actual network condition, it is also inefficient in energy usage. In existing routing protocols that use link quality estimation, beacons are exchanged periodically. Therefore, energy is consumed unnecessarily for the periodic beaconing when there is no data traffic. This is especially true if the events of interest are infrequent enough that there is no data traffic in the network most of the time [37].

To deal with the shortcomings of beacon-based link quality estimation and to avoid unnecessary beaconing, new mechanisms for link estimation and routing are desired.

Contributions of the paper. Using outdoor and indoor testbeds of 802.11b networks, we study the impact of environment, packet type, packet size, and interference pattern on the quality of

An extended abstract containing some preliminary results of this paper has appeared in IEEE INFOCOM 2006.

Hongwei Zhang is with the Department of Computer Science, Wayne State University, Detroit, Michigan 48202, U.S.A. E-mail: hzhang@cs.wayne.edu.

Anish Arora and Prasun Sinha are with the Department of Computer Science and Engineering, The Ohio State University, Columbus, OH 43210, U.S.A. E-mail: {anish, prasun}@cse.ohio-state.edu.

wireless links, and we characterize the inherent drawbacks of beacon-based link estimation. Our study shows that it is difficult (if even possible) to precisely estimate unicast link quality using broadcast beacons even if we make the length and transmission rate of beacons be the same as those of data packets. To address the drawbacks of beacon-based estimation, we propose to estimate link properties via MAC feedback for unicast data transmissions themselves. To this end, we define data-driven routing metrics that, using MAC latency for data transmissions, are similar to ETX/ETT but are estimated via unicast MAC feedback instead of broadcast beacons.

To implement data-driven link estimation and routing, we modify the Linux kernel and the WLAN driver *hostap* [5] to exfiltrate the MAC latency for each packet transmission, which is not available in existing systems. The exfiltration of MAC latency is reliable in the sense that it deals with the loss of MAC feedback at places such as *netlink* sockets and IP transmission control.

Building upon the capability of reliably fetching MAC latency for each packet transmission, we design a routing protocol *Learn on the Fly* (LOF) where ETX/ETT-type metrics are estimated without using periodic beacons. In LOF, control packets are used only rarely, for instance, during node boot-up. Upon booting up, a node initializes its routing engine by taking a few (e.g., 6) samples on the MAC latency to each of its neighbors; then the node adapts its routing decision solely based on the MAC feedback for data transmission, without using any control packet. To deal with temporal variations in link quality and possible imperfection in initializing its routing engine, the node probabilistically explores alternative neighbors at controlled frequencies.

Using an event traffic trace from the field sensor network of ExScal [6], we experimentally evaluate the design and the performance of LOF in a testbed of ∼195 Stargates [3] with 802.11b radios. We also compare the performance of LOF with that of beacon-based protocols, represented by the geography-unaware ETX [13], [35] and the geography-based PRD [33]. We find out that LOF reduces end-to-end MAC latency, reduces energy consumption in packet delivery, and improves route stability. Besides bursty event traffic, we evaluate LOF in the case of periodic traffic, and we find that LOF outperforms existing protocols in that case too. We also find that LOF significantly improves network throughput. The results corroborate the necessity, the feasibility, and the benefits of data-driven link estimation and routing.

Organization of the paper. In Section II, we study the shortcomings of beacon-based link quality estimation, and we analyze the feasibility of data-driven routing. Following that, we present a data-driven version of ETX/ETT-type routing metrics in Section III, and we design the protocol LOF in Section IV. We experimentally evaluate LOF in Section V, and we discuss the related work in Section VI. We make concluding remarks in Section VII.

II. Why data-driven link estimation and routing?

In this section, we first experimentally study the impact of packet type, packet length, and interference on link properties[1]. Then we discuss the shortcomings of beacon-based link property estimation and the concept of data-driven link estimation and routing.

[1]In this paper, the phrases *link quality* and *link property* are used interchangeably.

A. Experiment design

We set up two 802.11b network testbeds as follows.

Outdoor testbed. In an open field (see Figure 1), we deploy 29 Stargates in a straight line, with a 45-meter separation between any two consecutive Stargates. The Stargates run Linux with kernel 2.4.19. Each Stargate is equipped with a SMC 2.4GHz 802.11b wireless card and a 9dBi high-gain collinear omnidirectional antenna, which is raised 1.5 meters above the ground. To control the maximum communication range, the transmission power level of each Stargate is set as 35. (Transmission power level is a tunable parameter for 802.11b wireless cards, and its range is 127, 126, . . . , 0, 255, 254, . . . , 129, 128, with 127 being the lowest and 128 being the highest.)

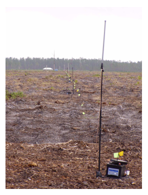
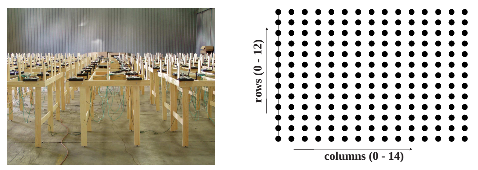


Figure 1. Outdoor testbed

Sensornet testbed Kansei. In an open warehouse with flat aluminum walls (see Figure 2(a)), we deploy 195 Stargates in a 15 $\times$ 13 grid (as shown in Figure 2(b)) where the separation between neighboring grid points is 0.91 meter (i.e., 3 feet). The



(a) Kansei (b) grid topology

Figure 2. Sensornet testbed Kansei

deployment is a part of the sensornet testbed Kansei [8]. For convenience, we number the rows of the grid as 0 - 12 from the bottom up, and the columns as 0 - 14 from the left to the right. Each Stargate is equipped with the same SMC wireless card as in the outdoor testbed. To create realistic multi-hop wireless networks similar to the outdoor testbed, each Stargate is equipped a 2.2dBi rubber duck omnidirectional antenna and a 20dB attenuator. We raise the Stargates 1.01 meters above the ground by putting them on wood racks. The transmission power level of each Stargate is set as 60, to simulate the low-to-medium density multi-hop networks where a node can reliably communicate with around 15 neighbors.

The Stargates in the indoor testbed are equipped with wall-power and outband Ethernet connections, which facilitate long-duration complex experiments at low cost. We use the indoor testbed for most of the experiments in this paper; we use the outdoor testbed mainly for justifying the generality of the phenomena observed in the indoor testbed.

Experiments. In the *outdoor testbed*, the Stargate at one end acts as the sender, and the other Stargates act as receivers. Given the constraints of time and experiment control, we leave complex experiments to the indoor testbed and only perform relatively simple experiments in the outdoor testbed: the sender first sends 30,000 1200-byte broadcast packets, then it sends 30,000 1200-byte unicast packets to each of the receivers.

In the *indoor testbed*, we let the Stargate at column 0 of row 6 be the sender, and the other Stargates in row 6 act as receivers.

To study the impact of interference, we consider the following scenarios (which are named according to the interference):

- *Interferer-free*: there is no interfering transmission. The sender first sends 30,000 broadcast packets each of 1200 bytes, then it sends 30,000 1200-byte unicast packets to each of the receivers, and lastly it broadcasts 30,000 30-byte packets.
- *Interferer-close*: one "interfering" Stargate at column 0 of row 5 keeps sending 1200-byte unicast packets to the Stargate at column 0 of row 7, serving as the source of the interfering traffic. The sender first sends 30,000 1200-byte broadcast packets, then it sends 30,000 1200-byte unicast packets to each of the receivers.
- *Interferer-middle*: the Stargate at column 7 of row 5 keeps sending 1200-byte unicast packets to the Stargate at column 7 of row 7. The sender performs the same as in the case of interferer-close.
- *Interferer-far*: the Stargate at column 14 of row 5 keeps sending 1200-byte unicast packets to the Stargate at column 14 of row 7. The sender performs the same as in the case of interferer-close.
- *Interferer-exscal*: In generating the interfering traffic, every Stargate runs the routing protocol LOF (as detailed in later sections of this paper), and the Stargate at the upper-right corner keeps sending packets to the Stargate at the left-bottom corner, according to an event traffic trace from the field sensor network of ExScal [6] . The traffic trace corresponds to the packets generated by a Stargate when a vehicle passes across the corresponding section of ExScal network. In the trace, 19 packets are generated, with the first 9 packets corresponding to the start of the event detection and the last 10 packets corresponding to the end of the event detection. Figure 3 shows, in sequence, the intervals between packets 1 and 2, 2 and 3, and so on. The sender performs the same as in the case of interferer-close.

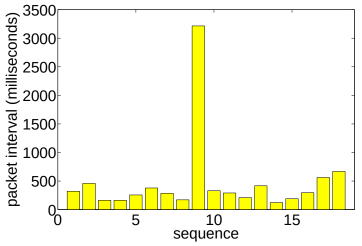


Figure 3. The traffic trace of an ExScal event

In all of these experiments, except for the case of interferer-exscal, the packet generation frequency, for both the sender and the interferer, is 1 packet every 20 milliseconds. In the case of interferer-exscal, the sender still generates 1 packet every 20 milliseconds, yet the interferer generates packets according to the event traffic trace from ExScal, with the inter-event-run interval being 10 seconds. (Note that the scenarios above are far from being complete, but they do give us a sense of how different interfering patterns affect link properties.)

