Accurate and Generic Sender Selection for Dissemination in Low-Power Wireless Networks

Zhiwei Zhao, *Member, IEEE,* Wei Dong, *Member, IEEE,* Jiajun Bu, *Member, IEEE,*
Tao Gu, *Senior Member, IEEE,* Geyong Min, *Member, IEEE,*

***Abstract*—Data dissemination is a fundamental service offered by low power wireless networks. Sender selection is the key to the dissemination performance and has been extensively studied. Sender impact metric plays a significant role in sender selection since it determines which senders are selected for transmission. Recent studies have shown that spatial link diversity has a significant impact on the efficiency of broadcast. However, the existing metrics overlook such impact. Besides, they consider only gains but ignore the costs of sender candidates. As a result, existing works cannot achieve accurate estimation of the sender impact. Moreover, they cannot well support data dissemination with network coding, which is commonly used for lossy environments. In this paper, we first propose a novel sender impact metric, namely γ, which jointly exploits link quality and spatial link diversity to calculate the *gain/cost* ratio of the sender candidates. Then we develop a generic sender selection scheme based on the γ metric (called γ-component) that can generally support both types of dissemination using native packets and network coding. Extensive evaluations are conducted through real testbed experiments and large-scale simulations. The performance results and analysis show that γ achieves far more accurate impact estimation than the existing works. In addition, the dissemination protocols based on γ-component outperform the existing protocols in terms of completion time and transmissions (by 20.5% and 23.1%, respectively).**

I. Introduction

Low power wireless networks have gained increasing importance for a variety of civil and military applications. A low power wireless network consists of a large number of small and inexpensive wireless nodes that integrate sensing, computation, and wireless communication capabilities [1]–[4]. Bulk data dissemination is used to disseminate a large data object to all network nodes reliably in a multi-hop manner. It is one of the key enabling services for software update, surveillance video distribution, etc. in low power wireless networks [5]–[7].

This work was supported by the Fundamental Research Funds for the Central Universities (No. ZYGX2016KYQD098 and No. 2016FZA5010), the National Science Foundation of China (No. 61602095 and No. 61472360), National Key Technology R&D Program (Grant No. 2014BAK15B02), CCF-Intel Young Faculty Researcher Program, CCF-Tencent Open Research Fund, China Ministry of EducationChina Mobile Joint Project under Grant No. MCM20150401 and the EU FP7 CLIMBER project under Grant Agreement No. PIRSES-GA-2012-318939. *(Corresponding author: Wei Dong)*

Z. Zhao is with the College of Computer Science and Engineering, University of Electronic Science and Technology of China, China. E-mail: zzw@uestc.edu.cn

W. Dong and J. Bu are with the College of Computer Science, Zhejiang University, China. E-mail: {dongw, bjj}@zju.edu.cn

T. Gu is with RMIT University, Australia. E-mail: tao.gu@rmit.edu.au

G. Min is with College of Engineering, Mathematics and Physical Sciences, University of Exeter, UK. E-mail: g.min@exeter.ac.uk

Existing dissemination protocols typically divide a large data object into multiple pages to enable the multi-hop pipeline transfer, and use the three-way handshake mechanism (ADV-REQ-DATA) to ensure data consistency. While some protocols [8]–[10] use native packets, others [11]–[14] use network coding to enhance the dissemination performance. However, the coding/decoding incurs considerable delay overhead. Network users choose whether to use coding based protocol or not, according to the wireless conditions. Specifically, when there is much spatial diversity, coding based protocols are preferred. Otherwise, native packets based protocols are preferred.

For both types of dissemination, sender selection plays a critical role for the transmission/delay performance, and has attracted much research attention [9], [10], [15]. The underlying principle of sender selection is to choose the best sender in a neighborhood that is expected to have the most impact in transmission. The sender impact metric is the key to select the best sender. MNP [9] uses the number of receivers as the selection metric. ECD [10] takes a step forward by considering link quality as well as the number of receivers. UFlood [15], a dissemination protocol for mesh networks, jointly considers the number of receivers, link quality and bit rate. When applied in low power networks, UFlood has the same sender selection metric with ECD since the bit rate is fixed.

