Learn on the Fly: Data-driven Link Estimation and Routing in Sensor Network Backbones

Hongwei Zhang Anish Arora Prasun Sinha
Computer Science and Engineering
The Ohio State University, USA
{zhangho, anish, prasun}@cse.ohio-state.edu

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 estimates link quality based on data traffic, and it chooses routes by way of a locally measurable metric ELD, the *expected MAC latency per unit-distance to the destination*. 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 existing 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 and enhances energy efficiency by a factor up to 2.37, which demonstrate the feasibility as well as potential benefits of data-driven link estimation and routing.**

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], [19], [28], estimating link quality is an important aspect of routing in wireless networks. Existing routing protocols [11], [12], [13], [23], [25] periodically exchange broadcast beacons between peers for link quality estimation. Nevertheless, 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 [10], [21]. Moreover, temporal correlations of link quality assume a complex pattern [24], which makes it even harder to precisely estimate unicast link quality via that of broadcast. Therefore, link quality estimated using periodic beacon exchange may not accurately apply for unicast data, which can negatively impact the performance of routing protocols.

In wireless sensor networks, a typical application 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 [26]. In this context, even if there were no discrepancy between the actual and the estimated link quality using periodic beacon exchange, the estimates tend to reflect link quality in the absence, rather than in the presence, of bursty data traffic. This is because: first, link quality changes significantly when traffic pattern changes (as we will show in Section II-B.2); and second, 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 estimation of link quality is not only limited in reflecting reality, 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 [26].

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 wireless links. The results show 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. Fortunately, we find that geography and the DATA-ACK handshake (available in the 802.11b MAC) make it possible to perform routing without using periodic beacons, in terms of information diffusion and data-driven link quality estimation respectively. To demonstrate the technique of data-driven link estimation and routing, we define a routing metric ELD, the *expected MAC latency per unit-distance to the* *destination*, which can be implemented in our 802.11 networks and works well in both our indoor testbeds and the large scale field experiment ExScal [6]. (Note: in principle, we could have used metrics such as ETX [11] or RNP [9] in data-driven routing, but this is not feasible given the existing 802.11 radios.)

To implement data-driven 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) which implements the ELD metric without using periodic beacons. In LOF, control packets are used only rarely, for instance, during the node boot-up. Upon booting up, a node initializes its routing engine by taking a few (e.g., 8) 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 switches its next-hop forwarder to another neighbor at controlled frequencies with a probability that this neighbor is actually the best forwarder.

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 existing protocols, represented by the geography-unaware ETX [11], [25] and the geography-based PRD [23]. We find 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. The results corroborate the feasibility as well as potential 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 the routing metric ELD 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 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, as well as 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.)

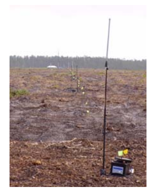
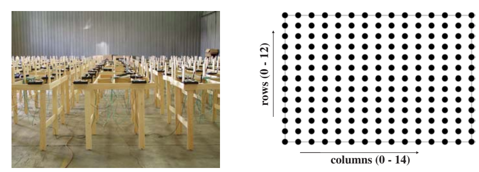


Fig. 1. Outdoor testbed

Indoor testbed. 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). 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.



(a) testbed (b) grid topology

Fig. 2. Indoor testbed

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.

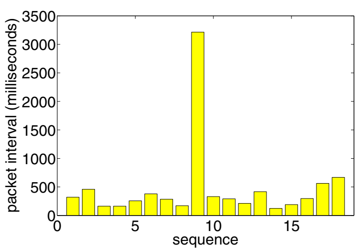


Fig. 3. The traffic trace of an ExScal event

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.)

