Dynamic Switching-based Data Forwarding for Low-Duty-Cycle Wireless Sensor Networks

Yu Gu *Member, IEEE,* and Tian He, *Member, IEEE*

Abstract—In this work, we introduce the concept of Dynamic Switch-based Forwarding (DSF) that optimizes the (i) expected data delivery ratio, (ii) expected communication delay, or (iii) expected energy consumption for low-duty-cycle wireless sensor networks under unreliable communication links. DSF is designed for networks with possibly unreliable communication links and predetermined node communication schedules. To our knowledge, these are the most encouraging results to date in this new research direction. In this paper, DSF is evaluated with a theoretical analysis, extensive simulation, and physical testbed consisting of 20 MicaZ motes. Results reveal the remarkable advantage of DSF in extremely low duty-cycle sensor networks in comparison to three well-known solutions (ETX [1], PRR×D [2] and DESS [3]). We also demonstrate our solution defaults into ETX in always-awake networks and DESS in perfect-link networks.

Index Terms—Wireless Sensor Networks, Low-Duty-Cycle Networks, Dynamic Data Forwarding.

1 Introduction

Wireless Sensor Networks (WSNs) have been proposed for use in many long-term applications such as military surveillance, assisted living, infrastructure monitoring and scientific exploration, which require a network lifespan that can range from a few months to several years. On the other hand, sensor devices (e.g., MicaZ and Telos) are normally equipped with limited power sources due to their small form factor and low-cost requirements. To resolve the conflict between limited energy and application lifetime requirements, it is necessary to reduce node communication and sensing duty cycles. With the growing gap between application requirements and the slow progress in battery capacity [5], there are an increasing number of extremely low duty-cycle sensor networks designed and deployed. Together with lossy radio links, these new networks impose new challenges for data forwarding protocols.

In this work, we focus on *low-duty-cycle sensor networks with unreliable communication links*, in which energy management protocols [6], [7], [8] schedule sensing and communication at each individual sensor device to enable a duty cycle of 10% or less. Essentially, during the operation of sensor applications, sensor nodes activate very briefly and stay in a dormant state for a very long period of time. Due to the devices' extremely limited energy budget, maintaining an always-awake communication backbone becomes infeasible. Consequently to forward a packet, a sender may experience *sleep latency* – the time spent waiting for the receiver to wake up.

In this paper we attempt to design a new data delivery method to optimize source-to-sink data delivery ratio, end-to-end (E2E) delay, or energy consumption under *unreliable* and *intermittent* connectivity within scheduled networks. The major intellectual contributions of this work are as follows:

- To the best of our knowledge, this is the first work to investigate the combined effect of *sleep latency* and *unreliable communication links*, which dramatically reduces the effectiveness of the existing solutions. A novel dynamic switch-based forwarding technique over time-dependent networks is proposed to achieve optimal expected delivery ratio (EDR), expected E2E delay (EED), or expected energy consumption (EEC), respectively. This technique is generic enough to allow flexible tradeoffs among these three key metrics.
- We extensively evaluate our solutions with 20 MicaZ motes experiments and 250-node simulation. The results from experiments and simulations show significantly better source-to-sink communication than several state-of-the-art solutions.

The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 describes the need for a new data forwarding technique in extremely low duty-cycle sensor networks. Section 4 articulates the network model and related assumptions. Section 5 introduces the detailed design of DSF and discusses related issues. Section 7 describes our system implementation and provides an evaluation on the TinyOS/Mote platform. Simulation results are presented in Section 8. Section 9 concludes the paper.

2 Related Work

The contribution of our work lies in the intersection of two important cutting-edge research topics. We demonstrate that the intriguing interaction between unreliable links and low-power duty-cycling necessitates a fundamentally new approach.

Link-Quality-Based Forwarding: Many recent works [9], [10] reveal that wireless communication links, especially for the low-power sensor devices, are extremely unreliable and have a significant impact on data delivery. In response

- *Yu Gu is with Singapore University of Technology and Design, Singapore. Tian He is with University of Minnesota, Twin Cities, USA. E-mail: jasongu@sutd.edu.sg, tianhe@cs.umn.edu*

A conference paper [4] containing some preliminary results of this paper has appeared in ACM SenSys 2007

to the reality of unreliable wireless links, several notable works have been done. De Couto et al. introduce the expected transmission count metric (ETX) to find high-throughput paths on multi-hop wireless networks [1]. Woo et al. show that cost-based routing using a minimum expected transmission metric obtains good performance in wireless sensor networks [11]. Seada et al. study the distance-hop trade-off for geographic routing in wireless sensor networks and show that the product of the packet reception rate (PRR) and the distance traversed toward the destination (D) is an optimal metric (PRR×D) for selecting a next-hop forwarder [12]. Lee et al. present SOFA, an on-demand solicitation-based forwarding protocol and show that SOFA outperforms the commonly used link estimation-based routing schemes implemented in TinyOS [13]. In ETF [14], Sang et al. exploit asymmetric wireless links and observe significant improvement of convergecast routing in sensor networks. In these works, the authors assume the constant availability of connectivity with no sleep latency, which may not be true in extremely low duty-cycle sensor networks.

Sleep-Latency-Based Forwarding: In the research direction of low duty-cycle networks, Dousse et al. provide a solid analysis of bounds of the delay for sending data from a node to a sink in the networks with completely uncoordinated node working schedules [15]. Lu et al. introduce various techniques for minimizing communication latency while providing energy-efficient periodic sleep cycles for nodes in wireless sensor networks [3]. Keshavarzian et al. introduce a multi-parent forwarding technique and propose a heuristic algorithm for assigning parents to the nodes in the network [16]. Lai and Paschalidis propose a minimal energy routing with latency guarantees in duty-cycled sensor networks [17]. Su et al. propose both on-demand and proactive algorithms for routing packets in intermittently connected sensor networks [18]. Several recent works studied multicast and flooding in low-duty-cycle sensor networks [19], [20]. More recently, Gu et al. study the delay control for low-duty-cycle sensor networks [21]. We note, however, that all these approaches in low duty-cycle networking assume perfect communication links.

We note that many MAC protocols, such as B-MAC [22], S-MAC [23] and RI-MAC [24], effectively deal with the issues of lossy radio links through FEC/ARQ and reduce duty-cycle through the Low-Power-Listening (LPL) [22]. More recently, Suriyachai et al. introduces a novel MAC protocol that incorporates topology control mechanisms to ensure timely data delivery and reliability control mechanisms to deal with inherently fluctuating wireless links [25]. These intelligent layer 2 protocols use implicit network information, such as packet transmissions, in order to optimize their underlying schedules or energy use. In this paper, we consider the dual of this problem by using information from layer 2 at the network layer to make better link selections.

In addition, there are many other related works on timely and reliable data forwarding in sensor networks. MMSPEED introduces a multi-path and multi-speed routing protocol for probabilistic QoS guarantee [26]. Dwarf achieves energy-efficient, robust and dependable forwarding by unicast-based partial flooding and delay-aware node selection [27]. WirelessHART with TSMP [28] is a deployed industry standard aiming to achieve timely and reliable transmission while reducing energy consumption. Munir et al. propose a scheduling algorithm that produces latency bounds of the real-time periodic stream sand accounts for both link bursts and interference [29].

To the best of our knowledge, no prior work has thoroughly studied the impact of both *lossy radio links* and *sleep latency* at the network layer. In this work, we reveal that these two issues are intrinsically correlated and that a new forwarding protocol can benefit from considering both.

3 MOTIVATION

Our work is motivated by the interesting intersection between sleep latency and unreliable communication links in wireless sensor networks.

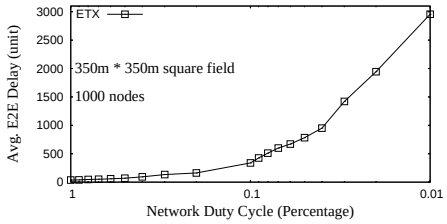


Fig. 1. E2E Delay vs. Network Duty Cycle

First, the state-of-the-art link-quality-based forwarding strategies such as ETX [1] and PRR×D [2] have demonstrated their superiority at improving network throughput and communication delay in traditional ad hoc and sensor networks. For both ETX and PRR×D, during a certain period of time each node usually has one fixed forwarding node for a destination. However, in extremely low duty-cycle scheduled sensor networks, metrics such as the expected transmission count (ETX) would suffer excessive delivery delays when waiting for the fixed receiver to wake up again if the ongoing packet transmission fails. Figure 1 shows the E2E delays from a randomly chosen source node to the sink node using ETX forwarding metrics under different network duty cycles in a randomly-generated network topology. The simulation setup is the same as in Section 8 and key simulation parameters are shown on the figure. The simulation was repeated 1000 times and the average value is reported in Figure 1 (in log-scale), which shows that as network duty cycle decreases, the E2E delay grows significantly. For example, at the duty cycle of 100%, the E2E delay of ETX is only 37.6 units of time. In contrast, when the duty cycle drops to 1%, the E2E delay increases to 2955.5 units of time, which is approximately an 80-fold performance degradation in end-to-end delay!

Second, sleep-latency-based forwarding [30], [3] ignores the reality that wireless radio quality is highly unreliable and that thus the optimality of their approaches holds only when the link quality in the network is perfect. Figure 2 shows the E2E delay from a randomly chosen source node

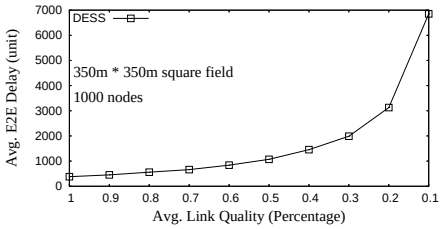


Fig. 2. E2E Delay vs. Average Link Quality

to the sink node using delivery methods proposed in [3] under different average link quality in a random generated network topology. As shown in the figure, the E2E delay increases from 380.0 to 6851.4 units of time while the average link quality decreases from 100% to 10%, which is approximate a 20-fold performance degradation, even though global scheduling information is available.