In the experiments, broadcast packets are transmitted at the basic rate of 1M bps, as specified by the 802.11b standard. Not focusing on the impact of packet rate in our study, we set unicast transmission rate to a fixed value (e.g., 5.5M bps). (We have tested different unicast transmission rates and observed similar phenomena.) For other 802.11b parameters, we use the default configuration that comes with the system software. For instance, unicast transmissions use RTS-CTS handshake, and each unicast packet is retransmitted up to 7 times until success or failure in the end.

B. Experimental results

For each case, we measure various link properties, such as packet delivery rate and the run length of packets successfully received without any loss in between, for each link defined by the sender - receiver. Due to space limitation, however, we only present the data on packet delivery rate here. The packet delivery rate is calculated once every 100 packets (we have also calculated delivery rates in other granularities, such as once every 20, 50 or 1000 packets, and similar phenomena were observed).

We first present the difference between broadcast and unicast when there is no interference, then we present the impact of interference on network conditions as well as the difference between broadcast and unicast.

1) Interferer free: Figure 4 shows the scatter plot of the delivery rates for broadcast and unicast packets at different distances in the outdoor testbed. From the figure, we observe the following:

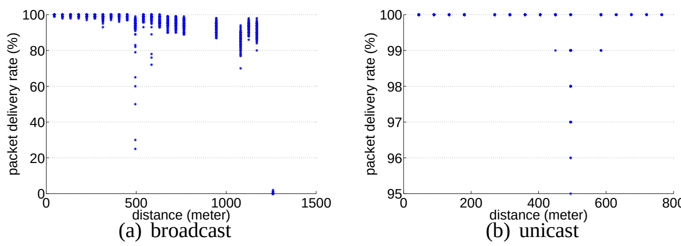
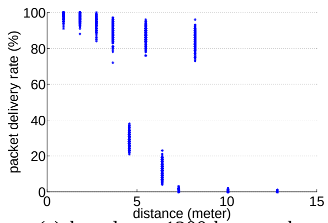


Figure 4. Outdoor testbed

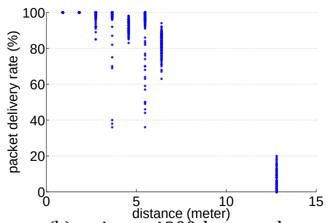
- Broadcast has longer communication range than unicast. This is due to the fact that the transmission rate for broadcast is lower, and that there is no RTS-CTS handshake for broadcast. (Note: the failure in RTS-CTS handshake also causes a unicast to fail.)
- For links where unicast has non-zero delivery rate, the mean delivery rate of unicast is higher than that of broadcast. This is due to the fact that each unicast packet is retransmitted up to 7 times upon failure.
- The variance in packet delivery rate is lower in unicast than that in broadcast. This is due to the fact that unicast packets are retransmitted upon failure, and the fact that there is RTS-CTS handshake for unicast. (Note: the success in RTS-CTS handshake implies higher probability of a successful unicast, due to temporal correlations in link properties [10].)

Similar results are observed in the indoor testbed, as shown in Figures 5(a) and 5(b). Nevertheless, there are exceptions at distances 3.64 meters and 5.46 meters, where the delivery rate of unicast takes a wider range than that of broadcast. This is likely due to temporal changes in the environment. Comparing Figures 5(a) and 5(c), we see that packet length also has significant impact on the mean and variance of packet delivery rate.

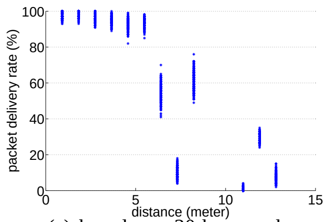
Implication. From Figures 4 and 5, we see that packet delivery rate differs significantly between broadcast and unicast, and the difference varies with environment, hardware, and packet length.



(a) broadcast: 1200-byte packet



(b) unicast: 1200-byte packet



(c) broadcast: 30-byte packet

Figure 5. Indoor testbed

2) Interference scenarios: To demonstrate how network condition changes with interference scenarios, Figure 6 shows the broadcast packet delivery rates in different interference scenarios. We see that broadcast packet delivery rate varies significantly (e.g., up to 39.26%) as interference patterns change. Thus, link properties estimated for one scenario may not apply to another.

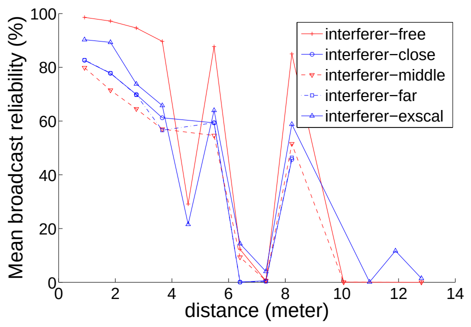


Figure 6. Network condition, measured in broadcast reliability, in different interference scenarios

Having shown the impact of interference patterns on network condition, Figure 7 shows how the difference between broadcast and unicast in the mean packet delivery rate changes as the interference and distance change. Given a distance and an interference scenario, the difference is calculated as $\frac{U-B}{B}$, where U and B denote the mean delivery rate for unicast and broadcast respectively. From the figure, we see that the difference is significant (up to 94.06%), and that the difference varies with distance. Moreover, the difference changes significantly (up to 103.41%) as interference pattern changes.

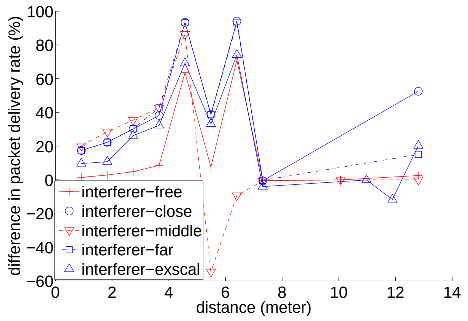


Figure 7. The difference between broadcast and unicast in different interference scenarios

Implication. For wireless sensor networks where data bursts are well separated in time and possibly in space (e.g., in bursty convergecast), the link properties experienced by periodic beacons may well differ from those experienced by data traffic. Moreover, the difference between broadcast and unicast changes as interference pattern changes.

C. Data-driven routing

To ameliorate the differences between broadcast and unicast link properties, researchers have proposed to make the length and transmission rate of broadcast beacons be the same as those of data packets, and then estimate link properties of unicast data via those of broadcast beacons by taking into account factors such as link asymmetry. ETX [13] has explored this approach. Nevertheless, this approach may not be always feasible when the length of data packets is changing; or even when the approach is feasible, it still does not guarantee that link properties experienced by periodic beacons reflect those in the presence of data traffic, especially in event-driven sensor network applications. Moreover, the existing method for estimating metrics such as ETX does not take into account the temporal correlations in link properties [10] (partly due to the difficulty of modeling the temporal correlations themselves [34]), which further decreases its estimation fidelity. For instance, Figure 8 shows the significant error[2] in estimating unicast delivery rate via that of broadcast under different interference scenarios when temporal correlations in link properties are not considered (i.e., assuming independent bit error and packet loss). Therefore, it is not trivial, if even possible, to precisely estimate link properties for unicast data via those of broadcast beacons.

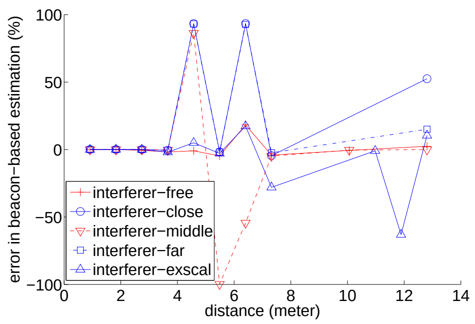


Figure 8. Error in estimating unicast delivery rate via that of broadcast

To circumvent the difficulty of estimating unicast link properties via those of broadcast, we propose to directly estimate unicast link properties via data traffic itself. More specifically, we propose to use MAC latency of unicast data transmissions as the basis of link estimation. In 802.11b MAC, every frame transmission is acknowledged by the receiver, thus the sender can determine if a

[2]The error is defined as actual unicast link reliability minus the estimated link reliability

transmission has succeeded by checking whether it receives the acknowledgment, and the sender can also determine how long each transmission takes, i.e., the MAC latency. Accordingly, nodes are able to get information on unicast MAC latency without using any beacons. In the mean time, Draves $et\ al.$ [15] have shown that MAC latency, as a routing metric, performs similar to the ETT metric (i.e., expected transmission time). Since ETT performs similar to the ETX metric (i.e., expected transmission count) [13] when radio transmission rate is fixed, MAC latency also performs similar to ETX. Therefore, MAC latency based metric not only enables data-driven link estimation but also represents the data-driven version of the commonly used routing metrics ETX and ETT which are estimated via broadcast beacons in many existing protocols [13], [14], [15], [33], [35]. (Note: we have also independently shown that routing metrics optimizing MAC latency optimize energy efficiency too [38].)

In what follows, we first present in Section III a data-driven version of the ETX/ETT metric using unicast MAC latency, then we present in Section IV the design of LOF which implements the metric without using periodic beacons.

