Exploring Link Correlation for Efficient Flooding in Wireless Sensor Networks

Ting Zhu, Ziguo Zhong, Tian He, and Zhi-Li Zhang
University of Minnesota, Twin Cities

Abstract

Existing flooding algorithms have demonstrated their effectiveness in achieving communication efficiency and reliability in wireless sensor networks. However, further performance improvement has been hampered by the assumption of link independence, a design premise imposing the need for costly acknowledgements (ACKs) from every receiver. In this paper, we present Collective Flooding (CF), which exploits the *link correlation* to achieve flooding reliability using the concept of *collective ACKs*. CF requires only 1-hop information from a sender, making the design highly distributed and scalable with low complexity. We evaluate CF extensively in real-world settings, using three different types of testbeds: a single hop network with 20 MICAz nodes, a multi-hop network with 37 nodes, and a linear outdoor network with 48 nodes along a 326-meter-long bridge. System evaluation and extensive simulation show that CF achieves the same reliability as the state-of-the art solutions, while reducing the total number of packet transmission and dissemination delay by $30 \sim 50\%$ and $35 \sim 50\%$, respectively.

1 Introduction

In wireless sensor networks, flooding is a protocol that delivers a message from one node to all the other nodes. Flooding is a fundamental operation for time synchronization [15], data dissemination [25, 26, 17, 10], group formation [14], node localization [39], and routing tree formation [6, 29].

Existing flooding algorithms [18, 34, 24, 12] have demonstrated their effectiveness in achieving communication efficiency and reliability in wireless sensor networks. Further performance improvement, however, has been hampered by the implicit assumption of *link independence* adopted in previous designs. In other words, existing flooding algorithms assume that the reception of a flooding message by multiple neighboring nodes is probabilistically independent of each other. Under such an assumption, it is necessary to have an acknowledgement (ACK) *directly* from the intended receiver for reliable flooding. This is because a node's ACK cannot be used to estimate the reception at other neighboring nodes if link independence is assumed.

However, *direct ACKs* per receiver may lead to high collision [11, 8], congestion [2], and possibly the ACK storm problem [24] in wireless networks. To address this inefficiency in ACKs, this work presents the first comprehensive study to exploit link correlation in the context of flooding design in wireless sensor networks. The driving idea behind our design is *collective ACKs*. Previously, a sender estimated whether a transmission was successful based only on the feedback from the *intended* receiver. Instead, the mechanism of *collective ACKs* allows the sender to infer the success of a transmission to a receiver based on the ACKs from *other* neighboring receivers by utilizing the link correlation among them. Specifically, we use the Conditional Packet Reception Probability (CPRP) as a metric to characterize the correlation among links. The CPRP is the probability of a node's successfully receiving a packet, given the condition that its neighbor has received the same packet. Based on the environment's stability, this metric is measured and calculated online among neighboring nodes using a form of hello message at an adaptive time interval (i.e., small interval when the environment is dynamic and large interval when the environment is stable).

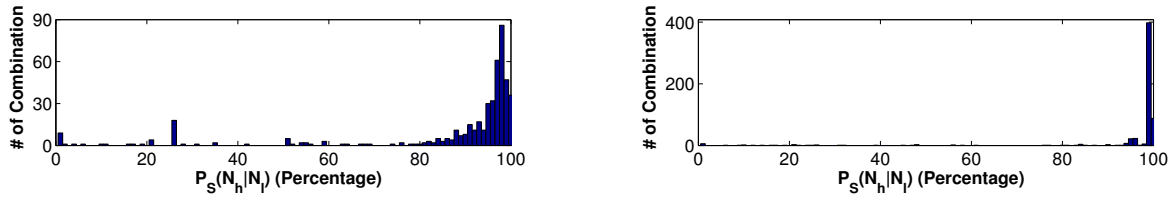
With link correlation information (CPRP) available among neighboring nodes, *collective ACKs* are achieved in an accumulative manner. The success of a transmission to a node (defined as the *coverage probability* of a node) is no longer a binary (0/1) estimation, but a probability value between 0 and 1. Using *collective ACKs*, a sender updates the coverage probability values of neighboring receivers whenever (i) it transmits or (ii) overhears a rebroadcast message. To improve efficiency, a transmission is considered necessary only when the coverage probability of a neighboring node has not reached a certain user-desired reliability threshold.

In addition to *collective ACKs*, we propose a dynamic forwarding technique to exploit link correlation further. In Collective Flooding, only a small set of nodes is selected dynamically as the forwarders of a flooding message via self-organized competition among neighboring nodes. Every node estimates its transmission effectiveness based on three factors: (i) neighborhood size, (ii) link quality, and (iii) link correlation among neighboring nodes. The most effective node will start to rebroadcast early to suppress the less effective nodes' rebroadcasts, consequently reducing the message redundancy.

In summary, our contributions are as follows:

(a) Correlation of Packet Reception among Receivers (b) Correlation of Packet Reception among Receivers

Figure 1. Correlation of Packet Reception



(a) Histogram of $P_S(N_h|N_l)$ (Indoor) (b) Histogram of $P_S(N_h|N_l)$ (Outdoor)

Figure 2. Distribution of Conditional Packet Reception Probability

• To our knowledge, *collective ACK* is a new concept that can improve the efficiency of reliable flooding operations. It transforms the traditional *direct* ACKs per receiver into *correlated* and *accumulative* ACKs.

• Although the phenomena of link correlation has been mentioned in the literature [31], we provide the first extensive study to exploit this phenomena for communication improvement. We reveal that link correlation can be used to achieve (i) *collective ACKs*, as well as (ii) efficient forwarder selection.

• Our design is simple and symmetric. Rebroadcast decisions at individual nodes are based on the coverage probability values of neighbors, which in turn are updated by overhearing rebroadcasts from their neighbors. All the operations only need 1-hop neighbors' information, making our protocol highly distributed and scalable.

• We evaluate our work extensively in multiple real-world testbeds and large scale simulation. The results indicate that our design is practical, reliable, and outperforms several existing state-of-the-art designs.

The rest of the paper is organized as follows: Section 2 presents the motivation behind the work. Section 3 introduces two key mechanisms. Section 4 describes the design. Sections 5 and 6 evaluate the work with testbeds and simulation. After discussing related work in Section 7, Section 8 concludes the paper.

2 Motivation

Previous studies on wireless links focus on packet receptions of individual receivers with single [31, 33, 4, 43, 22] or multiple [21] radios. Little systematic study has investigated the packet reception correlation among neighboring receivers. To fill the gap, this section reports our empirical study on wireless link correlation. More specifically, we observed the following phenomena, which serves as the foundation of this work.

Observation: *For packets transmitted from the same sender, if a packet is received by a node with a low packet reception ratio (PRR), it is highly probable that this packet is also received by the nodes with a high PRR.*

2.1 Experiment Setup

In our experiments, 42 MICAz nodes were used. The experiments were conducted with multiple randomly generated layouts under two different scenarios: (i) an open parking lot, and (ii) an indoor office. In each scenario, two types of experiments were conducted: Fixed Single Sender and Round Robin Sender. In the Fixed Single Sender experiment, the sender was placed in the center of the topology, while the other 41 nodes were randomly deployed as receivers. The sender broadcasted a packet in every 200*ms*. Each packet was identified by a sequence number. The total number of packets broadcasted was 6000. In the Round Robin Sender experiment, each node in turn broadcasted 200 packets with time intervals of 200*ms*. The receivers kept track of the received packets through the sender's ID and packet sequence number.

2.2 Correlated Packet Reception

In both indoor and outdoor experiments, we discovered that if a packet is received by a sensor node with low PRR, most of the time this packet is also received by the high PRR nodes. Figure 1(a) and 1(b) illustrate the first 600 packet receptions of two groups of three nodes in indoor and outdoor experiments, respectively. The locations of the nodes are shown in Figure 3 and Figure 4. The black bands correspond to the packets received at the nodes. Clearly, there exists a strong correlation of packet receptions among the neighboring nodes. For example, in Figure 1(a), given the two packets (sequence number 282 and 508) received by $N22$, these two packets were also received by $N29$ and $N23$. In order to quantify this correlation, we define the Conditional Packet Reception Probability (CPRP), as follows:

Definition: The *Conditional Packet Reception Probability is the probability that a high PRR node N_h receives a packet M from sender node S, given the condition that the packet M is received by a low PRR node N_l.*

We use $P_S(N_h|N_l)$ to denote CPRP, where N_h and N_l

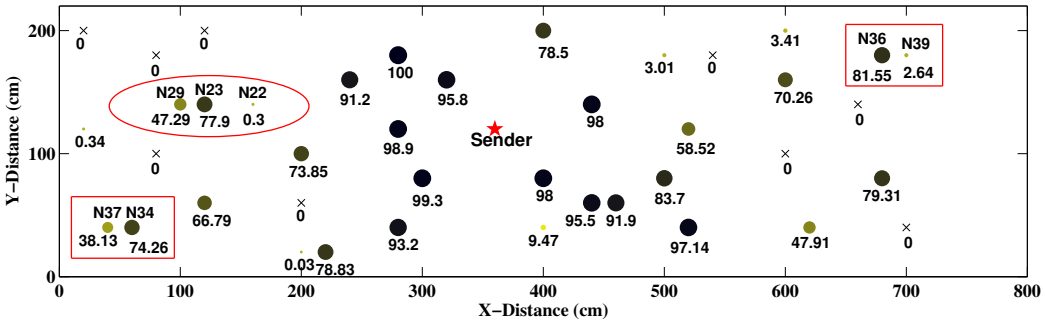


Figure 3. Packet Reception Ratios (PRR) of Individual Nodes in an Indoor Experiment

are neighboring receivers of the sender S. For example, in Figure 1(b), node $N31$ received 38 packets and 37 out of these packets were also received by node $N27$, so $P_S(N27|N31) = 97.4\%$. If the assumption on link independence holds, we would expect $P_S(N27|N31) = P_S(N27)$. However, this is not the case; as shown by the experiment, $P_S(N27)$ is 64.9% instead of 97.4%. This indicates a packet reception correlation between $N31$ and 27, which is also valid for node pairs $N31 \Leftrightarrow N24$ and $N27 \Leftrightarrow N24$.