Recent studies [16]–[19] show that packet receptions on adjacent wireless links much often correlated, which is different from the long held assumption of independent receptions. We observe that link correlation implicitly plays an important role on the efficiency of a sender's transmission by affecting the reception statuses at receivers. Consider that a sender transmits a page of packets to multiple receivers. In case that the sender's outbound links are strongly correlated, the receivers will exhibit similar reception statuses (i.e., if one receiver receives a packet, others are more likely to receive it, and vice versa). Consequently, re-transmission will be more efficient as the receivers are likely to request the same packets. In contrast, in case of weak link correlation, a sender should re-transmit more packets since each receiver may request different packets. Considering the lossy nature of low power wireless links, the re-transmission overhead is a critical criteria for sender's efficiency. Although the use of network coding can partially reduce the negative impact of weak link correlation, we observe that the impact on node reception statuses still greatly affects the transmission efficiency (Section III.B).

Unfortunately, the existing sender selection mechanisms

overlook the impact of link correlation, resulting in inaccurate sender selection and inefficient dissemination. In this paper, we exploit link correlation to design a novel sender selection scheme. We first propose a novel sender selection metric, named γ factor (γ for its sharp emission lines), which formally models the expected packet-level *gain/cost* ratio for each potential sender. We utilize the reception statuses in REQ messages received by a sender to extract link correlation and transmission progress information. The information, combined with link quality, is then used to estimate the utility of each requested packet to send, i.e., the number of successful receptions per packet transmission. This is different from the existing designs since they consider only the expected receptions but overlook the total number of transmissions. Under this estimation, γ is able to achieve accurate sender selection consistently with various link conditions, while the existing metrics of MNP, ECD and UFlood can achieve the same accuracy only under special conditions with weak link correlation.

We then propose a generic sender selection scheme based on the γ factor, named γ-component, which can be easily adopted in the existing dissemination protocols for accurate sender selection. We incorporate γ-component with two popular dissemination protocols—Deluge [8] and Rateless Deluge [11], and conduct extensive experiments to evaluate the effect of γ factor. The results show that 1) γ yields more accurate sender selection by 155.2%, 36.1% and 29.2% compared to the metrics used in Deluge, MNP, and ECD/UFlood, respectively. 2) By incorporating γ-component into Deluge and Rateless Deluge, the number of transmissions and completion time are reduced by 20.5% and 23.1%, respectively.

The major contributions of this paper include:

- An accurate sender selection metric (i.e., γ) is proposed for efficient bulk data dissemination. The new metric takes into account both link quality and link correlation.
- A lightweight and generic sender selection scheme based on γ, namely γ-component, is developed and incorporated into data dissemination for improving the protocol performance.
- Extensive testbed and simulations studies are conducted. The performance results demonstrate that γ is a more accurate sender selection metric and γ-based protocols outperform the existing protocols.

The remainder of this paper is organized as follows. Section II introduces the related work and background. Section III deserves the motivation of our work. Section IV presents the γ factor in detail. Section V presents the γ-component and its incorporation with existing protocols. Section VI evaluates the performance of dissemination protocols based on γ-component. Finally, Section VII concludes this paper.

II. Related works & Background

In this section, we first outline an overview of the existing bulk data dissemination protocols for low power wireless networks, and then present the details of the most related works, with a particular focus on sender selection.

We classify the existing bulk data dissemination protocols into two categories: protocols using native packets and protocols based on network coding. Network users choose whether to use coding based protocols or not, according to the network environmental conditions [20]. When there is much spatial diversity such as indoor environment, complex terrain, etc., coding based protocols are preferred. Otherwise, non-coding protocols are preferred.

Next, we provide more details for the most related works, with special focus on sender selection.

A. Native packets based dissemination

1) Deluge: Deluge is the standard bulk data dissemination protocol used in TinyOS [21]. It first segments a large object into multiple pages, each of which consists of multiple packets. It then transmits a page in one batch. NACK-based three-way handshake mechanism is used for ensuring reliability. Each node periodically broadcasts ADV messages to announce how many pages it can provide. If a node receives ADV messages that contain more pages, it randomly selects a sender to transmit REQ messages. The REQ messages contain bitmaps indicating the lost packets, such that the senders can transmit the requested packets. However, considering that different senders may be largely different in the transmission efficiency, the random selection may increase the overhead in terms of transmissions and delay.