Remarks. Although parameters such as Receiver Signal Strength Indicator (RSSI), Link Quality Indicator (LQI), and Signal to Noise Ratio (SNR) also reflect link reliability, it is difficult to use them as a precise prediction tool [7]. Moreover, the aforementioned parameters can be fetched only at packet receivers (instead of senders), and extra control packets are needed to convey these information back to the senders if we want to use them as the basis of link estimation; since control packets are usually exchanged at relatively low frequency (e.g., every 30 seconds), it is difficult to use this approach to precisely estimate in-situ link properties in real-time (such as in mission critical, event detection sensor network applications). Therefore, we do not recommend using these parameters as the core basis of data-driven routing.

The firmware of our SMC WLAN cards does not expose information on the number of retries of a unicast transmission, which would have been able to serve as the basis of data-driven link estimation too. As a part of our future work, we plan to examine this possibility (e.g., in IEEE 802.15.4 based mote networks) and study the corresponding protocol performance.

III. ELD: A DATA-DRIVEN VERSION OF ETX/ETT METRIC

In this section, we first define a data-driven routing metric ELD, the *expected MAC latency per unit-distance to destination*, then we analyze the sample size requirement in data-driven link estimation and routing.

A. A metric using unicast MAC latency and geography

To understand the benefits of data-driven link estimation, we can incorporate data-driven link estimation method with both geography-unaware, distance-vector routing protocol and geographic protocol. While it is not the objective of this paper to examine issues specific to geographic routing, geographic data-driven routing enables us to analyze, in a closed-form, the convergence speed of data-driven link estimation and routing as we show in Section III-B. Moreover, in sensor networks where nodes are mostly static and their precise geographic locations are readily available via devices such as GPS (as in ExScal [6]), geographic routing enables less frequent information diffusion as compared with distance-vector protocols[3], thus saving energy consumed in exchanging control packets. Therefore, we first present our study with a geographic, data-driven routing metric in most parts of this paper, and then we show in Section V-C that geographic data-driven routing performs similar to distance-vector data-driven routing in uniformly distributed networks.

Given that MAC latency performs, as a routing metric, similar to ETT (and ETX when radio transmission rate is fixed) [15], we define a greedy, geographic metric ELD, the *expected MAC latency per unit-distance to destination*, via which we hope to minimize the end-to-end MAC latency from source to destination. (Note that Lee $et\ al.$ [29] have shown the optimality of a metric similar to ELD in an idealized environment.) Specifically, given a sender S, a neighbor R of S, and the destination D as shown in Figure 9, we first calculate the *effective geographic progress* from S to D via R, denoted by $L_e(S,R)$, as $(L_{S,D} - L_{R,D})$, where $L_{S,D}$ denotes the distance between S and D, and $L_{R,D}$ denotes the distance between R and D. Then, we calculate, for the sender S, the *MAC latency per unit-distance to the destination* (LD) via R, denoted by $LD(S,R)$, as[4]

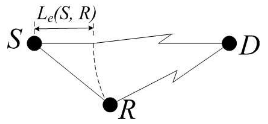


Figure 9. L_e calculation

$$\begin{cases} \frac{D_{S,R}}{L_e(S,R)} & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \quad (1)$$

where $D_{S,R}$ is the MAC latency from S to R. Therefore, the ELD via R, denoted as $ELD(S,R)$, is $E(LD(S,R))$ which is calculated as

$$\begin{cases} \frac{E(D_{S,R})}{L_e(S,R)} & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \quad (2)$$

For every neighbor R of S, S associates with R a rank

$$\langle ELD(S,R), var(LD(S,R)), L_{R,D}, ID(R) \rangle$$

where $var(LD(S,R))$ denotes the variance of $LD(S,R)$, and $ID(R)$ denotes the unique ID of node R. Then, S selects as its next-hop forwarder the neighbor that ranks the lowest among all the neighbors.

To understand what ELD implies for routing, we set up an experiment as follows: consider a line network formed by row 6 of the indoor testbed shown in Figure 2, the Stargate S at column 0 needs to send packets to the Stargate D at the other end (i.e., column 14). Using the data on unicast MAC latencies in the case of *interferer-free*, we show in Figure 10 the mean unicast MAC latencies and the corresponding ELD's regarding neighbors at different distances. From the figure, Stargate D, the destination which is 12.8 meters away from S, offers the lowest ELD, and S sends packets directly to D. From this example, we see that, using metric ELD, a node tends to choose nodes beyond the perfectly reliable communication range as forwarders, to reduce end-to-end MAC latency as well as energy consumption. Note that, however, the links chosen via ELD still have good average

[3]In geographic routing, for instance, control packets are usually needed only when the location of a node changes, which occurs infrequently in mostly static networks.

[4]Currently, we focus on the case where a node forwards packets only to a neighbor closer to the destination than itself.

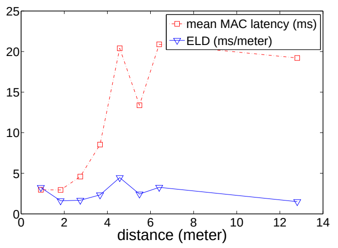


Figure 10. Mean unicast MAC latency and the ELD. Note that the sudden drop in MAC latency at distance 5.49m is due to the high link reliability for the corresponding link as shown in Figure 6; this non-distance-monotonic trend of link reliability and MAC latency is due to reasons such as location-specific signal fading and hardware heterogeneity [40].

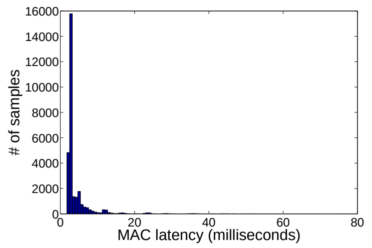


Figure 11. Histogram for unicast MAC latency

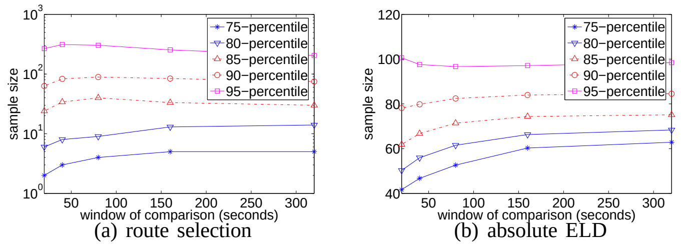


Figure 12. Sample size requirement

properties in terms of minimizing ELD, which is different from opportunistic routing such as ExOR [9] where links with transient good performance can be chosen; data-driven link estimation and opportunistic routing actually complement each other as we will discuss in Section VI.

Remark. ELD is a locally measurable metric based only on the geographic locations of nodes and information regarding the links associated with the sender S; ELD does not assume link conditions beyond the local neighborhood of S. In the analysis of geographic routing [33], however, a common assumption is *geographic uniformity* — that the hops in any route have similar properties such as geographic length and link quality. As we will show by experiments in Section V, this assumption is usually invalid. For the sake of verification and comparison, we derive another routing metric ELR, the *expected MAC latency along a route*, based on this assumption. More specifically, $ELR(S,R) =$

$$\begin{cases} E(D_{S,R}) \times \lceil \frac{L_{S,R}+L_{R,D}}{L_{S,R}} \rceil & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \qquad (3)$$

where $\lceil \frac{L_{S,R}+L_{R,D}}{L_{S,R}} \rceil$ denotes the number of hops to the destination, assuming equal geographic distance at every hop. We will show in Section V that ELR is inferior to ELD.

B. Sample size analysis

To understand the convergence speed of ELD-based routing and to guide protocol design, we experimentally study the sample size required to distinguish out the best neighbor in routing.

In our indoor testbed, let the Stargate at column 0 of row 6 be the sender S and Stargate at the other end of row 6 be the destination D; then let S send 30,000 1200-byte unicast packets to each of the other Stargates in the testbed, to get information (e.g., MAC latency and reliability) on all the links associated with S. The objective is to see what sample size is required for S to distinguish out the best neighbor.

First, we need to derive the *distribution model* for MAC latency. Figure 11 shows the histogram of the unicast MAC latencies for the link to a node 3.65 meters (i.e., 12 feet) away from S. (The MAC latencies for other links assume similar patterns.) Given the shape of the histogram and the fact that MAC latency is a type of "service time", we select three models for evaluation: exponential, gamma, and lognormal.[5] Against the data on the MAC latencies for all the links associated with S, we perform Kolmogorov-Smirnov test [23] on the three models, and we find that *lognormal* distribution fits the data the best.

[5]The methodology of LOF is independent of the distribution model adopted. Therefore, LOF would still apply even if better models are found later.

Therefore, we adopt lognormal distribution for the analysis in this paper. Given that MAC latency assumes lognormal distribution, the LD associated with a neighbor also assumes lognormal distribution, i.e., $log(LD)$ assumes normal distribution.

Because link quality varies temporally, the best neighbor for S may change temporally. Therefore, we divide the 30,000 MAC latency samples of each link into chunks of time span W_c, denoted as the *window of comparison*, and we compare all the links via their corresponding sample-chunks. Given each sample chunk for the MAC latency of a link, we compute the sample mean and sample variance for the corresponding $log(LD)$, and use them as the mean and variance of the lognormal distribution. When considering the i-th sample chunks of all the links ($i = 1, 2, \ldots$), we find the best link according to these sample chunks, and we compute the sample size required for comparing this best link with each of the other links as follows:

> Given two normal variates X_1, X_2 where $X_1 \sim N(\mu_1, \delta_1^2)$ and $X_2 \sim N(\mu_2, \delta_2^2)$, the sample size required to compare X_1 and X_2 at $100(1-\alpha)\%$ confidence level is $(\frac{Z_\alpha(\delta_1+\delta_2)}{\mu_1-\mu_2})^2$ $(0 \leq \alpha \leq 1)$, with Z_α being the α-quantile of a unit normal variate [24].