To analyze the CPRP among the pairwise receivers more systematically, we computed the CPRP for all node pairs with non-zero PRR values. In the indoor experiment, we had 32 non-zero PRR nodes, which generates $\frac{32\times31}{2} = 496$ combinations of $P_S(N_h|N_l)$. Figure 2(a) shows the distribution of these combinations. Figure 2(b) illustrates the distribution for the outdoor experiment.

Figure 2(a) and 2(b) show that the conditional packet reception probability $P_S(N_h|N_l)$ is collectively distributed close to 100%. This result verified our observation that if a packet is received by a low PRR node, this packet has a high probability of also being received by a high PRR node. Due to physical constraints, we only evaluated the link correlation by using MICAz platform. The background traffic or interference would also cause the link correlation on the other radio platforms [32].

2.3 Spatial Diversity in PRR

Besides the link correlation, we also confirmed that the packet reception ratios (PRR) of the receivers had a diverse spatial distribution [4]. Figure 3 shows the spatial distribution of PRR in the indoor Fixed Single Sender experiment. The centers of "•" and "x" indicate the nodes' locations. The larger the size of a "•", the higher the PRR value of this node, while "x" represents nodes that do not receive any packet. The numbers underneath the nodes' locations are Packet Reception Ratio (PRR) values.

Our Fixed Single Sender experiments show that even when two receivers are physically very close to each other, these receivers may have totally different PRRs. For example, in Figure 3, although $N36$ and $N39$ are located near each other at the upper-right corner, their PRRs are significantly different, 81.55% and 2.64%, respectively. There are many node pairs with such features, such as $N34$ and $N37$, $N23$ and $N22$. A similar phenomenon also occurs in the outdoor experiment, such as $N19$ and $N10$, shown in Figure 4.

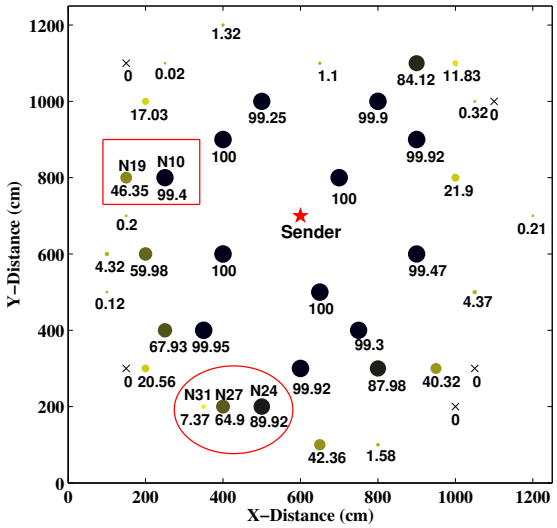


Figure 4. PRR(%) for Each Node (Outdoor)

2.4 Opportunity and Challenges on Flooding

While these observations would impact many protocol designs, this paper focuses on the flooding protocol design in particular.

Link Correlation: Existing flooding protocols did not take advantage of this correlated reception feature. As a result, direct ACKs from receivers is normally used when high reliability is desired. In other words, every receiver needs to send ACKs in response to the reception of a packet, leading to high communication overhead (when explicit ACKs are used) or high redundancy in rebroadcasting (when implicit ACKs are used). The research challenge here is how to exploit link correlation, so that the overhead of ACKs is reduced.

PRR Spatial Diversity: Section 2.3 shows that within the radio range of a sender, even when receivers are located close to each other, they may have dramatically different PRRs due to environmental effects such as multi-

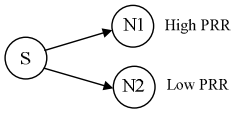


Figure 5. Collective ACKs

path fading. If a flooding protocol selects a fixed forwarder, this forwarder has to retransmit a large number of times to accommodate the receivers with low PRRs, introducing excessive duplicated reception for those receivers with high PRRs. The challenge here is how to reduce the impact of spatial diversity, so that the overhead of redundant transmissions is reduced.

In the rest of the paper, we present two corresponding mechanisms to deal with these two challenges respectively. We explain the ideas conceptually first in Section 3, followed by detailed design in Section 4.

3 Key Mechanisms in Collective Flooding

The main objective of collective flooding (CF) is to reduce redundant transmissions inside the network while providing reliable message dissemination. In CF, we call a node a *covered* node if it has already received the broadcasting packet. Covered nodes are responsible for rebroadcasting the packet to uncovered nodes in the network. In our design, rebroadcasting is used as an implicit ACK to the sender to save protocol overhead. We note that CF can be also applied when explicit ACKs are used. Specifically, there are two key mechanisms in the CF protocol:

• **Collective ACKs:** In CF, the overhearing of a node's rebroadcasting not only indicates that this node has received the packet, but also serves as a *collective ACK* of reception for some other neighboring nodes.

• **Dynamic Forwarder Selection:** The forwarder is selected dynamically through competition among nodes that have already received the broadcasting packet.

3.1 Benefit of Collective ACKs

The mechanism of *collective ACKs* allows a node to extract information about the status of its neighboring nodes via receiving or overhearing a packet from its neighbors. For example, in Figure 5, suppose that node S is a covered node while $N1$ and $N2$ are uncovered. They are within 1-hop communication range of each other, where $N1$ is a low PRR receiver of S and $N2$ is a high PRR receiver of S. When S broadcasts, if $N1$ receives the packet, in traditional flooding protocols without considering the correlation, $N1$ only knows that S is covered, but still considers $N2$ as uncovered until $N1$ overhears $N2$'s rebroadcasting.

CF takes a different approach. From $N1$'s viewpoint, a packet from S serves two purposes. First, it is a *direct ACK* that S is a covered node. Second, it also serves as a *collective ACK* to $N1$ that $N2$ has a reception probability of $P_S(N2|N1)$. Similarly, from S's viewpoint, if S later overhears the rebroadcasting (i.e., an implicit ACK) from $N1$, S not only gets a direct ACK that $N1$ is covered, but is also able to compute the coverage probability of $N2$ according to the link correlation metric $P_{N1}(N2|S)$. We note that in traditional designs, overhearing a packet serves only as a *direct ACK* that the packet sender is covered. In CF, the ACK is achieved in a *collective* manner, i.e., overhearing a packet serves as both direct and correlated ACKs from the packet sender and its neighbors.

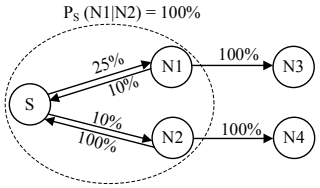


Figure 6. Example of Collective ACKs

Collective ACKs can greatly reduce the redundant transmission. For the sake of clarity, let us consider the simplified network shown in Figure 6. The link qualities from node S to $N1$ and $N2$ are 25% and 10%, respectively; the link qualities from $N1$ and $N2$ to S are 10% and 100%, respectively. We assume the CPRP $P_S(N1|N2) = 100\%$, which means that if $N2$ receives a packet from S, $N1$ also receives that packet.

In traditional flooding protocols, the sender S treats the receivers' packet receptions as independent. To provide reliable broadcasting, S needs to keep on transmitting until it receives ACKs or overhears the transmissions from both $N1$ and $N2$. Due to the low link quality from $N1$ back to S (10%), S might conduct many redundant retransmissions. In contrast, *collective ACKs* in CF allow node S to terminate the transmission earlier if $N2$ receives the flooding packet with a smaller number of retransmissions than expected. For example, if $N2$ receives the packet at the first attempt (*luckily*) and rebroadcasts, node S can immediately terminate the retransmission to $N1$, based on the assumption $P_S(N1|N2) = 100\%$. Therefore, in this case, the number of transmissions at node S can be reduced to one. As we can see from the above simplified example, *collective ACKs* can improve the efficiency of the reliable flooding protocol by utilizing the link correlation.

3.2 Benefit of Dynamic Forwarder Selection

As discussed in Section 2.4, a fixed-forwarder scheme has to accommodate the receiver with the lowest reception ratio, leading to high redundancy for the nodes with high reception ratios. To address this problem, in CF the covered nodes compete for becoming the forwarder node based on their *transmission effectiveness*, which is defined in Section 4.4 and calculated according to three factors: (i) neighborhood size, (ii) link quality, and (iii) coverage probability based on link correlation metric CPRP. A node's transmission is considered more effective if the node has more uncovered neighbors and good link qualities to them. This node wins the competition and rebroadcasts with the shortest back-off time. The nodes

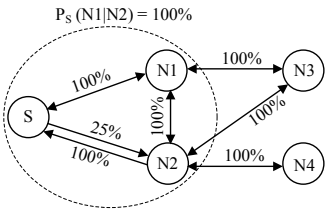


Figure 7. Example of Dynamic Forwarder Selection

transmission would change the transmission effectiveness value of this node and its neighboring nodes. Based on the transmission effectiveness, the forwarder is dynamically selected. This design avoids the same node being selected as the forwarder all the time.

