Learn on the Fly: Quiescent Routing in Wireless Sensor Networks

Hongwei Zhang, Anish Arora, Prasun Sinha
Technical Report: OSU-CISRC-4/05-TR20
The Ohio State University, USA

Abstract

In IEEE 802.11b network testbeds, we examine the drawbacks of estimating unicast link properties via those of broadcast packets. To redress these shortcomings, we design a beacon-free routing protocol *Learn on the Fly* (LOF). It chooses routes without assuming geographic uniformity by way of a locally measurable metric ELD, the *expected MAC latency per unit-distance toward the destination*; it estimates link quality solely based on data traffic without employing 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 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, enhances energy efficiency by a factor up to 2.37, and improves route stability by 2 orders of magnitude.

***Keywords*—experiment-based design and analysis, bursty convergecast, beacon-free geographic routing, data-driven link quality estimation, MAC latency, IEEE 802.11b, real time, energy, reliability**

1 Introduction

As the quality of wireless links, for instance, packet delivery rate, varies both temporally and spatially in a complex manner [6, 17, 23], estimating link quality is an important aspect of routing in wireless networks. Existing routing protocols [8, 9, 10, 19, 21] 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 [7, 18]. 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 [22]. In this context, even if there were no discrepancy between the actual and the estimated link quality using periodic beacon exchange, the estimate would 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 2.2.2). 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 the 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 [22].

To deal with the drawbacks 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 the assumptions of beacon-based routing do not hold well, and that we need to directly estimate unicast link quality without using broadcast beacons.

Fortunately, we find that geography and the DATA-ACK handshake (available in the 802.11b MAC) make beacon-free routing possible in terms of information diffusion and beacon-free link quality estimation. Using geography and MAC-layer transmission latency (simply referred to as MAC latency hereafter), we define a routing metric ELD, the *expected MAC latency per unit-distance toward the destination*. ELD is locally measurable and does not assume *geographic uniformity* — that the hops in any route have approximately the same geographic length . Via experiment-based analysis, we find that MAC latency has a *log-normal* distribution, and that the sample size required for ELD-based routing is relatively small (e.g., the 80-percentile is only 8). We also mathematically show that MAC latency is a good indicator of energy consumption in packet transmission.

To enable beacon-free routing, we modify the Linux kernel and the WLAN driver *hostap* [3] 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 in a beacon-free manner. In LOF, control packets are used only rarely, for instance, during the node boot-up. Upon booting up, a node ini-

This work was sponsored by DARPA contract OSU-RF #F33615-01-C-1901.

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

tializes 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 latency 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 [5], we experimentally evaluate the design and the performance of LOF in a testbed of 195 Stargates [1] with 802.11b radios. For instance, we investigate the validity of *geographic uniformity* which is assumed in literature [19], and we find that it usually does not hold and leads to inferior performance. We also compare the performance of LOF with that of existing protocols, represented by the geography-unaware ETX [8, 21] and the geography-based PRD [19]. We find that LOF, apart from guaranteeing 100% packet delivery, 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. As a side result, we compare the performance of ETX with that of PRD, which has not been done in the literature.

Organization of the paper. In Section 2, we study the shortcomings of beacon-based routing, and we analyze the feasibility of beacon-free routing. Following that, we present the routing metric ELD in Section 3, and we design the protocol LOF in Section 4. We experimentally evaluate LOF in Section 5, and we discuss the related work in Section 6. We make concluding remarks in Section 7.

2 Why beacon-free routing

Routing protocols that estimate link properties using periodic beacons implicitly assume the following:

I. Link properties estimated via broadcast beacons apply to unicast data packets;

II. Link properties experienced by periodic beacons reflect those in the presence of data traffic.

These assumptions are usually invalid in wireless networks, even though they hold well in wireline networks where links are reliable.

In this section, we experimentally investigate the invalidity of these assumptions in wireless networks, then we discuss the feasibility of beacon-free routing.

2.1 Experiment design

To check the validity of assumptions I and II and to study the impact of packet type, packet length, and interference on link properties, we set up two 802.11b network testbeds as follows.

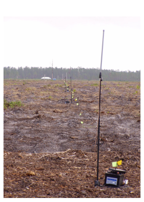
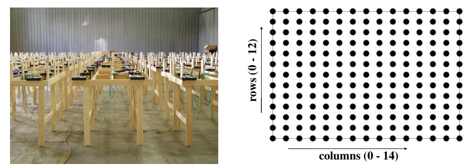


Figure 1: Outdoor testbed

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[1] of each Stargate is set as 35.

Indoor testbed. In an open warehouse (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, 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.



(a) testbed (b) grid topology

Figure 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 checking, in Section 2.2.1, 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. This experiment is designed to invalidate assumption I.

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.

[1] A tunable parameter for 802.11b wireless cards. The range for the power level is 127, 126, ..., 0, 255, 254, ..., 129, 128, with 127 being the lowest and 128 being the highest.

- *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 [5] . The traffic trace corresponds to the packets generated when a vehicle passes across a section of the 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.

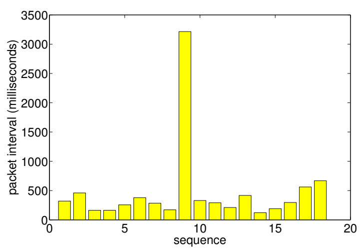


Figure 3: The traffic trace of an ExScal event

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. The above experiments are designed to invalidate both assumptions I and II. (Note that the scenarios above are far from being complete, but they would 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 and, without loss of generality, we configure the unicast transmission rate to be 11Mbps. For other 802.11b configurations, we use the default parameter values that come with the system software. For instance, each unicast packet is retransmitted up to 7 times until success or failure in the end.

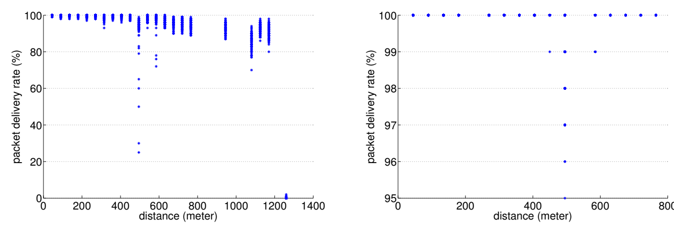
2.2 Experimental results

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 first present the difference between broadcast and unicast when there is no interference, then we present the impact of interference.