The main observation from our initial studies is that both the link quality and the duty cycle of sensor nodes can significantly impact end-to-end communication. Although link-quality-based forwarding [1], [2] and sleep-latency-based forwarding [30], [3] have demonstrated their effectiveness in their own contexts, they fail to deal with the combined effect exhibited in many real-world sensor network applications. This limitation motivates us to design a new data forwarding technique, which we discuss in the rest of the paper.

4 Models and Assumptions

Before presenting DSF in detail, we present the network model and assumptions used in this work. To simplify our description, we introduce DSF's design in a synchronized mode with discrete time. Later on, we explain why DSF works without time slots and only requires local synchronization. In other words, DSF works in CSMA networks where nodes are duty-cycled by upper-layer protocols such as sensing coverage [31] and power management [16], [32], [33].

4.1 Network Model

We assume a network with N sensor nodes. At a given point of time t a sensor node is in either an active or a dormant state. When a node is in the active state, it can sense and receive packets transmitted from neighboring nodes. When a node is in the dormant state, it turns off all function modules except a timer (for the purpose of waking itself up). In other words, a node can wake up to transmit a packet at any time, but can receive packets only when it is in its active state. Formally, we denote the network status at time t as $G(t) = (V, E(t))$, where V is a complete set of N nodes within the network, and $E(t)$ is a set of directed edges at time t. An edge $e(i, j)$ belongs to $E(t)$ if and only if (1) node n_i is a neighboring node of n_j, and (2) n_j is active and hence able to receive data at time t. Essentially, $G(t)$ represents the potential traffic flow within the network at time t. Obviously the connectivity of $G(t)$ varies with time. In other words, $G(t)$ is a time-dependent network.

We represent the states of each node n_i with a working schedule $\Gamma_i = (\omega_i, \tau)$.

- ω_i is an infinite binary string, in which 1 denotes the active state and 0 denotes the dormant state. Clearly, the duty cycle of a node is the percentage of 1's in the binary string. Since the working schedules of the sensor nodes are normally periodic (for sensing purposes), the infinite binary string ω_i can be described using a regular expression.
- The state transitions between active and inactive states are time-driven. We use τ to denote the time span a bit in the binary string ω_i.

We note that the simple 2-tuple (ω_i, τ) is generic enough to represent arbitrary sensor nodes working schedules. Theoretically, when $\tau \to 0$, ω_i can precisely characterize any on/off behavior of node n_i. For clarity of presentation, we begin our design with a simplified assumption that it takes time τ to transmit one packet and receive acknowledgment from a receiver. The assumption on the round-trip transmission time bound τ holds well when traffic/congestion is low, which is the case in extremely low duty-cycle sensor networks. In addition, B-MAC [22] has already used link-level implicit acknowledgement to support fixed round-trip transmission time.

4.2 Time-Expanded Network

To visualize the data delivery process in a time-dependent network $G(t) = (V, E(t))$, we replicate $G(t)$ with regular graphs $G = (V, E)$ along with the time dimension. We call this is a *time-expanded network*. In this section, for a given sensor network topology and node working schedules, we describe how we can build a corresponding time-expanded network. The resulting time-expanded network can help us better understand the data delivery method introduced in the rest of the paper.

Given a network $G(t) = (V, E(t))$ with n nodes and node working schedules $\Gamma_i = (\omega_i, \tau)$, where $i \in V$, we use the following rules to construct its corresponding time-expanded network.

- For any node $i \in V$ at time t, we build a distinct node N_{it}.
- For each newly built node N_{it}, if node j is a neighboring node of the node i and p is the position of first active bit in ω_j after time t, we build a directed edge from N_{it} to N_{jp} with a length of $(p - t)\tau$.
- At the destination node d, we connect all its time-expanded nodes to a null node with edge lengths of zero.

To illustrate the above network mapping rules, we provide a walk-through of time-expanded network construction from a time dependent graph. Figure 4 shows how to construct a time-expanded network from the linear time-dependent network shown in Figure 3. As shown in Figure 3, for node 1 at time 1, the only node that is within its communication range is node 2, and its first active state after time 1 appears at time 3, so in Figure 4, we build a directed edge from node 1 at time 1 to node 2 at time 3 with an edge length of 2τ. Similarly, we construct other edges in Figure 4. Finally, for the destination node 4, we connect all its time-expanded

nodes from time 1 to time 6 to a null node with edge lengths of zero.

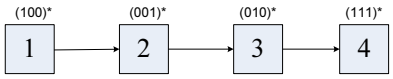


Fig. 3. A Linear Network

4.3 E2E Delay in Time-Expanded Network

Obviously, if all the nodes are in active states, end-to-end (E2E) delay in the above network model equals $H\tau$, where H is the minimum number of hops between a source and a destination. However, if nodes in a network have certain working schedules $\Gamma = \{\Gamma_1, \Gamma_2, ..., \Gamma_N\}$, transmission at each hop could be delayed by waiting for the intermediate receivers to wake up.

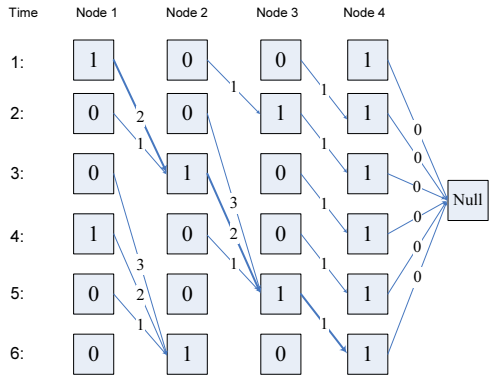


Fig. 4. Time-Expanded Network

To further illustrate the data delivery process in the extremely low duty-cycle network, Figure 4 demonstrates the process of delivering a packet from node 1 to node 4 when the packet is ready to be sent at time 1. For the sake of simplicity, in this example we assume all links are perfect with no packet loss and will discuss cases when the link quality is not perfect in detail in the following sections. At the first two time intervals, node 2 (the only neighbor of node 1) is in the dormant state, and thus no packets could be transmitted. At the third time interval, node 2 becomes active, which allows node 1 to transmit a packet to it, so the packet is delivered from node 1 to node 2. At the second hop, node 2 waits one time interval for node 3 to wake up, and the packet arrives at node 3 at time 5. Finally, since node 4 is active all the time, without any additional waiting, node 3 delivers the packet to node 4 at time 6. The total end-to-end delay therefore is 5 units of time.

5 Main Design

As shown in Section 4.3, when the link quality is perfect, the end-to-end delay is the sum of two types of delays: (1) the total transmission delay, which is the product of number of hops and τ, and (2) The sleep latency, which is the time spent on waiting for the receivers to wake up at each hop. However, the unreliable radio links between low-power sensor devices suggests that the packet transmission between a sender and a receiver would not always be 100% successful. As a result, the waiting time at each hop is highly impacted not only by the node working schedule but also by the link quality, which inspires us to design a dynamic switch-based data forwarding protocol.

Since every operation within an extremely low duty-cycle sensor network is time-dependent, for the sake of clarity we use the terms *node* and *time-expand node* interchangeably in the rest of the paper. We have organized this design section into five components. Section 5.1 describes the basic design of Dynamic Switch-based Forwarding (DSF). Section 5.2 analyzes the expected delivery ratio, E2E delay, and energy consumption, assuming the forwarding action is known *a priori*. Section 6 optimizes the forwarding action to achieve maximum delivery ratio, minimal delay, and energy efficiency, respectively.

5.1 The Basic Design of DSF

Differently than traditional data forwarding techniques such as ETX and PRR×D, we allow *multiple potential forwarding nodes* at each hop. For a given sink, each node maintains a sequence of forwarding nodes sorted in the order of the wake-up time associated with them. To start sending a packet, a node looks up the time associated with the first node in the sequence, wakes up at that time interval, and tries to send the packet. If the transmission is successful, forwarding is done. Otherwise, the node fetches the next wake-up time from the sequence and tries to send the packet again. This retransmission process over a single hop continues until the sending node confirms that the packet has been successfully received by one of forwarding nodes or the sending node reaches the end of the sequence and drops the packet.

Formally, we define the sequence of forwarding nodes at a node e as: S_n^e

Definition 1 (Forwarding Sequence S_n^e): S_n^e is a sequence of n nodes that can forward packets from node e to the sink. This sequence is sorted based on the wake-up time of the nodes. Formally, $S_n^e = (s_1^e, s_2^e, \cdots, s_n^e)$. Let $t(s_i^e)$ be the wake-up time of node s_i^e, Forwarding sequence S^e satisfies $t(e) < t(s_1^e) < t(s_2^e) < \cdots < t(s_n^e)$.

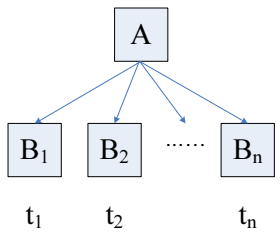


Fig. 5. Example of Dynamic Switching

Figure 5 demonstrates the packet transmission process between one sender and n nodes in its forwarding sequence. In Figure 5, node A has a packet to be sent and its forwarding sequence is $S_n^A = (B_1, B_2, \cdots, B_n)$. First, node A wakes up at time t_1 and tries to transmit the packet to the node B_1. If the data delivery is successful, node A ends the current packet forwarding session. However, if the

transmission fails, the node A wakes up again at time t_2 and tries to send the packet to the node B_2. This retransmission process continues with node A repeatedly trying to send the packet to the node in the sequence S_n^A. If the transmission fails at the last node B_n, node A drops the packet.

From the above example, we can see that the major advantage of dynamic switching is the use of a forwarding sequence to reduce the time spent on transmitting a packet successfully at each hop rather than waiting for a particular forwarding node to wake up again after failure, as in such solutions as ETX, PRR×D and DESS.