To illustrate the benefit further, we give an example to demonstrate the process of the dynamic forwarder selection. Again, let us consider a simplified scenario as in Figure 7. The link quality from source node (S) to $N2$ is 25%. All the other link qualities are 100%. $N3$ and $N4$ are 2 hops away from S. In order to minimize the total number of transmissions, traditional approaches, such as [18, 12], intend to select a fixed-forwarder among neighboring nodes according to their uncovered neighbor size, in other words, capability of covering more uncovered nodes. For example, in Figure 7, S selects $N2$ as a dedicated forwarder to rebroadcast the packet. The reason is that $N2$ has more uncovered neighbors ($N3$ and $N4$) than $N1$, which only has one uncovered neighbor ($N3$). However, due to the unreliable link between S and $N2$, a packet needs to be transmitted 4 times on average from S before it is received by $N2$. Then, another transmission is needed by $N2$ to cover $N3$ and $N4$. In total, an average of 5 transmissions are needed for a single network-wide broadcast.

In contrast, CF adopts a dynamic and opportunistic approach. After S broadcasts, S, $N1$, and $N2$ compete to be a forwarding node instead of using a dedicated forwarder. Based on the actual reception status, there are two cases:

Case 1: If $N2$ receives the packet (*luckily*) at first attempt, $N2$ can tell that $N1$ is covered based on CPRP $P_S(N1|N2) = 100\%$. After marking $N1$ as a covered node, $N2$ still has more uncovered neighbors ($N3$ and $N4$) than S and $N1$, and thus $N2$ wins the forwarder selection competition and rebroadcasts. After $N2$'s rebroadcasting, all the nodes update their neighbors' coverage probabilities and find that all their neighbors are covered. Therefore, the competition stops and no more transmissions are needed. The total number of transmissions for the network is 2.

Case 2: If $N2$ does not receive the packet, a competition occurs between S and $N1$. $N1$ is supposed to win because it has both $N2$ and $N3$ as uncovered nodes, while S has only $N2$ as an uncovered node. After $N1$'s broadcasting, $N2$ will receive the packet and win the competition because it is the only node that has uncovered neighbor $N4$. One more transmission is needed from $N2$ to cover $N4$. Therefore, the total number of transmissions for the network is 3.

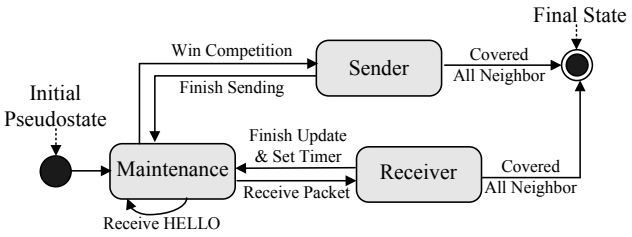


Figure 8. State Machine Diagram of CF

By introducing competition among the covered nodes, CF reduces the redundant transmissions. In the above example, even in the worst case (Case 2), CF only needs 3 transmissions, which is smaller than the traditional fixed-forwarder approaches' 5 times.

4 The Collective Flooding Protocol

This section presents the main design of the CF, which is a simple finite state machine. As shown in Figure 8, a node running CF is in one of three states at any time: (i) maintenance, (ii) receiver, and (iii) sender. Transitions between the states are triggered by events.

After the CF protocol is initiated, the node enters the maintenance state, in which all of its 1-hop neighbor information is periodically maintained. Here, two nodes are considered as neighbors if the link quality between them is larger than 0%. Whenever the node receives a broadcasting data packet, the node enters the receiver state and uses this packet as a *collective ACK* to update its neighbors' coverage probabilities. If the node has uncovered neighbors, it sets its back-off timer based on its transmission effectiveness, then goes back to the maintenance state. When the node's back-off timer fires, which means it wins the competition, it enters the sender state, in which it sends out the packet and updates its neighbors' coverage status; after that, it goes back to the maintenance state. This procedure repeats until the node estimates that all its neighbors are covered. In the rest of this section, we explain the operations in each state in detail.

4.1 Maintenance State

Wireless links in sensor networks are known to be dynamic. Therefore, maintenance is needed to keep track of the link quality. In CF, every node periodically sends out a hello message at an adaptive time interval T which is increased or decreased based on the link's stability. Every hello message is identified by the node ID and a packet sequence number. The hello message is used not only for 1-hop neighbor discovery, but also for updating the link qualities and calculating the Conditional Packet Reception Probability (CPRP) among neighboring nodes. While link quality calculation is straightforward, the calculation of CPRP deserves a little more explanation. Every node maintains a reception record of all hello messages from its neighboring nodes within a time window wT. In order to reduce the required mem-

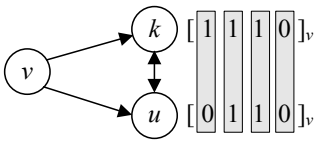


Figure 9. Example of Calculating CPRP

ory space and mitigate the overhead of control messages, the record is represented in a bitmap format (e.g., [0110]) for each neighbor. Such records are exchanged within a hello message every wT seconds among neighboring nodes. CPRP is calculated as follows:

$$P_v(k|u) = \frac{\sum_{i=1}^{w} B_{vk}(i) \,\&\, B_{vu}(i)}{\sum_{i=1}^{w} B_{vu}(i)} \quad (1)$$

Here v is the sender; k and u are the two receivers. $B_{vk}(i)$ is a bit representing node k's reception status of the ith hello message sent from node v. $B_{vk}(i) = 1$ if k receives this message, otherwise $B_{vk}(i) = 0$. For example, in Figure 9, a bitmap of $[1110]_v$ from node k indicates that k does not receive node v's 4th transmission. When node u receives this bitmap, it can use Equation 1 to calculate CPRP by performing the bit-wise *AND* operation with its own bitmap ($[0110]_v$). For example, The CPRP is calculated as $P_v(k|u) = \frac{1\&0+1\&1+1\&1+0\&0}{0+1+1+0} = 100\%$. We note that the length of the bitmap w strikes a balance between the control overhead and statistical confidence of the CPRP value.

4.2 Receiver State

A node enters the receiver state once it receives or overhears a broadcasting packet. Nodes in the receiver state compete to be selected as a forwarder. Without losing generality, suppose node u is the receiver of sender v. Node u maintains two pieces of information:

• **Coverage Probability:** This is the probability of a neighboring node's being covered in a broadcast from the viewpoint of a node. For example, $CP_u(k)$ is node u's estimated probability that u's neighbor k has received the broadcasting packet. Node u maintains $CP_u(k)$ for all its 1-hop neighboring nodes $k \in N(u)$, where $N(u)$ is u's neighboring node set.

• **Estimated Uncovered Node Set $U(u)$:** Here $U(u) \subseteq N(u)$. Initially, node u considers all of its 1-hop neighbors as uncovered. So for any node $k \in N(u)$, $CP_u(k) = 0$. u's uncovered node set is $U(u) = N(u)$.

Supposing node u receives a broadcasting packet M from its neighboring sender v, this packet serves two purposes. First, since the packet M is received from node v, u updates the coverage probability of v as $CP_u(v) = 1$, meaning that u is sure that v has already received the packet (note that this is actually a direct implicit ACK). Second, the packet also serves as a *collective ACK* for all other neighbors $k \in N(u)$. Based on the conditional packet reception probability $P_v(k|u)$ stored in its neighbor table, the coverage probability of other nodes $k \in \{N(u) - v\}$ is updated as follows:

$$CP_u(k) \leftarrow 1 - (1 - CP_u(k)) \cdot (1 - P_v(k|u)) \quad (2)$$

where the term $(1 - CP_u(k))$ is the probability that k had not received the packet M before v's forwarding; the term $(1 - P_v(k|u))$ is the probability of k's failure to receive M from v given the condition that u received M. So $1 - (1 - CP_u(k)) \cdot (1 - P_v(k|u))$ is the probability of node k's being covered either by (i) previous transmission in the network or (ii) current forwarding from v. We note that $CP_u(k)$ is the coverage probability estimated by node u. Formula 2 utilizes $P_v(k|u)$ to accumulate node u's confidence in treating k as a covered node. Namely, u's receiving from v also serves as a *collective ACK* for k. In the worst case when there is no link correlation, the conditional packet reception probability of a node will be equal to the link quality (i.e., $P_v(k|u) = P_v(k)$). In this case, our flooding protocol uses the link quality information (i.e., $P_v(k)$) to update the coverage probability via Formula 2.

When coverage probability $CP_u(k)$ reaches a user's pre-specified threshold $\alpha \leq 1$, node k is considered by node u as covered and is removed from node u's uncovered node set $U(u)$. If $U(u)$ is not empty, node u joins the competition for being the next local forwarder by setting its *back-off timer* according to its transmission effectiveness, which is detailed later in Section 4.4. If $U(u)$ is empty, node u exits the receiver state and completes its broadcasting mission, as shown in Figure 8.

We note that before the timer expires, if node u overhears the broadcast packet M again from one of its neighbors, u cancels the current running timer and repeats the above coverage probability updating procedure and resets its timer. If u's timer fires before all other competitors', u is selected as the forwarder. It enters the sender state to send out broadcast packet M and perform related updates, as explained in the next subsection.

4.3 Sender State

By winning the forwarder competition, node u enters the sender state and sends out the packet. Then, it updates the coverage probabilities of its uncovered neighbors $k \in U(u)$ with

$$CP_u(k) \leftarrow 1 - (1 - CP_u(k)) \cdot (1 - L(u,k)) \quad (3)$$

Here $L(u,k)$ is the link quality between u and k, so the term $(1 - L(u,k))$ indicates the probability of k's failure to receive the broadcasting packet from u. From u's point of view, k has a probability of $(1 - CP_u(k))$ of being uncovered before u's forwarding, and therefore the term $1 - (1 - CP_u(k)) \cdot (1 - L(u,k))$ shows the probability of the event that node k either was covered previously or is covered by u's current forwarding.