2.2.1 Interferer free

Figure 4 shows the scatter plot of the delivery rates for broadcast



(a) broadcast

(b) unicast

Figure 4: Outdoor testbed

and unicast packets at different distances in the outdoor testbed. From the figure, we observe the following:

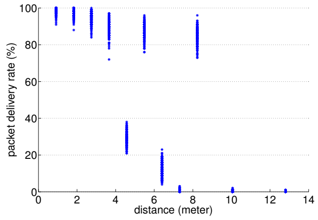
- Broadcast has longer communication range than unicast. This is due to the fact that the transmission rate for broadcast is lower, and that there is no RTS-CTS handshake for broadcast.
- For links where unicast has non-zero delivery rate, the mean delivery rate of unicast is higher than that of broadcast. 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.

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.

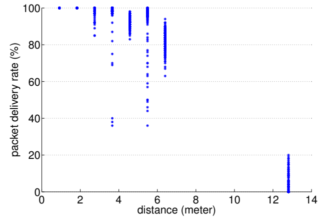
Implication. From Figures 4 and 5, we see that packet delivery rate differs significantly between broadcast and unicast, and the difference varies with environment, hardware, and packet length. Therefore, assumption I does not hold well even when there is no interference.

2.2.2 Interfering scenarios

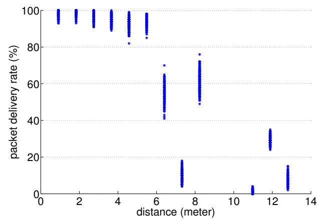
Figure 6 shows how the difference between broadcast and uni-



(a) broadcast: 1200-byte packet



(b) unicast: 1200-byte packet



(c) broadcast: 30-byte packet

Figure 5: Indoor testbed

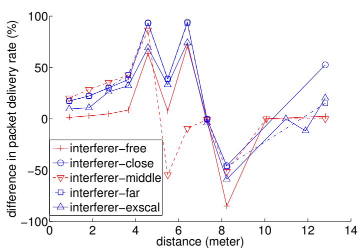
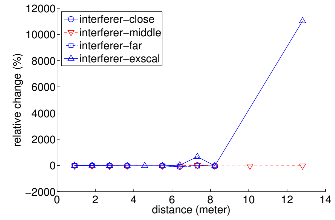
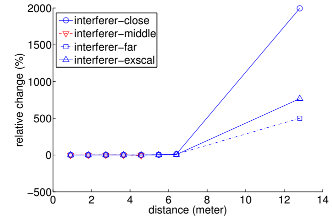


Figure 6: The difference between broadcast and unicast in different interfering scenarios

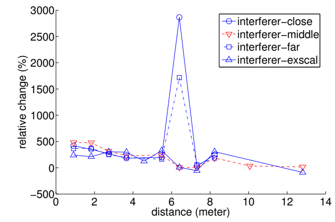


(a) broadcast

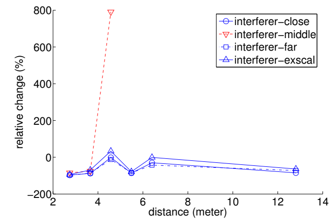


(b) unicast

Figure 7: Relative change in packet delivery rate



(a) broadcast



(b) unicast

Figure 8: Relative change in the COV of packet delivery rate

cast 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. This observation further demonstrates the invalidity of assumption I in wireless networks.

Figures 7 and 8 show the relative changes, when compared 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.

[2]COV is defined as the standard deviation divided by the mean [15].

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. Thus assumption II does not hold well. Moreover, the difference between broadcast and unicast changes as interference pattern changes.

2.3 Beacon-free routing

To estimate unicast link properties in spite of the difference between broadcast and unicast, there are two alternatives: estimating unicast link properties based on those of broadcast beacons but compensate for the difference; or estimating unicast link properties directly. From Section 2.2, we see that the difference assumes complex patterns, as factors such as environment, packet length, and interference pattern change. Therefore, it is not trivial, if even possible, to mathematically model the difference and to compensate for it in estimation. To circumvent this difficulty, we adopt the second approach, that is, directly estimating unicast link properties without using beacons.

Traditionally, beacons serve two major roles: diffusing information (e.g., the expected number of transmissions to the base station) and acting as the basis for link quality estimation. To enable beacon-free routing, there must be mechanisms performing these two tasks. In wireless sensor networks, beacon-free routing is enabled by the following facts:

- 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 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[3]. Also, the sender can determine how long each transmission takes (as to be explained in detail in Section 4.5), i.e., MAC latency. Therefore, the sender is able to get information on link quality without using any beacons.

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 in a beacon-free manner.

3 ELD: the routing metric

In this section, we first justify mathematically why MAC latency reflects link reliability and energy consumption, then we derive the routing metric ELD, the *expected MAC latency per unit-distance toward the destination*, and finally we analyze the consequent sample size requirement in routing.

3.1 MAC latency as the basis for route selection

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).

Intuitively, both the MAC latency and the energy consumption of a frame transmission depend on the link reliability. Therefore MAC latency certainly reflects link reliability and energy consumption. But to precisely characterize their relationships, we mathematically analyze them as follows, using 802.11b as an example. (Readers unfamiliar with the details of 802.11b could refer to [4], or simply skip the mathematical formulation and only check for the *pictorial representation*.)