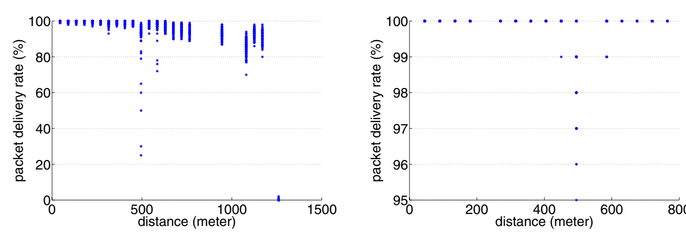
In the experiments, broadcast packets are transmitted at the basic rate of 1M bps, as specified by the 802.11b standard. Not focusing on the impact of packet rate in our study, we set unicast transmission rate to a fixed value (e.g., 5.5M bps). (We have tested different unicast transmission rates and 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 limitations, 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.

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:



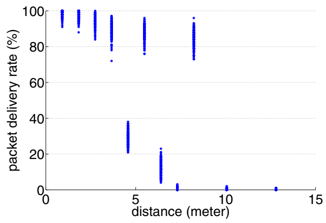
(a) broadcast (b) unicast

Fig. 4. Outdoor testbed

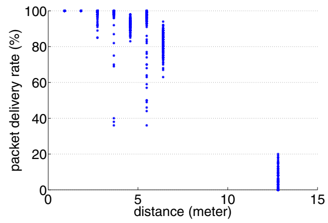
- Broadcast has longer communication range than unicast. This is due to the fact that the transmission rate for broadcast is lower, and that there is no RTS-CTS handshake for broadcast. (Note: the failure in RTS-CTS handshake also causes a unicast to fail.)
- For links where unicast has non-zero delivery 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 [9].)

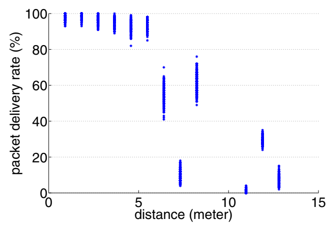
Similar results are observed in the indoor testbed, as shown in Figures 5(a) and 5(b). Nevertheless, there are exceptions



(a) broadcast: 1200-byte packet



(b) unicast: 1200-byte packet



(c) broadcast: 30-byte packet

Fig. 5. Indoor testbed

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.

2) Interfering scenarios: Figure 6 shows how the difference

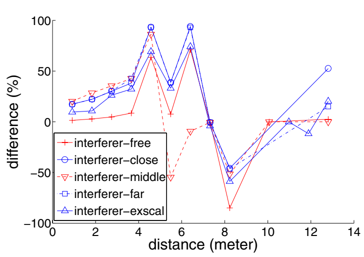
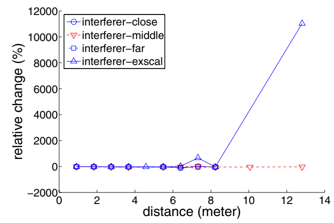


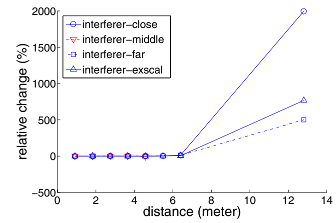
Fig. 6. The difference between broadcast and unicast in different interfering scenarios

between broadcast and unicast in the mean packet delivery rate changes as the interference and distance change. Given a distance and an interfering 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.

Figures 7 and 8 show the relative changes, when compared

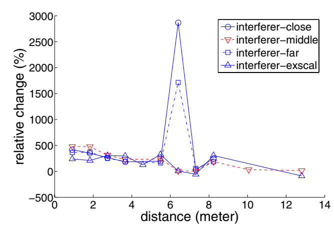


(a) broadcast

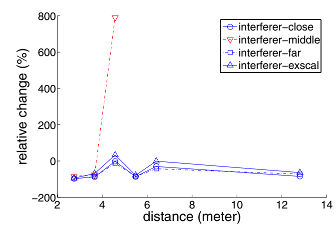


(b) unicast

Fig. 7. Relative change in packet delivery rate



(a) broadcast



(b) unicast

Fig. 8. Relative change in the COV of packet delivery rate

with the case of interferer-free, in packet delivery rate and its coefficient of variation (COV)[2] under different interfering scenarios. Given a distance and an interfering scenario, the relative change is calculated as $\frac{I-F}{F}$, where I and F denote the parameter value in the presence and in the absence of the interference respectively; if I or F is 0, we do not calculate the relative change since the value would be less meaningful. From the figures, we see that both the mean and the COV of packet delivery rate change significantly for broadcast when there is interference, yet the relative changes for unicast are much less. Moreover, the relative changes vary as interfering scenarios and distances vary.

