Messaging in Sensor Networks:

Addressing Wireless Communications and Application Diversity

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

Loren J. Rittle
Pervasive Platforms and Architectures Lab
Motorola Labs, USA
ljrittle@motorola.com

July 11, 2006

1 Introduction

In wireless sensor networks, which we refer to as *sensornet*s hereafter, nodes coordinate with one another to perform tasks such as event detection and data collection. Since nodes are spatially distributed, message passing is the basic enabler of node coordination in sensornets. This chapter deals with the sensornet system service that is responsible for message passing; this service includes consideration of routing and transport issues and we refer to it as the *messaging* service. Even though messaging has been studied extensively in existing wired networks such as the Internet, it remains an important problem for sensornets. This is primarily due to the complex dynamics of wireless communication, to resource constraints, and to application diversity, as we explain below.

Wireless communication is subject to the impact of a variety of factors such as fading, multi-path, environmental noise, and co-channel interference. As a result, wireless link properties (e.g., packet delivery rate) are dynamic and assume complex spatial and temporal patterns [41, 38]. The impact of these complex dynamics on messaging is substantial; for instance, it has led to changes in even the primitive concept of neighborhood [32]. Moreover, the mechanisms that deal with the dynamics are constrained in terms of their energy, network bandwidth, space, and time budget. Broadly speaking, the challenge then has been *how to provide dependable, efficient, and scalable messaging despite the complex dynamics of wireless links*.

Sensornets have a broad range of application, in science (e.g., ecology and seismology), engineering (e.g., industrial control and precision agriculture), and our daily life (e.g., traffic control and health care). The breadth of application domains diversifies sensornet systems in many ways, including their traffic patterns and quality of service (QoS) requirements. For instance, in data-collection systems such as those that observe ecology, application data is usually generated periodically, and the application can tolerate certain degrees of loss and latency in data delivery; but in emergency-detection systems such as those for industrial control, data is generated only when rare and emergent events occur, and the application requires that the data be delivered reliably and in real time. The implications of application diversity for messaging include:

- Application traffic affects wireless link properties due to interference among simultaneous transmissions. This impact can vary significantly across diverse traffic patterns [38].
- Different requirements of QoS and in-network processing pose different constraints on the spatial and temporal flow of application data [39].
- Messaging services that are custom-designed for one application can be unsuitable for another, as is evidenced by a study of Zhang et al [4].
- It is desirable that message services accommodate diverse QoS and in-network processing requirements.

Broadly speaking, the challenge then has been *how to provide messaging that accommodates and potentially adapts to diverse application traffic patterns and QoS requirements as well as supports diverse in-network processing methods*.

The design of messaging services that bridge the challenges of *both* the wireless communications and application diversity deserves further study and is the focus of this chapter. We particularly emphasize how messaging can be made aware of, or exploit, or adapt to the application characteristics. Specifically, we present an architecture and examine some algorithmic design issues for sensornet messaging in this context.

Our architecture, SMA, identifies a messaging component, TLR, that deals with wireless link dynamics and its interaction with application traffic patterns; it also identifies two other messaging components, AST and ASC, that support diversified application requirements (such as QoS and in-network processing). After discussing SMA, we present in detail one instantiation of the TLR component, the LOF routing protocol, in Section 3. LOF deals with link dynamics and its interaction with application traffic patterns in forming the basic routing structure. We discuss related work in Section 4, and we make concluding remarks in Section 5.

2 SMA: an architecture for sensornet messaging

To support diverse applications in a scalable manner, it is desirable to have a unified messaging architecture that identifies the common components as well as their interactions [39]. To this end, we first review the basic functions of sensornet messaging, based upon which we then identify the common messaging components and design the messaging architecture SMA.

2.1 Components of sensornet messaging

As in the case for the Internet, the objective of messaging in sensornets is to deliver data from their sources to their destinations. To this end, the basic tasks of messaging are, given certain QoS constraints (e.g., reliability and latency) on data delivery, choose the route(s) from every source to the corresponding destination(s) and schedule packet flow along the route(s). As we argued before, unlike wired networks, the design of messaging in sensornets is challenging as a result of wireless communication dynamics, resource constraints, and application diversity.

Given the complex dynamics of sensornet wireless links, a key component of sensornet messaging is precisely estimating wireless link properties and then finding routes of high quality links to deliver data traffic. Given that data traffic pattern affects wireless link properties due to interference among simultaneous transmissions [38], link estimation and routing should be able to take into account the impact of application data traffic, and we call this basic messaging component *traffic-adaptive link estimation and routing (TLR)*.