As in the receiver state, when the coverage probability $CP_u(k)$ of node k reaches a user-specified threshold α, node k is considered covered and is removed from the uncovered node set $U(u)$. If $U(u)$ is empty, node u terminates the flooding task; otherwise, node u joins the forwarder competition again by setting its back-off timer and returning to the maintenance state. We note that the

value of α is used as a threshold to terminate the retransmission at each node. Different α values can achieve different reliabilities for different applications.

4.4 Back-off Timer Design

The back-off timer is used to conduct dynamic forwarder selection. In the forwarder competition, the duration of the back-off timer is carefully set according to a combination of factors, including (i) neighborhood size, (ii) link quality, and (iii) neighbors' CPRPs. Intuitively, if a node has more uncovered neighbors with good link quality, this node should be the next forwarder and thus should have a short duration before the timer fires.

In CF, we define *Transmission Effectiveness (TE)* as a reference metric for setting the back-off time period.

Definition: $TE(u)$ *equals the number of uncovered nodes that are expected to be covered if the sender u transmits once.*

In general, the value of TE for node u can be calculated with the following Equation

$$TE(u) = \sum_{k \in U(u)} L(u,k) \cdot (1 - CP_u(k)) \qquad (4)$$

The meaning of Equation 4 is straightforward. The transmission effectiveness of u equals the summation of the probabilities of covering u's uncovered neighbors, namely the expected number of nodes to be covered by u's forwarding. The higher the $TE(u)$ value, the more effective node u's transmission is. For example, in a perfect link $(L(u,k) = 100\%)$ scenario, if node u has 2 uncovered nodes, the TE value of u is $TE(u) = 1 \times 1 + 1 \times 1 = 2$, meaning that if u transmits once, 2 neighbors get covered. In another example, if the link qualities from u to these uncovered nodes are all 50%, the TE value of u is $TE(u) = 0.5 \times 1 + 0.5 \times 1 = 1$, meaning that if u transmits once, one neighbor is expected to get covered. Since node u updates the value of $CP_u(k)$ whenever it sends or receives a broadcast packet from its neighbors, $TE(u)$ changes dynamically during the dissemination of the packet.

Every node continuously updates its TE value and sets the back-off timer based on the TE value. Intuitively, the higher the TE value, the smaller the back-off time period should be. The rationale behind this is that we always select the forwarder which is able to cover more nodes in the network with one transmission.

4.5 The Detailed Protocol

Combining all the design components, CF can be specified by the pseudo code shown in Protocol 1.

The design is simple and requires only 1-hop information. Each node maintains a state machine with three states. A state transition is triggered by the *event* of either receiving a broadcast packet (Line 4) or sending timer fired (Line 12). Lines 4 to 11 handle the event of receiving a packet. Lines 12 to 18 handle the timer fired event by sending out the packet and updating the coverage probability values of neighboring nodes. Lines 22 to 25 update the uncovered set of a node, and lines 26 to 29 determine whether the flooding task has been finished.

Protocol 1: Collective Flooding

```
1  Initially, U(u)←N(u), ∀ k ∈ U(u), CP_u(k)←0;
2  repeat
3      switch Event do
4          case u receives packet from v
5              for k ∈ U(u) do
6                  if k = v then CP_u(k)←1;
7                  else Updates CP_u(k) via Formula 2;
8                  end
9                  Call Update U(u);
10             end
11             Call Test U(u);
12         case timer fired
13             u sends out the packet;
14             for k ∈ U(u) do
15                 Update CP_u(k) via Formula 3;
16                 Call Update U(u);
17             end
18             Call Test U(u);
19         end
20     end
21 until U(u) = ϕ ;
22 Update U(u) method :
23 if CP_u(k) ≥ α then
24     U(u) ← U(u) − {k};
25 end
26 Test U(u) method :
27 if U(u) ≠ ϕ then Set back-off timer;
28 else Terminate the timer;
29 end
```

In summary, the CF protocol has three efficient features: (i) it can be implemented with a simple finite state machine with 3 states, which is suitable for resource constrained sensor nodes; (ii) it deals with the spatial diversity of packet reception with dynamic forwarder selection; and (iii) it reduces the communication redundancy through *collective ACKs*, eliminating costly direct ACKs from every receiver.

5 Implementation and Evaluation

We have implemented a complete version of CF on the TinyOS [27]/MICAz platform in nesC [5]. The following two protocols are also implemented as a baseline:

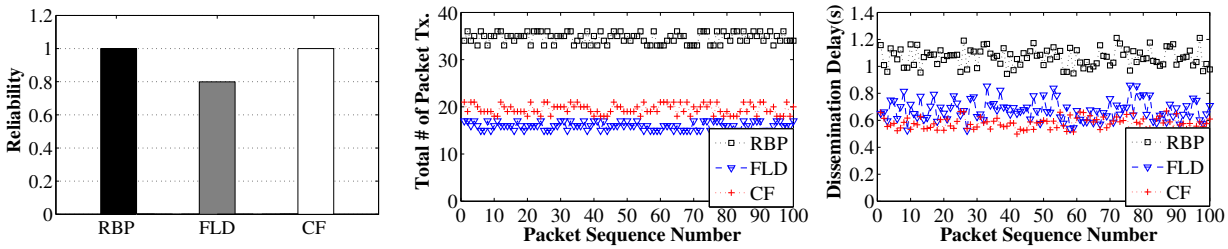
• **Standard Flooding (FLD):** Every node rebroadcasts the first-time received packet exactly once.

• **Reliable Broadcast Propagation (RBP):** RBP [34] was proposed in SenSys'06. As in standard flooding, in RBP, every node unconditionally rebroadcasts the first-time received packet once. Then the node adjusts the number of retries based on the neighborhood density. If there exists a bottleneck link from current node (N) to its downstream node (D), node N performs up to the maximum number of retries when it does not receive the ACK from D.

Four metrics are used to evaluate the protocols:

• **Reliability:** Reliability is quantified by the percentage of nodes in a network that receive the flooding packets.

• **Message Overhead:** Message overhead is measured



(a) Reliability (b) Message Overhead (c) Dissemination Delay

Figure 10. The Performance of Single Hop Indoor Experiment

by the total number of data packets transmitted during the experiment period. We do not count hello messages into the overhead for two reasons. First, the overhead of hello messages is highly environment-dependent. It can be very high in extremely dynamic environments, and very low in static environments. For example, in our indoor experiment, the average hourly hello messages are 73 packets per node during the day and 17 packets per node during the night. Second, most flooding designs need hello messages for neighbor information maintenance. We acknowledge that CF has extra overhead to exchange bitmap within a hello message every wT time interval to calculate CPRP. However, this overhead is independent of data traffic, hence the cost is amortized over multiple flooding operations.

• **Dissemination Delay:** Dissemination delay is the duration from the time that either the source initiates the packet to the time the last node receives the packet or no more nodes resend the packet for a single flood.

• **Load Balance:** This is indicated by the standard deviation of the number of packets transmitted per node per flood. This metric measures how evenly the rebroadcasting activities are distributed in the network.

5.1 Experiment Setup

During the experiments, we placed MICAz nodes in indoor and outdoor environments and tuned the transmission power to ensure the multi-hop communication between the source node and the other nodes. In the experiments, after deployment all the nodes were synchronized and started the neighbor discovery by sending out the hello messages. After all the nodes got the link quality and conditional packet reception ratio information about their 1-hop neighbors, a sender was selected to send out 100 data packets with a time interval of 10 seconds. For performance analysis purposes, in each data packet we included information such as hop count, time stamp, and the previous hop's node ID. Upon receiving the data packet, the intermediate node recorded this information in its flash memory. Every node also recorded the number of transmissions it conducted for each data packet, which was identified by the sequence number. For all the protocols, we maintained the same network placement during the experiments. Unless explicitly stated otherwise, we used the above default values in all the experiments.

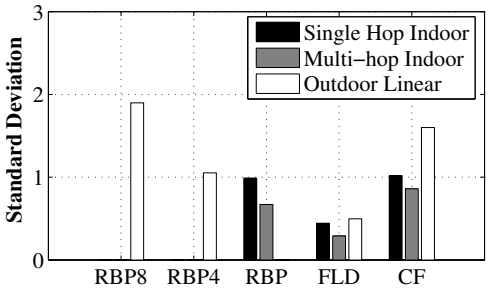
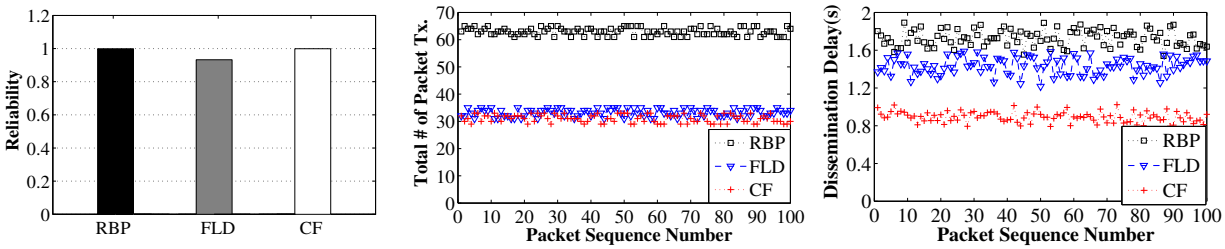


Figure 11. Standard Deviation

5.2 Single Hop Indoor Experiments

The indoor scenario represents potential applications including facility management [38], data center sensing [16], and structural monitoring [23]. In this experiment, one MICAz node was placed as a sender in the center of the indoor 7.5m $\times$ 2.5m testbed, and the other 19 MICAz nodes were randomly deployed around the sender. The transmission power was tuned to ensure that all of these 19 nodes were within the sender's transmission range, although not necessarily within each other's transmission range. The link qualities (i.e., packet reception ratios of the receivers) from the sender to these 19 nodes varied between 100% and 7%.

