

# Dynamic Switching-based Reliable Flooding in Low-Duty-Cycle Wireless Sensor Networks

Long Cheng<sup>\*†</sup>, Yu Gu<sup>\*</sup>, Tian He<sup>†</sup> and Jianwei Niu<sup>†</sup>

<sup>\*</sup>Singapore University of Technology and Design, Singapore

<sup>†</sup>University of Minnesota, Twin Cities, USA

<sup>‡</sup>State Key Lab of Software Development Environment, Beihang University, Beijing 100191, China

**Abstract**—Reliable flooding in wireless sensor networks (WSNs) is desirable for a broad range of applications and network operations, and has been extensively investigated. However, relatively little work has been done for reliable flooding in low-duty-cycle WSNs with unreliable wireless links. It is a challenging problem to efficiently ensure 100% flooding coverage considering the combined effects of low-duty-cycle operation and unreliable wireless transmission. In this work, we propose a novel dynamic switching-based reliable flooding (DSRF) framework, which is designed as an enhancement layer to provide efficient and reliable delivery for a variety of existing flooding tree structures in low-duty-cycle WSNs. The key novelty of DSRF lies in the dynamic switching decision making when encountering a transmission failure, where a flooding tree structure is dynamically adjusted based on the packet reception results for energy saving and delay reduction. DSRF is distinctive from existing works in that it explores both poor links and good links on demand. Through comprehensive performance comparisons, we demonstrate that, compared with the flooding protocol without DSRF enhancement, DSRF effectively reduces the flooding delay and the total number of packet transmission by 12% ~ 25% and 10% ~ 15%, respectively. Remarkably, the achieved performance is close to the theoretical lower bound.

## I. INTRODUCTION

Wireless sensor networks (WSNs) used for monitoring and surveillance purposes are usually expected to last over several months or years. Since sensor nodes normally have very limited energy supply, one of the primary design goals in WSNs is to extend network lifetime. Existing works [1], [2] have empirically demonstrated that 1) the radio communication dominates the power consumed in WSNs compared with the energy consumption for processing, and 2) idle listening consumes a significant amount of energy in radio communication, approximately the same as in receiving and sending mode. To reduce the power consumption of sensor nodes, a widely employed approach is to schedule each node's duty-cycle (also called wake/sleep cycle), i.e., periodically switching between active and dormant states according to a wakeup schedule.

Flooding is a fundamental network service in WSNs, such as disseminating code updates, remote network configuration, maintenance, and route discovery, where messages from a source node (the sink node) are propagated to all other nodes in a connected, multi-hop network [3]–[5]. An unreliable flooding can result in undesirable network performance problems, e.g., incomplete code update may cause conflict

of sensing/communication among nodes, and low network coverage of flooding can lead to failures of query-response applications [6]. Therefore, several recent works [4], [7]–[9] have been proposed to guarantee 100% deliverability of flooding in WSNs.

In low-duty-cycle WSNs, nodes stay in dormant state most of time for energy saving and wake up for a very short period of time on demand. This poses several challenges for efficient flooding. First, since receivers do not always wake up at the same time in an asynchronous low-duty-cycle WSN, flooding is essentially achieved through a number of unicasts [5], and thus very costly. To improve the energy efficiency, we can utilize the broadcast nature of radio transmissions and flood packets over a schedule-based spanning tree [4], where nodes with a common parent wake up simultaneously to receive the broadcast packet. Second, experimental results reveal that wireless connectivity is unreliable especially for low-power embedded devices [10], [11]. An individual broadcast message can be lost or only received by partial neighbors. Due to the low-duty-cycle operation, a sender may go to sleep before its downstream receivers forwarding the packet, thus, getting reception results through overhearing receivers' subsequent transmissions is also infeasible in low-duty-cycle WSNs [4]. Therefore, the ARQ (Automatic Repeat reQuest)-based mechanism is normally needed to guarantee the single-hop reliable broadcast, e.g., we can use the software ACK supported by CC2420 [12] radio stack to acknowledge a broadcast packet. In addition, the low-duty-cycle operation also poses challenges on the sleep latency [13]. The interval between consecutive retransmissions would be large since a receiver may need to wait for a long time until the sender wakes up again.

To overcome the challenges above, in this work, we propose a novel dynamic switching-based reliable flooding (DSRF) framework, addressing the problem of 100% reliable flooding in low-duty-cycle WSNs with unreliable communication links. DSRF is designed to enhance the flooding efficiency over a scheduled flooding tree, e.g., the hopcount-based flooding tree [14], ETX-based flooding tree [15], or the energy optimal flooding tree [3]. The key idea is to dynamically adjust the flooding tree structure based on successful packet reception results on a per packet basis.

Unlike previous cooperative forwarding works which mainly avoid transmissions over poor links by switching to better links [16], [17], *DSRF explores both poor links and good links*

on demand. It tends to utilize the *cost free* links provided by sibling nodes to receive its lost flooding packets, which indicates that *even switching to worse links can still bring performance gain*. Our major contributions are as follows:

- To the best of our knowledge, this is the first work to propose a dynamic switching mechanism for schedule-based reliable flooding in low-duty-cycle WSNs. A dynamic switching-based reliable flooding (DSRF) framework is presented, which is compatible with most common flooding tree structures.
- We investigate the impact of wakeup schedule sequence on the flooding efficiency. In order to maximize the switching benefit, we design a heuristic wakeup schedule ranking algorithm with a low computational complexity.
- Through comprehensive performance comparisons, we demonstrate the energy efficiency and flooding latency of our DSRF design.

The remainder of this work is organized as following. Section II presents the network model and assumptions. Section III introduces the DSRF design in detail. Evaluation results are presented in Section IV. In Section V, we survey related work briefly. Finally, Section VI concludes this paper.

## II. PRELIMINARIES

In this section, we describe the network model and assumptions used in this work.

### A. Low-Duty-Cycle Model

We consider a multi-hop WSN with  $N$  nodes deployed in the sensing field. Each sensor node operates at a low-duty-cycle (e.g., 1% or less). A node only switches to the active state when it has data packets to send, or it is scheduled to receive data packets.

The working cycle  $T$  is equally divided into fixed length slots with fine granularity, called time slots. Let  $\tau$  denote the duration of one time slot, which is the minimum time unit of a working cycle. Due to the low-duty-cycle operation, the length of active time unit  $\tau$  is normally very limited [4], and data transmission suffers from a long sleep latency when encountering transmission failures. Sleep latency refers to the time that a sender spends on waiting for the receiver to wake up [13].  $t_k$  denotes the  $k_{th}$  wakeup slots in one working cycle  $T$ . For an active slot, we distinguish the receiving and sending states, called a receiving slot or a sending slot. If a sender has no data packet to send, it may keep asleep in its sending slot to save energy.