With the basic communication structure provided by the TLR component, another important task of messaging is to adapt the structure and data transmission schedules according to application properties such as in-network processing and QoS requirements. Given the resource constraints in sensornets, application data may be processed in the network before it reaches the final destination to improve resource utilization (e.g., to save energy and to reduce data traffic load). For instance, data arriving from different sources may be compressed at an intermediate node before it is forwarded further. Given that messaging determines the spatial and temporal flow of application data and that data items from different sources can be processed together only if they meet somewhere in the network, messaging significantly affects the degree of processing achievable in the network [39, 16]. It is therefore desirable that messaging consider in-network processing when deciding how to form the messaging structure and how to schedule data transmissions. In addition, messaging should also consider application QoS requirements (e.g., reliability and latency in packet delivery), because messaging structure and transmission schedule determine the QoS experienced by application traffic[22, 30, 20]. In-network processing and QoS requirements tend to be tightly coupled with applications, thus we call the structuring and scheduling in messaging *application-adaptive structuring (AST)* and *application-adaptive scheduling (ASC)* respectively.

2.2 Architecture of sensornet messaging

These messaging components are coupled with wireless communication and applications in different ways and in different degrees, so we adopt two levels of abstraction in designing the architecture for sensornet messaging. The architecture, SMA (for *Sensornet Messaging Architecture*), is shown in Figure 1.1. At the lower level, traffic-adaptive

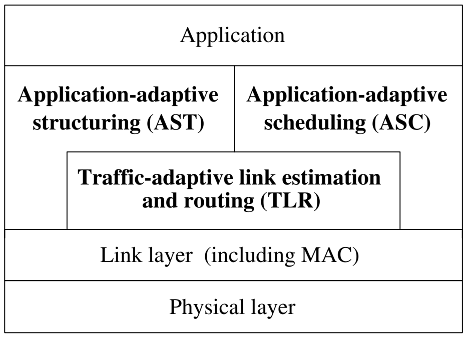

Application
Application-adaptive structuring (AST)
Application-adaptive scheduling (ASC)
Traffic-adaptive link estimation and routing (TLR)
Link layer (including MAC)
Physical layer

Figure 1.1: SMA: a sensornet messaging architecture

link estimation and routing (TLR) interacts directly with the link layer to estimate link properties and to form the basic routing structure in a traffic-adaptive manner. TLR can be performed without explicit input from applications, and TLR does not directly interface with applications. At the higher level, both application-adaptive structuring (AST) and application-adaptive scheduling (ASC) need input from applications, thus AST and ASC interface directly with applications. Besides interacting with TLR, AST and ASC may need to directly interact with link layer to perform tasks such as adjusting radio transmission power level and fetching link-layer acknowledgment to a packet transmission. In the architecture, the link and physical layers support higher-layer messaging tasks (i.e., TLR, AST, and ASC) by providing the capability of communication within one-hop neighborhoods.

In what follows, we elaborate on the individual components of SMA.

Traffic-adaptive link estimation and routing (TLR). To estimate wireless link properties, one approach is to use beacon packets as the basis of link estimation. That is, neighbors exchange broadcast beacons, and they estimate broadcast link properties based on the quality of receiving one another's beacons (e.g., the ratio of beacons successfully received, or the RSSI/LQI of packet reception); then, neighbors estimate unicast link properties based on those of beacon broadcast, since data are usually transmitted via unicast. This approach of beacon-based link estimation has been used in several routing protocols including ETX [35, 12].

We find that there are two major drawbacks of beacon-based link estimation. Firstly, it is hard to build high-fidelity models for temporal correlations in link properties [33, 31, 23], thus most existing routing protocols do not consider temporal link properties and assume instead independent bit error or packet loss. Consequently, significant estimation error can be incurred, as we show in [38]. Secondly, even if we could precisely estimate unicast link properties, the estimated values may only reflect unicast properties in the absence —instead of the presence— of data traffic, which matters since the network traffic affects link properties due to interference. This is especially the case in event-detection applications, where events are usually rare (e.g., one event per day) and tend to last only for a short time at each network location (e.g., less than 20 seconds). Therefore, beacon-based link estimation cannot precisely estimate link properties in a traffic-adaptive manner.