5.2 The Modeling of EDR, EED, and EEC

Given a known forwarding sequence S_n^e at a node e , we can model the expected delivery ratio, the expected E2E delay and the expected energy consumption for the node. Here, for the sake of clarity, we describe a scenario with a single sink node, that can be extended easily for scenarios with multiple sink nodes.

Formally, these three metrics are defined as:

Definition 2 (Expected Delivery Ratio $EDR_e(S_n^e)$): The expected delivery ratio at node e for a given forwarding sequence S_n^e, denoted by $EDR_e(S_n^e)$, is the expected packet delivery ratio from node e to the sink node (over multi-hop path).

Definition 3 (Expected E2E Delay $EED_e(S_n^e)$): The expected E2E Delay at node e for a given forwarding sequence S_n^e, denoted by $EED_e(S_n^e)$, is the expected data delivery delay for the packets sent by node e and received by the sink node (over multi-hop path).

Definition 4 (Expected Energy Consumption $EEC_e(S_n^e)$): The expected energy consumption at node e for a given forwarding sequence S_n^e, denoted by $EEC_e(S_n^e)$, is the expected energy consumption to deliver a packet from node e to the sink node (over multi-hop path). We note that since receiving (idle) energy is fixed for a given working schedule, we include only senders' transmission energy in EEC.

Our model for computing EDR, EED, and EEC values is distributed and can be executed at individual sensor nodes independently. At the sink node (b), obviously, its forwarding sequence is empty, the $EDR_b(\emptyset)$ value is 100% (i.e., no packet loss), while $EED_b(\emptyset)$ and $EEC_b(\emptyset)$ values are both zeros (i.e., no delay and no energy consumption). Consequently, we can obtain following initial equations:

$$EDR_b(\emptyset) = 1 \quad , \quad EED_b(\emptyset) = 0 \quad , \quad EEC_b(\emptyset) = 0 \qquad (1)$$

Let the bi-directional link quality p_{ei} denotes the success ratio of a round-trip transmission (DATA and ACK) between node e and the i^{th} forwarder in S_n^e. The link quality p_{ei} can be influenced by multiple factors such as transmission power and the distance between a sender and a receiver. We note that in extremely low duty-cycle sensor networks, traffic congestion is rare and hence has little effect on link quality.

The overall probability $P^e(i)$ that a packet transmission by node e is successful at the i^{th} forwarder (after $i-1$ failures) can be represented as:

$$P^e(i) = [\prod_{j=1}^{i-1}(1 - p_{ej})]p_{ei} \qquad (2)$$

Expected Delivery Ratio (EDR): Obviously, EDR value for node e is the sum of the product of the probability that the transmission is successful at a particular forwarder and its corresponding EDR value for all nodes in S_n^e. Assuming node e has n nodes in its forwarding sequence and letting EDR_i be the EDR value for the i^{th} forwarder (s_i^e) in S_n^e, we have the following recursive equation for $EDR_e(S_n^e)$.

$$EDR_e(S_n^e) = \sum_{i=1}^{n} P^e(i) EDR_i \qquad (3)$$

Expected E2E Delay (EED): The EED value of node e represents the expected delay for the packets sent by node e that reach the sink node b. Consequently, the probability that the packet transmission is successful at a certain forwarder is under the condition that the packet is delivered by one of the forwarders in S_n^e. Therefore, the conditional probability is $P^e(i)' = \frac{P^e(i)EDR_i}{EDR_e(S_n^e)}$. Letting EED_i be the EED value for the i^{th} forwarder in node e's forwarding sequence and d_i be the delay for node e to wait node s_i^e in S_n^e to wake up, $EED_b(S_n^e)$ can be represented as:

$$EED_e(S_n^e) = \textstyle\sum_{i=1}^{n} P^e(i)'(d_i + EED_i) \qquad (4)$$

Expected Energy Consumption(EEC): Similarly, let EEC_i be the EEC value for the i^{th} forwarder in S_n^e and that the expected energy consumption for successful packet transmission at node s_i^e is the sum of EEC_i and i units of energy consumption (note that energy wasted in $i-1$ failed transmission should be included as well). The probability that the retransmission of a packet reaches the i^{th} forwarder $P^e(i)'$ at node e is conditional on the data delivery ratio $EDR_e(S_n^e)$. Therefore, $P^e(i)' = \frac{P^e(i)EDR_i}{EDR_e(S_n^e)}$, and we can formulate the $EEC_e(S_n^e)$ as:

$$EEC_e(S_n^e) = \textstyle\sum_{i=1}^{n} P^e(i)'(i + EEC_i) \qquad (5)$$

The recursive calculation of EDR, EED and EEC can be implemented at individual nodes distributively. The main idea is to radiate known initial conditions ($EDR_b(\emptyset) = 1$, $EED_b(\emptyset) = 0$,$EEC_b(\emptyset) = 0$) from the sink node, so that the process of calculating EDR, EED and EEC values propagates outward from the sink nodes to the rest of the network.

6 Optimizing the Forwarding Sequence

In the previous section, we described the model for calculating EDR, EED, and EEC for a *given* forwarding sequence. In this section, we will discuss how we can obtain a forwarding sequence that is *optimal* in terms of the maximum expected data delivery ratio, minimum expected E2E delay, or minimum expected transmission energy consumption at individual sensor nodes, respectively.

In practical network settings, especially in low duty-cycle sensor networks, a sender should not endlessly retransmit

a packet because it would consume significant energy at the sending nodes. Therefore, we set the maximum time bound for a sender to retransmit a particular packet as T. Consequently, at node e, with known neighboring nodes and their corresponding working schedule Γ, we can have a full sequence of potential forwarding nodes that wake up before T.

Formally, let s_i^e be a next-hop node with wake-up time $t(s_i^e)$. Node e's full sequence S_m^e under the bound T is:

$$S_m^e = (s_1^e, s_2^e, \cdots, s_m^e) \ where \ s \in S_m^e \iff t(e) \leq t(s) \leq t(e)+T$$

6.1 Optimizing Expected Delivery Ratio (EDR)

Because the length of the potential forwarding sequence of a node is finitely subject to the maximum retransmission time interval T, under the reality of unreliable link quality among pairs of wireless sensor devices, packets sent by a source node may not all arrive the destination sink node. Therefore, when reliable transmission has the highest priority for a sensor network application, the optimization of the expected data delivery ratio (EDR) is critical.

Intuitively, in order to maximize the expected data delivery ratio at node e, we should try to send packets as long as one of the next-hop nodes is awake. The reasoning behind this is plausible, as since we want to maximize the expected data delivery ratio, we should take every opportunity to move the packet out of the sender. However, this intuition does not lead us to an optimal expected data delivery ratio, and Figure 6 presents a counterexample. In Figure 6, suppose the full forwarding sequence of the node S is $S_2^S = (A, B)$. If we choose both node A and B to form S's forwarding sequence S_2^S, according to the Equation 3, $EDR_e(S_2^S) = 10\%$. In contrast, if we choose only node B to be included in S_1^S, the corresponding $EDR_e(S_1^S) = 100\%$. Therefore, in order to optimize the expected data delivery ratio at a node e, we shall select a subsequence from the full sequence S_m^e. By definition, a subsequence can be obtained by removing some of the elements from the original sequence without disturbing the relative positions of the remaining elements. As an example, (B, E, D, G) is a subsequence of (A, B, E, C, F, D, H, G).

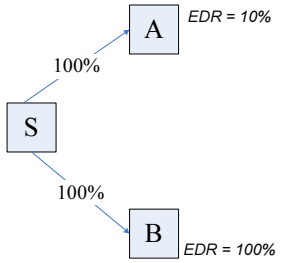


Fig. 6. Example for Selecting a Subset of Nodes in Potential Forwarding Sequence

To select an optimal subsequence S_{opt}^e from the full sequence S_m^e, we adopt a dynamic programming approach. Clearly, the last node s_m^e in S_m^e must be included in S_{opt}^e, since s_m^e provides the last chance for node e to retransmit before the packet is dropped. Starting from this optimal substructure, we can attempt to include nodes (one by one) from S_m^e ***backwardly*** into S_{opt}^e. If the inclusion of a node from S_m^e into S_{opt}^e increases $EDR_e(S_{opt}^e)$, we then add this node into S_{opt}^e permanently. Otherwise, we discard the node and try to add the next node. The above forwarding sequence selection process continues until we reach the node s_1^e in the full sequence S_m^e. The optimality of this dynamic programming algorithm is based on the fact that the optimal $EDR_e(S_{opt}^e)$ can be constructed efficiently from its optimal substructures. *The decisions made to include or exclude a* ***later*** *node in the forwarding sequence does not affect the optimality of decisions made to include or exclude* ***earlier*** *nodes and vice versa.* For each backward augmentation of the forwarding sequence, we guarantee the maximum data delivery ratio of the sequence between the newly augmented node and the last node. This forwarding sequence, then, serves as an optimal substructure for augmenting additional forwarders until the process reaches the first node in the sequence.

Let $S_{opt}^e(k)$ denote the optimal forwarding subsequence in terms of maximizing EDR metric from the sequence $S_k^e = (s_{m-k+1}^e, s_{m-k+2}^e, \ldots, s_m^e)$. Obviously, $S_{opt}^e(m)$ is the optimal subsequence we want to obtain.

We have the following initial optimal substructure:

$$\begin{aligned} S_{opt}^e(0) &= () \\ S_{opt}^e(1) &= (s_m^e) \end{aligned} \tag{6}$$

Building upon the previous optimal substructure, when we attempt to include the next node s_j^e in S_m^e into $S_{opt}^e(k-1)$ backwardly, there are two possible outcomes:

- According to the model of $EDR_e(S^e)$, if the appending of node s_j^e to $S_{opt}^e(k-1)$ increases the expected delivery ratio, we insert node s_j^e in front of the existing sequence $S_{opt}^e(k-1)$ to obtain $S_{opt}^e(k)$.
- If the inclusion of node s_j^e into sequence $S_{opt}^e(k-1)$ does not increase the data delivery ratio, the optimal forwarding sequence remains unchanged.