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 [11] has explored

[2]COV is defined as the standard deviation divided by the mean [17].

this approach. Nevertheless, this approach may not be always feasible when the length of data packets is changing; or even if the approach is always 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 [9] (partly due to the difficulty of modeling the temporal correlations themselves [24]), which further decreases its estimation fidelity. For instance, Figure 9 shows the significant error in

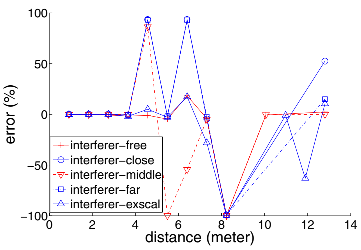


Fig. 9. Error in estimating unicast delivery rate via that of broadcast

estimating unicast delivery rate via that of broadcast under different interfering 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.

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. In this context, since we are not using beacons for link property estimation, we also explore the idea of not using periodic beacons in routing at all (i.e., beacon-free routing) to save energy; otherwise, beaconing requires nodes to wake up periodically even when there is no data traffic.

To enable data-driven routing, we need to find alternative mechanisms for accomplishing the tasks that are traditionally assumed by beacons: acting as the basis for link property estimation, and diffusing information (e.g., the cumulative ETX metric). In sensor network backbones, data-driven routing is feasible because of the following facts:

- **MAC feedback.** In MACs where every frame transmission is acknowledged by the receiver (e.g., in the 802.11b MAC), the sender can determine if a transmission has succeeded by checking whether it receives the acknowledgment. Also, the sender can determine how long each transmission takes, i.e., MAC latency. Therefore, the sender is able to get information on link properties without using any beacons. (Note: it has also been shown that MAC latency is a good routing metric for optimizing wireless network throughput [8].)
- **Mostly static network & geography.** Nodes are static most of the time, and their geographic locations are readily available via devices such as GPS. Therefore, we can use geography-based routing in which a node only needs to know the location of the destination and the information regarding its local neighborhood (such as the quality of the links to its neighbors). Thus, only the location of the destination (e.g., the base station in convergecast) needs to be diffused across the network. Unlike in beacon-based distance-vector routing, the diffusion happens infrequently since the destination is static most of the time. In general, control packets are needed only when the location of a node changes, which occurs infrequently.

In what follows, we first present the routing metric ELD which is based on geography and MAC latency, then we present the design of LOF which implements ELD 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 property estimation. Therefore, we do not recommend using these parameters as the core basis of data-driven routing, especially when senders need to precisely estimate in-situ link properties.
- Our objective in this paper is to explore the idea of data-driven link property estimation and routing, but it is not our objective to prove that geography-based routing is better than distance-vector routing. In principle, we could have used distance-vector routing together with data-driven link property estimation, but this would introduce extra control packets which we would like to avoid to save energy. (By the way, our study in our testbeds shows that geography-based data-driven routing has similar performance as that of distance-vector data-driven routing.)
- Conceptually, we could have also defined our routing metric based on other parameters such as ETX [11] or RNP [9]. Nevertheless, the firmware of our SMC WLAN cards does not expose information on the number of retries of a unicast transmission, which makes it hard to estimate ETX or RNP directly via data traffic. As a part of our future work, we plan to design mechanisms to estimate ETX and RNP via data traffic (e.g., in IEEE 802.15.4 based mote networks) and study the corresponding protocol performance.