2) MNP and ECD: MNP [9] and ECD [10] use a more accurate sender selection algorithm to improve the dissemination performance. In MNP, each sender counts for the number of unique requests it has received (using a reqCtr counter). The sender with the largest reqCtr will be selected and will start transmitting data packets. The rationale of MNP is clear: the sender with the largest reqCtr will likely serve more requesters, resulting in faster dissemination.

ECD further considers link quality. The sender with a large number of requesters and good link quality to those requesters will be favored. ECD selects senders with fewer transmissions compared to MNP. Formally, let u denote a sender, N_u^{req} denote the set of requesters of u, and q_{uv} denote the link quality from u to v. The sender selection metric used in MNP is calculated as

$$\varepsilon_{MNP} = |N_u^{req}| \tag{1}$$

while the sender selection metric used in ECD is

$$\varepsilon_{ECD} = \sum_{v \in N_u^{req}} q_{uv} \tag{2}$$

B. Coding based dissemination

1) Rateless Deluge and SYNAPSE: Both Rateless Deluge [11] and SYNAPSE [12] apply network coding to improve the dissemination performance in lossy environments. Rateless Deluge employs random linear code while SYNAPSE employs Fountain code. Instead of transmitting native packets, Rateless Deluge transmits encoded packets. Upon receiving a specified number of encoded packets, the receiver can decode the packets. Due to resource limitation of a low-power node, both the sender and receiver share the same seed for generating a sequence of random coefficients so that the message overhead

of the coefficients can be avoided. Encoding can only be performed on native packets. As such, the decoding cost may have a large impact on the performance since a forwarding node must decode a page of packets before it can prepare encode and transmit packets to the next hop. The random sender selection algorithm used in both Rateless Deluge and SYNAPSE is similar to the one in Deluge.

2) UFlood: UFlood [15] is a dissemination protocol combining both network coding and sender selection in wireless mesh networks. It addresses the problem of reliable dissemination of multiple packets to all network nodes. Different from Rateless Deluge and SYNAPSE, UFlood can encode encoded packets (e.g., encoded packet of the 2nd generation, 3rd generation, ...). The additional cost of the design includes the coefficients into the encoded packets. While it may be reasonable for wireless mesh nodes, it may not be affordable for low-power wireless nodes. UFlood uses the following sender selection metric.

$$\varepsilon_{UFlood} = \sum_{v \in N_u^{all}} q_{uv,b_u} \cdot b_u \cdot I_{uv} \tag{3}$$

where N_u^{all} is the set of all neighboring nodes (not only requesters), b_u is the optimal bit rate chosen by u, and I_{uv} is the variable indicating whether u's transmission is useful to v.

It is worth noting that UFlood does not employ the ADV-REQ-DATA handshake. Instead, each UFlood node periodically exchanges feedback messages containing a bit vector (only affordable in mesh networks), which indicates its own received and missing packets. As a result, I_{uv} plays an important role to identify whether q_{uv} should be accounted in u's utility: When node u receives feedback messages from node v, node u checks whether it has useful packets for node v. If yes, I_{uv} is set, and then q_{uv} is accounted for u's utility. This is equivalent to the case that node v sends an REQ message to node u such that q_{uv} is accounted for u's utility. Hence, Eq. (3) essentially calculates the sum of $q_{uv,b_u} \cdot b_u$ for *all requesters*. Besides, when applied in low power wireless networks equipped with 802.15.4 radios, b_u can be neglected since the bit rate cannot be adaptively changed. Therefore, the metric used in UFlood can be re-written as:

$$\begin{aligned} \varepsilon_{UFlood} &= \sum_{v \in N_u^{all}} q_{uv,b_u} \cdot b_u \cdot I_{uv} \\ &= \sum_{v \in N_u^{req}} q_{uv,b_u} \cdot b_u \\ &= b_u \cdot \sum_{v \in N_u^{req}} q_{uv,b_u} \end{aligned} \tag{4}$$