In the end, we have a set of sample sizes for each specific W_c. For a 95% confidence level comparison and route selection, Figure 12(a) shows the 75-, 80-, 85-, 90-, and 95-percentiles of the sample sizes for different W_c's. We see that the percentiles do not change much as W_c changes. Moreover, we observe that, even though the 90- and 95-percentiles tend to be large, the 75- and 80-percentiles are pretty small (e.g., being 2 and 6 respectively when W_c is 20 seconds), which implies that routing decisions can converge quickly in most cases. This observation also motivates us to use initial sampling in LOF, as detailed in Section IV-B.

Remarks. In the analysis above, we did not consider the temporal patterns of link properties (which are usually unknown). Had the temporal patterns been known and used in link estimation, the sample size requirement can be even lower.

By way of contrast, we also compute the sample size required to estimate the absolute ELD value associated with each neighbor. Figure 12(b) shows the percentiles for a 95% confidence level estimation with an accuracy of $\pm 5\%$ [24]. We see that, even though the 90- and 95-percentiles are less than those for route selection, the 75- and 80-percentiles (e.g., being 42 and 51 respectively when W_c is 20 seconds) are significantly greater than those for route selection. Therefore, when analyzing sample size requirement for routing, we should focus on relative comparison among neighbors rather than on estimating the absolute value, unlike what has been done in the literature [35].

IV. LOF: A DATA-DRIVEN PROTOCOL

Having defined the ETX/ETT-type data-driven routing metric ELD, we are ready to design protocol LOF for implementing ELD. Without loss of generality, we only consider a single destination, i.e., the base station to which every other node needs to find a route. In a nutshell, LOF works as follows: when a node boots up, it interacts with its neighbors to obtain the geographic location of the base station, as well as the IDs and locations of its neighbors; for every newly discovered neighbor, the node takes a few initial samples of the unicast MAC latency and initializes the ELD metric for the neighbor accordingly; after initialization, the node adapts its link estimation solely based on MAC feedback for unicast data transmissions; to address imperfect initial sampling and temporal link variation, the node also probabilistically explores alternative forwarders with controlled frequency. In what follows, we first elaborate on the individual components of LOF, then we discuss implementation issues such as reliably fetching MAC feedback.

A. Learning where we are

LOF enables a node to learn its neighborhood and the location of the base station via the following rules:

1) **[Issue request]** Upon boot-up, a node broadcasts M copies of *hello-request* packets if it is not the base station. A *hello-request* packet contains the ID and the geographic location of the issuing node. To guarantee that a requesting node is heard by its neighbors, we set M as 7 in our experiments.
2) **[Answer request]** When receiving a *hello-request* packet from another node that is farther away from the base station, the base station or a node that has a path to the base station acknowledges the requesting node by broadcasting M copies of *hello-reply* packets. A *hello-reply* packet contains the location of the base station as well as the ID and the location of the issuing node.
3) **[Handle announcement]** When a node A hears for the first time a *hello-reply* packet from another node B closer to the base station, A records the ID and location of B and regards B as a forwarder-candidate.
4) **[Announce presence]** When a node other than the base station finds a forwarder-candidate (and thus a path) for the first time, or when the base station boots up, it broadcasts M copies of *hello-reply* packets.

To reduce potential contention, every broadcast transmission mentioned above is preceded by a randomized waiting period whose length is dependent on node distribution density in the network. Note that the above rules can be optimized in various ways. For instance, rule 2 can be optimized such that a node acknowledges at most one *hello-request* from another node each time the requesting node boots up. Even though we have implemented quite a few such optimizations, we skip the details here since they are not the focus of this paper.

B. Initial sampling

Having learned the location of the base station as well as the locations and IDs of its neighbors, a node S needs to identify the best forwarder in routing. To jump start the route/forwarder selection process, S takes a few samples of the MAC latency for the link to every newly learned neighbor R before forwarding any data packets. The initial sampling is achieved by S sending a few unicast packets to every neighbor and then fetching the MAC feedback, based on which S initializes the ELD metric for every neighbor and selects the tentative next-hop forwarder.

The initial sampling gives a node a rough idea of the relative quality of the links to its neighbors, because, as traffic and interference pattern changes, the relative ranking in the mean MAC latency (and thus LD) among links changes less significantly than does MAC latency (and LD) itself as we show in Figure 13. Another reason for initial sampling is that, with

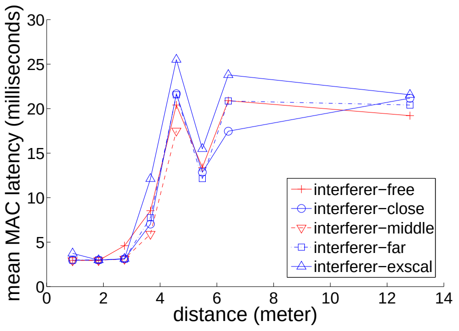


Figure 13. Mean unicast MAC latency in different interfering scenarios for the indoor experiments described in Section II-A

relatively small sample size, a node could gain a decent sense of the relative goodness of its neighbors as we show in the analysis of Section III-B. We set the initial sample size as 6 (i.e., the 80-percentile of the sample size when W_c is 20 seconds) in our experiments.

C. Data-driven adaptation

Via initial sampling, a node gets a rough estimation of the relative goodness of its neighbors. To improve its route selection for an application traffic pattern, the node needs to adapt its estimation of LD via the MAC feedback for unicast data transmission. (According to the analysis in Section III-B, route decisions converge quickly because of the small sample size requirement.) Since LD is lognormally distributed, LD is estimated by estimating $log(LD)$.

On-line estimation. To determine the estimation method, we first check the properties of the time series of $log(LD)$, considering the same scenario as discussed in Section III-B. Figure 14 shows a time series of the $log(LD)$ regarding a node 3.65 meters (i.e., 12 feet) away from the sender S (The $log(LD)$ for the other nodes assumes similar patterns.). We see that the time series fits well with the *constant-level model* [22] where the generating process is

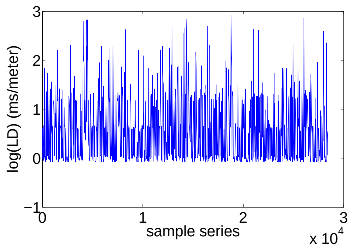


Figure 14. A time series of $log(LD)$

represented by a constant superimposed with random fluctuations. Therefore, a good estimation method is *exponentially weighted moving average* (EWMA) [22], assuming the following form

$$V_{i+1} \longleftarrow \alpha V_i + (1-\alpha)V' \tag{4}$$

where V_{i+1} is the next estimate of parameter V to be obtained, V_i is the current estimate, V' is the latest observation of V, and α is the weight ($0 \leq \alpha \leq 1$).

In LOF, when a new MAC latency and thus a new $log(LD)$ value with respect to the current next-hop forwarder R is observed, the V_i value in the right hand side of formula (4) may be quite old if R has just been selected as the next-hop and some packets have been transmitted to other neighbors immediately before. To deal with this issue, we define the *age factor* $\beta(R)$ of the current next-hop forwarder R as the number of packets that have been transmitted since V_i of R was last updated. Then, formula (4) is adapted to be the following:

$$V_{i+1} \longleftarrow \alpha^{\beta(R)} V_i + (1-\alpha^{\beta(R)})V' \tag{5}$$

(Experiments confirm that LOF performs better with formula (5) than with formula (4).)

Each MAC feedback indicates whether a unicast transmission has succeeded and how long the MAC latency l is. When a node receives a MAC feedback, it first calculates the age factor $\beta(R)$ for the current next-hop forwarder, then it adapts the estimation of $log(LD)$ as follows:

- If the transmission has succeeded, the node calculates the new $log(LD)$ value using l and applies it to formula (5) to get a new estimation regarding the current next-hop forwarder.
- If the transmission has failed, the node should not use l directly because it does not represent the latency to successfully transmit a packet. To address this issue, the node keeps track of the unicast delivery rate, which is also estimated using formula (5), for each associated link. Then, if the node retransmits this unicast packet via the currently used link, the expected number of retries until success is $\frac{1}{p}$, assuming that unicast failures are independent and that the unicast delivery rate along the link is p. Including the latency for this last failed transmission, the expected overall latency l' is $(1+\frac{1}{p})l$. Therefore, the node calculates the new $log(LD)$ value using l' and applies it to formula (5) to get a new estimation.

Route adaptation. As the estimation of LD changes, a node S adapts its route selection by the ELD metric. Moreover, if the unicast reliability to a neighbor R is below certain threshold (say 60%), S will mark R as dead and will remove R from the set of forwarder-candidates. If S loses all its forwarder-candidates, S will first broadcast M copies of *hello-withdrawal* packets and then restarts the routing process. If a node S' hears a *hello-withdrawal* packet from S, and if S is a forwarder-candidate of S', S' removes S from its set of forwarder-candidates and update its next-hop forwarder as need be. (As a side note, we find that, on average, only 0.9863 neighbors of any node are marked as dead in both our testbed experiments and the field deployment of LOF in project ExScal [6]. Again, the withdrawing and rejoining process can be optimized, but we skip the details here.)