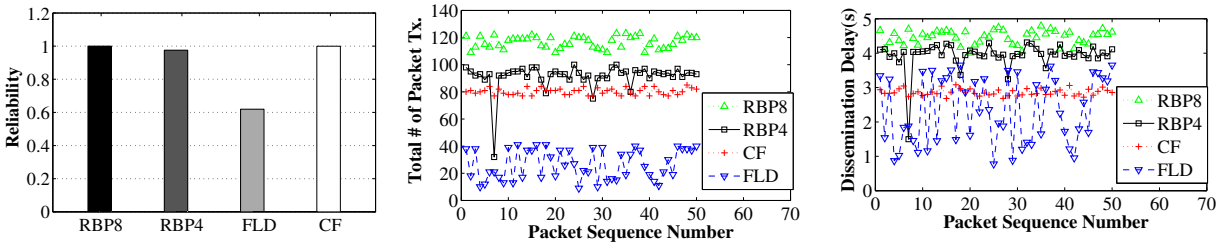
As shown in Figure 10(a), due to the unreliable wireless links, standard flooding (FLD) has only 79.9% reliability. CF, however, achieves the same 100% reliability as RBP. This is because in CF, based on the link quality and strong conditional packet reception ratio information, every node can accurately estimate whether all its 1-hop neighbors receive the packet. This accurate estimation also results in a lower number of transmissions inside the network. Figure 10(b) compares the total number of packets transmitted for every data packet initiated from the sender when running RBP, standard flooding (FLD), and CF protocols. The average values of the total number of packets transmitted for RBP, FLD, and CF are 34.64, 15.98, and 19.5, respectively. Compared with RBP, CF reduces 43.7% of the total number of packets transmitted, while maintaining the same 100% reliabil-



(a) Reliability (b) Message Overhead (c) Dissemination Delay

Figure 12. The Performance of Multi-hop Indoor Experiment

Figure 13. Outdoor Experiment Site: On a Bridge



(a) Reliability (b) Message Overhead (c) Dissemination Delay

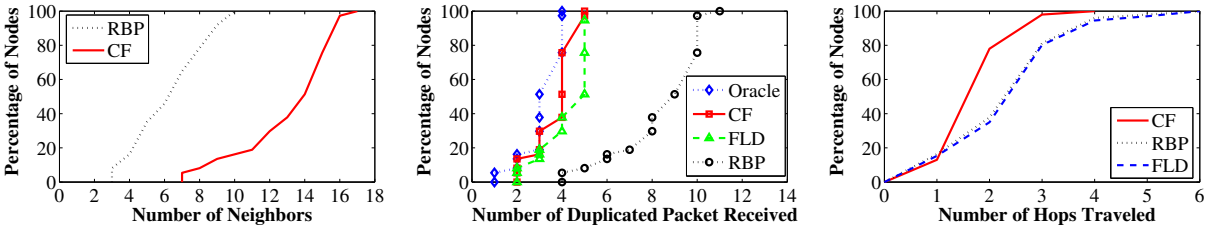
Figure 14. The Performance of Outdoor Linear Network Experiment

ity. Due to the low reliability of FLD, 20.1% of nodes do not receive the packet, and thus FLD has the smallest total number of transmissions.

Figure 10(c) compares the dissemination delays of the three protocols for all 100 data packets initiated by the sender. The average delay for CF is 0.571s, for RBP is 1.09s, and for FLD is 0.69s. Since in CF the node with the largest transmission effectiveness has the smallest back-off time, CF's dissemination delay is 47.6% less than RBP's. As shown in Figure 11, the standard deviations of CF and RBP are very close, but the average numbers of transmissions per node are different (1.73 for RBP, 0.975 for CF). The reason is that in RBP every node unconditionally rebroadcasts the first-time received packet once. In this way, RBP has a larger number of transmissions and a smaller standard deviation than CF.

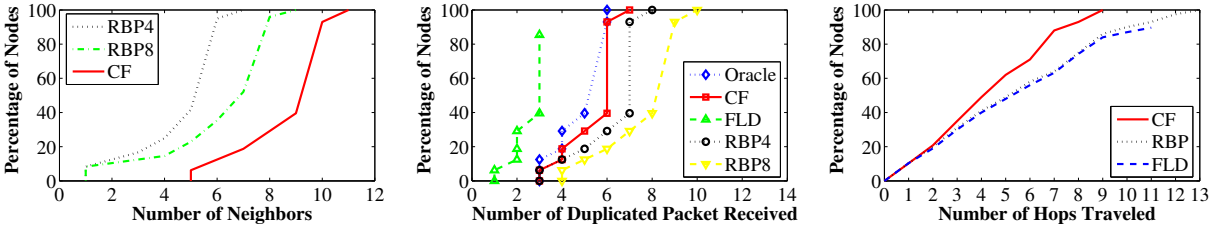
5.3 Multi-hop Indoor Experiments

To further investigate the performance of CF in a larger scale and denser network, a multi-hop indoor experiment was conducted. In this experiment, the MICAz node as a sender was placed in the bottom left boundary of the indoor 7.5m $\times$ 2.5m testbed and the other 36 MICAz nodes were randomly deployed on the testbed. Figure 12(a) shows that the reliability of CF, standard flooding (FLD), and RBP is 99.97%, 93.25%, and 99.89%, respectively. CF achieves the same reliability as RBP, but CF reduces the number of transmissions by 50.7%, as shown in Figure 12(b). The reduction is larger than the 43.7% reduction of number of transmissions in the sparser network (discussed in Section 5.2). This is because in the denser network, the node running CF has more 1-hop neighbors, which would further help the node accurately predict whether its 1-hop neighbors have received the packet. CF also has fewer transmissions than does standard flooding. For instance, the average values of the total number of packets transmitted by using RBP, standard flooding, and CF are 62.94, 33.21, and 31.04, respectively. CF has less number of packets transmitted, which translates to the less amount of energy consumption. In addition, CF does not prevent MAC layer energy management such as low power listening (LPL) [28] and SCP-MAC [41]. For example, in LPL, nodes briefly wake up to check channel activity without actually receiving data. If the channel is not idle, the node stays awake to receive data. Otherwise it immediately goes back to sleep. In this way, LPL protocols consume much less energy than they would listening for full contention period.



(a) CDF of Neighbor Size (b) CDF of Duplicated Packet Received (c) CDF of Number of Hops Traveled

Figure 15. Insight Analysis of Multi-hop Indoor Experiments



(a) CDF of Neighbor Size (b) CDF of Duplicated Packet Received (c) CDF of Number of Hops Traveled

Figure 16. Insight Analysis of Outdoor Experiments

Figure 12(c) compares the dissemination delay of these three protocols. The average delay for RBP, standard flooding, and CF is 1.73s, 1.41s, and 0.89s, respectively. By relying on the node with the largest transmission effectiveness to do the transmission first, the average delay of CF is 48.5% less than that of RBP. The standard deviation of CF, standard flooding, and RBP is 0.866, 0.288, and 0.669, respectively, as shown in Figure 11. CF has a slightly higher standard deviation than RBP. Compared with the sparser network (i.e., single hop indoor), the standard deviation of CF decreases in a denser network (i.e., multi-hop indoor). This is because in a denser network, when running CF, the nodes can more dynamically choose the path to propagate the packets.

5.4 Outdoor Linear Network Experiments

The outdoor experiment represents such potential applications as monitoring remote infrastructures or environments [30, 37, 42]. In the experiment, 48 MICAz nodes were deployed along a 326-meter-long bridge, as shown in Figure 13. For a fair comparison, we implemented two versions of RBP: RBP4 and RBP8. In RBP4, we set the bidirectional link quality threshold for two nodes to be considered neighboring nodes at 50%, and the maximum number of retries when an insufficient number of neighbors got the packet or an important neighbor did not get it at 4. To improve the reliability, in RBP8, the bidirectional link quality threshold was reduced to 30% and the number of retries was set at 8. $N1$ was selected to be the sender that initiated the broadcasting of the data packets.

As shown in Figure 14(a), the reliability of RBP8, RBP4, CF, and standard flooding is 99.96%, 97.6%, 99.93%, and 61.96%, respectively. With the link quality and conditional packet reception ratio information, CF achieves the same reliability as RBP8 and reduces the number of packets transmitted by 31.2% (shown in Figure 14(b)). The average values of the number of packets transmitted for RBP8, RBP4, CF, and standard flooding are 116.64, 91.54, 80.26, and 26.58, respectively.