### B. Schedule-based Flooding Tree

In a low-duty-cycle WSN, schedule-based flooding tree synchronizes neighboring nodes in order to align their active slots for sending and receiving, which can greatly reduce idle-listening time [18], [19]. Fig. 1 shows an example of the network topology of a schedule-based flooding tree.  $[r : t_i]$  means that a node's receiving slot is  $t_i$ , and  $[s : t_j]$  indicates that its sending slot is  $t_j$ . In such a tree, the sending slot of a parent node overlaps with all its child nodes' receiving slots,

and sibling nodes in the same level are usually scheduled to transmit data at different time slots to avoid potential collisions [20]. For node  $i$ , it maintains the neighbor set  $N(i)$ , child set  $C(i)$ , and its parent node  $f(i)$ .

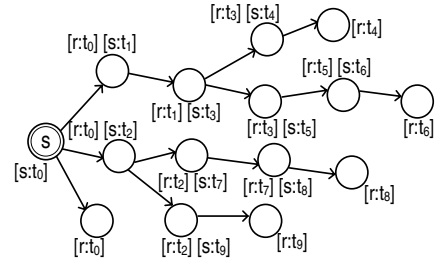


Fig. 1. In a schedule-based flooding tree, receivers align their receiving slots with the parent node's sending slot

### C. ARQ-based Reliable Flooding

We assume the ARQ-based mechanism is used to guarantee the flooding reliability in low-duty-cycle WSNs. If a node successfully receives a packet, it needs to return an ACK message to the sender. After collecting ACKs from all its receivers, the sender stops broadcasting that packet.

Since  $\tau$  is normally limited in low-duty-cycle setting, we assume  $\tau$  is assigned to complete a round-trip packet transmission, i.e., a data packet and a limited number of ACKs (up to  $M$ ) can be successfully transmitted [4]. This indicates that the maximum out-degree of a flooding tree is limited to  $M$ . For example, each ACK slot can be assigned to be  $864\mu s$  if we use the software ACK implementation for the CC2420 [12] radio stack.

### D. Assumptions

1) *Lossy and measurable links*: Existing works [10], [11] have shown that realistic link conditions in wireless networks can be highly unreliable due to many factors such as interference, attenuation, and channel fading. We assume that the MAC layer provides the link quality estimation service. There has been a lot of existing work on how to measure wireless link quality in an efficient and accurate manner [21], [22].

2) *Local time synchronization*: The local clocks of neighboring nodes are synchronized. Time synchronization for WSNs has been extensively studied in literatures, we can use a representative low-cost method, such as Glossy [23] or FTSP [24], to achieve efficient local time synchronization.

3) *Traffic Pattern*: The sink node has a sequence of data packets to flood to the whole network consecutively. Each packet has an identifier in the header. Each node maintains a data queue, incoming packets upon reception are queued until prior packets are delivered, following the FIFO (first in first out) policy. Since the traffic pattern discussed in this work is focused on the flooding traffic, we assume the queue size is large enough to buffer data packets injected by the sink node.

### III. MAIN DESIGN

This section first demonstrates a motivating example, then presents the DSRF architecture overview, followed by the detailed description of DSRF.

#### A. Motivation

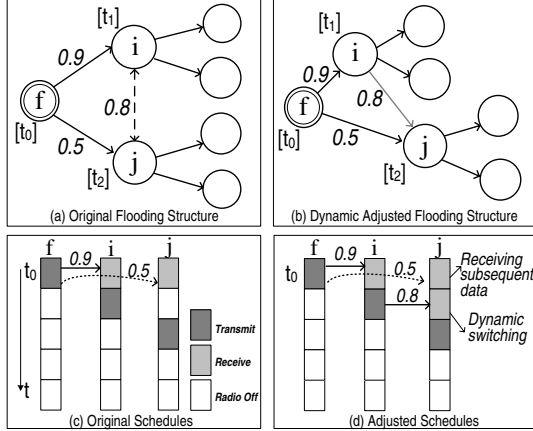


Fig. 2. Example of dynamic switching in DSRF

When the ARQ-based mechanism is used to guarantee the single-hop broadcast reliability over unreliable communication links, inevitably, the sender needs multiple retransmissions of the same packet to reach all child nodes, especially when the network density is high. Due to the low-duty-cycle operation, once a receiver fails to receive a packet, it has to wait for a long time until the sender wakes up again. Therefore, the traditional ARQ-based retransmission not only results in an undesirable high energy cost, but also suffers a large sleep latency.

Fig. 2 demonstrates a motivating example of DSRF design. As shown in Fig. 2(a), node  $f$  has two child nodes  $i$  and  $j$ . The weights of the directed edges are the corresponding PRRs (Packet Reception Ratio). Fig. 2(c) shows the original wakeup schedules, nodes  $f$ ,  $i$  and  $j$ 's sending slot is  $t_0$ ,  $t_1$  and  $t_2$ , respectively. When a packet is broadcasted from node  $f$  at time  $t_0$ , assume that  $j$  fails to receive the packet, but  $i$  receives it successfully. In this case,  $j$  has to wait for another  $T$  time when  $f$  wakes up again to launch the retransmission. Instead of waiting for one working cycle to receive the lost data packet,  $j$  can *dynamically switch* to  $i$  if it knows that  $i$  has already received the packet and switching can improve the network performance (we call  $i$  as a *helper* node).

If  $j$  decides to switch to  $i$ , it will keep awake at  $i$ 's sending slot to receive the lost packet. Fig. 2(b) and Fig. 2(d) show the adjusted tree structure and the adjusted wakeup schedules after switching, respectively.  $j$  assigns a receiving slot and synchronizes with  $i$ 's sending slot to receive the lost packet at  $t_1$ . In this example, since the sender has multiple data packets to flood,  $j$  still wakes up at  $t_0$  to receive the remaining packets from  $f$ . Let  $p_{fj}$  denote the PRR between a parent node  $f$  and its child node  $j$ , and  $p_{ij}$  denote the PRR between  $i$  and  $j$ . As

shown in Fig. 2,  $p_{fj} = 0.5, p_{ij} = 0.8$ . It is very likely that  $j$  can get the lost packet before its scheduled sending slot at  $t_2$ , given that  $p_{ij}$  is better than  $p_{fj}$ . Additionally, since  $f$  is released from the retransmission task to  $j$ , it can transmit the subsequent data packet in the next cycle, thus reducing the packet delivery delay.