III. ELD: THE ROUTING METRIC

In this section, we first formulate the routing metric ELD, the *expected MAC latency per unit-distance to the destination*, then we analyze the sample size requirement in routing.

A. A metric using MAC latency and geography

For convergecast in sensor networks (especially for event-driven applications), packets need to be routed reliably and in real-time to the base station. As usual, packets should also be delivered in an energy-efficient manner. Therefore, a routing metric should reflect link reliability, packet delivery latency, and energy consumption at the same time. One such metric that we adopt in LOF is based on MAC latency, i.e., the time taken for the MAC to transmit a data frame. (We have mathematically analyzed the relationship among MAC latency, energy consumption, and link reliability, and we find that MAC latency is strongly related to energy consumption in a positive manner, and the ratio between them changes only slightly as link reliability changes. Thus, routing metrics optimizing MAC latency would also optimize energy efficiency. Interested readers can find the detailed analysis in [27].)

Given that MAC latency is a good basis for route selection and that geography enables low frequency information diffusion, we define a routing metric ELD, *the expected MAC latency per unit-distance to the destination*, which is based on both MAC latency and geography. Specifically, given a sender S, a neighbor R of S, and the destination D as shown in Figure 10, 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[3]

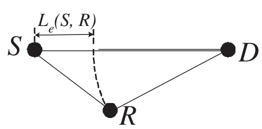


Fig. 10. L_e calculation

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

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

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

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. (Note: routing via metric ELD is a greedy approach, where each node tries to optimize the local objective. Like many other greedy algorithms, this method is effective in practice, as shown via experiments in Section V.)

To understand what ELD implies in practice, 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 11 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.

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

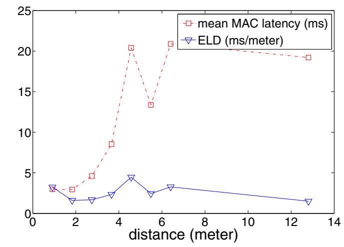


Fig. 11. Mean unicast MAC latency and the ELD

From this example, we see that, using metric ELD, a node tends to choose nodes beyond the reliable communication range as forwarders, to reduce end-to-end MAC latency as well as energy consumption.

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 [23], 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, $ELR(S,R) =$

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

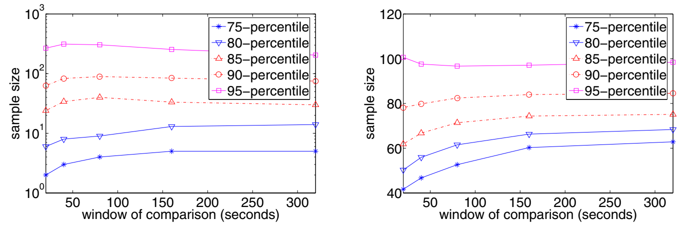
where $\lceil \frac{L_{S,R}+L_{R,D}}{L_{S,R}} \rceil$ denotes the number of hops to the destination, assuming equal geographic distance at every hop. We will show in Section V that ELR is inferior to ELD.

B. Sample size requirement

To understand the convergence speed of ELD-based routing and to guide protocol design, we have analyzed the sample size required to distinguish out the best neighbor in routing. (Due to the limitation of space, we relegate the detailed analysis to [27], and we only present the results here.)

Because link quality varies temporally, the best neighbor for a node may change temporally. Therefore, we compare links by periods of certain time span which we call *window of comparison* 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.

Remark. By way of contrast, we may 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 $\pm 5\%$.



(a) route selection (b) absolute ELD

Fig. 12. Sample size requirement

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 [25].

IV. LOF: A DATA-DRIVEN PROTOCOL

Having determined the routing metric ELD, we are ready to design protocol LOF for implementing ELD without using periodic beacons. 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.