We can see that ε_{UFlood} is essentially b_u times of ε_{ECD}. When using ε_{UFlood} as the sender selection metric, for the same contending senders, if a sender has the largest impact value of ε_{UFlood}, it will also have the largest impact value of ε_{ECD}. The selected sender will be identical for both metrics. As such, we use ε_{ECD} to denote both ECD's and UFlood's metrics in the following sections for simplicity. There are some more recent works [22], [23] that exploit constructive interference for dissemination. For example, the work [22] achieves far more efficient dissemination than Deluge by using both constructive interference and network coding. The establishment of constructive interference and the flooding structure may require stringent time synchronization.

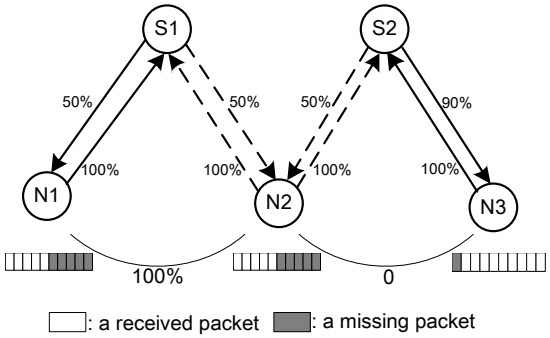


Fig. 1: A motivating example with native packets.

C. Short summary

Existing sender selection metrics evaluate a sender by calculating the expected number of receivers of the sender. There are two factors that introduce errors to these metrics: (i) They only consider the gain (i.e., the number of receivers) without considering the cost (i.e., the number of transmissions by the sender). A sender that have *many* neighboring nodes *with weak links* can be selected by these metrics, as the sum of its outbound link qualities can be large. However, such a sender's transmissions may not be efficient as it may require a large number of transmissions to cover its neighbors. (ii) They fail to consider link correlation or reception statuses at receivers. As to be demonstrated in Section III, link correlation greatly affects the transmission efficiency of a sender by impacting the reception statuses at receivers. In contrast, γ can (i) essentially calculate the *gain/cost ratio* of a sender and (ii) effectively make use of node reception statuses to exploit link correlation when evaluating a sender's effectiveness. Moreover, γ consistently achieves accurate sender selection in various conditions while other metrics can achieve the same accuracy only under special conditions. Besides, γ can be used for dissemination in both radio-always-on networks [8]–[10] and low-duty-cycled networks [24].

III. Motivating Examples

In this section, we use two examples to clearly present the motivation of our works.

A. Dissemination protocol using native packets

Figure 1 shows an example in which S1 and S2 are two potential senders while N1, N2 and N3 are three receivers. S1 and S2 intend to cover N1, N2 and N3 with a page of 10 packets. The quality for each directional link is indicated using a percentage. The percentages below S1 and S2 indicate the corresponding link correlation: the link correlation between S1→N1 and S1→N2 is 100%, indicating that when the transmission over S1→N1 fails, the transmission over S1→N2 also fails. The link correlation between S2→N2 and S2→N3

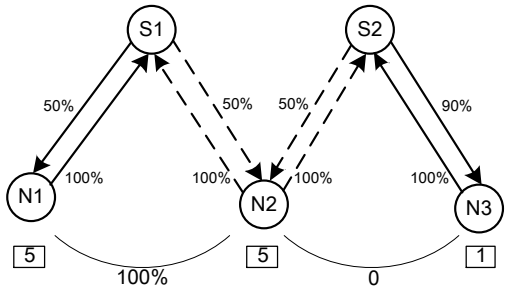


Fig. 2: A motivating example with network coding.

is 0, indicating that when the transmission over S2→N2 fails, the transmission over S2→N3 can succeed.

In each transmission round, the senders transmit all missing packets of the receivers in a batch. For example, if S1 intends to cover N1, it will transmit all N1's missing packets according to the received REQ from N1. At the end of the first round transmission, the reception statuses of the receivers in a given page are indicated by the blocks below, where a black block denotes a missing packet and a white block denotes a received packet. We can see that due to the impact of link correlation, N1 and N2 have the same missing packets while N2 and N3 have no common missing packets. The receivers respond with request (REQ) messages (carrying bit vectors indicating the reception statuses) for the missing packets.