We also observe that, given a flooding tree structure with the dynamic switching design, when encountering a transmission failure, a node tends to choose a sibling node with the best PRR as its parent node. One could argue that why not optimize the tree structure beforehand instead of switching at runtime, such as building an energy optimal tree [3]. We show in this work that even node  $j$  switching to a sibling node  $i$  with lower PRR could still save energy consumption and improve the latency. This is because,  $i$  is scheduled to forward packets to its child nodes no matter whether  $j$  switches or not, which we call a *cost free* link for  $j$  (at least one free transmission). In this case, DSRF can also be applied to an energy optimal tree to improve the flooding efficiency. This motivates us to design an architectural enhancement layer to provide efficient and reliable flooding over existing tree structures. Interestingly, from the evaluation results in Section IV, we observe that DSRF achieves considerable performance gain for the energy optimal tree [3] as well.

#### B. DSRF Architecture Overview

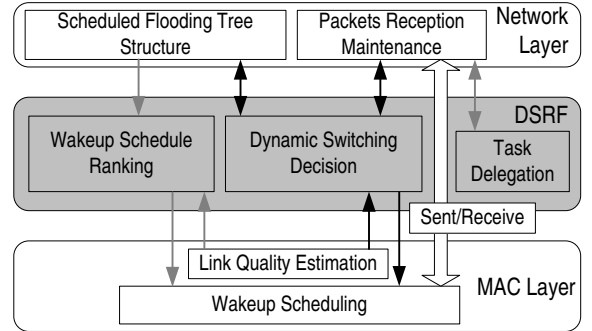


Fig. 3. DSRF Architecture Overview

The DSRF architecture is shown in Fig. 3. It consists of three core modules: wakeup schedule ranking, task delegation, and dynamic switching decision. For a parent node, given the available sending slots to be assigned to its child nodes, the wakeup schedule ranking module ranks the wakeup schedule of child nodes in order to maximize the switching gain. DSRF design takes the advantage of senders' spatial-temporal diversity. When a transmission failure occurs at a receiver, it makes a dynamic switching decision based on a cost analysis approach. Once deciding to switch to a sibling node (a helper node), it considers this helper node as its delegated parent node to receive the lost data packet, and dynamically adjusting the flooding tree structure and wakeup schedule. Then, it announces the switching decision to its original parent node and the helper node. The parent node updates the packet reception status and marks that the transmission task is handed

over. When a helper node receives the switching decision announcement, it adds the newly joined node as a receiver in its task delegation list, then notifies the packets reception maintenance module in the upper layer to take charge of the delivery. Once the lost packet is delivered to that receiver, the delegated task is removed from the task delegation list. Finally, the receiver will return to its normal duty-cycle schedule.

### C. DSRF Switching Procedure

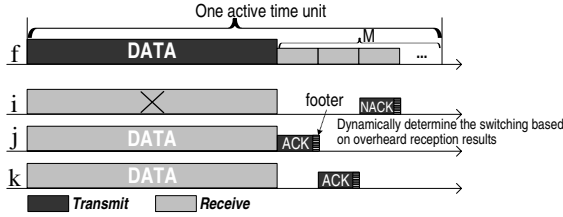


Fig. 4. Example of DSRF switching procedure

Since neighboring nodes with different parent nodes usually wake up at different sending slots in low-duty-cycle WSNs to avoid potential collisions [20], the switching naturally happens only between sibling nodes. In order to achieve the consensus among neighbors when switching happens, one straightforward approach is to assign a control message slot for the parent node, so that it makes the switching decision based on the collected ACKs and then broadcasts a message to announce the decision to all child nodes. However, this method is undesirable since it occupies one ACK slot and introduces extra overhead per transmission.

To reduce the communication overhead, we exploit the over-hearing feature of ACK transmissions, and design a consensus protocol for switching decision making in a lightweight manner. Fig. 4 shows an example of DSRF switching procedure. The parent node  $f$  broadcasts a data packet to its child nodes  $i$ ,  $j$  and  $k$ . Suppose  $j$  and  $k$  successfully receive the packet, while  $i$  fails to receive it. Following is the description of the DSRF switching procedure.

As shown in Fig. 4, each child node starts a backoff timer, where the value is inversely proportional to its priority in a predefined sequence, which will be discussed in Section III-E. When the timer expires, it will reply an ACK message, including the number of new child nodes it can accommodate, its sending slot, etc. Since node  $i$  fails to receive the packet from  $f$ , it overhears ACKs from other neighbors, looking for potential switching opportunities. Before its timer fires, according to the overheard ACKs, a switching decision is made locally based on a cost analysis approach, which will be introduced in detail in Section III-D. Suppose  $i$  decides to switch to  $j$ , a NACK (negative ACK) message is sent out on demand to announce the switching decision when its timer fires. In an ideal case, this NACK notifies the sender  $f$  as well as other sibling nodes that the transmission task will be shifted to a helper node  $j$ . Then,  $i$  considers  $j$  as its delegated parent node, and assigns one receiving slot to receive the lost

packet. If all child nodes either successfully receive the data or make switching decisions, the parent node  $f$  finishes the broadcasting task of the current data packet.

We notice that there may exist the hidden terminal problem if two non-neighboring sibling nodes return NACKs and they switch to the same helper node. For example,  $i$  and  $j$  may switch to  $k$  in the same active time slot, and  $j$  has a higher priority to return the ACK. To address this problem, in our design, if  $i$  does not overhear any ACK or NACK from  $j$  before its timer fires, it assumes that  $j$  switches to the same helper node  $k$ . Then, the switching decision will be reassessed, assuring that the number of child nodes of  $k$  does not exceed the limit  $M$ .

### D. Switching Decision Making

Since the energy cost required to listen or receive is about the same as the power consumption to transmit for typical sensor nodes [1], [2], we use the number of active time slots as the metric to evaluate the energy cost. The primary objective of the switching design is to improve the flooding energy efficiency, which means switching should only happen if it can reduce energy consumption. In some cases, a helper node may just enqueue a recently received packet since prior data packets have not been delivered. We also consider the queuing delay in making a switching decision.