Briefly speaking, LOF needs to accomplish two tasks: First, to enable a node to obtain the geographic location of the base station, as well as the IDs and locations of its neighbors; Second, to enable a node to track the LD (i.e., MAC latency per unit-distance to the destination) regarding each of its neighbors. The first task is relatively simple and only requires exchanging a few control packets among neighbors in rare cases (e.g., when a node boots up); LOF accomplishes the second task using three mechanisms: initial sampling of MAC latency, adapting estimation via MAC feedback for application traffic, and probabilistically switching next-hop forwarder.

In what follows, we elaborate on the individual components of LOF. (**Due to the limitation of space, we relegate to [27] the discussion on implementation issues of LOF: reliably fetching MAC feedback, reliable transport, node mobility, and neighbor-table size control**.)

A. Learning where we are

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

I. **[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.

II. **[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.

III. **[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.

IV. **[Announce presence]** When a node other than the base station finds a forwarder-candidate 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 II 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 needs to estimate the LDs regarding its neighbors. To design the estimation mechanism, let us first check Figure 13, which shows the

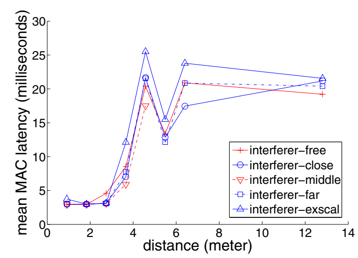


Fig. 13. MAC latency in the presence of interference

mean unicast MAC latency in different interfering scenarios for the indoor experiments described in Section II-A. We see that, even though MAC latencies change as interference pattern changes, the relative ranking in the mean MAC latency among links does not change much. Neither will the LDs accordingly.

In LOF, therefore, when a node S learns of the existence of a neighbor R for the first time, S takes a few samples of the MAC latency for the link to R before forwarding any data packets to R. The sampling is achieved by S sending a few unicast packets to R and then fetching the MAC feedback. The initial sampling gives a node a rough idea of the relative quality of the links to its neighbors, to jump start the data-driven estimation.

According to the analysis in Section III-B, another reason for initial sampling is that, with relatively small sample size, a node could gain a decent sense of the relative goodness of its neighbors. 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

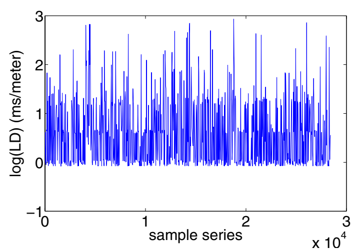


Fig. 14. A time series of $log(LD)$

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* [16] where the generating process is represented by a constant superimposed with random fluctuations. Therefore, a good estimation method is *exponentially weighted moving average* (EWMA) [16], assuming the following form

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

where V is the parameter to be estimated, 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 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 of R was last updated. Then, formula (4) is adapted to be the following:

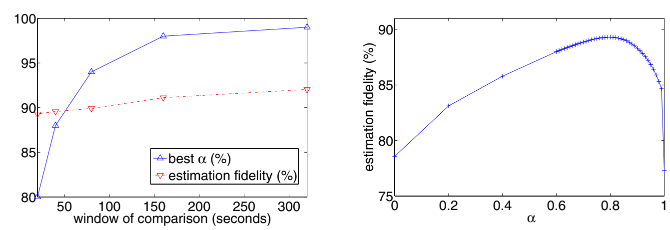
$$V \longleftarrow \alpha^{\beta(R)} V + (1-\alpha^{\beta(R)})V' \tag{5}$$

(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.

Another important issue in EWMA estimation is choosing the weight α, since it affects the stability and agility of estimation. To address this question, we again perform experiment-based analysis. Using the data from Section III-B, we try out different α values and compute the corresponding estimation fidelity, that is, the probability of LOF choosing the right next-hop forwarder for S. Figure 15(a) shows the best α value and the corresponding estimation fidelity for different windows of comparison. If the window of comparison is 20 seconds, for instance, the best α is 0.8, and the corresponding estimation fidelity is 89.3%. (Since the time span of the ExScal traffic trace is about 20 seconds, we set α as 0.8 in our experiments.)