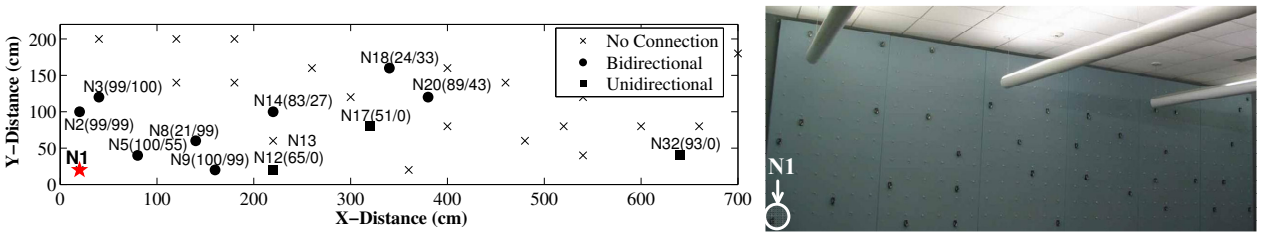
Due to the bottleneck links (further discussed in Section 5.5.4), in RBP4, the data packet with sequence number 7 is received by only 12 nodes in the network, which results in a deep drop of the total number of packets transmitted in Figure 14(b). As shown in Figure 14(c), the average dissemination delay of RBP8, RBP4, CF, and standard flooding is 4.46s, 3.93s, 2.85s, and 2.34s, respectively. The average delay of CF is 36% less than that of RBP8. As shown in Figure 11, the standard deviation of RBP8, RBP4, CF, and standard flooding is 1.9, 1.05, 1.6, and 0.5, respectively. Compared with the indoor experiments, in the outdoor experiment each node has fewer neighbors, resulting in more unbalanced transmissions among these nodes, which explains why the standard deviation of the outdoor experiment is larger.

5.5 System Insight Analysis

In the previous sections, we showed that CF has better performance than standard flooding and RBP. In this section, we explain why this is the case by revealing some system insights.

5.5.1 Number of Neighbors

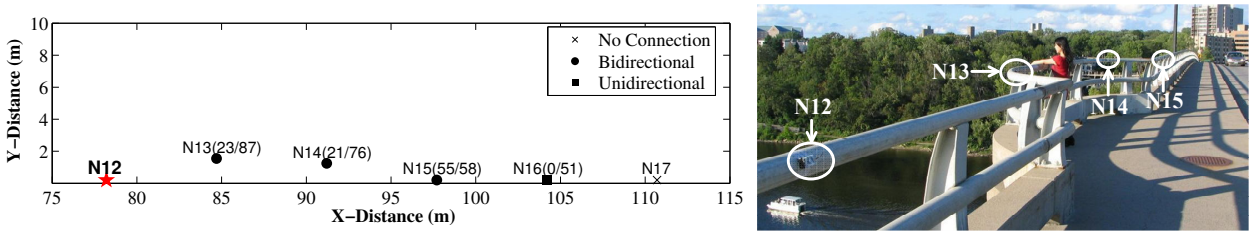
Figure 15(a) and 16(a) compare the CDF of each node's neighbor size when running CF and RBP for the multi-hop indoor and outdoor experiments. In RBP, two nodes are considered as neighbors if and only if the bidirectional link qualities between them are higher than a threshold. While in CF, there is no constraint on the link quality, thus nodes have more neighbors when running CF than when running RBP. The maximum number of neighbors that CF and RBP have is 17 and 9, respectively, for the indoor experiment, and 11 and 7, respec-



(a) Layout of Some Sensor Nodes in Indoor Experiment

(b) On-site View

Figure 17. Asymmetric Links in Indoor Experiment



(a) Layout of Some Sensor Nodes in Outdoor Experiment

(b) On-site View

Figure 18. Asymmetric and Bottleneck Links in Outdoor Experiment

tively, for the outdoor experiment. In CF, more neighbors indicates that more information can be utilized by the node to predict the coverage of its neighbors using *collective ACKs*.

5.5.2 *Prediction Accuracy*

In the previous section, we illustrated that CF has more 1-hop neighbors than RBP. In this section, we show that these 1-hop neighbors provide more information for the node running CF, which can more accurately predict whether its neighbors receive the packet. *Duplicated transmissions happen when the sender does not realize that the receiver already received the packet and retransmits the packet.* Both RBP and standard flooding do not utilize the information of conditional packet reception ratio to predict the packet reception of neighboring nodes, which results in a higher number of duplicated transmissions. Figure 15(b) and 16(b) show the CDF curve of the number of duplicated packets received for the same sequence number of the data packet by all the nodes running CF, standard flooding, and RBP, respectively. By tracing the logged data, we also include an Oracle solution, in which the node running CF, instead of doing the coverage probability estimation, stops transmission once its neighboring nodes receive the packet. From the figure, we can see that CF has a smaller number of duplicated packets received than does RBP in both the outdoor and indoor experiments. *The Oracle and CF curves are very close*, which indicates that the node running CF can accurately predict whether its neighbors have received the packet. Due to the accurate prediction, CF achieves the same reliability as RBP, while it has fewer duplicated transmissions.

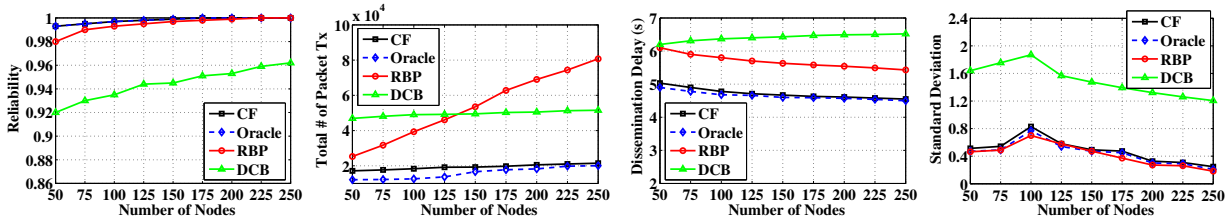
5.5.3 *Efficiency in Delivery Paths*

To trace the experiment, hop count information was attached to the data packet. Figure 15(c) and 16(c) shows the CDF of the number of hops the data packets traveled in order to reach the node. In CF, the node with the largest transmission effectiveness has the smallest back-off timer. This back-off mechanism significantly reduces the number of hops the data packets traveled. For example, the maximum number of hops for CF and RBP is 4 and 6, respectively, in the indoor experiment, and 9 and 13, respectively, in the outdoor experiment. Standard flooding has a similar path length as RBP, but in the outdoor experiment the maximum number of nodes covered by standard flooding was 41 out of 48. That is why the FLD curve terminates in the upper-right corner of Figure 16(c). Due to the smaller number of hops traveled, CF has a smaller dissemination delay than RBP.

5.5.4 *Asymmetric and Bottleneck Links*

Figure 17 shows the link qualities between node $N1$ and all its neighboring nodes. In the figure, a cross represents a node that does not have a direct link with $N1$; a square represents a node that has a unidirectional link with $N1$; and a round dot represents a node that has bidirectional links with $N1$. We use $N8(21/99)$ to represent that the link quality from $N1$ to $N8$ is 21%, while the link quality from $N8$ to $N1$ is 99%. Similar notation is used for all the other nodes. From the figure, we find a large number of asymmetric or even unidirectional links. If we run RBP, $N1$ selects only three nodes ($N2$, $N3$, and $N9$) as neighbors that have bidirectional link qualities higher than the threshold (60%). We also note that some nodes, such as $N13$, have shorter distances to $N1$ than $N17$ but have no connection to $N1$.

A similar phenomenon also happens in the outdoor experiment. Figure 18 shows the link qualities between node $N12$ and some of its neighboring nodes. Due to the environmental effect, the link quality from $N12$ to $N13$ is only 23%, while the backward link quality from $N13$ to $N12$ is 87%. Moreover, although the physical distance between $N12$ and $N13$ is closer than the distance between



(a) Reliability (b) Message Overhead (c) Dissemination Delay (d) Load Balance

Figure 19. Impact of Node Density

$N12$ and $N15$, the link quality from $N12$ to $N13$ is lower than the link quality from $N12$ to $N15$, which is 55%.

Since the bidirectional link qualities between node $N12$ and nodes $N13$ and $N14$ are below the thresholds of RBP4 and RBP8, which are 50% and 30%, respectively, both RBP4 and RBP8 exclude $N13$ and $N14$ from $N12$'s neighbor table. This introduces two effects: (i) $N13$ and $N14$ may not receive the packet from $N12$, leading to (ii) a bottleneck link between $N12$ and $N15$, because in RBP, the node only retries up to the maximum number of retransmissions if it does not hear the downstream node's ACK. For RBP4, the maximum number of retransmission is 4, meaning that in this specific topology, there is still a 4.1% probability that $N15$ will not receive the packet after $N12$'s 4th retry. As shown in Figure 14(b), the data packet with sequence number 7 could not be received by $N15$; thus the packet stopped propagation in the network.

As discussed in Section 3.1, the CF protocol can overcome the difficulties brought about by asymmetric links through the information on link quality and conditional packet receptions. In this scenario, as a receiver, $N12$ can estimate the packet reception probability of $N13$ and $N15$ based on overhearing $N14$'s transmission because $N14$ has larger transmission effectiveness than $N15$. Therefore, $N14$ transmits earlier than $N15$. Moreover, as a sender, $N12$ can also estimate the packet reception probabilities of $N13$ and $N15$ based on $N12$'s own transmission.

6 Simulation Evaluation

In order to understand the performance of the proposed CF scheme under numerous network settings, in this section we provide extensive simulation results. We compared the performance of CF with the following two state-of-the-art solutions and an Oracle approach:

• **Reliable Broadcast Propagation (RBP)** [34] by F. Stann et al. in SenSys'06.

• **Double-Covered Broadcast (DCB)** [18] by Wei Lou and Jie Wu in INFOCOM'04. In DCB, every node maintains 2-hop neighbor information. When a sender broadcasts a packet, it greedily selects the forwarders from its 1-hop neighbor set based on two criteria: (i) the rebroadcasts by the forwarders cover all the sender's 2-hop neighbors, and (ii) the sender's 1-hop non-forwarder neighbors need to be covered by at least two forwarders, including the sender itself.