Given a sender S and a receiver R where the link between them has non-zero reliability, we let $D_{S,R}$ and $P_{S,R}$ denote the MAC latency and the energy consumption for transmitting a unicast frame from S to R. Then, we are interested in calculating the expected values $E(D_{S,R})$ and $E(P_{S,R})$. For simplicity, we only consider the case where there is no interfering traffic, and we assume that the MAC continues to transmit a packet until it is successful. Let p_0 be the probability that a RTS-CTS handshake between S and R will fail (e.g., due to the loss of RTS or CTS) ($p_0 > 0$), p_1 be the probability that a DATA-ACK handshake between S and R will fail ($p_1 > 0$), and $C = p_0 + p_1 - p_0 p_1$. Then, we have

$$E(D_{S,R}) = (1-p_0)(1-p_1)(464+t_d) + (\tfrac{1-p_1}{p_1})T \quad (\mu s) \qquad (1)$$

where

$$\begin{aligned} t_d &= \text{time taken to transmit the DATA frame (in microseconds);} \\ T &= (502 + \tfrac{(134+2t_d)(1-p_0)p_1}{C})\tfrac{2C^2-C^3}{(1-C)^2} + \\ &\quad \tfrac{158720C^8-138880C^9}{(1-C)^2} + \tfrac{(-t_d-482)C^2}{C} - \\ &\quad \tfrac{119040C^8}{1-C} - \tfrac{1004p_0^2-502p_0^3}{(1-p_0)^2} + \\ &\quad \tfrac{-158720p_0^8+138880p_0^9}{(1-P_0)^2} + \tfrac{(t_d+482)p_0^2}{1-p_0} + \\ &\quad \tfrac{119040p_0^8}{1-p_0} + 155\sum_{k_0=2}^{7}((2C)^{k_0} - (2p_0)^{k_0}). \end{aligned}$$

Derivation sketch for formula (1). Assume a frame transmission from S to R takes k_1 rounds of DATA-ACK handshakes and k_0 rounds of RTS-CTS handshakes. Clearly, $k_0 \geq k_1$. Then, the MAC latency $D_{S,R}(k_0, k_1)$ can be decomposed into the following three components:

- The latency $I(k_0, k_1)$ due to the initial DIFS before any RTS-CTS-DATA-ACK handshake. We have $I(k_0, k_1) = DIFS\ (\mu s)$.
- The latency $CB(k_0, k_1)$ due to the contention avoidance backoffs: there are $(k_0 - k_1)$ RTS-CTS handshake failures, $(k_1 - 1)$ DATA-ACK handshake failures, and $(k_0 - k_1) + (k_1 - 1) = k_0 - 1$ contention backoffs. Therefore, we have

$$\begin{aligned} CB(k_0,k_1) &= (k_0-k_1)(CTSTimeout + DIFS) + \\ &\quad (k_1-1)(ACKTimeout + DIFS) + \\ &\quad \textstyle\sum_{i=1}^{k_0-1} BT_i \end{aligned}$$

  where $CTSTimeout = t_{rts} + t_{cts} + 2SIFS$, $ACKTimeout = t_d + t_{ack} + SIFS + DIFS$, and BT_i is the value of the i^{th} contention backoff timer.
- The latency $DT(k_0, k_1)$ due to the normal RTS-CTS-DATA-ACK procedure. We have

$$\begin{aligned} DT(k_0,k_1) &= (k_0-k_1)t_{rts} + \\ &\quad k_1(t_{rts} + SIFS + t_{cts}) + \\ &\quad (SIFS + t_d + SIFS + t_{ack}) \quad (\mu s) \end{aligned}$$

  where t_{rts}, t_{cts}, and t_{ack} denote the time taken to transmit a RTS, a CTS, and an ACK respectively.

Therefore, we have

$$\begin{aligned} D_{S,R}(k_0,k_1) &= I(k_0,k_1) + CB(k_0,k_1) + DT(k_0,k_1) \\ &= k_0(DIFS + t_{rts} + CTSTimeout) + \\ &\quad k_1(-CTSTimeout + ACKTimeout + \\ &\quad t_{cts} + 3SIFS + t_d + t_{ack}) - \\ &\quad ACKTimeout + \textstyle\sum_{i=1}^{k_0-1} BT_i \end{aligned}$$

Now, let us calculate the probability $P(k_0, k_1)$ that a transmission from S to R takes k_0 rounds of RTS-CTS handshake and k_1 rounds of DATA-ACK handshake. We have $P(k_0, k_1) =$

$$\begin{aligned} & P\{(k_0-k_1) \text{ RTS-CTS failure out of } k_0 \text{ times}\}\times \\ & P\{(k_1-1) \text{ DATA-ACK failures followed by a success}\} \\ &= \begin{cases} (\binom{k_0}{(k_0-k_1)} p_0^{k_0-k_1}(1-p_0)^{k_1}) \times (p1^{k_1-1}(1-p_1)) & \text{if } k_0 \geq k_1 \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} \binom{k_0}{k_1}(\frac{1-p_1}{p_1})p_0^{k_0}(\frac{(1-p_0)p_1}{p_0})^{k_1} & \text{if } k_0 \geq k_1 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

[3]Even though this method is not perfect when an acknowledgment frame can get lost, it works well in practice given the low probability of losing an acknowledgment frame.

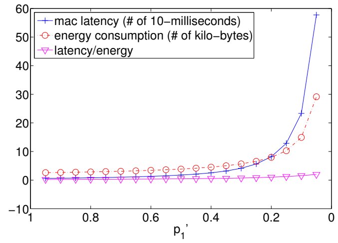


Figure 9: MAC latency as an indicator of energy consumption

Therefore, we have

$$\begin{aligned} E(D_{S,R}) &= E_{k_0,k_1}(D_{S,R}(k_0,k_1)) \\ &= \sum_{k_0=1}^{\infty}\sum_{k_1=1}^{k_0} D_{S,R}(k_0,k_1)P(k_0,k_1) \end{aligned}$$

Applying 802.11b parameters, such as t_{rts} and $SIFS$, to the above formula, we could arrive at formula (1). Due to the limitation of space, we skip the detail here.

□

For energy consumption, we have

$$E(P_{S,R}) = \frac{1-p_1}{p_1(1-C)^2}(34C + (l_d + 14)p_1(1-p_0)(2-C)) \quad (bytes) \tag{2}$$

where

$$l_d = \text{the length of the DATA frame (in number of bytes).}$$

Derivation sketch for formula (2). Let k_0, k_1, and $P(k_0, k_1)$ be the same as in the "derivation sketch for formula (1). Let $B_{rts-cts}$ be the length, in number of bytes, of a RTS-CTS pair, and $B_{data-ack}$ be the length of a DATA-ACK pair. Then, if a transmission from S to R takes k_0 rounds of RTS-CTS handshakes and k_1 rounds of DATA-ACK handshakes ($k_0 \geq k_1$), the energy $P_{S,R}(k_0, k_1)$ consumed, in number of bytes transmitted, is $k_0B_{rts-cts} + k_1B_{data-ack}$.