(a) best α

(b) sensitivity analysis

Fig. 15. The weight α in EWMA

For sensitivity analysis, Figure 15(b) shows how the estimation fidelity changes with α when the window of comparison is 20 seconds. We see that the estimation fidelity is not very sensitive to changes in α over a wide range. For example, the estimation fidelity remains above 85% when α changes from 0.6 to 0.98. Similar patterns are observed for the other windows of comparison too. The insensitivity of estimation fidelity to α guarantees the robustness of EWMA estimation in different environments.

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. Probabilistic neighbor switching

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, 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 probabilistic neighbor switching 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 $P_{ns}(R')$. On the other hand, the probabilistic neighbor switching is exploratory and optimistic in nature, therefore it should be used only for good neighbors. In LOF, neighbor switching only considers the set of neighbors that are not marked as dead.

In what follows, we explain how to determine the switching probability $P_{ns}(R')$ and the switching 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.

Switching probability. At the moment of neighbor switching, 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, the probability $P_h(R_i)$ of a neighbor R_i being actually better than all the neighbors that ranks lower than itself, and the probability $P_{ns}(R_i)$ of a neighbor R_i being actually the best forwarder. Due to the limitation of space, we relegate the detailed derivation to [27].

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

$$I_{ns}(R_0) = C \times P_{ns}(R_0) \tag{6}$$

where C is a constant being equal to $(N \times K)$, with N being the number of active neighbors that S has, and K being a constant reflecting the degree of temporal variations in link quality. We set K to be 20 in our experiments.

The switching probabilities and the switching interval are re-calculated each time the next-hop forwarder is changed.

V. Experimental evaluation

Via testbeds and field deployment, we experimentally evaluate the design decisions and the performance of LOF. First, we present the experiment design; then we discuss the experimental results.

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, as well as scenarios where the data traffic is periodic; LOF has also been used in the backbone network of ExScal. Due to the limitation of space, we relegate the details to [27].

Protocols studied. We study the performance of LOF in comparison with that of beacon-based routing, where the latest development is represented by ETX [11], [25] and PRD [23]: (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 [13], where a route with the minimum *expected transmission time* is used. ETT is similar to *MAC latency* as used in 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. The design of PRD is based on the analysis that assumes geographic-uniformity.

By their original proposals, ETX and PRD use broadcast beacons in estimating the respective routing metrics. In this paper, we compare the performance of LOF with that of ETX and PRD as originally proposed in [11] and [23], without considering the possibility of directly estimating metrics ETX and PRD via data traffic. This is because the firmware of our SMC WLAN cards does not expose information on the number of retries of a unicast transmission. (As a part of our future work, we plan to design mechanisms to estimate ETX and PRD via data traffic and study the corresponding protocol performance.) In our experiments, metrics ETX and PRD are estimated according to the method originally proposed in [11] and [23]; for instance, broadcast beacons have the same packet length and transmission rate as those of data packets. Since it has been shown that ETX and PRD perform better than protocols based on metrics such as RTT (round-trip-time) and hop-count [12], [23], we do not study those protocols in this paper.

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 probabilistic neighbor switching;
- *L-sd*: considers, in probabilistic neighbor switching, the neighbors that have been marked as dead;
- *L-se*: performs probabilistic neighbor switching after every packet transmission.

(We have also studied the performance of geography-unaware distance-vector routing using data-driven estimation trying to minimize the sum of MAC latency along routes, and we found that the performance is similar to that of LOF, except that more control packets are used.)

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 [27], 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 [11], [13].
- *Energy efficiency*: energy spent in delivering a packet to the base station.

B. Experimental results

MAC latency. Using boxplots, Figure 16 shows the end-to-end 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 switching technique.