Formally, let $s_j^e \oplus S$ denote inserting node s_j^e to the front of the sequence S, the corresponding recursive equation for $S_{opt}^e(k)$ can be represented as:

$$S_{opt}^e(k) = \begin{cases} S_{opt}^e(k-1) & EDR_e(S_{opt}^e(k-1)) > \\ & EDR_e(s_j^e \oplus S_{opt}^e(k-1)) \\ s_j^e \oplus S_{opt}^e(k-1) & Otherwise \end{cases} \tag{7}$$

6.1.1 Detailed Algorithm for Optimizing EDR

In the previous section, we discussed recursive equations for optimizing forwarding sequence for EDR. In this section, we introduce detailed dynamic programming algorithm that implements earlier mathematical formulations. The complete algorithm is shown in Algorithm 1. Firstly, according to the known wake-up time of neighboring nodes, we form a full sequence $S - m^e$ as the input of our optimizing algorithm. Then following Equation 6, we construct an initial optimal substructure (Line 1 to Line 2). From Line 4 to Line 14, we perform the task of adding forwarding node backwardly and decide whether a node should be included in the optimal forwarding sequence or not. Line 5 to Line 9 constructs a temporary forwarding sequence

Algorithm 1 Forwarding sequence optimization for EDR at a time expanded node e

Input: Full sequence S_m^e, with length m.
Output: The optimal forwarding sequence S^e

1: $S_{opt}^e(0) \leftarrow ()$
2: $S_{opt}^e(1) \leftarrow (s_m^e)$
3: $i \leftarrow 2$
4: **for** j from $(m-1)$ downto 1 **do**
5: **if** $t(s_j^e) \neq t(s_{opt}^e(i-1))$ **then**
6: $S \leftarrow s_j^e \oplus S_{opt}^e(i-1)$.
7: **else**
8: $S \leftarrow s_j^e \oplus (S_{opt}^e(i-1) - s_{opt}^e(i-1))$.
9: **end if**
10: **if** $EDR_e(S) > EDR_e(S_{opt}^e(i-1))$ **then**
11: $S_{opt}^e(i) \leftarrow S$
12: **else**
13: $S_{opt}^e(i) \leftarrow S_{opt}^e(i-1)$
14: **end if**
15: $i \leftarrow i+1$
16: **end for**

with the inclusion of a new node from full sequence. According to rules described in Equation 7, we decide the new optimal substructure (Line 10 to line 14). This selection process continues until we have tried every node in the full sequence. Therefore, the complexity of this algorithm is proportional to the length of full sequence, and can be expressed as $\mathcal{O}(DT)$, where D is the density of next-hop nodes and T is the maximum per-hop delay allowed.

6.2 Optimizing Expected E2E Delay (EED)

In many sensor network applications, such as military surveillance, target tracking and infrastructure monitoring, the delay for the source-to-sink communication is critical to the performance of the system.

We note that if there is no bound on the expected delivery ratio (EDR) for the forwarding sequence, the optimal forwarding sequence in terms of minimizing delay can be trivially achieved by including only a single node j which has the minimum ($d_j + EED_j$) value among all nodes in S_m^e (Equation 4). However, with such a quick-and-dirty solution, especially when the link quality between node e and node j is low, node e may suffer from an extremely low packet delivery ratio to the sink node and consequently may cause the whole network to be unavailable. Therefore, it is important to minimize the EED metric for the node e under the constraint that the EDR metric of the forwarding sequence is greater than a certain bound R. The bound R must be less or equal to the optimal EDR value that could be achieved at the node e.

Similarly to maximizing EDR, we also adopt a dynamic programming approach to select a subset of nodes in S_m^e *backwardly* to optimize EED. But in contrast, the last node in S_m^e is no longer guaranteed to be the optimal initial optimal substructure, since the inclusion of the node may increase the expected E2E delay. Instead, to optimize EED, we need to try every node in the full sequence S_m^e as the last node in the optimal subsequence. For example, if we suppose $S_m^e = (B, E, D, G)$, we need to obtain optimal subsequences from (B, E, D, G), (B, E, D), (B, E), and (B) with G, D, E, B chosen, respectively.

Suppose node s_{last}^e is selected as the last node and $S_{opt}^e(last, k)$ represents the optimal forwarding subsequence in terms of EED chosen from the sequence $S_k^e(last) = (s_{last-k+1}^e, s_{last-k+2}^e, \ldots, s_{last}^e)$, where $k \leq last$ and $last \in \{1, 2, \ldots, m\}$.

For each last node, we have the following initial optimal substructure for $S_{opt}^e(last, k)$ in terms of minimal EED:

$$\begin{aligned} S_{opt}^e(last, 0) &= () \\ S_{opt}^e(last, 1) &= (s_{last}^e) \end{aligned} \tag{8}$$

Similar to the recursive equations for maximizing EDR, for each node s_j^e in $S_k^e(last)$, the optimized forwarding sequence for EED is:

$$S_{opt}^e(last, k) = \begin{cases} S_{opt}^e(last, k-1) & EED_e(S_{opt}^e(last, k-1)) < \\ & EED_e(s_j^e \oplus S_{opt}^e(last, k-1)) \\ s_j^e \oplus S_{opt}^e(last, k-1) & Otherwise \end{cases} \tag{9}$$

After having all $S_{opt}^e(last, last)$ where $last \in \{1, 2, \ldots, m\}$, we chose the forwarding sequence with the minimal EED value, under the constraint that $EDR \geq R$.

6.2.1 Detailed Algorithm for Optimizing EED

As shown in 2, similar to the forwarding sequence optimization for EDR, we first build a full sequence of forwarding nodes as the input of our algorithm. Then as explained in previous section, while optimizing EED, we cannot ensure that the last node in the full sequence forms the initial optimal substructure, therefore we need to try every node in the full sequence as the last node in the optimal subsequence (Line 1). With an initial subsequence, we proceed with our dynamic programming algorithm that is identical to the the optimization of EDR except for applying EED model in Line 11 when we calculate metric value for a given forwarding sequence (Line 2 to Line 17). After generating all forwarding sequences, we select the one that yields minimal EED value while achieving EDR bound R (Line 20 to Line 24). Since we repeat forwarding sequence optimization for EDR m times, the complexity of this algorithm is $\mathcal{O}(Tm^2)$.

6.3 Reducing Expected Energy Consumption (EEC)

For applications such as scientific exploration, the difficulty of entering the sensing field and the corresponding high cost of system deployment calls for the longevity of the system, making energy conservation the highest priority for the system design. Similarly to the optimization of EED, if we do not have a bound on the expected delivery ratio, the optimal forwarding sequence for the minimal EEC would include only one node with the smallest EEC value in S_m^e and may also experience an extremely low source-to-sink data delivery ratio. Therefore, in this section we reduce EEC under the constraint that EDR of the forwarding sequence is above threshold R.

Algorithm 2 Forwarding sequence optimization for EED at a time expanded node e

Input: Full sequence S^e_m, with length m.
Input: Data Delivery Ratio Bound R
Output: The optimal forwarding sequence S^e

1: **for** $i = 1$ to m **do**
2: $S_opt^e(i, 0) \leftarrow ()$
3: $S_opt^e(i, 1) \leftarrow (s^e_i)$
4: $j \leftarrow 2$
5: **for** k from $i - 1$ downto 1 **do**
6: **if** $t(s^e_k) \neq t(s^e_{opt}(i, j - 1))$ **then**
7: $S \leftarrow s^e_k \oplus S^e_{opt}(i, j - 1)$.
8: **else**
9: $S \leftarrow k \oplus S^e_{opt}(i, j - 1) - s^e_{opt}(i, j - 1)$.
10: **end if**
11: **if** $EED_e(S) < EED_e(S^e_{opt}(i, j - 1))$ **then**
12: $S^e_{opt}(i, j) \leftarrow S$
13: **else**
14: $S^e_{opt}(i, j) \leftarrow S^e_{opt}(i, j - 1)$
15: **end if**
16: $j \leftarrow j + 1$
17: **end for**
18: **end for**
19: $MinEED \leftarrow \infty$
20: **for** $i = 1$ to m **do**
21: **if** $EED_e(S^e_{opt}(i, i)) < MinEED$ and $EDR_e(S^e_{opt}(i, i)) \geq R$ **then**
22: $S^e \leftarrow S^e_{opt}(i, i)$
23: $MinEED \leftarrow EED_e(S^e)$
24: **end if**
25: **end for**

Unlike optimizing EED, in Equation 5, where i represents the index of forwarding node in the forwarding sequence, the i value changes for each already selected forwarding node as we backwardly add early nodes. In other words, the decisions made to include or exclude an **early** node in the forwarding sequence **does affect** the expected energy of **later** nodes. Lacking an optimal substructure, we can only choose either an exhaustive search (in the case that a forwarding sequence is small) or a greedy heuristic algorithm. We found that the greedy case for EEC is actually very effective. The main idea of the greedy algorithm is that starting with an empty optimal forwarding sequence, we continuously add the unselected node in S^e_m that results in a minimal increase in EEC into the optimal forwarding sequence until the EDR of the optimal forwarding sequence reaches R. Empirical results indicate that the greedy algorithm obtains optimal results 85% of the time and the suboptimal results are within 5% of the optimal values.