D. Exploratory neighbor sampling

Given that the initial sampling is not perfect (e.g., covering 80% instead of 100% of all the possible cases) and that wireless link quality varies temporally (e.g., from initial sampling to actual data transmission), the data-driven adaptation alone may miss using good links, simply because they were relatively bad when tested earlier and they do not get chance to be tried out later on. Therefore, we propose exploratory neighbor sampling in LOF. That is, whenever a node S has consecutively transmitted $I_{ns}(R_0)$ number of data packets using a neighbor R_0, S will switch its next-hop forwarder from R_0 to another neighbor R' with probability $P_{ns}(R')$ (so that the link quality to R' can be sampled). On the other hand, the exploratory neighbor sampling is optimistic in nature, and it should be used only for good neighbors. In LOF, exploratory neighbor sampling only considers the set of neighbors that are not marked as dead.

In what follows, we explain how to determine the sampling probability $P_{ns}(R')$ and the sampling interval $I_{ns}(R_0)$. For convenience, we consider a sender S, and let the neighbors of S be $R_0, R_1, \ldots, R_N$ with increasing ranks.

Sampling probability. At the moment of neighbor sampling, a better neighbor should be chosen with higher probability. In LOF, a neighbor is chosen with the probability of the neighbor actually being the best next-hop forwarder. We derive this probability in three steps: the probability $P_b(R_i, R_j)$ of a neighbor R_i being actually better than another one R_j (given by formula (6)), the probability $P_h(R_i)$ of a neighbor R_i being actually better than all the neighbors that ranks lower than itself (given by formula (7)), and the probability $P_{ns}(R_i)$ of a neighbor R_i being actually the best forwarder (given by formula (8)).

Given S and its two neighbors R_i and R_j, we approximate $P_b(R_i, R_j)$ with $P\{LD(S,R_i) > LD(S,R_j)\}$, which equals $P\{log(LD(S,R_i)) > log(LD(S,R_j))\}$. As discussed in Section III-B, $log(LD(S,R_i))$ as well as $log(LD(S,R_j))$ has a normal distribution. Assume $log(LD(S,R_i)) \sim N(\mu_i, \delta_i^2)$, $log(LD(S,R_j)) \sim N(\mu_j, \delta_j^2)$, and that $log(LD(S,R_i))$ is independent of $log(LD(S,R_j))$, then we have

$$P_b(R_i, R_j) = G\left(\frac{\mu_j - \mu_i}{\sqrt{\delta_i^2 + \delta_j^2}}\right) \tag{6}$$

where $G(x) = 1 - \Phi(x)$, with

$$\Phi(x) = \begin{cases} \frac{1}{2}\text{erfc}(-x/\sqrt(2)) & x \leq 0 \\ 1 - \frac{1}{2}\text{erfc}(x/\sqrt(2)) & x > 0 \end{cases}$$
$$\text{erfc}(x) \approx \left(\frac{1}{1+x/2}\right)\text{exp}\left(-x^2 + P\left(\frac{1}{1+x/2}\right)\right)$$
$$P(x) = 0.17087277x^9 - 0.82215223x^8 + 1.48851587x^7 - 1.13520398x^6 + 0.27886807x^5 - 0.18628806x^4 + 0.09678418x^3 + 0.37409196x^2 + 1.00002368x - 1.26551223.$$

Knowing $P_b(R_j, R_k)$ for every j and k, we compute $P_h(R_i)$

($i = 1, \ldots, N$) inductively as follows:

$$\begin{aligned} &P_h(R_1) = P_b(R_1, R_0); \\ &P_h(R_i) \approx P_b(R_i, R_0) \times \textstyle\prod_{j=1}^{i-1}(1 - (P_b(R_j, R_i) + \\ &\qquad\qquad (P_h(R_j) - 1) \times P_b(R_0, R_i))) \\ &\qquad\qquad\qquad (i = 2, \ldots, N) \end{aligned} \tag{7}$$

Then, we compute the sampling probability as follows:

$$\begin{aligned} &P_{ns}(R_0) = P_b(R_0, R_1) \times \textstyle\prod_{j=2}^{N}(1 - P_h(R_j)); \\ &P_{ns}(R_i) = P_h(R_i) \times \textstyle\prod_{j=i+1}^{N}(1 - P_h(R_j)) \\ &\qquad\qquad (i = 1, \ldots, N-1); \\ &P_{ns}(R_N) = P_h(R_N) \end{aligned} \tag{8}$$

(**Interested readers can find the derivation of formulas (6), (7), and (8) in [38].**)

Because of the approximation in formula (7), $\sum_{i=0}^{N} P_{ns}(R_i)$ may not equal to 1. To address this issue, we normalize the $P_{ns}(R_i)$'s ($i = 0, \ldots, N$) such that their sum is 1.

Sampling interval. The frequency of neighbor sampling should depend on how good the current next-hop forwarder R_0 is, i.e., the sampling probability $P_{ns}(R_0)$. In LOF, we set the sampling interval $I_{ns}(R_0)$ to be proportional to $P_{ns}(R_0)$, that is,

$$I_{ns}(R_0) = K \times (N \times P_{ns}(R_0)) \tag{9}$$

where N is the number of active neighbors that S has,[6] and K is a constant being proportional to the stability of link quality as measured in the expected number of data packets that will be generated in each period of stable link quality. We set K to be 20 in our experiments.

The above method of setting the sampling probability and the sampling interval ensures that better forwarders will be sampled with higher probabilities, and that the better the current forwarder is, the lower the frequency of exploratory neighbor sampling. The sampling probabilities and the sampling interval are re-calculated each time the next-hop forwarder is changed.

E. Implementation issues

In this subsection, we discuss implementation issues of LOF.

MAC feedback exfiltration. In LOF, both the status and the MAC latency are required for every unicast transmission. Yet the default Linux WLAN driver *hostap* [5] only signals failed unicast transmissions, and it does not signal the unicast MAC latency. Therefore, we modify the Linux kernel and the hostap driver such that the transmission status, whether success or failure, is always signaled and the MAC latency is reported too. Since we implement LOF, using EmStar [4], as a user-space process, MAC feedback is sent to the LOF process via *netlink* sockets and */proc* file system [21].

Given that the LOF process executes in user-space and that packet transmission is supported via UDP sockets in EmStar, there is memory copying in the procedure between the LOF process sending a packet and the hostap driver transmitting the corresponding 802.11b MAC frame(s). Thus, one issue is how to map a data transmission at the user-space with the frame transmission at the driver and thus the MAC feedback. Fortunately, the data buffers in EmStar, Linux TCP/IP stack, hostap driver, and the SMC WLAN card are managed in the first-in-first-out (FIFO) manner. Therefore, as long as we make sure that each data transmission from the LOF process can be encapsulated in a single MAC frame, each MAC feedback can be mapped with the corresponding data transmission if there is no loss of MAC feedback.

Nevertheless, we find that, under stressful conditions, MAC feedback may get lost in two ways:

- A MAC feedback will be dropped in *netlink* sockets if the socket buffer overflows.
- If there is no valid ARP (Address Resolution Protocol) entry regarding the unicast destination, a data packet is dropped at the IP layer (without informing the application layer) before even getting to the hostap driver, which means that no MAC feedback will be generated and thus "lost".

To deal with possible loss of MAC feedback, LOF adopts the following two mechanisms:

- To avoid buffer overflow at *netlink* sockets, LOF enforces flow control within a node by enforcing an upper bound on the number of data transmissions whose MAC feedback has not come back. (This upper bound is set to 7 in our experiments.)
- After each data transmission, LOF checks the kernel ARP table to see if there is a valid entry for the destination of this unicast packet. In this way, LOF is able to decide whether a MAC feedback will ever come back and act accordingly.

Via the stress tests in both testbeds and outdoor deployment, we find that the above mechanisms guarantee the reliable delivery of MAC feedback.

We implement LOF at user-space for the sake of safety and easy maintenance. As a part of our future work, we are exploring implementing LOF in kernel space to see if the process of reliably fetching MAC feedback can be simplified.

Reliable transport. MAC feedback helps not only in link quality estimation but also in reliable data transport. For example, upon detecting a failed transmission via the MAC feedback, a node can retransmit the failed packet via a new next-hop forwarder. On the other hand, the transmission status carried in a MAC feedback only reflects the reliability at the MAC layer. To guarantee end-to-end reliability, we need to make sure that packet delivery is reliable at layers above MAC: First, we need to guarantee the liveness of the LOF routing process, which is enabled by the EmStar process monitoring facility *emrun* in our current implementation; Second, the sender of a packet transmission guarantees that the packet is received by the hostap driver, using the transmission status report from EmStar; Third, sender-side flow control guarantees that there is no queue overflow at the receiver side.

Node mobility. Given that nodes in most sensor networks are static, LOF is not designed to support high degree of mobility. Nevertheless, LOF can deal with infrequent movement of nodes in the following simple manner:

- If the base station moves, the new location of the base station is diffused across the network;
- If a node other than the base station moves, it first broadcast M copies of *hello-withdrawal* packets, then it restarts its routing process.