We examine the performance of the existing sender selection metrics. Using ε_{ECD}, S2 will be first selected as the sender. This is because the impact of S2 ($\varepsilon_{ECD}(S2) = 0.9 + 0.5 = 1.4$) is larger than the impact of S1 ($\varepsilon_{ECD}(S1) = 0.5 + 0.5 = 1$). After selected as a sender, S2 starts transmitting the requested data packets to N2 and N3. The expected number of transmissions to cover N2 and N3 is $\frac{1}{0.9} + \frac{5}{0.5} = 11.1$. Note that we call a node covered only when it has received the whole page of packets. When N2 and N3 are both covered, N1 still has missing packets. To cover N1, S1 needs $\frac{5}{0.5} = 10$ transmissions. The total number of transmissions to cover all the receivers is 10+11.1=21.1.

When the random selection algorithm and MNP's metric are used, we cannot decide the sending priority of S1 and S2. Because both S1 and S2 have 2 receivers such that $\varepsilon_{MNP} = 2$ for both senders. Actually, S1 is the better one to be first selected. When S1 is selected, as N1 and N2 are requesting the same five packets, the expected number of transmissions to cover N1 and N2 is $\frac{5}{0.5} = 10$. Then N3 still needs 1 packet and S2 needs to cover N3 with $\frac{1}{0.9} = 1.1$ transmissions. The total number of transmissions to cover all the receivers is 10+1.1=11.1 $<$ 21.1.

With this example, we conclude that considering link quality alone cannot accurately estimate a sender's impact.

B. Dissemination protocols using network coding

Figure 2 shows a similar example in which S1 and S2 are two potential senders while N1, N2 and N3 are three receivers. When network coding is used, an encoded packet transmission is useful to a node as long as the node did not receive a sufficient number of encoded packets (i.e., 10 encoded packets in the example) [11] [12]. As a result, the reception statuses of the receivers are indicated by the number of missing packets, instead of a request bit vector. The numbers inside the rectangles indicate how many encoded packets are needed by the corresponding receiver to recover a given page. Another difference is that the number of transmissions by the sender is decided by the worst link from the sender to the receivers.

After the first round of transmission, the receivers receive some packets while still needing a certain number of packets to decode an entire page.

We also examine the performance of the existing sender selection algorithms. Using ECD's metric (or UFlood's metric), S2 will be first selected as a sender. This is because the impact of S2 ($\varepsilon_{ECD}(S2) = 0.5 + 0.9 = 1.4$) is larger than that of S1 ($\varepsilon_{ECD}(S2) = 0.5 + 0.5 = 1$). As a packet transmission is useful to a node as long as the node has not received a sufficient number of packets, the number of transmissions is decided by the worst link, i.e., S2→N2. The expected number of transmissions of S2 to cover both N2 and N3 is $\frac{5}{0.5} = 10$. After that, N1 still needs five encoded packets. To cover N1, S1 should transmit $\frac{5}{0.5} = 10$ packets. In total, the number of transmissions to cover N1, N2 and N3 is 10+10=20.

Similarly, both the random selection algorithm and MNP's metric cannot decide the sending priority of S1 and S2 in this example. Actually, S1 is the better one to be first selected. When S1 is selected, as N1 and N2 both request five packets, the expected number of transmissions to cover N1 and N2 is 5/0.5=10. After that, S2 needs only $\frac{1}{0.9} = 1.1$ transmissions to cover N3. The total number of transmissions is 10+1.1=11.1 $<$ 20.

From both case studies, we conclude that *link correlation affects the performance of bulk data dissemination protocols by affecting the reception statuses of receivers and it should be incorporated into the metric design in the sender selection algorithm for evaluating the effectiveness of a sender's broadcast*.

IV. The γ factor

Based on the above observation, we design γ, an accurate sender selection metric incorporating both link quality and reception statuses (i.e., bit vectors in REQ messages). The key idea of γ is to utilize the node reception statuses to accurately estimate the expected *gain/cost ratio* of a potential sender (i.e., the average expected number of receptions for each requested packet transmission). γ favors senders with the better link quality and stronger link correlation.