6.3.1 Detailed Algorithm for Reducing EEC

In Algorithm 3, we first construct a full sequence of forwarding nodes according to wake-up time of neighboring nodes for node e. Then starting with an empty forwarding sequence (Line 1), we select a node from the full sequence, which yields a minimal EEC value after its inclusion in the forwarding sequence (Line 3 to Line 13). Above node

Algorithm 3 Forwarding sequence optimization for EEC at a time expanded node e

Input: Full sequence S^e_m, with length m.
Input: Data Delivery Ratio Bound R
Output: The optimal forwarding sequence S^e

1: $S^e \leftarrow ()$
2: **repeat**
3: $MinEEC \leftarrow \infty$
4: **for** $i = 1$ to m **do**
5: **if** NOT $s^e_i \in S^e$ **then**
6: $S \leftarrow s^e_i \cup S^e$
7: **if** $EEC_e(S) < MinEEC$ **then**
8: $j \leftarrow i$
9: $MinEEC \leftarrow EEC_e(s)$
10: **end if**
11: **end if**
12: **end for**
13: $s^e \leftarrow s^e_j \cup S^e$
14: **until** $EDR_e(S^e) \leq R$

selection process continues until the EDR value of current forwarding sequence reaches bound R. Since in the worst case we need to add all nodes in the full sequence to the final forwarding sequence, the complexity of this greedy algorithm is $\mathcal{O}(DT)$.

6.4 The Impact of EDR Constraints on Optimality

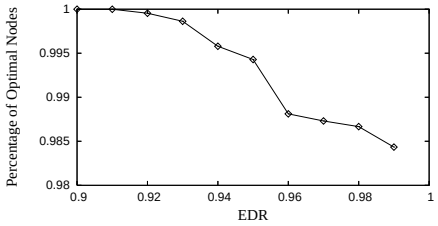


Fig. 7. Percentage of Optimality vs. EDR

We note that the EDR bound R imposes a non-convex constraint on the EED and EEC optimization problems. To optimize the forwarding sequence efficiently, the optimization processes described in Sections 6.2 and 6.3 first identify an optimal forwarding sequence under unconstrained search space. If the resulting sequence satisfies the EDR bound R, it is also an optimal solution to the original constrained problem. However, it is also possible that the resulting sequence violates the constraint especially when the EDR bound R is very high. In this case, we select the optimal EDR forwarding sequence from S^e_i, where i is the minimal value leads to $EDR_e(S^e) \geq R$ to satisfy the constraints (instead of achieving optimal EED or EEC).

Obviously, if the percentage of constraint violation is high, our solution is not effective. To evaluate this issue, we studied the impact of a high EDR bound on the optimality of our solution. Figure 7 shows the percentage of optimality under different EDR bounds. Clearly, our solution is very effective in identifying optimal solutions. For example, even with a 99% delivery ratio, 98.4% solutions are optimal.

6.5 Special Cases: ETX and DESS

We note that when nodes in the network are always active with no sleeping schedules, our EDR, EED, and EEC metrics and corresponding forwarding sequences default into those of the ETX solution. In addition, when all radio links among neighboring nodes are perfect, EDR, EED, and EEC default into those in the DESS solution. To a certain degree, we argue that EDR, EED, and EEC metrics are more generic data forwarding metrics, considering both link quality and sleep latency. In other words, ETX and DESS are two special cases of a more generic DSF solution. To validate this empirically, we will show such a convergence in the evaluation section later.

6.6 On Link Quality Change

The measurement of link quality plays an important role in our DSF design. In practice, however, link quality is affected by many environmental factors and changes over time. To achieve low-cost and accurate link quality estimation, we can adopt state-of-the-art solutions such as MultiHopLQI [34] and Four-bit link estimation [35]. In addition, through many empirical studies [36], [37], many researchers have revealed that although link quality changes noticeably over a long period of time, changing rate is slow. Therefore measurements of the link quality can be updated at a relatively large interval (e.g., once every ten minutes), which further reduce the system overhead for link quality estimation.

7 Implementation and Evaluation

We have implemented a complete version of the DSF forwarding scheme on the TinyOS/Mote platform in nesC with 20 MicaZ motes. To compare performance, we also implemented ETX [1] on the motes. The major components of DSF implementation include neighbor discovery, link quality measurement, the forwarding sequence optimization algorithms discussed in Section 6, and data forwarding with an optimized forwarding sequence.

We use FTSP [38] for the purpose of time synchronization among motes and Deluge [39] for the purpose of wireless reprogramming. The compiled image occupies 27,398 bytes of code memory and 1,137 bytes of data memory.

This testbed experiment was repeated multiple times with different node placement and working schedules. The results show the similar trend that resulted in all the experiments, and we report one collected dataset from the experiments in the following subsection.

7.1 Performance Comparison

In this section, we describe and compare the empirical E2E delay and energy consumption for DSF and ETX. In the experiment, the source node sends 100 packets to the sink node with DSF of optimal EED and ETX forwarding scheme, respectively.

Figure 8 shows the E2E data delivery delay for DSF and ETX. The packets in the figure are sorted according to their E2E delay, making it clear that ETX experiences heavy penalties when its single-hop transmission has failed, since it has to wait for the fixed forwarding node to wake up again. In contrast, when DSF encounters a single-hop transmission failure, its capability to dynamically switch the forwarding node significantly reduces the E2E delay. For instance, among 100 sent packets, the maximal E2E data delivery delays for DSF and ETX are 4317ms and 15426ms respectively, while the average delays are 849ms and 3942ms.

In addition to the E2E delay, we are also interested in the energy consumption of the two comparing protocols. Figure 9 demonstrates the energy consumption (number of transmissions for a single packet delivery) for DSF and ETX. From the figure, we can see that ETX incurs a smaller number of transmissions than DSF. For example, all of the packet deliveries for ETX finished with a maximum of 9 transmissions, while about 84% of the packets for DSF arrived at the sink node within 9 transmissions. However, the DSF shows a better delay-energy efficiency than ETX. With the same 9 transmissions, the delay for DSF and ETX is 1785ms and 15426ms, respectively.

7.2 System Insights

In this section, we investigate the internal state of each sensor node and reveal the corresponding statistics for DSF.

Figure 10 demonstrates the greater diversity of forwarder link qualities for DSF over those for ETX. While almost all ETX forwarders have link qualities above 50%, the distribution of forwarder link qualities for DSF is roughly uniform and ranges from 3% to 97%. Such diversity in forwarder link qualities for DSF, along with its smaller E2E delay, leads us to conclude that unreliable links are also helpful in reducing E2E delays in low duty-cycle sensor networks.

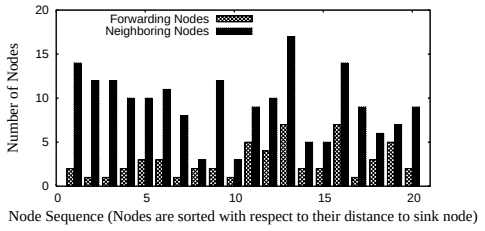


Fig. 11. Number of Forwarding Nodes vs. Number of Neighboring Nodes

Figure 11 shows the relationship between the number of nodes in the forwarding sequence and the number of available neighboring nodes for each sensor node in the experiment. The node sequence is ordered by the node's distance to the sink node. From this figure, we can see that most nodes have more than one node in their forwarding sequence. We also observe that generally, as the node's distance to the sink node increases, the number of forwarding nodes in the forwarding sequence also increases, since in order to maintain a certain data delivery ratio, the more distant nodes normally need to select more of their neighboring nodes. For example, the average number of forwarding nodes for the first 10 nodes is 1.8 nodes, while the value for the last 10 nodes is 3.8 nodes.

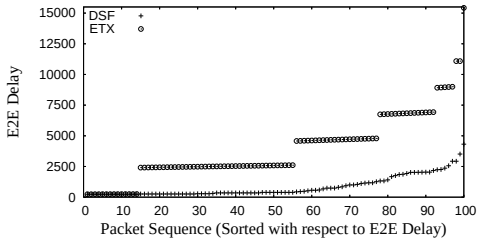


Fig. 8. E2E Data Delivery Delay

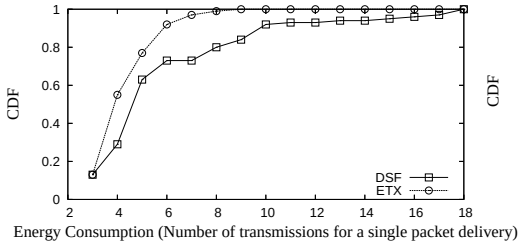


Fig. 9. Energy Consumption

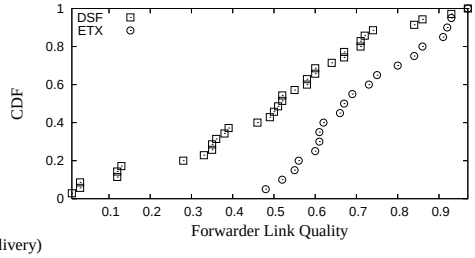


Fig. 10. Diversity in Forwarder Link Qualities

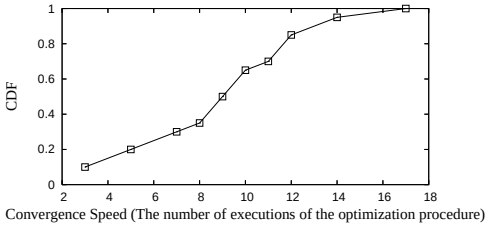


Fig. 12. DSF Convergence Speed

In addition to studying the distribution of the forwarding nodes, we also investigated how fast each node converges to its optimal forwarding sequence. To track the convergence speed of the DSF, we recorded the number of times that each node executed its forwarding sequence optimization procedure, as shown in Figure 12. There we can see that the forwarding sequence optimization process at all nodes converges within 18 executions of the optimization procedure. Furthermore, the number of executions of the optimization procedure at individual nodes is proportional to the number of neighboring nodes. This observation is also consistent with our complexity analysis for forwarding sequence optimization procedures.

8 Large-Scale Simulation

The results of the following system evaluation indicates that our proposed approaches can be efficiently implemented on resource-constrained sensor nodes and demonstrates their effectiveness in improving source-to-sink wireless communication between sensing nodes and sink. However, this evaluation was restricted to a limited design space. In order to understand the performance of the proposed scheme under numerous network settings, in this section, we provide simulation results with 250 nodes. We compared the performance of DSF with following state-of-the-art solutions:

- ETX [1] by Douglas S. J. De Couto et al. in Mobicom'03.
- PRR×D [2] by Karim Seada et al. in SenSys'04.
- DESS [3] by Gang Lu et al. in INFOCOM'05.