• **Oracle:** In addition to the state-of-the-art solutions, we also include a theoretical "best-case" bound provided by an Oracle. In the Oracle approach, we assume there exists a perfect cost-free ACK in CF, so instead of doing the coverage probability estimation, the node will exactly know whether or not its neighbors have received the packet. We note that the Oracle approach is not optimal, but it serves as a good baseline.

6.1 Simulation Setup

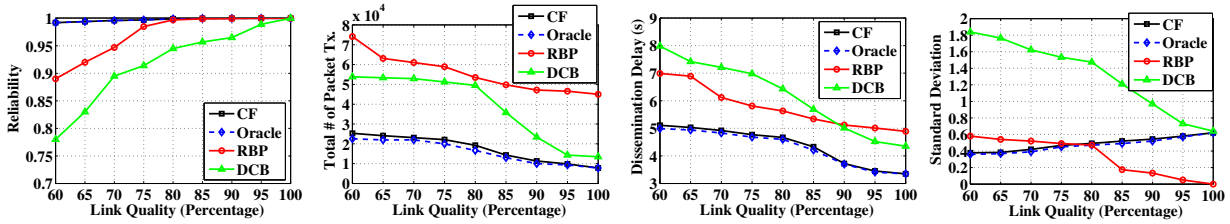
We simulated our design with ns-2. Our simulation MAC layer provided multiple access with collision avoidance. The MAC layer worked in the broadcast mode with no ACKs and retransmissions. The radio model was implemented based on our empirical data, which has the CPRP feature as described in Section 2.

In the simulation, we randomly deployed 250 sensor nodes in a 200m$\times$200m square field. A source node was positioned near the boundary of the field, and the source sent out the data packet with 29 bytes payload every 10 seconds. The total simulation time was set at 3200 seconds. In order to avoid the initialization bias of the system state on the broadcast operation, the source did not send out the data packet in the first 100 seconds, but exchanged only hello messages between neighboring nodes to establish the neighborhood information. Similarly, to make sure that all the broadcast packets propagated throughout the network, the source stopped sending out the data packet after 3100 seconds. Every data point on a graph represents the averaged value of 10 runs, and 95% confidence intervals for the data are within 2$\sim$8% of the mean shown. Unless explicitly stated otherwise, we used the above default values in our simulation.

6.2 Impact of Node Density

In this experiment, we analyzed the effect of node density by varying the number of nodes in the field from 50 nodes to 250 nodes.

Figure 19(a) shows that the reliability of all the protocols increases as the network density increases. When the node density varies, CF has more than 99% reliability, while the mean value of reliability of DCB varies from 0.92 to 0.962. This is because DCB uses only two forwarding nodes to cover the non-forwarding nodes. If the transmissions from both of the forwarders failed, the non-forwarding node will not be covered. When the node density is low, RBP has lower reliability than CF. The reason is that in RBP two nodes are considered as neigh-



(a) Reliability (b) Message Overhead (c) Dissemination Delay (d) Load Balance

Figure 20. Impact of Lossy Links

bors only when the link quality between them is larger than a set threshold (60% according to [34]). In a sparse network, some nodes in a sparse area may not be considered as neighboring nodes of any other nodes in the network. Those in sparse area nodes thus will have a lower chance of receiving the packet from their neighbors when running RBP. However, in CF the node estimates its neighbors' packet receptions based on the link quality information. As long as these nodes are physically connected, CF can provide high reliability.

Figure 19(b) shows that when the network density increases, the total number of packets transmitted linearly increases in RBP but slightly increases in both DCB and CF. The reason is that in RBP, every node needs to retransmit the packet that it receives for the first time. DCB does not provide an optimal algorithm such that the number of forwarders is the minimum and the non-forward nodes are covered exactly by only two forwarders. In DCB, when the network density increases, the number of forwarders slightly increases, which results in an increase in the total number of packet transmissions. In CF, when the network density increases, every node needs to cover more neighbors, which results in the slight increase in the total number of packets transmitted. More neighboring nodes also helps the node predict its neighbors' coverage probability, which results in the decrease in the gap between CF and Oracle in Figure 19(b).

Figure 19(c) shows that when the network density increases, the end-to-end delay decreases in CF and RBP but increases in DCB. This is because when the network density increases, the number of retransmissions used in CF and RBP decreases, while in DCB, when the network density increases, the number of forwarders slightly increases. Every forwarder needs to do back-off and retransmits the packet if it does not hear the retransmissions from its successors that are selected as forwarders.

Figure 19(d) shows that when the network density increases, the standard deviation of the number of data packets transmitted per node for all the protocols first increases and then decreases. This is because when the node density increases, the network will switch from *forwarder dominating* to *non-forwarder dominating* for all the protocols. DCB has the highest deviation, which is because it always selects the forwarders to do the retransmission. In high-density networks, the number of transmissions needed by the nodes is balanced, but RBP has a slightly lower standard deviation than CF. This is because RBP requires every node to do retransmission at least once, while in CF some nodes do not need to do retransmission if *collective ACKs* provide sufficient evidence that their neighboring nodes are covered already.

6.3 Impact of Lossy Links

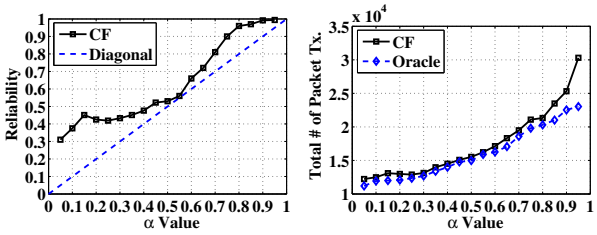
In this experiment, we analyze the effects of varying link qualities. The average link quality varies from 60% to 100%.

Figure 20(a) and Figure 20(b) show that as the link quality increases, the total number of packets transmitted decreases while the reliability increases for all the protocols. When the average link quality is 60%, the total number of packets transmitted by CF, DCB, and RBP is 25323, 53812, and 74132, respectively, while the mean value of reliability is 0.992, 0.78, and 0.89 for CF, DCB, and RBP, respectively. The total number of packets transmitted in RBP is 2.9 times more than that in CF. In order to let DCB achieve higher reliability, we set the maximum number of retransmissions at 4 for the forwarding nodes, which causes the flat period of DCB in Figure 20(b) when the link quality increases from 60% to 80%.

Figure 20(c) shows that the end-to-end delay decreases for all the protocols as the link quality increases. This is because the better the link quality, the fewer backoffs and retries needed by all the protocols. Figure 20(d) shows that when the link quality increases, the standard deviation decreases for RBP and DCB. For RBP, when the link quality is 100%, every node still needs to retransmit the packet that is received for the first time, which results in the standard deviation value of 0, while for CF, the standard deviation increases as the link quality increases. This is because as link quality increases, the nodes that do retransmission become centralized. When the link quality equals 100%, CF becomes the protocol that relies on the nodes with high connectivity to do the retransmission.

6.4 Impact of the Reliability Threshold

CF uses α to control the reliability desired by the users. Technically, the α value is the threshold that is used by each node to check whether its neighbors can be considered as covered. In this section, we evaluate the impact of α value. The total number of nodes in our sim-



(a) Reliability (b) Message Overhead

Figure 21. Impact of α Value

ulation is 150. α value varies from 0.05 to 0.95 with step 0.05. Figure 21(a) shows that the reliability curve of CF is above the diagonal line, indicating that CF satisfies the user requirement well. We note that the closer the reliability curve of CF from the diagonal line, the better CF can track the requirement. Figure 21(a) shows the largest difference between desired and actual α values is about 30.1% when $\alpha = 0.15$. The average difference is about 12.9%, which is a satisfactory performance under high link dynamics.

Figure 21(b) shows that as the α value increases, the total number of packets transmitted also increases. When the α value increases from 0.55 to 0.95, the gap between CF and Oracle also increases. For example, when the α value is 0.95, the difference in the total number of packets transmitted between CF and Oracle is 7280.

Figure 21(a) and 21(b) hint that setting α value to be 0.9 is a good choice for reliable flooding. It achieves the reliability of 0.992 with 4987 fewer packets transmitted than when the α is 0.95, indicating that approaching 100% reliability would be prohibitively expensive under unreliable communication environments.

7 State of the Art

The literature in flooding protocol designs can be classified into two categories: deterministic approaches and probabilistic approaches.

In the deterministic approaches, a fixed node within a connected dominating set is determined as a forwarding node. These approaches are also called fixed-forwarder approaches. In these approaches, the connected dominating set is calculated by using global or local information. It has been proved [20] that the creation of a minimum connected dominating set (MCDS) is NP-complete, so most approaches [12, 18] attempt to find a good approximation to the MCDS. Double-Covered Broadcast (DCB) [18] provides high reliability when the packet loss ratio is low. Compared with CF, DCB introduces more overhead to maintain 2-hop neighbor information. Its reliability is affected by the link quality between the forwarders.

In a probabilistic approach, when a node receives a packet, it forwards the packet with probability p. The value of p is determined by relevant information gathered at each node. Simple probabilistic approaches, such as [24], predefine a single probability for every node to rebroadcast the received packet. When running the above protocols in a network with different node densities, the nodes in a dense area may receive a lot of redundant transmissions. More complicated and efficient protocols, such as distance-based and location-based [40] schemes, use either area or precise position information to reduce the number of redundant transmissions.

Although the probabilistic scheme without ACKs provides a good stochastic result, it has relatively low reliability under unreliable wireless environments. The gossip-based approach [13] provides high reliability, using multiple rounds of message exchanges. Moreover, instead of overhearing, it needs the exchange of metadata. For applications, such as reprogramming an entire sensor network, perfectly propagating the code to all the nodes in the network is required. Trickle [26] uses gossiping and link-layer broadcasting to propagate small code updates. RBP [34] is used for the applications such as routing and resource discovery. In ADB [35], the node uses the footer in DATA and ACK frames to make the rebroadcast decisions. However, under unreliable wireless environment, the loss of an ACK from a receiver will cause the sender to treat the receiver as a 100% uncovered node and redundant transmissions are conducted. Moreover, such explicit ACKs may cause collision [11] in dense networks.