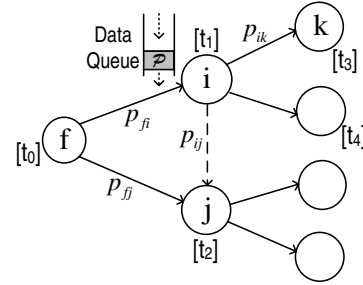


Fig. 5. Example of dynamic switching decision making

Fig. 5 shows an example of the dynamic switching decision making. The parent node  $f$  transmits a packet  $\mathcal{P}$  to child nodes  $i$  and  $j$  at  $t_0$ ,  $i$  and  $j$ 's sending slot is  $t_1$  and  $t_2$ , respectively. Suppose node  $j$  fails to receive the packet  $\mathcal{P}$  from  $f$ , we derive a lower bound of the expected retransmission cost.

$$C_{rx}^l = \frac{1}{p_{fj}} \cdot \left\{ \frac{\tau}{n} + \tau \right\} \quad (1)$$

where  $p_{fj}$  is the PRR between  $f$  and  $j$ .  $n$  is the number of child nodes which have not received the data packet from  $f$  based on  $j$ 's incomplete estimation. The cost of a transmission consists of the sending cost and receiving cost. Since  $j$  can only overhear ACKs from part of its sibling nodes, for any retransmission,  $\frac{\tau}{n}$  is the lower bound of the shared sending cost of  $f$  in terms of  $j$ . And  $\tau$  is the receiving cost of  $j$ . Therefore, Eq. 1 represents a lower bound of the expected energy cost if  $f$  retransmits the packet to  $j$  in the following working cycles.

As shown in Fig. 5, given a predefined sequence of returning ACKs for  $i$ 's child nodes, let  $p_{ik}$  denote the link quality of

the first node in the sequence. From the DSRF switching procedure, we know that the first node in the sequence has no chance to switch. Assume  $j$  overhears an ACK from  $i$ . The following information is contained in  $i$ 's ACK message: data identifier, its sending slot, the number of available ACK slots, a bit flag indicating whether there are prior packets in the queue ahead of packet  $\mathcal{P}$ , and  $p_{ik}$ . In our design,  $j$  only considers  $i$  as a potential helper node if  $i$  transmits  $\mathcal{P}$  in its immediate sending slot. Suppose  $j$  decides to switch to  $i$  to receive the lost packet, we derive an upper bound of the transmission cost with the switching decision.

$$\mathcal{C}_{sw}^u = \frac{1}{p_{ij}} \cdot 2\tau - \min\left\{\frac{1}{p_{ij}}, \frac{1}{p_{ik}}\right\} \cdot \tau \quad (2)$$

where  $p_{ij}$  denotes the PRR between  $i$  and  $j$ .  $2\tau$  is the typical cost of a transmission, and  $\frac{1}{p_{ik}}$  is the number of minimum expected free transmissions that  $i$  will broadcast the packet to its child nodes regardless of whether  $j$  switches to  $i$  or not. Since  $\min\left\{\frac{1}{p_{ij}}, \frac{1}{p_{ik}}\right\} \cdot \tau$  is the minimum energy saving  $i$  can provide to  $j$ , Eq. 1 is an upper bound of the transmission cost after switching.

Switching is expected to save energy, thus we have the switching decision criteria as in Ineq. 3. For node  $i$ , if there are multiple sibling nodes it can switch to, it will choose the one which can maximize the energy saving.

$$\mathcal{C}_{rx}^l - \mathcal{C}_{sw}^u > 0 \quad (3)$$

**Theorem 1.** *When  $i$  transmits the lost data packet in its immediate sending slot, if  $p_{ij}$  is larger than  $\frac{2p_{fj}}{p_{fj}+1}$ , the switching must be a beneficial decision. If  $p_{ij}$  is smaller than  $\frac{p_{fj}}{2}$ , no beneficial switching decision can be achieved, i.e., the lower bound of  $p_{ij}$  is  $\frac{p_{fj}}{2}$  for  $j$  making a switching decision to  $i$ .*

*Proof:* From Eq. 1 and Eq. 2, we know that the range of  $\mathcal{C}_{rx}^l$  is  $[\frac{\tau}{p_{fj}}, \frac{2\tau}{p_{fj}}]$ , and the range of  $\mathcal{C}_{sw}^u$  is  $[\frac{\tau}{p_{ij}}, \frac{2\tau}{p_{ij}} - \tau]$ . To ensure that  $\mathcal{C}_{sw}^u < \mathcal{C}_{rx}^l$  unconditionally, the upper range of  $\mathcal{C}_{sw}^u$  must be smaller than the lower range of  $\mathcal{C}_{rx}^l$ . We have  $(\frac{2\tau}{p_{ij}} - \tau) < \frac{\tau}{p_{fj}}$ , i.e.,  $p_{ij} > \frac{2p_{fj}}{p_{fj}+1}$ .

When the lower range of  $\mathcal{C}_{sw}^u$  is larger than the upper range of  $\mathcal{C}_{rx}^l$ , there is no beneficial switching possibility. In this case, we have  $\frac{\tau}{p_{ij}} > \frac{2\tau}{p_{fj}}$ . That is,  $p_{ij} < \frac{p_{fj}}{2}$ . ■

From Theorem 1, a node can quickly filter out ineligible sibling nodes when making a switching decision. It also theoretically indicates that, *utilizing worse links may also bring performance gain in DSRF.*

### E. Wakeup Schedule Ranking

1) *Impact of Wakeup Schedule Sequence:* From above discussion, if node  $j$  switches to a helper node  $i$  whose immediate sending slot is later than that of  $j$ , this means  $j$  has to wait for another working cycle, thus incurring a long sleep latency. In our design, given a wakeup schedule sequence of a set of sibling nodes (i.e., the sequence of sending slots), a node should only switch to a sibling node who ranks ahead of it in the sequence, so that it can have a chance to receive the lost packet before its subsequent sending slot.

With dynamic switching, different wakeup schedule sequences lead to different probabilities that all child nodes receive the data packet *in one working cycle*. A higher probability of finishing the transmission task in one working cycle means less retransmissions are needed in the following working cycles, thus improving the energy efficiency and delivery latency. Given the available sending slots to be assigned to a set of child nodes, we shall rank the wakeup schedule to maximize the switching gain. Note that the assigned sending slots are not necessarily consecutive, it depends on the slots allocated by a certain slot allocation mechanism to these child nodes with a common parent. Since leaf nodes do not need to be assigned sending slots, we only consider non-leaf nodes.

2) *Heuristic Wakeup Schedule Ranking Algorithm:* For a parent node  $f$  broadcasting a packet  $\mathcal{P}$ , let  $P\{(n)\}$  denote the probability that all its  $n$  non-leaf child nodes have already received  $\mathcal{P}$  in one working cycle. We define the *optimal wakeup schedule ranking* problem, which is to find a permutation of the wakeup schedule sequence to maximize  $P\{(n)\}$ , in which the order indicates that nodes will wakeup to transmit packet in sequence. Let  $\pi$  be any one permutation, we have

$$P\{(n)|\pi\} = \prod_{k=1}^n p_k^* \quad (4)$$

where  $n$  is the number of  $f$ 's non-leaf child nodes.  $p_k^*$  denotes the probability of the  $k_{th}$  node receiving the packet before its sending slot with the consideration of dynamic switching.