Therefore, we have

$$\begin{aligned} E(P_{S,R}) &= E_{k_0,k_1}(P_{S,R}(k_0,k_1)) \\ &= \sum_{k_0=1}^{\infty}\sum_{k_1=1}^{k_0} P_{S,R}(k_0,k_1)P(k_0,k_1) \end{aligned}$$

Applying 802.11b parameters, such as $B_{rts-cts}$, to the above formula, we could arrive at formula (2). Due to the limitation of space, we skip the detail here.

□

To draw Formulas (1) and (2), we let the probability p_1' that a DATA-ACK handshake will succeed represent link reliability (i.e., $p_1' = 1 - p_1$). Letting $k = \frac{\text{length(DATA+ACK)}}{\text{length(RTS+CTS)}}$, and assuming that bit errors are independent, we have $p_1 = 1 - (1 - p_0)^k$. Thus,

$$p_0 = 1 - \sqrt[k]{1 - p_1} = 1 - \sqrt[k]{p_1'} \tag{3}$$

Based on equations (1), (2), (3), and assuming that t_d is 1200, Figure 9 presents a visual characterization of the expected MAC latency, the expected energy consumption, and the ratio between them, as link reliability changes.

From Figure 9, we see 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. Note that, as link reliability becomes too low, the rate of increase in MAC latency is slightly faster compared to energy consumption. This is because the contention window for the random backoff in MAC increases exponentially. In practice, however, this scenario may not happen, because extremely low link reliability only leads to transmission failures due to the upper limit on the number of retries (whose default value is 7).

Previous work has argued the advantages of using routing metrics Expected Transmission Count (ETX) [21, 8] and Expected Transmission Time (ETT) [10, 9], which are similar to MAC latency, from perspectives such as reducing self-interference and increasing throughput. Our analysis complements theirs by mathematically showing the relationships among MAC latency, energy consumption, and link reliability.

3.2 ELD: a geography-based routing metric

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 toward 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 toward the destination* (LD) via R, denoted by $LD(S, R)$, as[4]

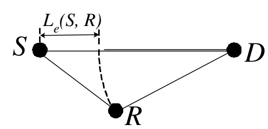


Figure 10: Calculate L_e

$$\begin{cases} \frac{D_{S,R}}{L_e(S,R)} & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \tag{4}$$

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 calculates as

$$\begin{cases} \frac{E(D_{S,R})}{L_e(S,R)} & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \tag{5}$$

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. Because MAC latency and LD are lognormally distributed (to be discussed in Section 3.3), the ranks are compared via their logarithmic values in protocol LOF.

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

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

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. 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.

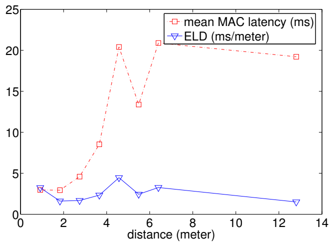


Figure 11: Mean unicast MAC latency and the ELD

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 literature of geographic routing [19], however, a common assumption is that the hops in any route have similar properties such as geographic length and link quality. As we will show by experiments in Section 5, 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, $ELR(S,R) =$

$$\begin{cases} E(D_{S,R}) \times \lceil \frac{L_{S,R}+L_{R,D}}{L_{S,R}} \rceil & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \quad (6)$$

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 5 that ELR is inferior to ELD.

3.3 Sample size requirement

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 12 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 [14] on the three models, and we find that *lognormal* distribution fits the data the best.

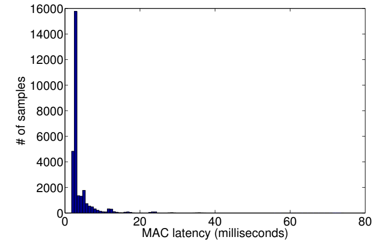


Figure 12: Histogram for unicast MAC latency

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.

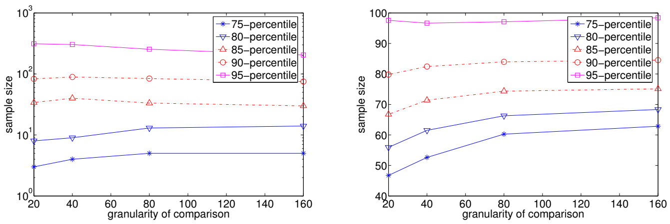
Because link quality varies temporally, the best neighbor for S may change temporally. Therefore, we divide the 30,000 MAC latency samples of each link into chunks of length L_c, denoted as the *granularity of comparison*, and we compare all the links via their corresponding sample-chunks. Given each 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 ($i = 1, 2, ..., \lceil \frac{30000}{L_c} \rceil$), 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 [15].

In the end, we have a set of sample sizes for each specific L_c. For a 95% confidence level comparison and route selection, Figure 13(a) shows the 75-, 80-, 85-, 90-, and 95-percentiles of the sample sizes for different L_c's. We see that the percentiles do not change much as L_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 3 and 8 respectively when L_c is 20), 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 4.2.

Remark. By way of contrast, we may also compute the sample size required to estimate the absolute ELD value associated with each neighbor. Figure 13(b) shows the percentiles for a 95% confidence level estimation with an accuracy of ±5%. 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 47 and 56 respectively when L_c is 20) are significantly greater than those for route selection. Therefore, when analyzing sample size re-

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



(a) route selection

(b) absolute ELD

Figure 13: Sample size requirement

quirement 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 [21].

4 LOF: the beacon-free protocol

Having determined the routing metric ELD, we are ready to design the protocol LOF for implementing ELD in a beacon-free manner. 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, enabling a node to obtain the geographic location of the base station, as well as the IDs and locations of its neighbors; Second, enabling a node to track the LD (i.e., MAC latency per unit-distance toward 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 this section, we first elaborate on the individual components of LOF, then we discuss implementation issues of LOF such as reliably fetching MAC feedback.

4.1 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. 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.