(Note that a node can detect the movement of itself with the help of a GPS device.)

[6]Since $P_{ns}(R_0)$ tends to decrease as N increases (when the other factors such as network setup are fixed), we introduce N in the formula to avoid having too small $I_{ns}(R_0)$.

Neighbor-table size control. Compared with Berkeley motes, Stargates have relatively large memory and disk size (e.g., 64MB RAM and 32MB flash disk). Therefore, we adopt a very simple method of neighbor-table size control: keeping the best next-hop forwarders according to their ranks. In our experiments, we set the maximum neighbor table size as 20. A more detailed study of the best neighborhood management scheme for Stargates is beyond the scope of this paper.

V. Experimental evaluation

Via testbeds and field deployment, we experimentally evaluate the design decisions and the performance of LOF.

A. Experiment design

Network setup. In our indoor testbed as shown in Figure 2, we let the Stargate at the left-bottom corner of the grid be the base station, to which the other Stargates need to find routes. Then, we let the Stargate S at the upper-right corner of the grid be the traffic source. S sends packets of length 1200 bytes according to the ExScal event trace as discussed in Section II-A and Figure 3. For each protocol we study, S simulates 50 event runs, with the interval between consecutive runs being 20 seconds. Therefore, for each protocol studied, 950 (i.e., 50×19) packets are generated at S.

We have also tested scenarios where multiple senders generate ExScal traffic simultaneously, the data traffic is periodic, or the network assumes a random instead of a grid topology; LOF has also been used in the backbone network of ExScal. We discuss them in Section V-C, together with other related experiments.

Protocols studied. We study the performance of LOF in comparison with that of beacon-based routing, represented by ETX [13], [35] and PRD [33]: (For convenience, we do not differentiate the name of a routing metric and the protocol implementing it.)

- *ETX*: expected transmission count. It is a type of geography-unaware distance-vector routing where a node adopts a route with the minimum ETX value. Since the transmission rate is fixed in our experiments, ETX routing also represents another metric ETT [15], where a route with the minimum *expected transmission time* is used. Since ETT performs similar to *MAC latency* as shown in [15], ETX routing performs similar to MAC latency based routing and is the distance-vector, beacon-based counterpart of LOF.
- *PRD*: product of packet reception rate and distance traversed to the destination. Unlike ETX, PRD is geography-based. In PRD, a node selects as its next-hop forwarder the neighbor with the maximum PRD value. PRD can be regarded as the geographic counterpart of ETX, since $\frac{1}{PRD}$ approximates the *expected ETX per unit-distance to destination*. Accordingly, PRD can be regarded as a beacon-based counterpart of LOF.

In our experiments, metrics ETX and PRD are estimated according to the method originally proposed in [13] and [33]. For instance, broadcast beacons have the same packet length and transmission rate as those of data packets.

To verify some important design decisions of LOF, we also study different versions of LOF as follows:

- *L-hop*: assumes geographic-uniformity, and thus uses metric ELR, as specified by formula (3), instead of ELD;
- *L-ns*: does not use the method of exploratory neighbor sampling;
- *L-sd*: considers, in exploratory neighbor sampling, the neighbors that have been marked as dead;
- *L-se*: performs exploratory neighbor sampling after every packet transmission.

We will discuss distance-vector data-driven routing in Section V-C.

For easy comparison, we have implemented all the protocols mentioned above in EmStar [4], a software environment for developing and deploying wireless sensor networks.

Evaluation criteria. Reliability is one critical concern in convergecast. Using the techniques of reliable transport discussed in Section IV-E, all the protocols guarantee 100% packet delivery in our experiments. Therefore, we compare protocols in metrics other than reliability as follows:

- *End-to-end MAC latency*: the sum of the MAC latency spent at each hop of a route. This reflects not only the delivery latency but also the throughput available via a protocol [13], [15].
- *Energy efficiency*: energy spent in delivering a packet to the base station.

B. Experimental results

MAC latency. Using boxplots, Figure 15 shows the end-to-end

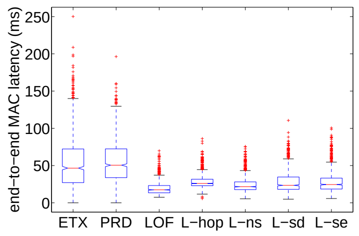


Figure 15. End-to-end MAC latency

MAC latency, in milliseconds, for each protocol. The average end-to-end MAC latency in both ETX and PRD is around 3 times that in LOF, indicating the advantage of data-driven link quality estimation. The MAC latency in LOF is also less than that of the other versions of LOF, showing the importance of using the right routing metric (including not assuming geographic uniformity) and neighbor sampling technique.

To explain the above observation, Figures 16, 17, 18, and 19

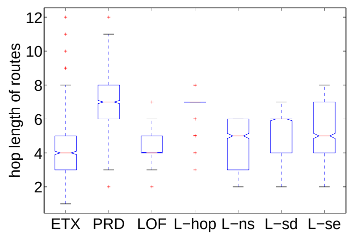


Figure 16. Number of hops in a route

show the route hop length, per-hop MAC latency, average per-hop geographic distance, and the coefficient of variation (COV) of

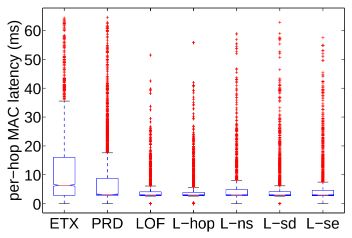


Figure 17. Per-hop MAC latency

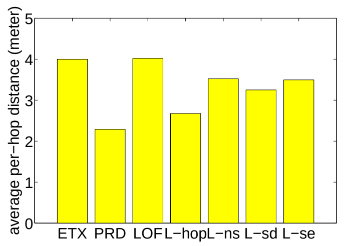


Figure 18. Average per-hop geographic distance

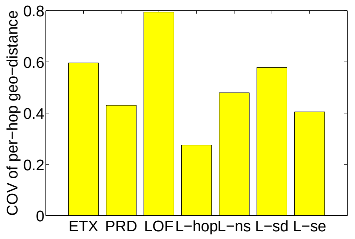


Figure 19. COV of per-hop geographic distance in a route

per-hop geographic distance. Even though the average route hop length and per-hop geographic distance in ETX are approximately the same as those in LOF, the average per-hop MAC latency in ETX is about 3 times that in LOF, which explains why the end-to-end MAC latency in ETX is about 3 times that in LOF. In PRD, both the average route hop length and the average per-hop MAC latency is about twice that in LOF.

From Figure 19, we see that the COV of per-hop geographic distance is as high as 0.2754 in L-hop. Therefore, the assumption of geographic uniformity is invalid, which partly explains why L-hop does not perform as well as LOF. Moreover, the fact that the COV value in LOF is the largest and that LOF performs the best tend to suggest that the network state is heterogeneous at different locations of the network.

Energy efficiency. Given that beacons are periodically broadcasted in ETX and PRD, and that beacons are rarely used in LOF, it is easy to see that more beacons are broadcasted in ETX and PRD than in LOF. Therefore, we focus our attention only on the number of unicast transmissions required for delivering data packets to the base station, rather than on the broadcast overhead. To this end, Figure 20 shows the number of unicast transmissions averaged over the number of packets received at the base station, denoted as $numPkt$. $numPkt$ in ETX and PRD is 1.49 and 2.37 times that in LOF respectively, showing again the advantage of data-driven instead of beacon-based link quality estimation. $numPkt$ in LOF is also less than that in the other versions of LOF. For instance, $numPkt$ in L-hop is 2.89 times that in LOF.

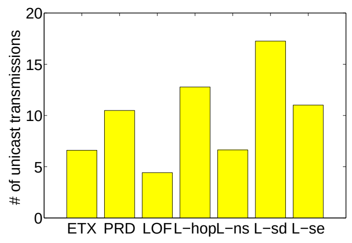


Figure 20. Number of unicast transmissions per packet received

Note that $numPkt$ is high in L-sd; this is mainly because 1) the nodes closer to the base station have fewer forwarder candidates and thus their dead neighbors are sampled with non-negligible, higher frequencies, yet 2) these nodes need to transmit much more data than nodes farther away from the base station (due to the nature of convergecast), and 3) transmissions toward dead neighbors tend to fail and need retransmission. $numPkt$ in L-ns is similar to that in ETX; nonetheless, L-ns still performs better than ETX by incurring less end-to-end and per-hop MAC latency (as shown in Figures 15 and 17) and by using more reliable links (as to be shown in Figure 21). Therefore, data-driven link estimation and routing, even without exploratory neighbor sampling (as in L-ns), still outperforms beacon-based approaches.

Given that the SMC WLAN card in our testbed uses Intersil Prism2.5 chipset which does not expose the information on the number of retries of a unicast transmission, Figure 20 does not represent the actual number of bytes sent. Nevertheless, given Figure 17 and the fact that MAC latency and energy consumption are positively related (as discussed in Section III-A), the above observation on the relative energy efficiency among the protocols still holds.