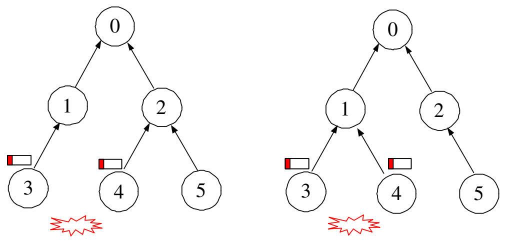
To address the limitations of beacon-based link estimation, Zhang et al [38] propose the LOF routing protocol (for *Learn on the Fly*) that estimates unicast link properties via MAC feedback[1] for data transmissions themselves without using beacons. Since MAC feedback reflects in-situ the network condition in the presence of application traffic, link estimation in LOF is traffic-adaptive. LOF also addresses the challenges of data-driven link estimation to routing protocol design, such as uneven link sampling (i.e., the quality of a link is not sampled unless the link is used in data forwarding). It has been shown that, compared with beacon-based link estimation and routing, LOF improves both the reliability and energy efficiency in data delivery. More importantly, LOF quickly adapts to changing traffic patterns, and this is achieved without any explicit input from applications.

The TLR component provides the basic service of automatically adapting link estimation and routing structure to application traffic patterns. TLR also exposes its knowledge of link and route properties (such as end-to-end packet delivery latency) to higher level components AST and ASC, so that AST and ASC can optimize the degree of in-network processing while providing the required QoS in delivering individual pieces of application data.

[1]The MAC feedback for a unicast transmission includes whether the transmission has succeeded and how many times the packet has been retransmitted at the MAC layer.

Application-adaptive structuring (AST). One example of application-adaptive structuring is to adjust messaging structure according to application QoS requirements. For instance, radio transmission power level determines the communication range of each node and the connectivity of a network. Accordingly, transmission power level affects the number of routing hops between any pairs of source and destination and thus packet delivery latency. Transmission power level also determines the interference range of packet transmissions, and thus it affects packet delivery reliability. Therefore, radio transmission power level (and thus messaging structure) can be adapted to satisfy specific application QoS requirements, and Kawadia and Kumar have studied this in [22].

Besides QoS-oriented structuring, another example of application-adaptive structuring is to adjust messaging structure according to the opportunities of in-network processing. Messaging structure determines how data flows spatially, and thus affects the degree of in-network processing achievable. For instance, as shown in Figure 1.2(a), nodes 3 and



(a) Before adaptation (b) After adaptation

Figure 1.2: Example of application-adaptive structuring

4 detect the same event simultaneously. But the detection packets generated by nodes 3 and 4 cannot be aggregated in the network, since they follow different routes to the destination node 0. On the other hand, if node 4 can detect the correlation between its own packet and that generated by node 3, node 4 can change its next-hop forwarder to node 1, as shown in Figure 1.2(b). Then the packets generated by nodes 3 and 4 can meet at node 1, and be aggregated before being forwarded to the destination node 0.

In general, to improve the degree of in-network processing, a node should consider the potential in-network processing achievable when choosing the next-hop forwarder. One way to realize this objective is to adapt the existing routing metric. For each neighbor k, a node j estimates the utility $u_{j,k}$ of forwarding packets to k, where the utility is defined as the reduction in messaging cost (e.g., number of transmissions) if j's packets are aggregated with k's packets. Then, if the cost of messaging via k without aggregation is $c_{j,k}$, the associated messaging cost $c'_{j,k}$ can be adjusted as follows (to reflect the utility of in-network processing):

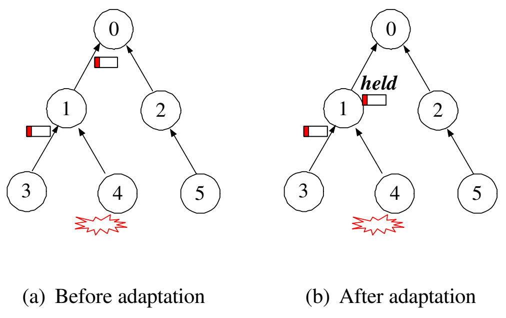
$$c'_{j,k} = c_{j,k} - u_{j,k}$$

Accordingly, a neighbor with the lowest adjusted messaging cost is selected as the next-hop forwarder.

Since QoS requirements and in-network processing vary from one application to another, AST needs input (e.g., QoS specification and utility of in-network processing) from applications, and it needs to interface with applications directly.