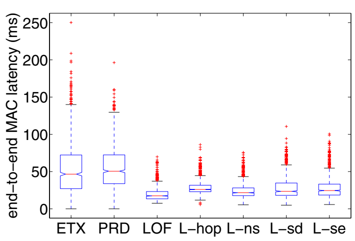


Fig. 16. End-to-end MAC latency

To explain the above observation, Figures 17, 18, 19, and 20 show the route hop length, per-hop MAC latency, average per-hop geographic distance, and the coefficient of variation (COV) of 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.

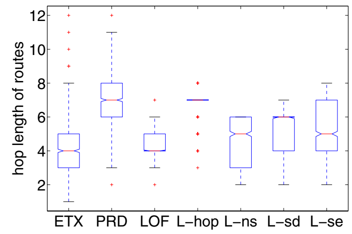


Fig. 17. Number of hops in a route

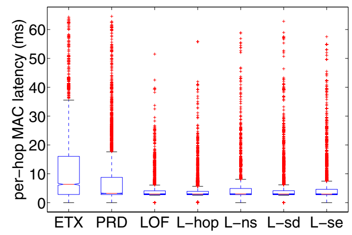


Fig. 18. Per-hop MAC latency

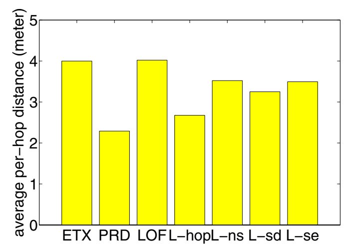


Fig. 19. Average per-hop geographic distance

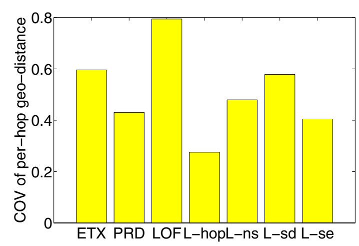


Fig. 20. COV of per-hop geographic distance in a route

From Figure 20, we see that the COV of per-hop geographic distance is as high as 0.4305 in PRD and 0.2754 in L-hop. Therefore, the assumption of geographic uniformity is invalid, which partly explains why PRD and L-hop do 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 21 shows the

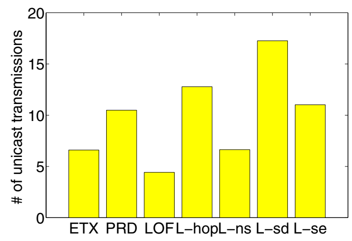


Fig. 21. Number of unicast transmissions per packet received

number of unicast transmissions averaged over the number packets received at the base station. The number of unicast transmissions per packet received 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. The number of unicast transmissions per packet received in LOF is also less than that in the other versions of LOF. For instance, the number of unicast transmissions in L-hop is 2.89 times that in LOF.

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 21 does not represent the actual number of bytes sent. Nevertheless, given Figure 18 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 22 shows the

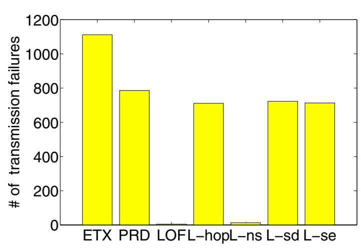


Fig. 22. Number of failed unicast transmissions

number 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 19 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.

VI. Related work

The literature on routing in ad hoc and wireless networks is quite rich. In this section, we only review those related most closely to LOF.

Link properties in 802.11b mesh networks and dense wireless sensor networks have been well studied in [7], [19], and [28]. 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 [21] and [10]. Our work complements [21] and [10] 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 [21] and [10]. [10] mentioned the difficulty of getting MAC feedback and thus focused on the method of beacon-based link estimation. Our work complements [10] 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, [21] and [10] 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 [14], [11], [25], we do not directly compare LOF to [21] and [10], 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 [11], [25] and ETT/WCETT [13] have been proposed and shown to perform well in real-world wireless networks [12]. The geography-based metric PRD [23] 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 the data-driven protocol LOF where unicast link properties are estimated via the data traffic itself.