To explain the above observation, Figure 21 shows the number

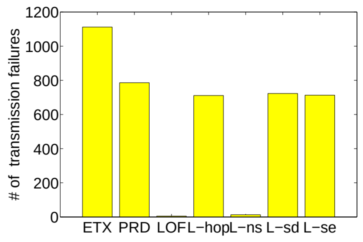


Figure 21. Number of failed unicast transmissions

of failed unicast transmissions for the 950 packets generated at the source. The number of failures in ETX and PRD is 1112 and 786 respectively, yet there are only 5 transmission failures in LOF. Also, there are 711 transmission failures in L-hop. Together with Figures 18 and 5(b), we see that there exist reliable long links, yet only LOF tends to find them well: ETX also uses long links, but they are not reliable; L-ns uses reliable links, but they are relatively shorter.

C. Other experiments

Multiple senders and periodic traffic. Besides the scenario of 1 source event traffic which we discussed in detail in the last subsection, we have performed experiments where the Stargate at the upper-right corner and its two immediate grid-neighbors simultaneously generate packets according to the ExScal traffic trace. We have also experimented with periodic traffic where 1 or 3 Stargates (same as those in the case of event traffic) generate 1,000 packets each, with each packet being 1200-byte long and the inter-packet interval being 500 milliseconds. In these experiments, we have observed similar patterns in the relative protocol performance as those in the case of 1 source event traffic. For conciseness, we only present the end-to-end MAC latency for these three cases, as shown in Figures 22, 23, and 24.

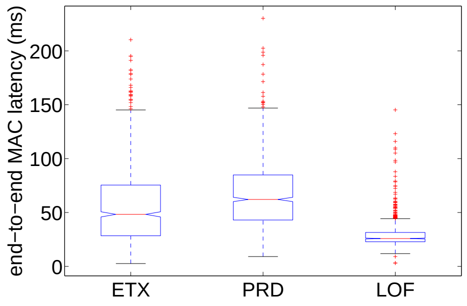


Figure 22. End-to-end MAC latency: event traffic, 3 senders

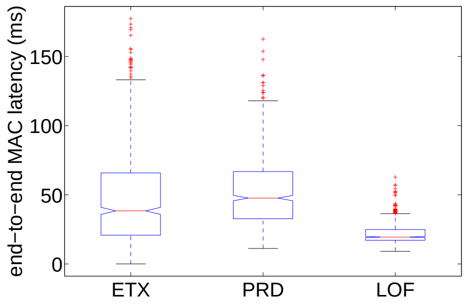


Figure 23. End-to-end MAC latency: periodic traffic, 1 sender

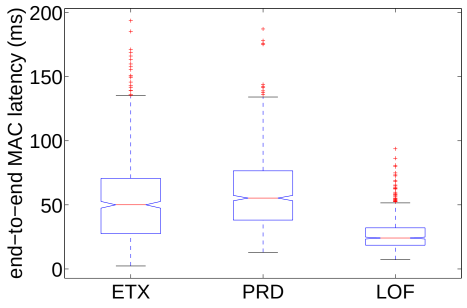


Figure 24. End-to-end MAC latency: periodic traffic, 3 senders

Distance-vector data-driven routing and network throughput. We have so far focused on the geographic version of data-driven link estimation and routing in this paper, so that we do not need periodic control packets at all. Nonetheless, we can also adopt the classical distance-vector routing (based on distributed Bellman-Ford algorithm) with data-driven link estimation. The distance-vector version of data-driven link estimation and routing is useful in cases where high-precision location information is not readily available or the nodes in a network are not uniformly distributed. In distance-vector data-driven routing, the routing metric is the distance-vector counterpart of ELD, that is, end-to-end MAC latency which Draves *et al.* [15] have shown to perform virtually identical to the ETT metric. We have implemented the distance-vector data-driven routing protocol *LOF-dv* in EmStar, and we experimentally measure its performance in Kansei. For the case where there is one node generating data packets according to the ExScal traffic trace, Figure 25 shows how the end-to-end MAC

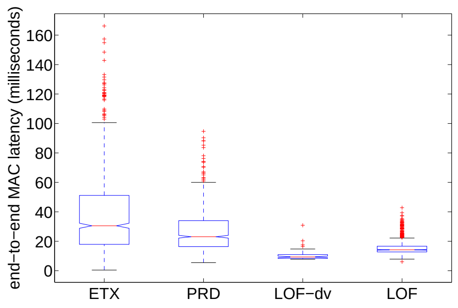


Figure 25. End-to-end MAC latency: event traffic, 1 sender. (Note: the experiments are done with a 15×6 subgrid of Kansei.)

latency in LOF-dv compares with that in other protocols. We see that LOF-dv has similar performance as LOF, and LOF-dv performs better than beacon-driven routing ETX and PRD. We observe similar patterns in terms of other evaluation metrics (such as energy efficiency) and experimentation setup (such as multiple senders and periodic traffic).

To understand the impact of different protocols on network capacity, we also measure the network throughput by letting the corner source node generating data packets as fast as possible. Each data packet contains a payload of 1200 bytes. Figure 26 shows the throughput in different protocols. We see that LOF

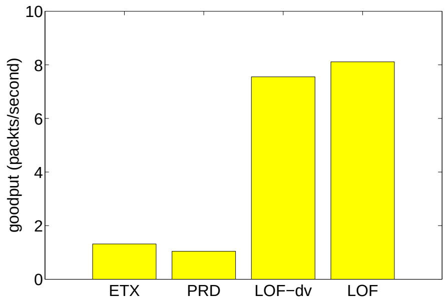


Figure 26. Network throughput (Note: the experiments are done with a 15×6 grid of Kansei.)

and LOF-dv yield similar network throughput, and they both significantly improves the network throughput of ETX and PRD (e.g., up to a factor of 7.78). Our current experimental facility limits the highest achievable one-hop throughput to be 50 packets/second, thus the highest achievable multi-hop throughput is 7.14 packets/second [30]. We see that both LOF and LOF-dv achieve a throughput very close to the highest possible network throughput.

The data presented in Figures 25 and 26 are for experiments executed on a 15×6 grid of Kansei. When executing these experiments, we were unable to access the complete grid of Kansei due to some maintenance issues. But we believe the observations will carry over to other network setups including the complete grid of Kansei.

Random network topology. Besides grid topology, we also evaluate the performance of different protocols in random network topology. To generate random topology, we randomly select 50 Stargates out of the 15×6 Kansei grid where each Stargate is selected with equal probability, and we keep the Stargate at the left-bottom corner of the 15×6 grid as the base station. For the case where a node farthest away from the base station generates packets according to the ExScal traffic trace, Figure 27 shows the end-to-end MAC latency in different protocols. We see

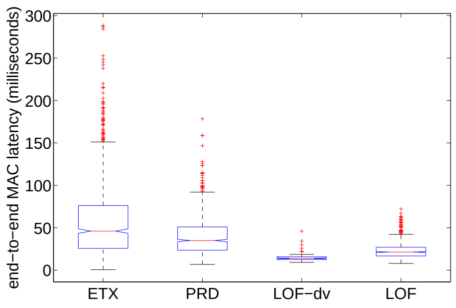


Figure 27. End-to-end MAC latency: random network topology

that data-driven protocols LOF and LOF-dv outperform beacon-based protocols ETX and PRD in this case too. Similarly, LOF and LOF-dv also have higher energy efficiency than ETX and PRD. Compared with the case of grid topology, the MAC latency incurred in the random topology is slightly larger; this is because the generated random network topology is sparser than and is a subgraph of the grid topology, thus the optimal routes in the grid topology may not exist in the random topology, and, for the same source node, the optimal route in the random topology may perform worse than the optimal route available in the grid topology.

Field deployment. Based on its well-tested performance, LOF has been incorporated in the ExScal sensor network field experiment [6], where 203 Stargates were deployed as the backbone network, with the inter-Stargate separation being around 45 meters. LOF successfully guaranteed reliable and real-time convergecast from any number of non-base Stargates to the base station in ExScal, showing not only the performance of the protocol but also the stability of its implementation.

VI. Related work

Link properties in 802.11b mesh networks and dense wireless sensor networks have been well studied in [7], [27], and [39]. They have observed that wireless links assume complex properties, such as wide-range non-uniform packet delivery rate at different distances, loose correlation between distance and packet delivery rate, link asymmetry, and temporal variations. Our study on link properties complements existing works by focusing on the differences between broadcast and unicast link properties, as well as the impact of interference pattern on the differences.

Differences between broadcast and unicast and their impact on the performance of AODV have been discussed in [31] and [12]. Our work complements [31] and [12] by experimentally studying the differences as well as the impact of environment, distance, and interference pattern on the differences, which were not the focus of [31] and [12]. [12] mentioned the difficulty of getting MAC feedback and thus focused on the method of beacon-based link estimation. Our work complements [12] by developing techniques for reliably fetching MAC feedback, which build the foundation for data-driven link estimation and routing. To improve the performance of AODV, [31] and [12] also discussed reliability-based mechanisms (e.g., RSSI- and SNR-based ones) for blacklisting bad links. Since it has been shown that reliability-based blacklisting does not perform as well as ETX [17], [13], [35], we do not directly compare LOF to [31] and [12], instead we compare LOF to ETX.

Recently, great progress has been made regarding routing in wireless sensor networks as well as in mesh networks. Routing metrics such as ETX [13], [35] and ETT/WCETT [15] have been proposed and shown to perform well in real-world wireless networks [14]. The geography-based metric PRD [33] has also been proposed for energy-efficient routing in wireless sensor networks. Nevertheless, unicast link properties were still estimated using broadcast beacons in these works. Our work differs from existing approaches by experimentally demonstrating the difficulty of precisely estimating unicast link properties via those of broadcast beacons, and proposing to estimate unicast link properties via the data traffic itself.