In general, γ can be calculated as follows:

$$\gamma_i = \frac{G_i}{C_i} \tag{5}$$

where G_i is the gain (i.e., the expected packet receptions) of node i, and C_i is the cost (i.e., the number of transmissions)

of node i. Next, we present the calculation of γ for both non-coding and coding based dissemination.

A. Notations

The notations are listed as follows.

- $\mu[i]$ denotes the reception utility of the i-th packet.
- γ_u denotes node γ value of u.
- N_u^{all} denotes the set of neighboring nodes of node u.
- N_u^{req} denotes the set of nodes who have sent requests to node u.
- $|N_u|$ denotes the size of N_u, i.e., the number of nodes that have sent requests to node u.
- q_{uv} denotes the link quality from node u to node v.
- R_{vu} denotes the request vector from node v to node u.
- $|R_{vu}|$ denotes the length of R_{vu}. For example, $|R_{vu}| = 48$ in Deluge since a default Deluge page consists of 48 packets.
- R_{vu}[i] denotes the i-th bit of R_{vu}. The value 1 denotes the corresponding packet is lost at v and needs to be retransmitted by u and 0 denotes the corresponding packet is correctly received at v.
- R_u denotes the combined request vector at node u.
- $|R_u|$ denotes the length of R_u.
- R_u[i] = $\vee_{v \in N_u} R_{vu}$[i]. If $R_u[i] = 1$, the i-th packet in the current page needs to be (re-)transmitted.
- n_v denotes the number of missing packets of node v within the receiving page.

B. Description of γ

We describe the calculation of γ in detail for both native packets and network coding based dissemination. When a potential sender receives requests from its receivers, we calculate γ to evaluate the effectiveness of the sender. A larger γ indicates a more effective sender, of which the transmissions are likely to be more efficient.

1) γ for dissemination with native packets: We first calculate the expected number of receptions for each requested packet of a sender, and then calculate γ for the sender.

Intuitively, when a packet is useful to more receivers, the packet's transmission is more beneficial. Hence, we define the utility of a packet to be the expected number of the receptions at all the neighboring nodes that need the packet. Suppose the i-th packet in the current page is requested, the reception utility is

$$\mu[i] = \sum_{v \in N_u} R_{vu}[i] \cdot q_{uv} \tag{6}$$

It is worth noting that only receptions at receivers who have requested the i-th packet (i.e., $R_{vu}[i] = 1$) are included in the calculation since other receivers have already received the packet ($R_{vu}[i] = 0$) and thus the packet is useless to those receivers.

As sender u needs to transmit each packet i with $R_u[i] = 1$, the total number of transmissions is $M = \sum_{i=1}^{|R_u|} R_u[i]$. We define the average number of receptions for each packet transmission as γ to reflect the gain/cost ratio as follows:

$$\gamma_u = \frac{\sum_{i=1}^{|R_u|} R_u[i] \cdot \mu[i]}{M} \tag{7}$$

The cost is the number of transmissions M, and the benefit is the sum of utilities of all requested packets. It is obvious that a larger γ value means a more effective potential sender with higher gain/cost ratio. We revisit the example in Figure 1, with Eq. (7), $\gamma_{S1} = 1$ and $\gamma_{S2} = 0.57 < \gamma_{S1}$. Hence, the better sender S1 is selected with γ.

2) γ for dissemination with network coding: γ works the same way for network coding based protocols, i.e., the average utility of each packet transmission. The key difference is that when network coding is used, a packet is useful for more receivers as compared to that with native packets, which partly reduces the impact of link correlation. This significantly affects the calculation of γ for dissemination with network coding.

When network coding is employed, a packet is useful to a receiver as long as the receiver did not receive sufficient number of linear-independent encoded packets. As a result, a sender needs to send a total number of $M = \max_{(v \in N_u^{req})} n_v$ packets, which are the cost of the sender.

Now we estimate the gain of the M transmissions. For each transmission, we consider the following two cases for the transmission utility of the i-th packet transmission ($1 \le i \le M$).