For the  $k_{th}$  node in the sequence, it has  $k-1$  potential switching candidates. If failed to receive packet  $\mathcal{P}$  from  $f$ , the  $k_{th}$  node will switch to a node which has received the packet and has the highest PRR to it. For ease of calculation, we descendingly order the first  $k-1$  sibling nodes by the PRR to the  $k_{th}$  node when calculating  $p_k^*$ . We have

$$p_1^* = p_{f1} \\ p_k^* = p_{fk} + \bar{p}_{fk} \cdot \sum_{r=1}^{k-1} \left\{ \left( \prod_{m=0}^{r-1} \bar{p}_{fm} \right) \cdot p_{fr} \cdot p_{rk} \right\}, \quad k > 1 \quad (5)$$

where  $p_{fk}$  is the PRR between  $f$  and the  $k_{th}$  child node,  $r$  is the index of descending sequence of the first  $k-1$  sibling nodes.  $p_{fr}$  denotes the PRR between  $f$  and the  $r_{th}$  child node, and  $p_{rk}$  denotes the PRR between the  $r_{th}$  node and the  $k_{th}$  node.  $\bar{p}_{fm} = 1 - p_{fm}$ ,  $\bar{p}_{f0} = 1$ .  $(\prod_{m=0}^{r-1} \bar{p}_{fm}) \cdot p_{fr} \cdot p_{rk}$  is the probability that the  $k_{th}$  node receives the lost packet  $\mathcal{P}$  from the  $r_{th}$  node, given that all the first  $r-1$  sibling nodes fail to receive the packet, while the  $r_{th}$  node receives it. The rationale is that, if the  $k_{th}$  node fails to receive a packet from  $f$ , it prefers to switching to the  $1_{st}$  child node since the link quality between them is the highest. If the  $1_{st}$  child node also fails to receive the packet, it switches to the  $2_{nd}$  child node, and so on.

However, it is difficult to obtain the optimal wakeup schedule ranking, since sibling nodes usually have different ranking sequences to maximize respective  $p_k^*$ . A straightforward way to get the optimal wakeup schedule ranking is to enumerate all possible permutations and search the  $\max P\{(n)\}$ , which runs

---

**Algorithm 1:** Wakeup schedule ranking at parent node

---

**Input:** Child node set  $C(f)$ ,  $\forall i, j \in C(f)$ ,  $p_{fi}$ ,  $p_{ij}$ , and  $n$

**Output:** A permutation of  $C(f)$  (wakeup schedule sequence)

```
1  $C(v_f)$  are descendingly sorted according to PRR to  $f$ ,
   denote the permutation as  $\pi_n$ ;
2  $P\{(n)\}_{max} = P\{(n)|\pi_n\}$ ;
3 for  $k = n - 1; k \geq 1; k - -$  do
4   Append node  $k$  to the end of the sequence temporarily,
   denote the permutation as  $\pi_k$ ;
5   if  $P\{(n)|\pi_k\} \geq P\{(n)\}_{max}$  then
6      $P\{(n)\}_{max} = P\{(n)|\pi_k\}$ ;
7     Append node  $k$  to the end;
8   else
9     Continue;
10 Return the final permutation;
```

---

in  $O(n!)$  at each parent node. Together with recursively calculating  $p_k^*$  for each permutation, the computation complexity of this solution would be high for the resource constrained sensor nodes. Therefore, we propose an efficient heuristic wakeup scheduling algorithm with a low computational cost, as shown in Algorithm 1.

Since the node with the worst link quality to its parent node is the broadcasting bottleneck, we name it as the *bottleneck* node. By placing it at the end of a wakeup sequence, the probability of receiving a data packet before its sending slot for the bottleneck node is maximized. Intuitively, we first sort the child nodes in a descending order according to the PRR to the parent node (in Line 1). Then, we temporarily append the second last node to the end of the sequence (in Line 4). If the new permutation increases  $P\{(n)\}$ , we formally append it to the end of the sequence (in Line 7). Otherwise, we do not change its position. We observe that when calculating  $p_k^*$ , the gain involving more helper nodes becomes marginal. Therefore, in order to further reduce the computation complexity, we only consider the first potential helper node with the best PRR. That is,  $p_k^* \approx p_{fk} + \bar{p}_{fk} \cdot p_{fr} \cdot p_{rk}$ , where the  $r_{th}$  node ranks before the  $k_{th}$  node and has the highest PRR to it. The algorithm greedily checks each node one by one until reaching the first node, which runs in  $O(n^2)$ . Finally, an optimized wakeup schedule sequence is returned.

In DSRF switching procedure, since each node can only make a switching decision before its ACK returning backoff timer fires, the node which returns an ACK at the end has more switching chances. Therefore, the sequence of returning ACKs discussed in Section III-C is the same as the wakeup schedule sequence.

#### F. Fault Tolerance in DSRF

Since the wireless links are unreliable, we also consider the possibilities of ACK and NACK transmission failures.

1) *The loss of ACK:* To reduce the impact of ACK transmission failure, nodes whose ACK returning backoff timers are still pending, utilize overhearing to learn the reception results

correspondingly. And the reception results will be encoded in their ACK footers. For example, if M is 8, we use 8 bits to indicate the reception results. In this way, even if the parent node fails to receive an ACK, overhearing sibling nodes may implicitly confirm the successful reception. This method can also improve the robustness for the pure ARQ-based retransmission mechanism. A parent node will retransmit the packet if it does not receive the packet reception result for a child node.

2) *The loss of NACK:* DSRF can also recover from the loss of NACK. When a parent node  $f$  fails to receive a NACK from  $i$ , it just retransmits to  $i$  in the next working cycle. If a NACK from  $i$  is lost at a helper node  $j$ , it does not know  $i$ 's switching decision. Thus, it will not take charge of the forwarding task for  $i$ . In this case, the list of intended receivers is piggybacked in a data packet, where the short intra-group addresses for child nodes is used to reduce the message overhead. If  $i$  overhears the lost data, it will keep silent since it is not in the list. If  $i$  has not received the lost packet for a certain time (e.g., receiving a subsequent data packet), it will send a retransmission request for the lost data packet to its original parent node  $f$ . Then,  $f$  will retransmit the data to  $i$ . In this way, DSRF is able to recover from the control message transmission failures.