4.2 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 14, which shows the mean unicast

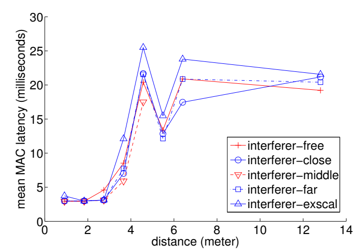


Figure 14: MAC latency in the presence of interference

MAC latency in different interfering scenarios for the indoor experiments described in Section 2.1. 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 samples the MAC latency of the link to R before forwarding any data packets to R. The sampling is achieved by S sending 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 3.3, 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 8 (i.e., the 80-percentile of the sample size when L_c is 20) in our experiments.

4.3 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.

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 3.3. Figure 15 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* [13] where the generating process is represented by a constant superimposed with random fluctuations. Therefore, a good estimation method is *exponentially weighted moving average* (EWMA) [13], assuming the following form

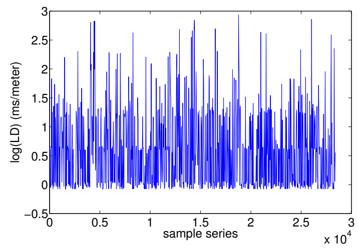


Figure 15: A time series of $log(LD)$

$$V \longleftarrow \alpha V + (1-\alpha)V' \tag{7}$$

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 (7) 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 (7) is adapted to be the following:

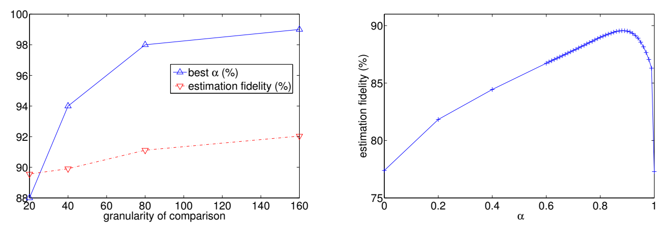
$$V \longleftarrow \alpha^{\beta(R)} V + (1-\alpha^{\beta(R)})V' \tag{8}$$

(Through experiments, we observe that formula (8) endows LOF better performance than formula (7).)

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 (8) 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 (8), 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 (8) to get a new estimation.

Another key issue in the EWMA estimation is choosing the right 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 3.3, 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 16(a) shows the best α value and the corresponding estimation fidelity for different granularities of comparison. If the granularity of comparison is 20, for instance, the best α is 0.88, and the corresponding estimation fidelity is 89.56%. (Since the ExScal traffic trace contains 19 packets, we set α as 0.88 in our experiments.)



(a) best α (b) sensitivity analysis

Figure 16: The weight α in EWMA

For sensitivity analysis, Figure 16(b) shows how the estimation fidelity changes with α when the granularity of comparison is 20. 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 granularities of comparison too. The insensitivity of estimation fidelity to α guarantees the robustness of the estimation method. Hence we may not need to change α when network environment changes.

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 [5]. Again, the withdrawing and rejoining process can be optimized, but we skip the details here.)

4.4 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 (given by formula (9)), 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 (10)), and the probability $P_{ns}(R_i)$ of a neighbor R_i being actually the best forwarder (given by formula (11)).

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 3.3, $log(LD(S,R_i))$ 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

$$P_b(R_i, R_j) = G\left(\frac{\mu_j - \mu_i}{\sqrt{\delta_i^2 + \delta_j^2}}\right) \qquad (9)$$

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

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

Derivation sketch for formula (9). Since $log(LD(S,R_i)) \sim N(\mu_i, \delta_i^2)$, $log(LD(S,R_j)) \sim N(\mu_j, \delta_j^2)$, and $log(LD(S,R_i))$ is independent of $log(LD(S,R_j))$, it is easy to show that $Z' = log(LD(S,R_i)) - log(LD(S,R_j)$ is a normal variate with mean $(\mu_i - \mu_j)$ and variance $(\delta_i^2 + \delta_j^2)$. Therefore, $Z = \frac{Z' - (\mu_i - \mu_j)}{\sqrt{\delta_i^2 + \delta_j^2}}$ is a standard normal variate. Thus,

$$\begin{aligned} P_b(R_i, R_j) &= P\{log(LD(S,R_i)) > log(LD(S,R_j))\} \\ &= P[Z' > 0] \\ &= P\left[Z > \frac{-(\mu_i - \mu_j)}{\sqrt{\delta_i^2 + \delta_j^2}}\right] \end{aligned}$$

By Andrew's method [20], $P\left[Z > \frac{-(\mu_i - \mu_j)}{\sqrt{\delta_i^2+\delta_j^2}}\right] = G\left(\frac{\mu_j - \mu_i}{\sqrt{\delta_i^2+\delta_j^2}}\right)$, where $G(x) = 1 - \Phi(x)$, with

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

□

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:

$$\begin{aligned} P_h(R_1) &= P_b(R_1, R_0); \\ P_h(R_i) &\approx P_b(R_i, R_0) \times \prod_{j=1}^{i-1}(1 - (P_b(R_j, R_i) + (P_h(R_j) - 1) \times P_b(R_0, R_i))) \\ & \qquad (i = 2, \ldots, N) \end{aligned} \qquad (10)$$

Derivation sketch for formula (10). Let $P_b(\langle R_{k_0}, R_{m_0}\rangle, \ldots, \langle R_{k_l}, R_{m_l}\rangle)$ denote the probability that R_{k_0} is better than R_{m_0}, …, and R_{k_l} is better than R_{m_l}, and let $P_b(\langle R_i, R_j\rangle | \langle R_{k_0}, R_{m_0}\rangle, \ldots, \langle R_{k_l}, R_{m_l}\rangle)$ denote the probability R_i being better than R_j given that R_{k_0} is better than R_{m_0}, …, and R_{k_l} is better than R_{m_l}. Then, $P_h(R_i)$ ($i = 1, \ldots, N$) is computed inductively as follows:

$$\begin{aligned} P_h(R_1) &= P_b(R_1, R_0); \\ P_h(R_i) &= P_b(R_i, R_0) \times P_b(\langle R_i, R_1\rangle | \langle R_i, R_0\rangle) \times \ldots \\ & \quad P_b(\langle R_i, R_j\rangle | \langle R_i, R_0\rangle, \ldots, \langle R_i, R_{j-1}\rangle) \times \ldots \\ & \quad P_b(\langle R_i, R_{i-1}\rangle | \langle R_i, R_0\rangle, \ldots, \langle R_i, R_{i-2}\rangle) \\ &= P_b(R_i, R_0) \times \prod_{j=1}^{i-1} P_b(\langle R_i, R_j\rangle | \langle R_i, R_0\rangle, \ldots, \langle R_i, R_{j-1}\rangle) \\ &= P_b(R_i, R_0) \times \prod_{j=1}^{i-1}(1 - P_b(\langle R_j, R_i\rangle | \langle R_i, R_0\rangle, \ldots, \langle R_i, R_{j-1}\rangle)) \\ &= P_b(R_i, R_0) \times \prod_{j=1}^{i-1}(1 - P_b(\langle R_j, R_i\rangle, \langle R_j, R_0\rangle, \ldots, \langle R_j, R_{j-1}\rangle)) \\ &= P_b(R_i, R_0) \times \prod_{j=1}^{i-1}(1 - (P_b(R_j, \text{all of } R_0 \text{ to } R_{j-1}) \times \\ & \quad P_b(\text{any of } R_0 \text{ to } R_{j-1}, R_i) + \\ & \quad P_b(R_j, R_i) \times P_b(R_i, \text{all of } R_0 \text{ to } R_{j-1}))) \\ &= P_b(R_i, R_0) \times \prod_{j=1}^{i-1}(1 - (P_h(R_j) \times P_b(\text{any of } R_0 \text{ to } R_{j-1}, R_i) + \\ & \quad P_b(R_j, R_i) \times P_b(R_i, \text{all of } R_0 \text{ to } R_{j-1}))) \\ &= P_b(R_i, R_0) \times \prod_{j=1}^{i-1}(1 - (P_b(R_j, R_i) + \\ & \quad (P_h(R_j) - 1) P_b(\text{any of } R_0 \text{ to } R_{j-1}, R_i))) \\ &\approx P_b(R_i, R_0) \times \prod_{j=1}^{i-1}(1 - (P_b(R_j, R_i) + \\ & \quad (P_h(R_j) - 1) \times P_b(R_0, R_i))) \\ & \qquad (i = 2, \ldots, N) \end{aligned}$$

□

Then, we compute the switching probability as follows:

$$\begin{aligned} P_{ns}(R_0) &= P_b(R_0, R_1) \times \prod_{j=2}^{N}(1 - P_h(R_j)); \\ P_{ns}(R_i) &= P_h(R_i) \times \prod_{j=i+1}^{N}(1 - P_h(R_j)) \\ & \qquad (i = 1, \ldots, N-1); \\ P_{ns}(R_N) &= P_h(R_N) \end{aligned} \qquad (11)$$

Derivation sketch for formula (11). Inductively,

$$\begin{aligned} P_{ns}(R_0) &= P_b(R_0, R_1) \times P_b(\langle R_0, R_2\rangle | \langle R_0, R_1\rangle) \times \ldots \\ & \quad P_b(\langle R_0, R_N\rangle | \langle R_0, R_1\rangle, \ldots, \langle R_0, R_{N-1}\rangle) \\ &= P_b(R_0, R_1) \times \prod_{j=2}^{N} P_b(\langle R_0, R_j\rangle | \langle R_0, R_1\rangle, \ldots, \langle R_0, R_{j-1}\rangle) \\ &= P_b(R_0, R_1) \times \prod_{j=2}^{N}(1 - P_b(\langle R_j, R_0\rangle | \langle R_0, R_1\rangle, \ldots, \langle R_0, R_{j-1}\rangle)) \\ &= P_b(R_0, R_1) \times \prod_{j=2}^{N}(1 - P_h(R_j)); \\ P_{ns}(R_i) &= P_h(R_i) \times \prod_{j=i+1}^{N} P_b(\langle R_i, R_j\rangle | \langle R_i, R_0\rangle, \ldots, \langle R_i, R_{j-1}\rangle) \\ &= P_h(R_i) \times \prod_{j=i+1}^{N}(1 - P_h(R_j)) \\ & \qquad (i = 1, \ldots, N-1); \\ P_{ns}(R_N) &= P_h(R_N) \end{aligned}$$

□

Because of the approximation in formula (10), $\sum_{i=0}^{N} P_{ns}(R_i)$ 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.

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,

$$I_{ns}(R_0) = C \times P_{ns}(R_0) \quad (12)$$

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.

4.5 Implementation issues

In this subsection, we discuss implementation issues of LOF.

MAC feedback exfiltration. In LOF, both the status and the MAC latency for every unicast transmission are required. Yet the default Linux WLAN driver *hostap* [3] only signals for 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 that the MAC latency is reported too. Since we implement LOF, using EmStar [2], as a user-space process, MAC feedback is sent to the LOF process via *netlink* sockets and */proc* file system [12].

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.)

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.

5 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.

5.1 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 2.1 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 tested scenarios of multiple senders and periodic traffic, and LOF has also been used in the backbone network of ExScal. We discuss them in Section 5.3.

Protocols studied. We study the performance of LOF in comparison with that of beacon-based routing, where the latest development is represented by ETX [8, 21] and PRD [19]: (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 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 [10], where a route with the minimum *expected transmission time* is used.
- *PRD*: product of packet reception rate and distance traversed toward 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. In its design and analysis, PRD assumes *geographic-uniformity*, that the hops in any route have approximately the same geographic length.

Both ETX and PRD use broadcast beacons in estimating the respective routing metrics. 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 [9, 19], 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 (6), 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.

For easy comparison, we have implemented all the protocols mentioned above in EmStar [2], 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 4.5, all the protocols guarantee 100% packet delivery according to 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 [8, 10].
- *Energy efficiency*: energy spent in delivering a packet to the base station.
- *Route stability*: the number as well as the degree of route changes, and the stability of end-to-end packet delivery.