As a flooding protocol, CF is different from ExOR [3] which is a data forwarding protocol. In ExOR, all the nodes work together to forward the data from a source to a single destination. The "batch map" used in ExOR serves as an explicit ACK for each received packet, which is different from the *collective ACKs* which are achieved in an accumulative manner.

Despite this rich literature, the existing approaches do not exploit link correlation for performance improvement. Using link correlation, *collective ACK* becomes the key difference between CF and previous work. More specifically, the CF protocol has two new contributions: (i) instead of using an implicit or explicit ACK [1], each node dynamically estimates and accumulates its neighbors' coverage status through *collective ACKs* by using the correlation of packet receptions among neighboring nodes; (ii) the forwarders are dynamically selected in a distributed fashion based on the nodes' realtime estimations of their neighbors' packet reception status. These two features lead to reliable and efficient message dissemination. Finally, we note that the concept of *collective ACKs* is independent of specific protocol designs. It could be used as an add-on feature to other routing [19, 7, 36, 44] and flooding [9] protocols. We leave this as future work.

8 Conclusions

In this paper, we propose CF to provide efficient and reliable message dissemination service with low complexity. We demonstrate that CF is effective through two

main mechanisms: *collective ACKs* and dynamic forwarder selection. Both mechanisms take advantage of link correlation among neighboring receivers. This is the first work that transforms the direct ACKs per receiver into a collective one. This unique design noticeably reduces the redundancy in rebroadcasting, as shown in our evaluation. We fully implemented and evaluated the CF protocol in several testbeds including a single hop network with 20 MICAz nodes, a multi-hop network with 37 MICAz nodes, and a linear outdoor network with 48 nodes along a 326-meter-long bridge. We also performed extensive simulation with various network configurations to reveal its performance at scale. The results show that the CF protocol has low overhead, low dissemination delay, and high reliability in unreliable wireless environments. Conceptually, the design of CF protocol is generic enough to be applied to wireless mesh networks and other stationary wireless networks. However, it is necessary to systematically investigate the implication of running CF in these types of networks in the future.

9 Acknowledgements

This work was supported in part by NSF grants CNS-0917097, CNS-0845994, and CNS-0626609. We also received partial support from InterDigital and Microsoft Research. We thank our shepherd Dr. Philip Levis and the reviewers for their valuable comments.

10 References

[1] M. Afanasyev, D. G. Andersen, and A. C. Snoeren. Efficiency through Eavesdropping: Link-layer Packet Caching. In *NSDI '08*.

[2] B. Hull, K. Jamieson, and H. Balakrishnan. Mitigating Congestion in Wireless Sensor Networks. In *SenSys '04*.

[3] S. Biswas and R. Morris. ExOR: Opportunistic Multi-Hop Routing for Wireless Networks. In *SIGCOMM '05*.

[4] C. Reis, R. Mahajan, D. Wetherall and J. Zahorjan. Measurement-based Models of Delivery and Interference. In *SIGCOMM '06*.

[5] D. Gay, P. Levis, R. Behren, M. Welsh, E. Brewer, and D. Culler. The nesC Language: A Holistic Approach to Networked Embedded Systems. In *PLDI '03*.

[6] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection Tree Protocol. In *SenSys '09*.

[7] Y. Gu, T. Zhu, and T. He. ESC: Energy Synchronized Communication in Sustainable Sensor Networks. In *ICNP '09*.

[8] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan. Understanding and Mitigating the Impact of Rf Interference on 802.11 Networks. In *SIGCOMM '07*.

[9] S. Guo, Y. Gu, B. Jiang, and T. He. Opportunistic Flooding in Low-Duty-Cycle Wireless Sensor Networks with Unreliable Links. In *MobiCom '09*.

[10] J. Considine, F. Li, G. Kollios, and J. W. Byers. Approximate Aggregation Techniques for Sensor Databases. In *ICDE '04*.

[11] J. Padhye, S. Agarwal, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of Link Interference in Static Multi-hop Wireless Networks. In *IMC '05*.

[12] J. Wu, W. Lou, and F. Dai. Extended Multipoint Relays to Determine Connected Dominating Sets in MANETs. In *IEEE Transcations on computing*, 55(3): 334-347, 2006.

[13] L. Alvisi, J. Doumen, R. Guerraoui, B. Koldehofe, H. Li, R. van Renesse, G. Tredan. How Robust Are Gossip-Based Communication Protocols? In *SIGOPS Operating Systems Review 41,5*, 2007.

[14] L. Luo, T. F. Abdelzaher, T. He, and J. A. Stankovic. Envirosuite: An environmentally immersive programming framework for sensor networks. In *TECS*, 5(3): 543-576, 2006.

[15] C. Lenzen, P. Sommer, and R. Wattenhofer. Optimal Clock Synchronization in Networks. In *SenSys '09*.

[16] C. M. Liang, J. Liu, L. Luo, A. Terzis, and F. Zhao. DCNet: A High-Fidelity Data Center Sensing Network. In *SenSys '09*.

[17] K. Lin and P. Levis. Data Discovery and Dissemination with DIP. In *IPSN '08*.

[18] W. Lou and J. Wu. Double-Covered Broadcast (DCB): A Simple Reliable Broadcast Algorithm in MANETs. In *INFOCOM '04*.

[19] M. Caesar, M. Castro, E. Nightingale, G. O'Shea, and A. Rowstron. Virtual Ring Routing: Network Routing Inspired by DHTs. In *SIGCOMM '06*.

[20] M. V. marathe, H. Breu, H.B. Hunt III, S.S. Ravi, and D.J. Rosenkrantz. Simple Heuristics for Unit Disk Graphs. In *Networks*, 25: 59-68, 1995.

[21] A. Miu, H. Balakrishnan, and C. E. Koksal. Improving Loss Resilience with Multi-Radio Diversity in Wireless Networks. In *MobiCom '05*.

[22] M. Z. Murtaza, J. Heidemann, and F. Stann. Studying the Spatial Correlation of Loss Patterns Among Communicating Wireless Sensor Nodes. In *Technical Report ISI-TR-2007-626, USC/Information Sciences Institute*, 2007.

[23] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A Wireless Sensor Network for Structural Monitoring. In *SenSys '04*.

[24] S. Ni, Y. Tseng, Y. Chen, and J. Sheu. The Broadcast Storm Problem in a Mobile Ad Hoc Network. In *MobiCom '99*.

[25] P. Levis, and D. Culler. Mate: A Tiny Virtual Machine for Sensor Networks. In *ASPLOS X*, 2002.

[26] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code maintenance and propagation in wireless sensor networks. In *NSDI '04*.

[27] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. TinyOS: An Operating System for Wireless Sensor Networks. In *the book Ambient Intelligence*, 2005.

[28] J. Polastre, J. Hill, and D. Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *SenSys '04*.

[29] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica. Beacon-Vector Routing: Scalable Point-to-Point Routing in Wireless Sensor Networks. In *NSDI '05*.

[30] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Health Monitoring of Civil Infrastructures Using Wireless Sensor Networks. In *IPSN '07*.

[31] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis. An Empirical Study of Low Power Wireless. In *SING Tech Report, SING-08-03*, Oct. 2008.

[32] K. Srinivasan, M. Jain, J. Choi, T. Azim, E. Kim, P. Levis, and B. Krishnamachari. The k-Factor: Inferring Protocol Performance Using Inter-link Reception Correlation. In *Technical Report SING-09-02*.

[33] K. Srinivasan, M. Kazandijeva, S. Agarwal, and P. Levis. The β-factor: Measuring Wireless Link Burstiness. In *SenSys '08*.

[34] F. Stann, J. Heidemann, R. Shroff, and M. Z. Murtaza. RBP: Robust Broadcast Propagation in Wireless Networks. In *SenSys '06*.

[35] 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 *SenSys '09*.

[36] T. Zhu and M. Yu. A Dynamic Secure QoS Routing Protocol for Wireless Ad Hoc Networks. In *Sarnoff '06*.

[37] T. Zhu, Z. Zhong, Y. Gu, T. He, and Z. Zhang. Leakage-Aware Energy Synchronization for Wireless Sensor Networks. In *MobiSys '09*.

[38] V. Singhvi, A. Krause, C. Guestrin, J. James, H. Garrett, and H. S. Matthews. Intelligent Light Control Using Sensor Networks. In *SenSys '05*.

[39] K. Whitehouse and D. Culler. A Robustness Analysis of Multi-hop Ranging-based Localization Approximations. In *IPSN '06*.

[40] Y. B. Ko, and N. Vaidya. Flooding-based Geocasting Protocols for Mobile Ad Hoc Networks. In *MONET*, 2002: 471-480.

[41] W. Ye, F. Silva, and J. Heidemann. Ultra-Low Duty Cycle MAC with Scheduled Channel Polling. In *SenSys '06*.

[42] Z. Zhong, T. Zhu, D. Wang, and T. He. Tracking with Unreliable Node Sequence. In *INFOCOM '09*.

[43] G. Zhou, T. He, and J. A. Stankovic. Impact of Radio Irregularity on Wireless Sensor Networks. In *MobiSys '04*.

[44] T. Zhu and M. Yu. A Secure Quality of Service Routing Protocol for Wireless Ad Hoc Networks. In *GLOBECOM '06*.