Application-adaptive scheduling (ASC). One example of application-adaptive scheduling is to schedule packet transmissions to satisfy certain application QoS requirements. To improve packet delivery reliability, for instance, lost packets can be retransmitted. But packet retransmission consumes energy, and not every sensornet application needs 100% packet delivery rate. Therefore, the number of retransmissions can be adapted to provide different end-to-end packet delivery rates while minimizing the total number of packet transmissions [8]. To provide differentiated timeliness guarantee on packet delivery latency, we can also introduce priority in transmission scheduling such that urgent packets have high priority of being transmitted [36]. Similarly, data streams from different applications can be ranked so that transmission scheduling ensures differentiated end-to-end throughput to different applications [15].

Besides QoS-oriented scheduling, another example of application-adaptive scheduling is to schedule packet transmissions according to the opportunities of in-network processing. Given a messaging structure formation, transmission scheduling determines how data flows along the structure temporally and thus the degree of in-network processing achievable. To give an example, let us look at Figure 1.3(a). Suppose node 4 detects an event earlier than node 3 does.



(a) Before adaptation (b) After adaptation

Figure 1.3: Example of application-adaptive scheduling

Then the detection packet from node 4 can reach node 1 earlier than the packet from node 3. If node 1 immediately forwards the packet from node 4 after receiving it, then the packet from node 4 cannot be aggregated with that from node 3, since the packet from node 4 has already left node 1 when the packet from node 3 reaches node 1. On the other hand, if node 1 is aware of the correlation between packets from nodes 3 and 4, then node 1 can hold the packet from 4 after receiving it (as shown in Figure 1.3(b)). Accordingly, the packet from node 3 can meet that from node 4, and these packets can be aggregated before being forwarded.

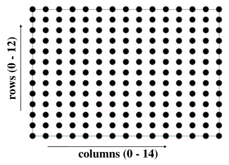
In general, a node should consider both application QoS requirements and the potential in-network processing when scheduling data transmissions, so that application QoS requirements are better satisfied and the degree of in-network processing is improved. Given that in-network processing and QoS requirements are application specific, ASC needs to directly interface with applications to fetch input on parameters such as QoS specification and utility of in-network processing.

Due to the limitation of space, we only discuss TLR in detail in the remainder of this chapter. A detailed study of ASC can be found in [39]; AST is still an open issue for further exploration.

Remark. It is desirable that the components TLR, AST, and ASC be deployed all together to achieve the maximal network performance. That said, the three components can also be deployed in an incremental manner while maintaining the benefits of each individual component, as shown in [38, 39].

3 Data-driven link estimation and routing

As briefly discussed in Section 2, the major drawbacks of beacon-based link estimation and routing are twofold: Firstly, it is hard to precisely estimate unicast link properties via those of broadcast, especially when temporal correlation in link properties is not considered; Secondly, network condition changes with traffic patterns, and beacon-based link estimation may not be able to converge. To get an intuitive sense of how traffic patterns affect network condition and how temporal correlation affects the fidelity of beacon-based link estimation, we experimentally measure the packet delivery rate of broadcast and unicast in the presence of different traffic patterns, and we calculate the error incurred in link estimation if temporal correlations in link properties are not considered. We conduct the experiments in the sensornet testbed Kansei [7]. Kansei is deployed in an open warehouse with flat aluminum walls (see Figure 1.4(a)), and it consists of 195 Stargates[2] organized into a 15 $\times$ 13 grid (as shown in Figure 1.4(b)) where the separation between neighboring grid points is 0.91 meter (i.e., 3 feet). Each Stargate is equipped with the same SMC wireless



(a) Kansei (b) Grid topology

Figure 1.4: Sensornet testbed Kansei

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. Figure 1.5 shows that network condition, measured in broadcast reliability, varies significantly with traffic

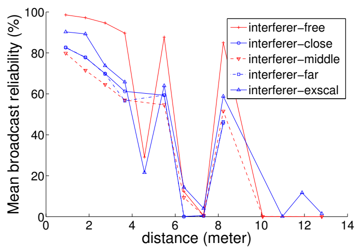


Figure 1.5: Network condition, measured in broadcast reliability, in the presence of different traffic patterns

patterns. Figure 1.6 shows the significant error[3] in estimating unicast delivery rate via that of broadcast under different