Similar to LOF, SPEED [15] also used MAC latency and geographic information in route selection. In parallel with our work, [20] proposed NADV which also uses information from MAC layer. While focusing on real-time packet delivery and a general framework for geographic routing, [15] and [20] did not focus on the protocol design issues in data-driven link estimation and routing. They did not study the differences between broadcast and unicast link properties. They did not consider the importance of appropriate *probabilistic neighbor switching* either. SPEED switches next-hop forwarders after every packet transmission (as in L-se), and NADV does not perform probabilistic neighbor switching (as in L-ns), both of which degenerate network performance as shown in Section V. Complementary to SPEED and NADV, moreover, we have analyzed the small sample size requirement in LOF, which shows the feasibility of data-driven link estimation. While [15] and [20] have evaluated their methods via simulation, we have studied the systems issues in reliably fetching MAC feedback and evaluated LOF via experiments in real networks with realistic traffic trace. Finally, [20] did not compare the performance of NADV with that of ETX and PRD.

The problem of local minimum or geographic void has been dealt with in routing protocols such as GPSR [18]. 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 [22]. 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 probabilistic neighbor switching. 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 [9]. Given the limitation of our 802.11 radios, we have not applied the technique of data-driven link estimation to metrics such as ETX [11] or RNP [9]. We plan to explore these directions in our future work.

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. LOF also helps in enhancing network security, since the network is less exposed. More detailed study of the impact of data-driven routing on energy efficiency and security is a part of our future work.

Acknowledgment

This work was sponsored by DARPA contract OSU-RF #F33615-01-C-1901. We thank Vinayak Naik and Emre Ertin for their help in the discussion and experimentation. We appreciate the help from UCLA EmStar team for their help in answering questions regarding EmStar. We also thank the anonymous referees for their comments on 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] B. Awerbuch, D. Holmer, and H. Rubens. High throughput route selection in multi-rate ad hoc wireless networks. Technical report, Johns Hopkins University, 2003.
[9] 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.
[10] I. Chakeres and E. Belding-Royer. The utility of hello messages for determining link connectivity. In *WPMC*, 2002.
[11] 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.
[12] R. Draves, J. Padhye, and B. Zill. Comparison of routing metrics for static multi-hop wireless networks. In *ACM SIGCOMM*, pages 133–144, 2004.
[13] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *ACM MobiCom*, pages 114–128, 2004.
[14] 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.
[15] T. He, J. Stankovic, C. Lu, and T. Abdelzaher. SPEED: A stateless protocol for real-time communication in sensor networks. In *IEEE ICDCS*, 2003.
[16] F. Hillier and G. Lieberman. *Introduction to Operations Research*. McGraw-Hill, 2001.
[17] R. Jain. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, Inc., 1991.
[18] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *ACM MobiCom*, pages 243–254, 2000.
[19] D. Kotz, C. Newport, and C. Elliott. The mistaken axioms of wireless-network research. Technical Report TR2003-467, Dartmouth College, Computer Science, July 2003.
[20] S. Lee, B. Bhattacharjee, and S. Banerjee. Efficient geographic routing in multihop wireless networks. In *ACM MobiHoc*, pages 230–241, 2005.
[21] 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.
[22] 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.
[23] 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.
[24] A. Willig. A new class of packet- and bit-level models for wireless channels. In *IEEE PIMRC*, 2002.
[25] 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.
[26] H. Zhang, A. Arora, Y. R. Choi, and M. Gouda. Reliable bursty convergecast in wireless sensor networks. In *ACM MobiHoc*, 2005.
[27] H. Zhang, A. Arora, and P. Sinha. Learn on the fly: Quiescent routing in sensor network backbones. Technical Report OSU-CISRC-7/05-TR48, The Ohio State University (http://www.cse.ohio-state.edu/∼zhangho/publications/LOF-TR.pdf), 2005.
[28] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *ACM SenSys*, pages 1–13, 2003.