## IV. EVALUATION

In this section, we present performance evaluation results. We implement DSRF as an enhancement layer for reliable schedule-based flooding in the ns-2 simulator [25], and compare the performance with two other baseline solutions below.

- **Overhearing:** A receiver always overhears all potential transmissions from sibling nodes whose sending slots are scheduled earlier than its own in one working schedule. But it has to acknowledge the packet reception result to its original parent node when the parent node wakes up even if it has overheard the data packet from a sibling node.
- **Reliable Flooding w/o DSRF:** The ARQ-based mechanism is used to guarantee flooding reliability without DSRF enhancement.

#### A. Simulator details

1) *Simulation setting:* We generate 100 random topologies for each node density setting, ranging from 100 to 300, where the node density is defined as the number of nodes deployed in a  $200m \times 200m$  field. The node transmission range is set as 40 m. In all simulations, the sink node floods 10 data packets. We use the Nakagami fading model and derive the PRR which is related with the distance between two nodes [26]. The sink node is positioned at bottom left (0m,0m). We set the node working schedule length  $T$  equals  $200\tau$ , where one active time unit  $\tau$  is 50 ms. We implement a simple yet effective wakeup slot allocation and distribution algorithm, constructing a sequential collision-free schedule-based flooding tree. The data packet length is set to 100 bytes. All the results have been averaged over 100 rounds, and the related standard deviations are provided as error bars.

2) *Flooding tree structures*: Three different tree structures are built in a distributed manner, where  $M$  is set as 7, as shown in the following.

- *Hop-based Spanning Tree* [14]: Each node chooses a neighbor as its parent node with the minimum number of hops to reach the sink. Since only hopcounts are considered while link qualities are neglected, it is likely that a node chooses its parent node with a very unreliable link.
- *ETX-based Spanning Tree* [15]: It takes the effect of link quality on routing performance into account. Each node chooses a neighbor as its parent node with the minimum accumulated ETX value to reach the sink.
- *Heuristic Energy Optimal Tree* [3]: Each node selects a neighbor which has the best link quality (PRR) among those neighboring nodes while ensuring the network’s loop-free property.

3) *Evaluation metrics*: We use four main performance metrics for evaluation:

- *Flooding Delay*: it is defined as the total time taken for a sink node to flood a certain number of data packets.
- *Average Delivery Delay*: it is calculated by averaging the delivery delay per-node per-packet. This metric reflects the data forwarding delay at each node in the process of flooding.
- *Energy Efficiency*: it is measured as the total number of data transmissions for flooding a certain number of data packets to the entire network. Since in the low-duty-cycle model, if a sender has no data to transmit, it will stay dormant in its sending slot for energy saving. This metric is proportional to the total number of active sending slots that nodes have data packets to transmit.
- *Effective Duty-Cycle*: the average fraction of time that a non-leaf-node is in active state during one working cycle. This metric is another factor reflecting the energy efficiency.

### B. Performance Overview

We present the evaluation results under three different tree structures from Fig. 6 to Fig. 8, respectively. Given a tree structure, for each data packet, we derive a lower bound of the number of data transmissions to show the maximum room for performance improvement. It is calculated as the sum of the theoretical minimum expected transmissions for all non-leaf-nodes, i.e.,  $\sum \frac{1}{p_{f1}}$ , where  $p_{f1}$  is the highest PRR value of a non-leaf-node (sender) towards its child nodes. Since the first node in the sequence has no chance to switch,  $\sum \frac{1}{p_{f1}}$  is a lower bound of the expected transmissions that a sender finishes the transmission of a packet to all its child nodes. We have the following observations concerning these results.

1) *Flooding Latency*: From Fig. 6(a) to Fig. 8(a), compared with the reliable flooding w/o DSRF, DSRF reduces the flooding delay under all three different tree structures from 12% to 25%. And as the network density increases from 100 to 300, the delay reduction significantly increases. For the

hop-based spanning tree, it achieves around 12% reduction on average. When flooding over the ETX-based spanning tree, DSRF provides around 18% improvement. While DSRF shows more than 25% performance gain for the heuristic energy optimal tree. Interestingly, DSRF shows advantages over the *Overhearing* approach on the flooding latency in most cases. Since DSRF enables a sender finishing the transmission task earlier, as shown from Fig. 6(b) to Fig. 8(b), it also effectively reduces the average delivery delay for all three different tree structures.

2) *Energy Efficiency*: As shown from Fig. 6(c) to Fig. 8(c), DSRF effectively reduces the total number of packet transmissions by 10% ~ 15% under all three different tree structures. With the lower bound of data transmissions, for the ETX-based flooding tree and the energy optimal tree, DSRF achieves very close performance to the theoretical lower bound. The main reason is that DSRF reduces a parent node’s retransmissions by utilizing *cost free* links provided by sibling nodes.

From Fig. 6(d) to Fig. 8(d), the sender’s average effective duty-cycle in DSRF is much lower than that of the *Overhearing* approach, and it is close to the original duty-cycle setting (reliable flooding w/o DSRF).

### C. Insights

Firstly, two questions are raised towards these evaluation results. 1) Are the 12% and 18% improvements on the flooding latency below our expectation? 2) Why does DSRF yield even better performance under the energy optimal trees? Intuitively, for the hop-based spanning tree and ETX-based spanning tree, DSRF should have more chances to switch to better links to improve the flooding latency and energy efficiency. However, careful scrutiny of the tree structures reveals that the performance gain of DSRF is actually remarkable. As shown in Table I, the average non-leaf-node out-degree is quite low for the hop-based tree and ETX-based tree. Therefore, it does not always have chance to switch with such a low average out-degree. While for the heuristic energy optimal tree, DSRF has more switch chances.