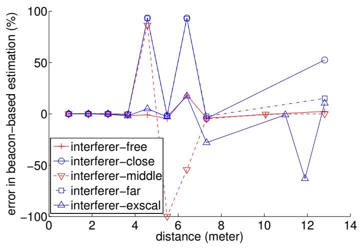


Figure 1.6: Error in estimating unicast delivery rate via that of broadcast

[2]Stargates[2] use IEEE 802.11 as its radio, and have been adopted as the backbone nodes in many sensornet systems including ExScal [9] and MASE [1].

traffic scenarios when temporal correlations in link properties are not considered (i.e., assuming independent bit error and packet loss) [38]. 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 sensornet backbones, data-driven routing is feasible because of the following facts:

- **MAC feedback.** In MACs where every packet transmission is acknowledged by the receiver (e.g., in 802.11b and 802.15.4MACs), 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 [10].)

- **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 and the performance of LOF which implements ELD without using periodic beacons.

Remarks:

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

- 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 [6]. 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.
- Routing metric can also be based on other parameters such as ETX [12] or RNP [11], and detailed study has been reported in [40]. Due to the limitation of space, however, we only present the routing metric that is based on MAC latency in this chapter.

3.1 ELD: the routing metric

For messaging in sensornets (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 [37].)

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 1.7, 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

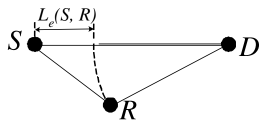


Figure 1.7: L_e calculation

$LD(S, R)$, as[4]

$$\begin{cases} \frac{D_{S,R}}{L_e(S,R)} & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \tag{1.1}$$

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

$$\begin{cases} \frac{E(D_{S,R})}{L_e(S,R)} & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \tag{1.2}$$

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 3.3.)

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 1.4, 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 1.8 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.

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

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

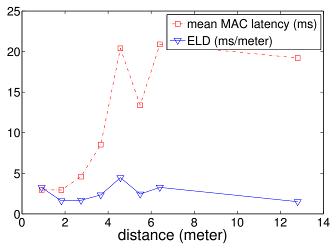


Figure 1.8: Mean unicast MAC latency and the ELD

of S. In the analysis of geographic routing [29], 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 3.3, 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, $ELR(S, R) =$

$$\begin{cases} E(D_{S,R}) \times \lceil \frac{L_{S,R}+L_{R,D}}{L_{S,R}} \rceil & \text{if } L_{S,D} > L_{R,D} \\ \infty & \text{otherwise} \end{cases} \tag{1.3}$$

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 3.3 that ELR is inferior to ELD.

3.2 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.

3.2.1 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 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 detailed discussion here since they are not the focus of this chapter.

3.2.2 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 1.9, which shows the mean unicast MAC latency in different interfering scenarios. 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.

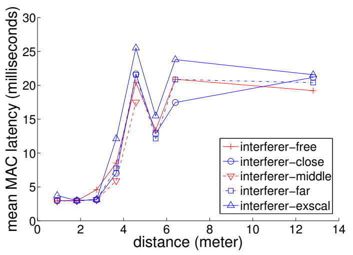


Figure 1.9: MAC latency in the presence of interference

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.

3.2.3 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 $log(LD)$. Figure 1.10 shows a time series of the $log(LD)$ regarding a node 3.65 meters (i.e., 12 feet) away from a sender. We

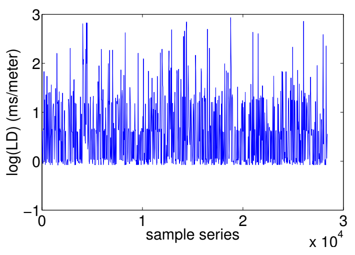


Figure 1.10: A time series of $log(LD)$

see that the time series fits well with the *constant-level model* [19] where the generating process is represented by a constant superimposed with random fluctuations. Therefore, a good estimation method is *exponentially weighted moving average* (EWMA) [19], assuming the following form

$$V \longleftarrow \alpha V + (1 - \alpha)V' \tag{1.4}$$

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 (1.4) 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 (1.4) is adapted to be the following:

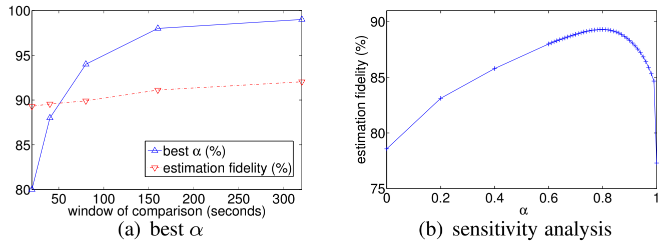
$$V \longleftarrow \alpha^{\beta(R)}V + (1 - \alpha^{\beta(R)})V' \tag{1.5}$$

(Experiments confirm that LOF performs better with formula (1.5) than with formula (1.4).)