5.2 Experimental results

MAC latency. Using boxplots[6], Figure 17 shows the end-to-end

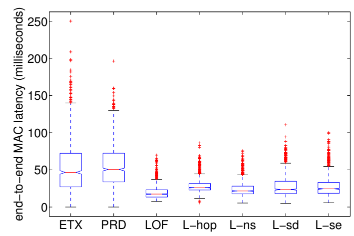


Figure 17: End-to-end MAC latency

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 both the data-driven link quality estimation and the decision of not assuming geographic uniformity. 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 and neighbor switching technique.

To explain the above observation, Figures 18, 19, 20, and 21

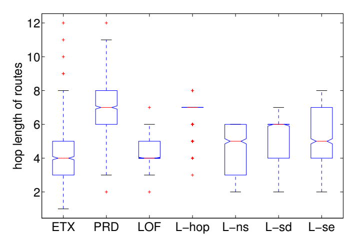


Figure 18: Number of hops in a route

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

[6] Boxplot is a nice tool for describing the distribution of a data sample:

- The lower and upper lines of the "box" are the 25th and 75th percentiles of the sample. The distance between the top and bottom of the box is the interquartile range.
- The line in the middle of the box is the sample median.
- The "whiskers", lines extending above and below the box, show the extent of the rest of the sample. If there is no outlier, the top of the upper whisker is the maximum of the sample, and the bottom of the lower whisker is the minimum. An outlier is a value that is more than 1.5 times the interquartile range away from the top or bottom of the box. An outlier, if any, is represented as a plus sign.
- The notches in the box shows the 95% confidence interval for the sample median.

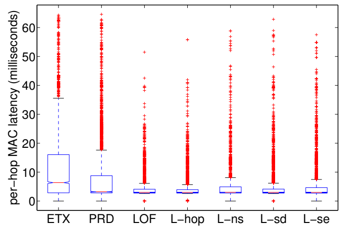


Figure 19: Per-hop MAC latency

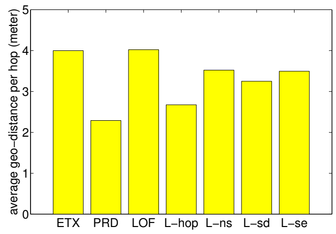


Figure 20: Average per-hop geographic distance

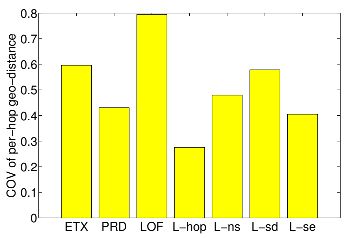


Figure 21: COV of per-hop geographic distance in a route

In PRD, both the average route hop length and the average per-hop MAC latency is about twice that in LOF.

From Figure 21, 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 22 shows the 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 22 does not represent the actual number of bytes sent. Nevertheless, given Figure 19 and the fact that MAC latency and energy consumption are positively related (as discussed in Section 3.1), the above observation on the relative energy efficiency among the protocols still holds.

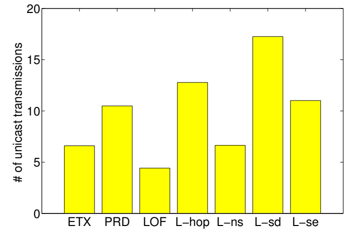


Figure 22: Number of unicast transmissions per packet received

To explain the above observation, Figure 23 shows the 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 20 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.

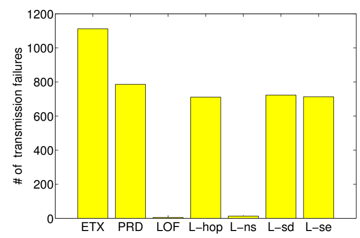


Figure 23: Number of failed unicast transmissions

Besides degenerating energy efficiency, transmission failures also increase queue accumulation, which can lead to reduction in network throughput. Figure 24 shows the maximum of the average queue length at the nodes involved in packet delivery. The maximum queue length in ETX and PRD is 12.64 and 5.22 times that in LOF respectively. The maximum queue length in L-hop is also 5.22 times that in LOF, again showing the negative impact of the invalid assumption — geographic uniformity.

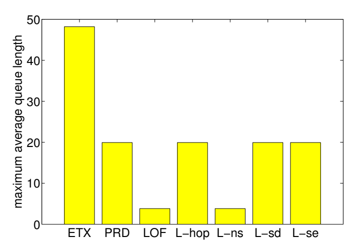


Figure 24: Maximum average queue length

Route stability. Figure 25 shows the average number of route

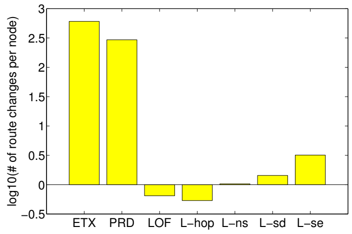


Figure 25: Average number of route changes per node

changes at each node. For readability in spite of the sharp difference in the values across protocols, we present the common logarithm (i.e., base 10) of the values along the y-axis. We see that the average number of route changes in ETX and PRD is 2 orders of magnitude greater than that in LOF. As a result, packets tend to be delivered in order in LOF but not in ETX and PRD, as shown in Figure 26 where the reorder distance of a packet p_0 is

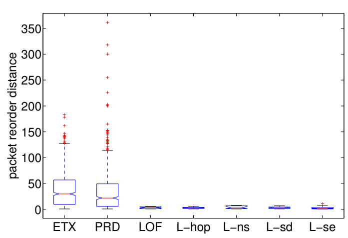


Figure 26: Packet reorder distance

the number of packets that are generated later than p_0 but reach the base station earlier than p_0.