8.1 Simulation Setup

In the simulation, we deployed 250 sensor nodes randomly in a 150m×150m square field. A sink was positioned in the center of the deployment field, and each sensor node sent its packet to the sink over multiple hops. The radio model was implemented according to [40], which considers both temporal and spatial oscillations of the radio links and has several adjustable parameters. Except as otherwise specified, we set these parameters strictly according to the CC2420 radio hardware specification [41]. These parameters accurately reflect the performance of MicaZ motes in that they have the same modulation method, encoding method, frame length and path loss exponent.

In all experiments, we set the sender retransmission time bound T equals 200τ, which is also the length of the node working schedule. Each experiment was repeated 30 times with different random seeds, node deployments, and node working schedules. Data collected at each node was obtained by averaging 1000 source-to-sink communications. The 95% confidence intervals are within 1∼10% of the means.

8.2 Performance Evaluation

This section compares the data delivery ratio, E2E delay and energy consumption per delivered packet of source-to-sink communications among DSF, ETX, PRR×D, and DESS under different link qualities and duty cycles.

For the simulation of different link qualities, we first used CC2420 radio specifications to obtain the neighbor table for each sensor node, then set the pairwise link quality according to the simulation configurations.

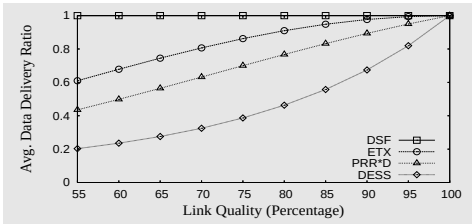
In following three subsections, evaluation figures for optimizing metrics are shaded to highlight their performances.

8.2.1 Optimizing Expected Delivery Ratio

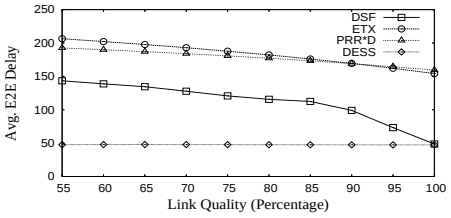
In this section, we examine the performance difference among DSF with optimal EDR, ETX, PRR×D, and DESS under different link qualities and duty cycles.

Varying Link Qualities: Figure 13(a) shows the data delivery ratio among the four compared schemes. The figure clearly shows that under the low link qualities, ETX, PRR×D and DESS can deliver only a very small portion of packets, while DSF with optimal EDR is able to deliver most of the packets to the sink node. For example, when the link quality is 55%, DSF delivers 99.9% of packets, while ETX, PRR×D, and DESS deliver only 61.0%, 43.5% and 20.3% of packets, respectively. Therefore, when the data delivery ratio is the primary design goal of a sensornet application, DSF would be a good choice for the system.

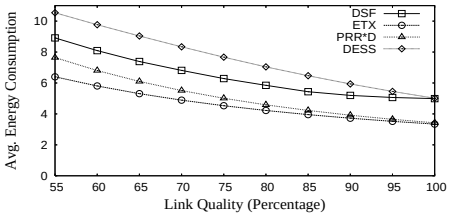
Figure 13(b) and Figure 13(c) show the corresponding E2E delay and energy consumption for four schemes. From Figure 13(b), we observe that DESS has the smallest and most constant E2E delay at all link qualities because at each hop, DESS would attempt to transmit its packet to the forwarder only once on the shortest delay path during



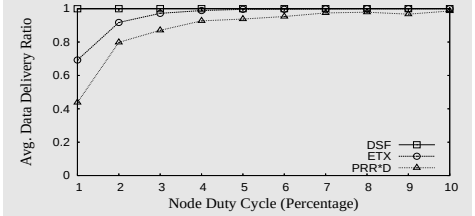
(a) Delivery Ratio vs. Link Quality



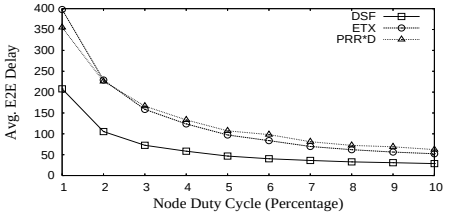
(b) E2E Delay vs. Link Quality



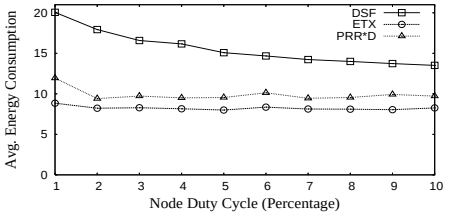
(c) Energy vs. Link Quality



(d) Delivery Ratio vs. Duty Cycle

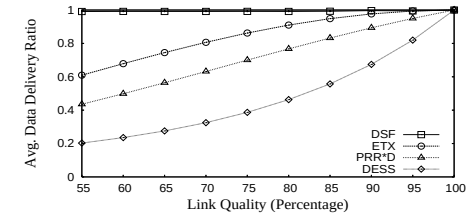


(e) E2E Delay vs. Duty Cycle

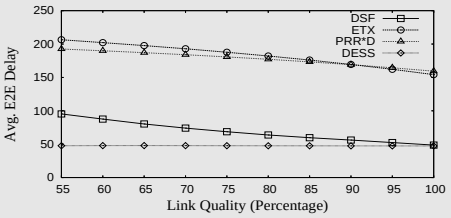


(f) Energy vs. Duty Cycle

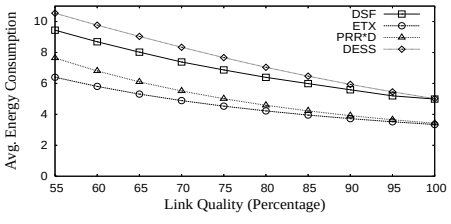
Fig. 13. Optimizing Expected Delivery Ratio (EDR)



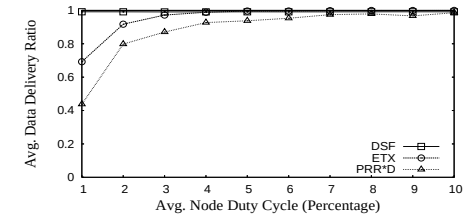
(a) Delivery Ratio vs. Link Quality



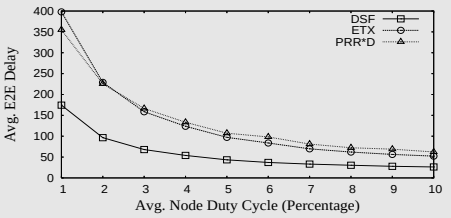
(b) E2E Delay vs. Link Quality



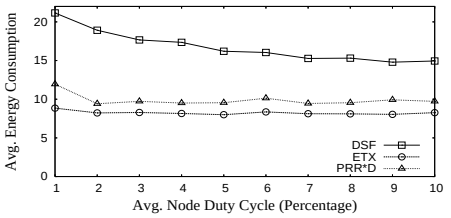
(c) Energy vs. Link Quality



(d) Delivery Ratio vs. Duty Cycle

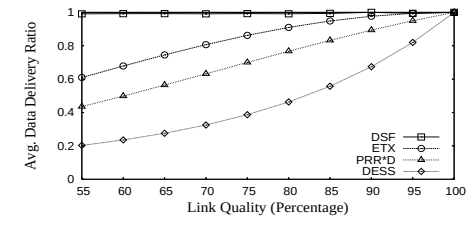


(e) E2E Delay vs. Duty Cycle

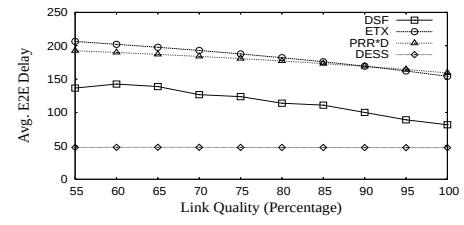


(f) Energy vs. Duty Cycle

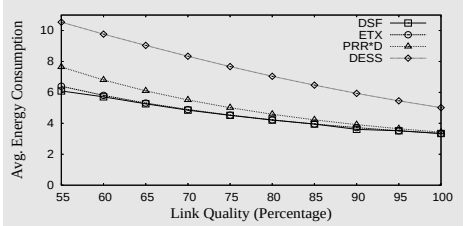
Fig. 14. Optimizing Expected E2E Delay (EED)



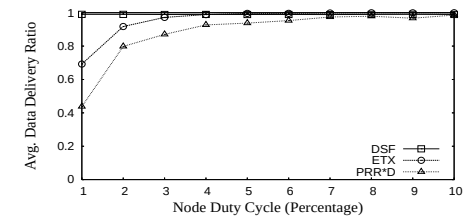
(a) Delivery Ratio vs. Link Quality



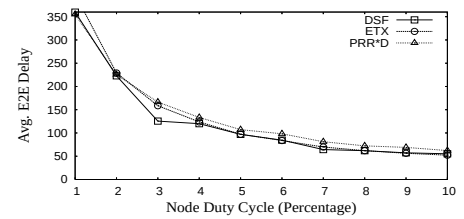
(b) E2E Delay vs. Link Quality



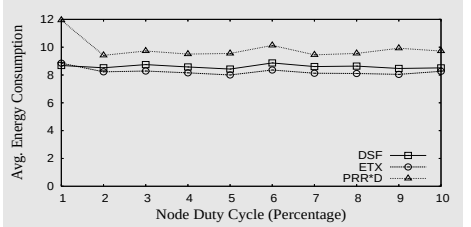
(c) Energy vs. Link Quality



(d) Delivery Ratio vs. Duty Cycle



(e) E2E Delay vs. Duty Cycle



(f) Energy vs. Duty Cycle

Fig. 15. Reducing Expected Energy Consumption (EEC)

one round of the node working schedule. Therefore, all the packets for DESS that reach the sink node are those for which every single-hop transmission is successful with one single attempt, and that consequently represent the minimal possible delivery delay, which is a constant value. At the same time, however, DESS experiences the largest packet loss among the four compared schemes. DSF, on the contrary, has the largest data delivery ratio though a smaller E2E delay than ETX and PRR×D. However, DSF's high data delivery ratio also incurs energy penalties.