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 (1.5) 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 (1.5), 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 (1.5) 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 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 1.11(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

Figure 1.11: The weight α in EWMA

For sensitivity analysis, Figure 1.11(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 [9]. Again, the withdrawing and rejoining process can be optimized, but we skip the details here.)

3.2.4 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, 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. (Interested readers can find the detailed derivation in [37].

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,

$$I_{ns}(R_0) = C \times P_{ns}(R_0) \tag{1.6}$$

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.

3.3 Experimental evaluation

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

3.3.1 Experiment design

Network setup. In testbed Kansei as shown in Figure 1.4, 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 [38]. 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. Interested readers can find the detailed discussion in [37].

Protocols studied. We study the performance of LOF in comparison with that of beacon-based routing, where the latest development is represented by ETX [12, 35] and PRD [29]: (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 [14], 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 [12] and [29], 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. In our experiments, metrics ETX and PRD are estimated according to the method originally proposed in [12] and [29]; for instance, broadcast beacons have the same packet length and transmission rate as those of data packets.

To verify some important design decisions of LOF, we also study different versions of LOF as follows:[5]

- *L-hop*: assumes geographic-uniformity, and thus uses metric ELR, as specified by formula (1.3), instead of ELD;

[5]Note: we have 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.

- *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 [3], a software environment for developing and deploying sensornets.

Evaluation criteria. Reliability is one critical concern in convergecast. Using the techniques of reliable transport discussed in [37], 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 [12, 14].
- *Energy efficiency*: energy spent in delivering a packet to the base station.

3.3.2 Experimental results

MAC latency. Using boxplots, Figure 1.12 shows the end-to-end MAC latency, in milliseconds, for each protocol.

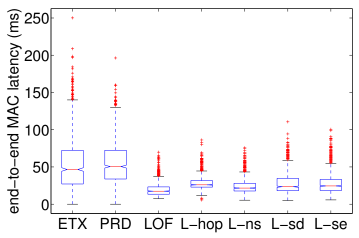


Figure 1.12: End-to-end MAC latency

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.

To explain the above observation, Figures 1.13, 1.14, 1.15, and 1.16 show the route hop length, per-hop MAC

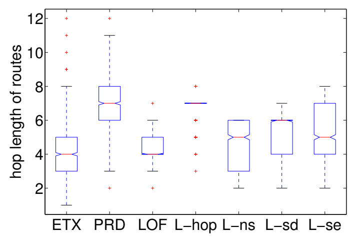


Figure 1.13: Number of hops in a route

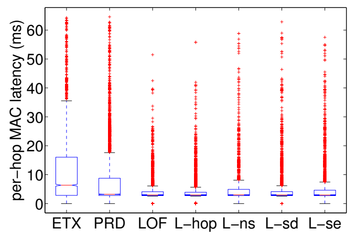


Figure 1.14: Per-hop MAC latency

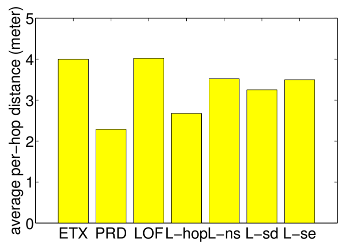


Figure 1.15: Average per-hop geographic distance

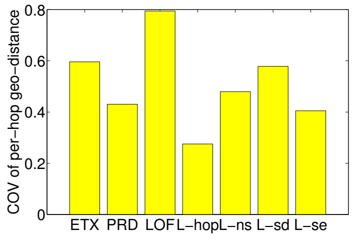


Figure 1.16: COV of per-hop geographic distance in a route

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.

From Figure 1.16, 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 1.17 shows the number of unicast transmissions averaged over the

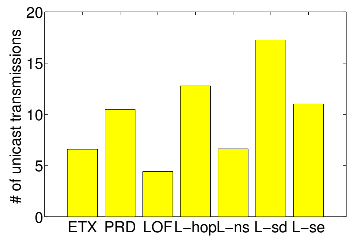


Figure 1.17: Number of unicast transmissions per packet received

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 1.17 does not represent the actual number of bytes sent. Nevertheless, given Figure 1.14 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.