(1) For receiver k with $n_k \ge i$ (i.e., the number of the missing packets of node k is larger than that of i, which means the i-th packet is useful to node k if the packet is linear-independent with the received packets), the utility of packet to k is $q_{uk} \times (1 - p_l)$, where p_l denotes the probability that the encoded packet is linear-dependent with the received packets and can be calculated according to [11]. When using the default setting of random linear code in Rateless Deluge, $p_l = 0.00392$.

Hence, the utility to all these receivers with $n_k \ge i$ is:

$$\mu_1[i] = \sum_{k:n_k \ge i} q_{uk}(1 - p_l) \tag{8}$$

(2) For receiver k with $n_k < i$, the i-th packet is useful to node k only if k has not received n_k packets during the previous i-1 transmissions. We denote $P_{i-1}(k)$ as the probability that node k has not received n_k packets after i-1 transmissions. It is the sum of probabilities that k receives 0 up to $n_k - 1$ packets.

$$P_{i-1}(k) = \sum_{m=0}^{n_k - 1} \binom{i-1}{m} \cdot q_{uk}^m \cdot (1 - q_{uk})^{i-1-m} \tag{9}$$

The utility to all these receivers with $n_k < i$ is:

$$\mu_2[i] = \sum_{k:n_k < i} P_{i-1}(k) \cdot q_{uk}(1 - p_l) \tag{10}$$

Hence, the utility of the i-th packet is the sum of utilities to both the above kinds of receivers:

$$\mu[i] = \mu_1[i] + \mu_2[i] \tag{11}$$

According to the definition of γ, we calculate γ as follows:

$$\gamma_u = \frac{\sum_{i=1}^{M} \mu[i]}{M} \tag{12}$$

Let us revisit the example shown in Figure 2, with Eq. (12), $\gamma_{S1} = 1$ and $\gamma_{S2} = 0.699998 \approx 0.7 < \gamma_{S1}$, and the better sender S1 is selected with γ.

We can see that γ does not directly exploit link correlation for impact modeling. Instead, it uses the reception statuses at receivers to estimate the expected number of receptions for each requested packet. While link correlation provides an estimation of the number of common missing packets, the reception statuses directly provide the ground truth of the number of common missing packets (which is affected by link correlation). Hence, our calculation based on the reception statuses is reasonable and more accurate. As a result, γ favors the senders with strong correlated outbound links.

V. The γ-component

In this section, we present γ-component, a generic sender selection scheme for bulk data dissemination. We first present the designs of two essential modules to enable γ based sender selection: γ estimation and transmission contention. We then integrate these two modules into γ-component (i.e., sender selection implementation based on γ). Finally, we apply γ-component to both Deluge and Rateless Deluge (namely γ-Deluge and γ-Rateless Deluge).

A. γ Estimation

Before each round of transmission during the dissemination, the γ values of potential senders are updated for sender selection. According to Eq. (7) and Eq. (12), a sender needs to collect reception statuses and link quality information for γ estimation. We first discuss how to obtain these information.

(1) Obtaining R_{vu} and n_v. We get R_{vu} and n_v in the REQ message sent by uncovered nodes when missing packets are detected. There are two differences between γ's REQ mechanism and Deluge's REQ mechanism. First, multiple potential senders overhear the REQ message and may be responsible for sending requested packets in REQ that is not designated for them. This enlarges the set of potential senders so that we select the best one. Second, we note that in Deluge [8], a REQ message may be suppressed if another REQ message for the same page is overheard. This mechanism, however, will lead to biased estimation in our protocol design. For this reason, we require that the uncovered nodes send REQ messages unless there is an ongoing page transmission.

The setting of the estimation period for sending and receiving REQ messages depends on the time window during which nodes randomize the transmissions of REQ messages (as will be discussed in Section V.A.3). If the time window is too small, the probability of REQ collisions will increase. On the other hand, if the time window is too large, it will cause a long dissemination delay. In this paper, we set the estimation period to be consistent with Deluges default settings in order for a fair comparison.