TABLE I  
AVG. NON-LEAF-NODE OUT-DEGREE

Trees	Node Density				
	100	150	200	250	300
Hop-based Spanning Tree	2.26	2.47	2.61	2.66	2.73
ETX-based Spanning Tree	2.01	2.08	2.12	2.14	2.16
Heuristic Energy Optimal Tree	2.27	2.70	2.96	3.15	3.23

Secondly, the *Overhearing* approach is supposed to receive data packets earlier than DSRF. However, it is interesting that DSRF outperforms the *Overhearing* approach in most cases, both in terms of the flooding latency and transmission efficiency. After examining the simulation traces, we find two reasons behind the results. 1) When a child node fails to receive a data packet from its parent but overhears it from a sibling node, it still has to wait until its parent node wakes up in the next working cycle. That is why the average delivery

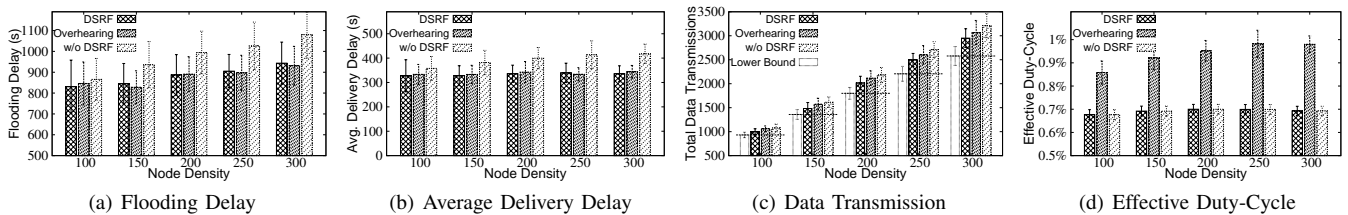


Fig. 6. Hop-based Spanning Tree

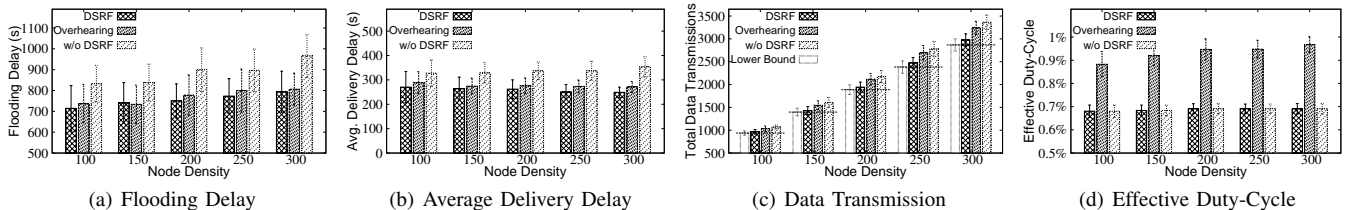


Fig. 7. ETX-based Spanning Tree

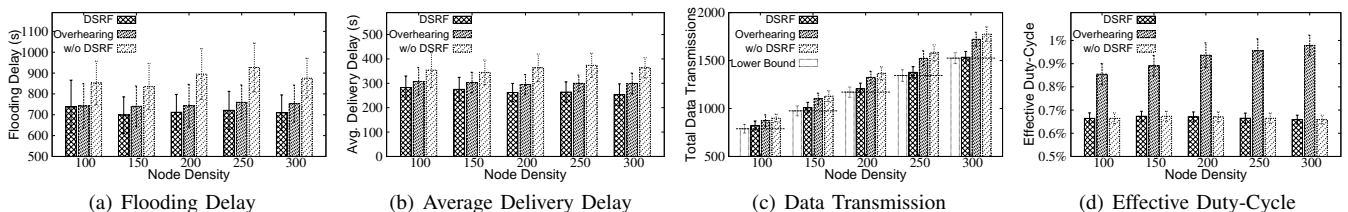


Fig. 8. Heuristic Energy Optimal Tree

delay of *Overhearing* is higher than that of DSRF on average, as shown from Fig. 6(b) to Fig. 8(b). In addition, since the average non-leaf-node out-degree is comparatively low, the effective overhearing probability is also limited. However, as the node density increases, *Overhearing* can provide a shorter flooding delay, as shown in Fig. 6(a). 2) We surprisingly find that *Overhearing* can only reduce the delay in the first several data transmissions. As more data packets are flooded from the source, it is more likely that sibling nodes keep transmitting the data packets that a receiver has already received, while the packet it intends to overhear is still queued at sibling nodes. This is because that nodes follow a FIFO policy to rebroadcast data packets, successfully received incoming packets will be queued until prior packets are delivered.

Thirdly, we explore the reason why the average effective duty-cycle in DSRF is quite close to the original duty-cycle setting. Due to the average non-leaf-node out-degree is low in our simulation, switching does not happen frequently. Take the ETX-based tree for example, the average number of switching time ranges from 50 to 262 for flooding 10 packets over the whole network with the node density increased from 100 to 300. Since each switching only adds one extra active time slot, DSRF operates with a very low overhead. We observe that the performance gain brought by DSRF is directly related to the average out-degree of a scheduled flooding tree. The higher the average out-degree, the more the performance gain.

## V. RELATED WORK

Reliable flooding has been extensively investigated in the research community. It is always required for distribution of code or related data over the entire network [6], e.g., Deluge [8] or CORD [9]. However, there is relatively little research on the reliable flooding in low-duty-cycle WSNs with unreliable wireless links. Flash Flooding [27] and Glossy [28] use concurrent transmissions over interference for rapidly flooding WSNs, which can significantly increase network throughput, enhance packet transmission reliability, and to reduce flooding latency. However, the concurrent transmission based flooding suffers the scalability problem. The packet reception performance degrades as the node density or the size of the network increases [29]. Besides, without explicit or implicit acknowledgements from receivers, it is difficult to guarantee 100% flooding reliability.

Flooding in low-duty-cycle WSNs has recently attracted much attention. Guo et al. propose an energy optimal tree-based Opportunistic Flooding [3] for WSNs with unreliable wireless links and asynchronous working schedules. Based on the delay distribution along a flooding tree, it makes a probabilistic forwarding decision at each sender. ADB [16] optimizes the broadcast for asynchronous duty-cycle WSNs through transmission task delegation, so as to avoid transmissions over poor links. Lai et al. [30] present hybrid-cast, an asynchronous and multihop broadcasting protocol, which reduces redundant transmission via delivery deferring



and online forwarder selection. Wang et al. [15] extend the dynamic switch-based unicasting design in [13], [31] to the flooding scenario, then transform the duty-cycle-aware broadcast problem into a shortest path problem. Guo et al. [4] then propose to schedule nodes of common parents waking up simultaneously to adapt existing flooding-tree-based designs for low-duty-cycle networks, in which the link correlation [32] is explored to improve the broadcasting energy efficiency. Our work complements the scheduled flooding tree-based design, e.g., [4], by combining the duty cycle configuration with the cooperative forwarding paradigm, to optimize the flooding efficiency.

## VI. CONCLUSION

In this work, a dynamic switching-based reliable flooding (DSRF) framework is presented for schedule-based reliable flooding in low-duty-cycle WSNs. Unlike previous works sharing a similar cooperative forwarding idea, which mainly avoid transmissions over poor links by switching to better links, DSRF explores both poor links and good links. We conduct a comprehensive evaluation of DSRF through ns-2 based simulation. The efficiency of our design has been demonstrated through performance comparisons with alternative approaches under three different tree structures. Evaluation results show that DSRF improves the flooding efficiency in different tree structures, and the performance gain brought by DSRF is directly related to the average out-degree of flooding trees. For the ETX-based flooding tree and the energy optimal tree, the performance achieved by DSRF is very close to the theoretical lower bound.