To explain the above observation, Figure 1.18 shows the number of failed unicast transmissions for the 950 packets

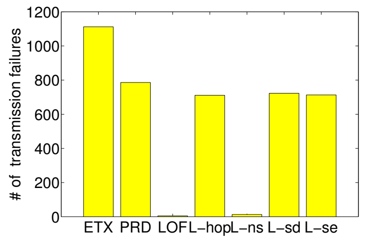


Figure 1.18: Number of failed unicast transmissions

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 Figure 1.15, 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.

Based on its well-tested performance, LOF has been incorporated in the ExScal sensornet field experiment [9], where 203 Stargates were deployed as the backbone network with the inter-Stargate separation being around 45 meters. LOF has successfully guaranteed reliable and real-time convergecast from any number of non-base Stargates to the base station in ExScal.

4 Related work

Architecture. The Sensornet Protocol (SP) [28] provides a unified link layer abstraction for sensornets. As SP focuses on the interface between link and network layers, SP can be incorporated into the SMA architecture which focuses on higher layer architecture. Woo et al [34] discuss network support for query processing in sensornets, in particular, issues such as query-oriented routing, efficient rendezvous for storage and correlation, and unified in-network processing. Their focus is on query processing, as opposed to the architectural and algorithmic issues involved in supporting a broader range of applications, such as distributed signal processing and computing. Impala [26] considers adapting the communication protocol to changing network conditions and application requirements, using the Adaptation Finite State Machine (AFSM). Nahrstedt et al [27] consider the provision of application-specific QoS in ubiquitous environments, via a framework for QoS specification and compilation, QoS setup, and QoS adaptation. SMA and LOF complement the proposals in [26] and [27] by focusing on issues such as application-adaptive link estimation, structuring, and scheduling, which obviate the human from the loop.

In Internet related research, the concepts of Application Oriented Networking (AON) [5] and Application-driven Networking [21, 17] are being explored to enable coordination among disparate applications, to enforce application-specific policies, to improve visibility of information flow, and to enhance application optimization and QoS. While these concepts are generic enough to be applied to sensornets, the techniques employed and the problems faced in the Internet context differ from those in sensornets due to differences in both technology ad application domains. For instance, severe resource constraints are unique to sensornets and are not major issues in the Internet.

Routing. Link properties in sensornets and 802.11b mesh networks have been well studied [6, 24, 41]. These works 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.

In addressing the challenges of wireless communication, great progress has recently been made regarding routing in sensornets and mesh networks. Routing metrics such as ETX [12, 35] and ETT/WCETT [14] have been proposed and shown to perform well in real-world wireless networks [13]. The geography-based metric PRD [29] has also been proposed for energy-efficient routing in sensornets. Unicast link properties are estimated using broadcast beacons in these works. LOF differs from existing approaches by avoiding the difficulty of precisely estimating unicast link properties via those of broadcast beacons.

As in LOF, SPEED [18] uses MAC latency and geographic information for route selection. Concurrently with our work, [25] proposes NADV which also uses information from MAC layer. While they do focus on real-time packet delivery and a general framework for geographic routing, [18] and [25] did not focus on the protocol design issues in data-driven link estimation and routing. Nor do they consider the importance of appropriate *probabilistic neighbor switching* . 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 is shown in Section 3.3.

5 Concluding remarks

Being a basic service for sensornets, messaging needs to deal with the dynamics of wireless communication, to support diverse applications, and to cope with the interaction between link dynamics and application traffic patterns. For dependable, efficient, and scalable messaging in sensornets, we have proposed the sensornet messaging architecture SMA. SMA employs two levels of abstraction: the lower level component TLR deals with wireless link dynamics and its interaction with traffic patterns in a unified manner; the higher level components AST and ASC directly interface with applications and support application-specific QoS requirements and in-network processing.

SMA is a first step towards the unified sensornet messaging architecture. As technologies and applications evolve, it is expected that the architecture and its components will be enriched. The benefits of TLR and ASC have been demonstrated experimentally, yet there are still many fundamental problems associated with modeling the stability and optimality of data-driven link estimation and routing, as well as those of application-adaptive structuring and scheduling. In the context of link estimation and routing, there are also interesting systems issues deserving further exploration, for instance, comparison of different routing metrics (e.g., number-of-transmission based vs. MAC latency based) and metric evaluation mechanisms (e.g., distance-vector routing vs. geographic routing). Another important issue in sensornets is power management. It has significant implications to the design of sensornet architecture and algorithms, and should be studied systematically too.