From Figure 13(c), we can see that DSF has a slightly higher energy consumption per delivered packet than ETX and PRR×D since it attempts more transmissions and delivers more packets than these schemes. DESS ignores the link quality completely, has a very low data delivery ratio, and wastes much energy on transmitting packets that do not arrive at the sink node, therefore having the largest energy consumption per delivered packet. For instance, at a link quality of 55%, the per-delivered packet energy consumption for DSF, ETX, PRR×D, and DESS is 8.91, 6.40, 7.64 and 10.54, respectively.

Varying Duty Cycles: Figure 13(d) reports the data delivery ratio under different node duty cycles. It shows that under all node duty cycles, DSF with optimal EDR has a higher data delivery ratio than ETX and PRR×D. As the node duty cycle increases, the data delivery ratio for all schemes increases as well. For example, the delivery ratio for DSF, ETX, and PRR×D increases from 99.9%, 69.3%, and 43.8% to 100%, 99.9%, and 98.6%, respectively, when duty cycle increases from 1% to 10%. Figure 13(e) shows that the corresponding E2E delay for DSF is smaller than the other two baseline schemes, even with a higher data delivery ratio. Figure 13(f) shows again that the high data delivery ratio of DSF results in higher energy consumption.

8.2.2 Optimizing Expected E2E Delay

In this section, we examine the performance difference among DSF with optimal EED, ETX, PRR×D, and DESS under different link qualities and duty cycles. For optimal EED at each node, we set the data delivery ratio bound as 99%.

Varying Link Qualities: Figure 14(b) shows the end-to-end delay for four forwarding schemes under different link qualities. At link qualities less than 100%, the E2E delay is larger for DSF than for DESS, for the reason mentioned in the previous subsection. Meanwhile, the E2E delay for DSF is much smaller than for ETX and PRR×D. For example, at a link quality of 90%, the E2E delay for DSF, ETX, and PRR×D is 56.2, 169.4, and 178.3, respectively. When the link quality reaches 100%, the results for DSF with optimal EED converges with those of DESS. In Figure 14(c), we can see that the energy consumption for DSF is still higher than that for ETX and PRR×D. However, we also observe that DSF is more delay-energy efficient than the other schemes. For example, when the link quality is 80%, the per-energy delay for DSF, ETX, PRR×D, and DESS is 10.07, 47.41, 50.36, and 14.61, respectively.

Varying Duty Cycles: Figure 14(e) shows the end-to-end communication delay under different node duty cycles. There we can see that DSF has a smaller delay than the baseline schemes under all duty cycles while retaining a high data delivery ratio (Figure 14(d)). The overall energy consumption for DSF is still higher than that for the other schemes. However, as mentioned before, the per-energy delay for DSF is much smaller than for ETX and PRR×D. For example, at a duty cycle of 5%, the per-energy delay for DSF, ETX, and PRR×D is 3.82, 14.08, and 16.33, respectively.

8.2.3 Reducing Expected Energy Consumption

This section presents the performance differences among DSF with optimal EEC, ETX, PRR×D, and DESS under different link qualities and duty cycles. For an optimal EEC at each node, we set the data delivery ratio bound as 99%.

Varying Link Qualities: In Figure 15(c), energy consumption for DSF approaches the ETX at all link qualities while maintaining high data delivery ratio. For example, when link quality is 70%, the energy consumption for DSF, ETX, PRR×D, and DESS is 4.21, 4.21, 4.59, and 7.04, respectively. When link quality approaches 100%, DSF converges to the ETX in terms of energy consumption. In addition, with equivalent energy consumption, the E2E delay for DSF is smaller than for ETX and PRR×D. At a link quality of 80%, the E2E delay for DSF, ETX, and PRR×D is 113.95, 182.13, and 197.18, respectively. Interestingly, we notice that under optimal EEC, DSF does not converge to the DESS when link quality reaches 100%, because when optimizing EEC, DSF would seek the delivery path with the minimum number of transmissions instead of the minimum E2E Delay.

Varying Duty Cycles: Figure 15(f) shows the energy consumption under different node duty cycles. From the figure, we observe that the energy consumption for DSF approaches that of ETX and is better than that of PRR×D. With a higher data delivery ratio (Figure 15(d)) and comparable energy consumption, the end-to-end delay for DSF is still smaller than for the baseline schemes.

8.3 Insights

In the previous section, we saw the significant improvement of the source-to-sink communication for DSF over ETX, PRR×D, and DESS. In this section, we reveal the underlying reasons why DSF provides better performance than those state-of-the-art solutions.

8.3.1 Diversity in Link Quality

Both ETX and PRR×D generally prefer reliable links and try to avoid highly unstable links. While this intuitive approach holds well in traditional wireless networks, we saw that as node duty cycle decreases, the delay of such schemes becomes excessive since the time spent on waiting for the forwarder to wake up again is no longer tolerable. Figure 16 shows the CDF curve of the forwarder's link qualities for 200 randomly sampled senders from DSF, ETX, and PRR×D. From the figure, we can see that the distribution of DSF link quality is roughly uniform, with no obvious range being favored, while ETX and PRR×D select much more reliable links. This observation strengthens our understanding

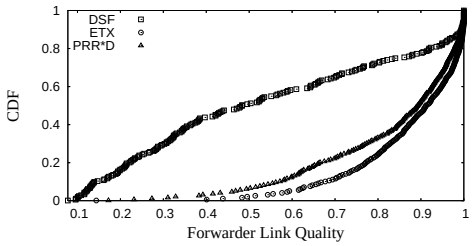


Fig. 16. Diversity in Forwarder Link Qualities (Average # of Neighbors=6)

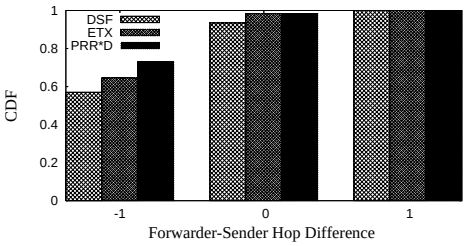


Fig. 17. Forwarder-Sender Hop Differences (Average # of Neighbors=6)

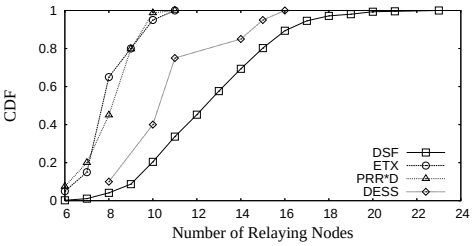


Fig. 18. Diversity in Delivery Paths (# of Neighbors=6)

that unreliable links are as useful as highly reliable links for minimizing the source-to-sink communication delay in low duty-cycle networks.

8.3.2 Implications of Packet Forward Direction

The metric of PRR×D tries to balance the distance advanced from the sender to a forwarder and the link quality between them. Similarly, the minimized ETX and DESS paths also prefer to move the packet forward. However, this is not the case for EED. Figure 17 shows the forwarder-sender hop difference for 200 random sampled nodes from DSF, ETX and PRR×D. In contrast to other schemes, from Figure 17, we observe that a DSF sender may transmit its packet to a forwarder with smaller hop, same hop or even larger hop. More specifically, for the 200 sampled nodes, only 57.1% forwarders have smaller hop count than the sender, while 36.4% and 6.5% forwarders have same and larger hop count respectively. In contrast, ETX and PRR×D forward 64.6% and 73.2% packets to the smaller hop-count nodes while almost never send packets to larger hop-count nodes.

Adding the observations from Figure 16 and Figure 17, we can conclude that link quality, hop counter or the combination of the two have little implications on the selection of forwarders in order to minimize the E2E delay in the extremely low-power sensor networks.

8.3.3 Diversity in Delivery Paths

In the previous subsection, we demonstrated that picking low-quality links is beneficial in low duty-cycle sensor networks for reducing the source-to-sink communication delay. In this section, we show the greater diversity of delivery paths for DSF over those for ETX, PRR×D, and DESS. In the simulation setup, 150 nodes are deployed in a $160m \times 160m$ field. Forty source nodes on the edge of the field send their packets to the sink node located in the center of the field. In Figure 18, we show the number of nodes that relay the packets sent by the source nodes during 100-packet delivery processes for DSF, ETX, PRR×D, and DESS. Clearly, DSF explores a much larger neighbor space than the other three schemes in these 100 packet transmission processes. For example, the maximum number of relaying nodes for DSF, ETX, PRR×D, and DESS is 23, 11, 11, and 16, respectively. Furthermore, in Figure 19, we also visualize the delivery paths for DSF, ETX, PRR×D and DESS for a source node in the southeast corner of the field. In Figure 19, we plot the nodes that relay the packets sent by the source in 10 packet delivery processes. From the Figure, we can see that DSF clearly explores a much more larger neighbor space than other three schemes in these 10 packet transmission processes. These two sets of figures again demonstrates DSF's adaptability to the presence of unreliable radio links and the low duty-cycle of sensor nodes.

9 Conclusion

In this work, we propose a dynamic switch-based forwarding (DSF) scheme for extremely low duty-cycle sensor networks, which addresses the combined effect of unreliable radio links and sleep latency in data forwarding. We derive a distributed model for data delivery ratio (EDR), E2E delay (EED), and energy consumption (EEC) at individual nodes and optimize the forwarding action in terms of these three metrics. To evaluate the performance of DSF, we have fully implemented the DSF in a network of 20 MicaZ motes and performed extensive simulation with various network configurations. The results demonstrate that DSF significantly improves source-to-sink communication over several state-of-the-art solutions in low duty-cycle sensor networks with unreliable radio links.