Similar to LOF, MAC feedback is also used in protocols such as SPEED [20], NADV [29], EAR [26], four-bit-estimation [16], and CARP [28]. Nonetheless, these protocols did not focus on protocol design issues specific to data-driven link estimation and routing. For instance, they did not consider the importance of appropriate *exploratory neighbor sampling*. SPEED switches next-hop forwarders after every packet transmission (as in L-se), and NADV, EAR, four-bit-estimation, and CARP do not perform exploratory neighbor sampling (as in L-ns), both of which degenerate network performance as shown in Section V. Complementary to these study, moreover, we have systematically characterized the differences between broadcast and unicast link properties, and we have analyzed the small sample size requirement in LOF, showing both the necessity and feasibility of data-driven link estimation.

Rather than selecting the next-hop forwarder before data transmission, opportunistic routing protocols that take advantage of spatial diversity in wireless transmission have been proposed [9], [11], [19], [36]. In these protocols, the forwarder is selected, through coordination among receivers, in a reactive manner after data transmission. Link estimation can still be helpful in these protocols since it can help effectively select the best set of listeners [9]. Therefore, findings of this paper can be useful in opportunistic routing too. Recently, DSF [18] was proposed to address the impact of duty cycling on wireless routing. DSF also uses the metric ETX for a link, therefore the technique of data-driven link estimation can be incorporated into DSF to improve the precision of estimating link and path ETX.

The problem of local minimum or geographic void has been dealt with in routing protocols such as GPSR [25]. In this paper, therefore, we have not considered this problem since it is independent of our major concerns — data-driven link estimation and routing. As a part of our future work, we plan to incorporate techniques of dealing with geographic void into LOF, by adapting the definition of "effective geographic progress" (in Section III-A) and routing around void. The impact of localization errors on geographic routing has been studied in [32]. In LOF, we adopted a separate software component that fine tunes the GPS readings to reduce localization inaccuracy, as also used in the field experiment ExScal [6].

VII. Concluding remarks

Via experiments in testbeds of 802.11b networks, we have demonstrated the difficulties of precisely estimating unicast link properties via broadcast beacons. To circumvent the difficulties, we have proposed to estimate unicast link properties via data traffic itself, using MAC feedback for data transmissions. To this end, we have modified the Linux kernel and *hostap* WLAN driver to provide feedback on the MAC latency as well as the status of every unicast transmission, and we have built system software for reliably fetching MAC feedbacks. Based on these system facilities, we have demonstrated the feasibility as well as potential benefits of data-driven routing by designing protocol LOF. LOF mainly used three techniques for link quality estimation and route selection: initial sampling, data-driven adaptation, and exploratory neighbor sampling. With its well tested performance and implementation, LOF has been successfully used to support convergecast in the backbone network of ExScal, where 203 Stargates have been deployed in an area of 1260 meters by 288 meters.

In this paper, we have focused on data-driven link estimation and routing in 802.11 networks. But we believe that the concept of data-driven link estimation also applies to other sensor networks such as those using IEEE 802.15.4 radios, since temporal correlation in link properties also leads to estimation inaccuracy in these networks [10]. Besides saving energy by avoiding periodic beaconing, LOF facilitates greater extent of energy conservation, because LOF does not require a node to be awake unless it is generating or forwarding data traffic. We plan to explore these directions in our future work.

Acknowledgment

We thank Dr. Vinayak Naik and Dr. Emre Ertin for their help in experimentation, and we thank the UCLA EmStar team for their help with EmStar. We also thank Dr. Yongguang Zhang and the anonymous referees for their helpful comments on an earlier draft of this paper.

References

[1] Broadband seismic network, mexico experiment (mase). http://research.cens.ucla.edu/.

[2] Codeblue: Wireless sensor networks for medical care. http://www.eecs.harvard.edu/∼mdw/proj/codeblue/.

[3] Crossbow technology inc. http://www.xbow.com/.

[4] EmStar: Software for wireless sensor networks. http://cvs.cens.ucla.edu/emstar/.

[5] Linux WLAN driver *hostap*. http://hostap.epitest.fi/.

[6] Exscal project. http://www.cse.ohio-state.edu/exscal, 2004.

[7] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. In *ACM SIGCOMM*, pages 121–132, 2004.

[8] A. Arora, E. Ertin, R. Ramnath, M. Nesterenko, and W. Leal. Kansei: A high-fidelity sensing testbed. *IEEE Internet Computing*, March 2006.

[9] S. Biswas and R. Morris. ExOR: Opportunistic multi-hop routing for wireless networks. In *ACM SIGCOMM*, 2005.

[10] A. Cerpa, J. Wong, M. Potkonjak, and D. Estrin. Temporal properties of low power wireless links: Modeling and implications on multi-hop routing. In *ACM MobiHoc*, pages 414–425, 2005.

[11] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading structure for randomness in wireless opportunistic routing. In *ACM SIGCOMM*, 2007.

[12] I. Chakeres and E. Belding-Royer. The utility of hello messages for determining link connectivity. In *WPMC*, 2002.

[13] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *ACM MobiCom*, pages 134–146, 2003.

[14] R. Draves, J. Padhye, and B. Zill. Comparison of routing metrics for static multi-hop wireless networks. In *ACM SIGCOMM*, pages 133–144, 2004.

[15] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *ACM MobiCom*, pages 114–128, 2004.

[16] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis. Four-bit wireless link estimation. In *ACM HotNets*, 2007.

[17] O. Gnawali, M. Yarvis, J. Heidemann, and R. Govindan. Interaction of retransmission, blacklisting, and routing metrics for reliability in sensor network routing. In *IEEE SECON*, pages 34–43, 2004.

[18] Y. Gu and T. He. Data forwarding in extremely low duty-cycle sensor networks with unreliable communication links. In *ACM SenSys*, 2007.

[19] T. He, B. M. Blum, Q. Cao, J. A. Stankovic, S. H. Son, and T. F. Abdelzaher. Robust and timely communication over highly dynamic sensor networks. *Real-Time Systems Journal*, 37, 2007.

[20] T. He, J. Stankovic, C. Lu, and T. Abdelzaher. SPEED: A stateless protocol for real-time communication in sensor networks. In *IEEE ICDCS*, 2003.

[21] T. Herbert. *The* Linux TCP/IP *Stack: Networking for Embedded Systems*. Charles River Media, 2004.

[22] F. Hillier and G. Lieberman. *Introduction to Operations Research*. McGraw-Hill, 2001.

[23] M. Hollander. *Nonparametric statistical methods*. Wiley, 1999.

[24] R. Jain. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, Inc., 1991.

[25] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *ACM MobiCom*, pages 243–254, 2000.

[26] K.-H. Kim and K. G. Shin. On accurate measurement of link quality in multi-hop wireless mesh networks. In *ACM MobiCom*, 2006.

[27] D. Kotz, C. Newport, and C. Elliott. The mistaken axioms of wireless-network research. Technical Report TR2003-467, Dartmouth College, Computer Science, July 2003.

[28] R. Krishnan, A. Raniwala, and T. cker Chiueh. Design of a channel characteristics-aware routing protocol. In *IEEE INFOCOM miniconference*, 2008.

[29] S. Lee, B. Bhattacharjee, and S. Banerjee. Efficient geographic routing in multihop wireless networks. In *ACM MobiHoc*, pages 230–241, 2005.

[30] J. Li, C. Blake, D. D. Couto, H. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *ACM MobiCom*, pages 61–69, 2001.

[31] H. Lundgren, E. Nordstrom, and C. Tschudin. Coping with communication gray zones in ieee 802.11b based ad hoc networks. In *ACM WoWMoM*, pages 49–55, 2002.

[32] K. Seada, A. Helmy, and R. Govindan. On the effect of localization errors on geographic face routing in sensor networks. In *IEEE-ACM IPSN*, 2004.

[33] K. Seada, M. Zuniga, A. Helmy, and B. Krishnamacari. Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks. In *ACM SenSys*, 2004.

[34] A. Willig. A new class of packet- and bit-level models for wireless channels. In *IEEE PIMRC*, 2002.

[35] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *ACM SENSYS*, pages 14–27, 2003.

[36] F. Ye, G. Zhong, S. Lu, and L. Zhang. GRAdient Broadcast: A robust data delivery protocol for large scale sensor networks. In *IEEE IPSN*, 2003.

[37] H. Zhang, A. Arora, Y. R. Choi, and M. Gouda. Reliable bursty convergecast in wireless sensor networks. In *ACM MobiHoc*, 2005.

[38] H. Zhang, A. Arora, and P. Sinha. Learn on the fly: Data-driven link estimation and routing in sensor network backbones. Technical Report WSU-CS-DNC-07-TR1, Wayne State University (http://www.cs.wayne.edu/∼hzhang/group/TR/DNC-07-TR1.pdf), 2007.

[39] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *ACM SenSys*, pages 1–13, 2003.

[40] M. Zuniga and B. Krishnamachari. An analysis of unreliability and asymmetry in low-power wireless links. *ACM Transactions on Sensor Networks*, 3(2), 2007.