References

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

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

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

[4] Rmase. http://www2.parc.com/isl/groups/era/nest/Rmase/default.html.

[5] Application-oriented networking. http://www.cisco.com/, 2005.

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

[8] A. Arora and et al. A Line in the Sand: A wireless sensor network for target detection, classification, and tracking. *Computer Networks (Elsevier)*, 46(5), 2004.

[9] A. Arora, R. Ramnath, E. Ertin, P. Sinha, S. Bapat, V. Naik, V. Kulathumani, H. Zhang, H. Cao, M. Sridhara, S. Kumar, N. Seddon, C. Anderson, T. Herman, N. Trivedi, C. Zhang, M. Gouda, Y. R. Choi, M. Nesterenko, R. Shah, S. Kulkarni, M. Aramugam, L. Wang, D. Culler, P. Dutta, C. Sharp, G. Tolle, M. Grimmer, B. Ferriera, and K. Parker. Exscal: Elements of an extrem scale wireless sensor network. In *IEEE RTCSA*, 2005.

[10] B. Awerbuch, D. Holmer, and H. Rubens. High throughput route selection in multi-rate ad hoc wireless networks. Technical report, Johns Hopkins University, 2003.

[11] 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.

[12] 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.

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

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

[15] C. T. Ee and R. Bajcsy. Congestion control and fairness for many-to-one routing in sensor networks. In *ACM SenSys*, pages 148–161, 2004.

[16] K.-W. Fan, S. Liu, and P. Sinha. On the potential of structure-free data aggregation in sensor networks. In *IEEE INFOCOM*, 2006.

[17] J. Follows and D. Straeten. *Application-Driven Networking: Concepts and Architecture for Policy-based Systems*. IBM, December 1999.

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

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

[20] B. Hull, K. Jamieson, and H. Balakrishnan. Mitigating congestion in wireless sensor networks. In *ACM SenSys*, pages 134–147, 2004.

[21] IBM. *Application Driven Networking: Class of Service in IP, Ethernet and ATM Networks*. IBM, August 1999.

[22] V. Kawadia and P. R. Kumar. Principles and protocols for power control in ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 23(5):76–88, 2005.

[23] A. Konrad, B. Zhao, and A. Joseph. A markov-based channel model algorithm for wireless networks. *Wireless Networks*, 9:189–199, 2003.

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

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

[26] T. Liu and M. Martonosi. Impala: A middleware system for managing autonomic, parallel sensor systems. In *ACM PPoPP*, 2003.

[27] K. Nahrstedt, D. Xu, D. Wichadakul, and B. Li. Qos-aware middleware for ubiquitous and heterogeneous environments. *IEEE Communications Magazine*, 2001.

[28] J. Polastre, J. Hui, P. Levis, J. Zhao, D. Culler, S. Shenker, and I. Stoica. A unifying link abstraction for wireless sensor networks. In *ACM SenSys*, pages 76–89, 2005.

[29] 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.

[30] C. Wan, S. Eisenman, and A. Campbell. CODA: Congestion detection and avoidance in sensor networks. In *ACM SenSys*, pages 266–279, 2003.

[31] H. S. Wang and N. Moayeri. Finite-state markov channel - a useful model for radio communication channels. *IEEE Transactions on Vehicular Technology*, 44(1):163–171, 1995.

[32] K. Whitehouse, C. Sharp, E. Brewer, and D. Culler. Hood: A neighborhood abstraction for sensor networks. In *USENIX/ACM MobiSys*, 2004.

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

[34] A. Woo, S. Madden, and R. Govindan. Networking support for query processing in sensor networks. *Communications of the ACM*, 47(6):47–52, 2004.

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

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

[37] 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.

[38] H. Zhang, A. Arora, and P. Sinha. Learn on the fly: Data-driven link estimation and routing in sensor network backbones. In *IEEE INFOCOM*, 2006.

[39] H. Zhang, L. J. Rittle, and A. Arora. Application-adaptive messaging in sensor networks. Technical Report OSU-CISRC-6/06-TR63, The Ohio State University, 2006.

[40] H. Zhang, L. Sang, and A. Arora. Link estimation and routing in sensor networks: Beacon-based or data-driven? Technical Report OSU-CISRC-6/06-TR64, The Ohio State University, 2006.

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

Index