## ACKNOWLEDGEMENTS

This work was supported in part by SUTD SRG ISTD 2010 002, Singapore-MIT IDC IDD61000102a, IDG31100106a Grant, NRF2012EWT-EIRP002-045, Research Fund of the State Key Laboratory of Software Development Environment under Grant No. BUAA SKLSDE-2012ZX-17, National Natural Science Foundation of China under Grant No. 61170296 and 61190120, Program for New Century Excellent Talents in University under Grant No. NECT-09-0028, and US National Science Foundation (NSF) Grant CNS-0917097.

## REFERENCES

- [1] B. Hohlt, L. Doherty, and E. Brewer, "Flexible power scheduling for sensor networks," in *Proc. IPSN '04*, 2004, pp. 205–214.
- [2] L. Tang, Y. Sun, O. Gurewitz, and D. Johnson, "Pw-mac: An energy-efficient predictive-wakeup mac protocol for wireless sensor networks," in *Proc. INFOCOM '11*, 2011, pp. 1305–1313.
- [3] S. Guo, Y. Gu, B. Jiang, and T. He, "Opportunistic flooding in low-duty-cycle wireless sensor networks with unreliable links," in *Proc. MobiCom '09*, 2009, pp. 133–144.
- [4] G. Shuo, S. M. Kim, T. Zhu, Y. Gu, and T. He, "Correlated flooding in low-duty-cycle wireless sensor networks," in *Proc. ICNP '11*, 2011, pp. 383–392.
- [5] Z. Li, M. Li, J. Liu, and S. Tang, "Understanding the flooding in low-duty-cycle wireless sensor networks," in *Proc. ICPP '11*, 2011, pp. 673–682.
- [6] F. Stann, J. Heidemann, R. Shroff, and M. Z. Murtaza, "Rbp: robust broadcast propagation in wireless networks," in *Proc. SenSys '06*, 2006, pp. 85–98.

- [7] A. Hassanzadeh, R. Stoleru, and J. Chen, "Efficient flooding in wireless sensor networks secured with neighborhood keys," in *Proc. WiMob '11*, 2011, pp. 119–126.
- [8] L. Huang and S. Setia, "Cord: Energy-efficient reliable bulk data dissemination in sensor networks," in *Proc. INFOCOM '08*, 2008, pp. 574–582.
- [9] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proc. SenSys '04*, 2004, pp. 81–94.
- [10] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *Proc. SenSys '03*, pp. 1–13.
- [11] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links," in *Proc. SECON '04*, pp. 517–526.
- [12] "CC2420 Datasheet." [Online]. Available: [www.ti.com](http://www.ti.com)
- [13] Y. Gu and T. He, "Dynamic switching-based data forwarding for low-duty-cycle wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 12, pp. 1741–1754, 2011.
- [14] D. B. Johnson, D. A. Maltz, and J. Broch, "Ad hoc networking," 2001, ch. DSR: the dynamic source routing protocol for multihop wireless ad hoc networks, pp. 139–172.
- [15] W. Feng and L. Jiangchuan, "On reliable broadcast in low duty-cycle wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 5, pp. 767–779, May 2012.
- [16] Y. Sun, O. Gurewitz, S. Du, L. Tang, and D. B. Johnson, "Adb: an efficient multihop broadcast protocol based on asynchronous duty-cycling in wireless sensor networks," in *Proc. SenSys '09*, 2009, pp. 43–56.
- [17] Q. Cao, T. Abdelzaher, T. He, and R. Kravets, "Cluster-based forwarding for reliable end-to-end delivery in wireless sensor networks," in *Proc. INFOCOM '07*, 1928–1936.
- [18] V. Salmani and P. H. Chou, "Pushing the throughput limit of low-complexity wireless embedded sensing systems," in *Proc. SUTC '10*, 2010, pp. 181–188.
- [19] Y. Cao, S. Guo, and T. He, "Robust multi-pipeline scheduling in low-duty-cycle wireless sensor networks," in *Proc. INFOCOM '12*, 2012, pp. 361–369.
- [20] X. L. Huang and B. Bensaou, "On max-min fairness and scheduling in wireless ad-hoc networks: analytical framework and implementation," in *Proc. MobiHoc '01*, 2001, pp. 221–231.
- [21] K.-H. Kim and K. G. Shin, "On accurate measurement of link quality in multi-hop wireless mesh networks," in *Proc. MobiCom '06*, 2006, pp. 38–49.
- [22] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis, "Four-bit wireless link estimation," in *Proc. HotNets VI*, 2007.
- [23] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with glossy," in *Proc. IPSN '11*, 2011, pp. 73–84.
- [24] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *Proc. SenSys '04*, 2004, pp. 39–49.
- [25] "Network simulator." [Online]. Available: <http://www.isi.edu/nsnam/ns/>
- [26] L. Cheng, J. Cao, C. Chen, J. Ma, and S. Das, "Exploiting geographic opportunistic routing for soft qos provisioning in wireless sensor networks," in *Proc. MASS '10*, 2010, pp. 292–301.
- [27] J. Lu and K. Whitehouse, "Flash flooding: Exploiting the capture effect for rapid flooding in wireless sensor networks," in *Proc. INFOCOM '09*, 2009, pp. 2491–2499.
- [28] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with glossy," in *Proc. IPSN '11*, 2011, pp. 73–84.
- [29] Y. Wang, Y. He, X. Mao, Y. Liu, Z. Huang, and X.-Y. Li, "Exploiting constructive interference for scalable flooding in wireless networks," in *Proc. INFOCOM '12*, 2012, pp. 2104–2112.
- [30] S. Lai and B. Ravindran, "On multihop broadcast over adaptively duty-cycled wireless sensor networks," in *Proc. DCOSS '10*, 2010, pp. 158–171.
- [31] Y. Gu and T. He, "Data forwarding in extremely low duty-cycle sensor networks with unreliable communication links," in *Proc. SenSys '07*, 2007, pp. 321–334.
- [32] K. Srinivasan, M. Jain, J. I. Choi, T. Azim, E. S. Kim, P. Levis, and B. Krishnamachari, "The  $\kappa$  factor: inferring protocol performance using inter-link reception correlation," in *Proc. MobiCom '10*, 2010.