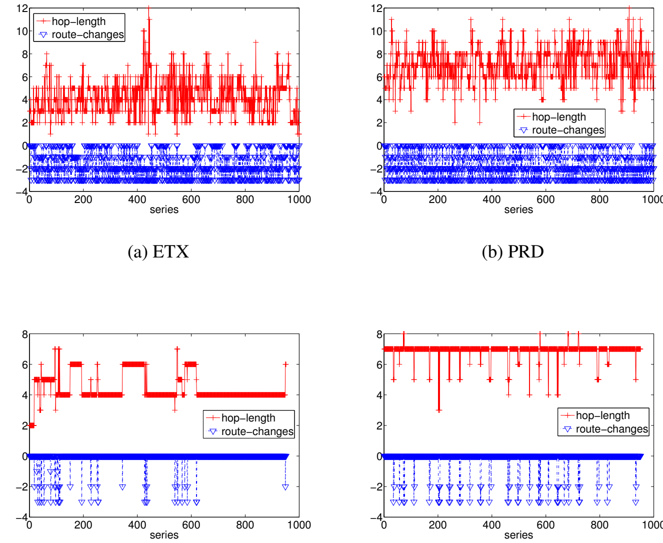
To understand how route changes, we measure the degree of route changes and how the hop-length of routes change. The degree of route change is measured as follows:

- 0: the route taken by a packet is the same as that taken by the previous packet;
- -1: the route taken by a packet is different from that taken by the previous packet, but they are of equal hop length;
- -2: the route taken by a packet is longer, in hop-length, than that taken by the previous packet;
- -3: the route taken by a packet is shorter, in hop-length, than that taken by the previous packet.

Due to space limitations, we only present, in Figure 27, the time series of the hop-length and the degree of route change for protocols ETX, PRD, LOF, and L-hop. LOF seldom changes route; yet route changes frequently in ETX and PRD, even when the routes are of equal hop-length.

5.3 Other experiments

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 from the ExScal traffic trace.



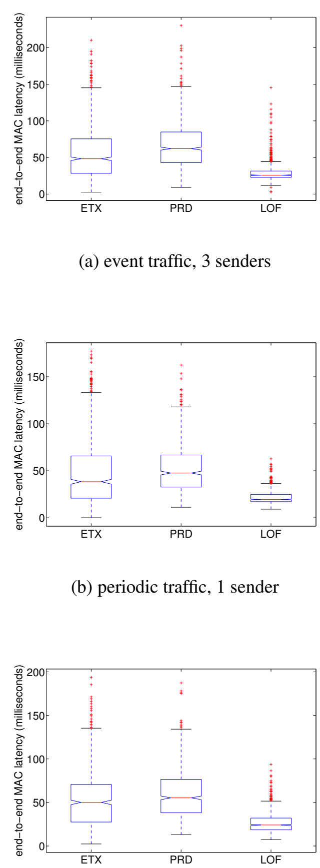
(a) ETX

(b) PRD

(c) LOF

(d) L-hop

Figure 27: Time series of route changes



(a) event traffic, 3 senders

(b) periodic traffic, 1 sender

(c) periodic traffic, 3 senders

Figure 28: End-to-end MAC latency

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 Figure 28.

With its well-tested performance, the implementation of LOF has been used in the field sensor network project ExScal [5] where 203 Stargates and 985 XSM motes are deployed in an area of 1260 meters by 288 meters. The 203 Stargates form the backbone network of ExScal to support reliable and real-time communication among the 985 XSM motes deployed for target detection, classification, and tracking. LOF has 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.

6 Related work

There is a rich literature on routing in ad hoc and wireless networks. 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 [6], [17], and [23]. 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 the 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 [18] and [7]. Our work complements theirs by experimentally study the differences as well as the impact of environment, distance, and interference pattern on the differences, which were not the focus of [18] and [7]. [7] mentioned the difficulty of getting MAC feedback and thus sticked to the method of beacon-used link estimation. Our work complements it by developing techniques for reliably fetching MAC feedback, which build the foundation for beacon-free link estimation as well as routing. To improve the performance of AODV, [18] and [7] also discussed reliability-based mechanisms (e.g., SNR-based ones) for blacklisting bad links. Since it has been shown that reliability-based blacklisting does not perform as well as ETX [11, 8, 21], we do not directly compare LOF to [18] and [7], 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 [8, 21] and ETT/WCETT [10] have been proposed and shown to perform well in real-world wireless networks [9]. The geography-based metric PRD [19] has also been proposed for energy-efficient routing in wireless sensor networks. Nevertheless, unicast link properties were still estimated based on those of broadcast beacons in these works, and [19] did not experimentally verify the assumption of "geographic uniformity" — that the hops in any route have approximately the same geographic length. Our work differs from the existing approaches by experimentally demonstrating the difficulty of precisely estimating unicast link properties via those of broadcast beacons, experimentally invalidating the assumption of geographic uniformity, and proposing the beacon-free protocol LOF where unicast link properties are estimated via the data traffic itself and geographic uniformity is not assumed. Another side result of our work is the comparison between the geography-unaware ETX routing and the geography-based PRD routing, which has not been done in the literature.

The problem of local maximum or geographic void has been dealt with in routing protocols such as GPSR [16]. We have not considered this problem in LOF, since it is orthogonal to our major concern — data-driven link quality estimation as well as routing. As a part of our future work, we plan to incorporate techniques developed by GPSR into the implementation of LOF.

7 Concluding remarks

Via experiments in testbeds of 802.11b networks, we have demonstrated the weakness of beacon-based link quality estimation as well as routing. We have also shown that the assumption of *geographic uniformity* is invalid in geographic routing.

To address the issues, 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 theses system facilities, we have demonstrated the feasibility of beacon-free routing by designing protocol LOF. It uses three main techniques for link quality estimation and route selection: initial sampling, data-driven adaptation, and probabilistic neighbor switching. Not assuming geographic uniformity, LOF uses the locally measurable routing metric ELD, the *expected MAC latency per unit-distance toward the destination*. 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.

Besides saving energy by avoiding periodic beaconing, the beacon-free nature of 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. The beacon-free nature of LOF also helps in enhancing network security, since the network is less exposed. More detailed study of the impact of beacon-free routing on energy efficiency and security is a part of our future work.

Acknowledgment

We thank Vinayak Naik and Emre Ertin for their help in the discussion and experimentation. We also appreciate the help from UCLA EmStar team for their help in answering questions regarding EmStar. The help and support from our ExScal team are always appreciated.

References

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

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

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

[4] Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. In *ANSI/IEEE Std 802.11*. 1999.

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

[6] 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.

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

[8] 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.

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

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

[11] 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.

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

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

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

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

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

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

[18] 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.

[19] 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.

[20] R. Thisted. *Elements of statistical computing*. Chapman & Hall, Ltd., 1988.

[21] 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.

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

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