Acknowledgement

This work was supported in part by Singapore University of Technology and Design grant SRG-ISTD-2010-002, NSF grant CNS-0917097, CNS-0845994 and CNS-0720465. We also received partial support from InterDigital Company.

References

[1] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A High Throughput Path Metric for Multi-Hop Wireless Routing," in *MOBICOM'03*, 2003.

[2] K. Seada, M. Zuniga, A. Helmy, and B. Krishnamachari, "Energy-efficient Forwarding Strategies for Geographic Routing in Lossy Wireless Sensor Networks," in *SenSys '04*, 2004.

[3] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel, "Delay Efficient Sleep Scheduling in WIreless Sensor Networks," in *INFOCOM'05*, 2005.

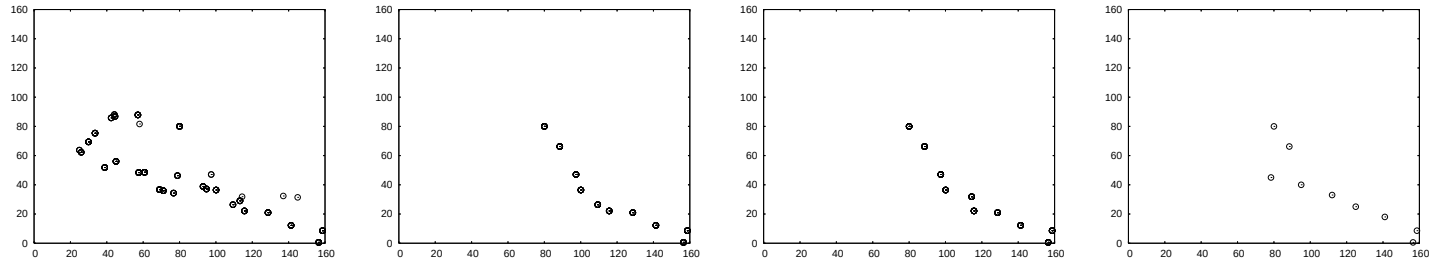
[4] Y. Gu and T. He, "Data Forwarding in Extremely Low Duty-Cycle Sensor Networks with Unreliable Communication Links," in *SenSys '07*, 2007.

[5] H. Kiehne, *Battery Technology Handbook*. Marcel Dekker, 2003.

[6] J. Jeong, Y. Gu, T. He, and D. Du, "VISA: Virtual Scanning Algorithm for Dynamic Protection of Road Networks," in *INFOCOM '09*, 2009.

[7] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks," in *SenSys'03*, 2003.

[8] G. S. Kasbekar, Y. Bejerano, and S. Sarkar, "Lifetime and coverage guarantees through distributed coordinate-free sensor activation," in *MobiCom '09*, 2009.



(a) EED Delivery Path (b) ETX Delivery Path (c) PRR×D Delivery Path (d) DESS Delivery Path

Fig. 19. Delivery Paths

[9] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance In Dense Wireless Sensor Networks," in *Sensys 03*, 2003.

[10] G. Zhou, T. He, and J. A. Stankovic, "Impact of Radio Irregularity on Wireless Sensor Networks," in *MobiSys'04*, June 2004.

[11] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," in *SenSys 2003*, 2003.

[12] M. Zamalloa, K. Seada, B. Krishnamachari, and A. Helmy, "Efficient geographic routing over lossy links in wireless sensor networks," *ACM TOSN*, vol. 4, no. 3, 2008.

[13] S. Lee, K. J. Kwak, and A. T. Campbell, "Solicitation-Based Forwarding for Sensor Networks," in *SECON'06*, 2006.

[14] L. Sang, A. Arora, and H. Zhang, "On Exploiting Asymmetric Wireless Links via One-way Estimation," in *MobiHoc '07*, 2007.

[15] O. Dousse, P. Mannersalo, and P. Thiran, "Latency of wireless sensor networks with uncoordinated power saving mechanisms," in *MobiHoc '04*, 2004.

[16] A. Keshavarzian, H. Lee, and L. Venkatraman, "Wakeup Scheduling in Wireless Sensor Networks," in *MobiHoc '06*, 2006.

[17] W. Lai and I. C. Paschalidis, "Sensor network minimal energy routing with latency guarantees," in *MobiHoc '07*, 2007.

[18] L. Su, C. Liu, H. Song, and G. Cao, "Routing in intermittently connected sensor networks," in *ICNP*, 2008.

[19] L. Su, B. Ding, Y. Yang, T. Abdelzaher, G. Cao, and J. Hou, "ocast: Optimal multicast routing protocol for wireless sensor networks," in *ICNP '09*, 2009.

[20] F. Wang and J. Liu, "Duty-cycle-aware broadcast in wireless sensor networks," in *INFOCOM'09*, 2009.

[21] Y. Gu, T. He, M. Lin, and J. Xu, "Spatiotemporal delay control for low-duty-cycle sensor networks," in *RTSS*, 2009.

[22] J. Polastre and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," in *SenSys'04*, November 2004.

[23] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *INFOCOM*, 2002.

[24] Y. Sun, O. Gurewitz, and D. B. Johnson, "Ri-mac: a receiver-initiated asynchronous duty cycle mac protocol for dynamic traffic loads in wireless sensor networks," in *SenSys '08*, 2008.

[25] P. Suriyachai, J. Brown, and U. Roedig, "Time-critical data delivery in wireless sensor networks," in *DCOSS '10*, 2010.

[26] M.-E. Felemban, M.-C.-G. Lee, and M.-E. Ekici, "Mmspeed: Multipath multi-speed protocol for qos guarantee of reliability and timeliness in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 6, 2006.

[27] M. Strasser, A. Meier, K. Langendoen, and P. Blum, "Dwarf: delay-aware robust forwarding for energy-constrained wireless sensor networks," in *DCOSS'07*, 2007.

[28] K. Pister and L. Doherty, "Tsmp: Time synchronized mesh protocol," in *DSN'08*, 2008.

[29] S. Munir, S. Lin, E. Hoque, S. M. S. Nirjon, J. A. Stankovic, and K. Whitehouse, "Addressing burstiness for reliable communication and latency bound generation in wireless sensor networks," in *IPSN '10*, 2010.

[30] Q. Cao, T. Abdelzaher, T. He, and J. Stankovic, "Towards Optimal Sleep Scheduling in Sensor Networks for Rare Event Detection ," in *IPSN'05*, 2005.

[31] S. He, J. Chen, D. Yau, H. Shao, and Y. Sun, "Energy-efficient capture of stochastic events by global- and local-periodic network coverage," in *MobiHoc*, 2009.

[32] Y. Wu, S. Fahmy, and N. B. Shroff, "Energy Efficient Sleep/Wake Scheduling for Multi-Hop Sensor Networks: Non-Convexity and Approximation Algorithm," in *INFOCOM*, 2007.

[33] W. Shu, X. Liu, Z. Gu, and S. Gopalakrishnan, "Optimal sampling rate assignment with dynamic route selection for real-time wireless sensor networks," in *RTSS '08*, 2008.

[34] *MultiHopLQI*, TinyOS, http://www.tinyos.net/tinyos-1.x/tos/lib/MultiHopLQI.

[35] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis, "Four-bit wireless link estimation," in *Hotnet'07*, 2007.

[36] S. Lin, G. Zhou, K. Whitehouse, Y. Wu, J. A. Stankovic, and T. He, "Towards stable network performance in wireless sensor networks," in *RTSS '09*, 2009.

[37] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis, "Understanding the causes of packet delivery success and failure in dense wireless sensor networks," in *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, 2006.

[38] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The Flooding Time Synchronization Protocol," in *SenSys'04*, 2004.

[39] J. Hui and D. Culler, "The Dynamic Behavior of a Data Dissemination Protocol for Network Programming at Scale," in *SenSys'04*, 2004.

[40] M. Zuniga and B. Krishnamachari, "Analyzing the Transitional Region in Low Power Wireless Links," in *IEEE SECON'04*, 2004.

[41] *CC2420 Product Information and Data Sheet*, Chipcon, avaiable at http://www.chipcon.com/.

Yu (Jason) Gu is currently an assistant professor in the Pillar of Information System Technology and Design at the Singapore University of Technology and Design. He received the Ph.D. degree from the University of Minnesota, Twin Cities in 2010. Dr. Gu is the author and co-author of over 20 papers in premier journals and conferences. His publications have been selected as graduate-level course materials by over 20 universities in the United States and other countries. His research includes Networked Embedded Systems, Wireless Sensor Networks, Cyber-Physical Systems, Wireless Networking, Real-time and Embedded Systems, Distributed Systems, Vehicular Ad-Hoc Networks and Stream Computing Systems. Dr. Gu is a member of ACM and IEEE.

Tian He is currently an associate professor in the Department of Computer Science and Engineering at the University of Minnesota-Twin City. He received the Ph.D. degree under Professor John A. Stankovic from the University of Virginia, Virginia in 2004. Dr. He is the author and co-author of over 90 papers in premier sensor network journals and conferences with over 4000 citations. His publications have been selected as graduate-level course materials by over 50 universities in the United States and other countries. Dr. He has received a number of research awards in the area of sensor networking, including four best paper awards (MSN 2006 and SASN 2006, MASS 2008, MDM 2009). Dr. He is also the recipient of the NSF CAREER Award 2009 and McKnight Land-Grant Professorship 2009-2011. Dr. He served a few program chair positions in international conferences and on many program committees, and also currently serves as an editorial board member for four international journals including ACM Transactions on Sensor Networks. His research includes wireless sensor networks, intelligent transportation systems, real-time embedded systems and distributed systems, supported by National Science Foundation and other agencies. Dr. He is a member of ACM and IEEE.