(2) Obtaining q_{uv}. To estimate link quality, we incorporate the LEEP link estimation protocol [25] into our design. LEEP is a passive link estimation protocol that can be invoked in proactive protocols to update neighbors' link quality. It has shown to be effective in many protocols [10], [26], [27]

We attach the LEEP header (containing a seqno) to ADV, REQ, and DATA messages. Each node uses these messages to estimate the inbound link quality from neighboring nodes. Note that DATA messages are broadcasted in a batch to all neighboring nodes and can be used for inbound link estimation. This process effectively calibrates estimated link quality via control-plane messages. Moreover, we attach the LEEP footer (containing node IDs and their inbound link quality) to ADV messages. Therefore, the outbound link quality can be obtained by periodically exchanging the inbound link quality encompassed in the ADV messages.

Combining the requests sent by uncovered nodes and the link quality to the requesters, we can estimate a sender's impact according to Eq. (7) when using native packets or Eq. (12) when using network coding.

(3) REQ collection. The requirement that all uncovered nodes send REQ messages may lead to REQ collisions, especially in dense networks. The impact of REQ collisions is two-fold:

- For γ estimation, the REQ collisions will lead to incomplete feedback collection, which will lead to inaccurate sender impact estimation and sender selection.
- The REQ collisions may cause large delay, which might balance the reduced transmission delay by sender selection.

Density aware REQ mechanism

To deal with the REQ collisions, we propose a density aware REQ mechanism. The key idea is to adjust the REQ timer according to the network density to keep the collision probability under a threshold. We first model the relationship between REQ timer and the collision probability as follows.

Suppose there are N contending nodes in a neighborhood, and we denote the REQ timer as T_{REQ}. According to [28], the REQ attempt probability is $P_a = \frac{2}{T_{REQ}+1}$. The probability of a successful transmission of N contending nodes in a timer duration is the probability that only one node attempts to send an REQ in the timer period,

$$P_s = NP_a(1-P_a)^{N-1} \tag{13}$$

The probability that no nodes attempt to send an REQ in the timer duration is,

$$P_n = (1-P_a)^N \tag{14}$$

With the above probabilities, we can calculate the collision probability in the timer duration as follow.

$$P_c = 1 - P_s - P_n \tag{15}$$

We can see that the collision probability increases along with the network density. Then, we set the collision probability P_c to be under a threshold C_{thr}, and we can get the relationship between number of contending nodes and the timer duration as follows.

$$N \cdot \frac{2}{1+T_{REQ}} \cdot (1 - \frac{2}{1+T_{REQ}})^{N-1} + (1 - \frac{2}{1+T_{REQ}})^N = 1 - C_{thr} \tag{16}$$

When we set C_{thr} to be a constant, e.g., 0.5%, the REQ backoff timer can be obtained according to the network density with Eq. (16).

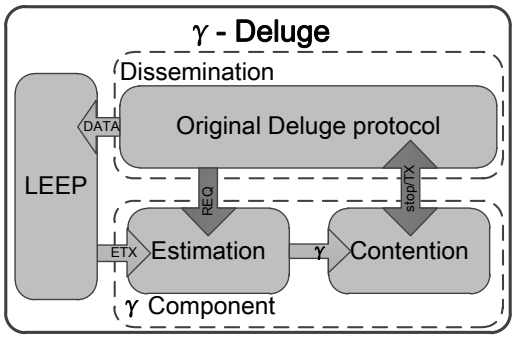


Fig. 3: The γ-component.

Fig. 4: The 8×10 testbed.

We also set a threshold Th_{req} for the REQ timer to avoid too large timer delay in very dense networks. As a result, the REQ backoff timer is $min(Th_{req}, T_{REQ})$.

Another challenge for γ estimation is how to align the estimation periods of different nodes within a neighborhood. If the senders' estimations start at different time, the result would be unfair. To address this problem, we adopt a similar approach as in [10]. We attach a `pendingPktNum` field to each data packet, indicating the number of remaining packets that the sender intends to transmit. With this information and the expected transmission time for a single packet, we can estimate the end time of the ongoing page transmission.

B. Transmission Contention

All nodes that overhear the REQ messages and have the requested page are potential senders. We should select the best sender in the potential senders, which has the largest γ value.

There are two alternatives for transmission contention: active contention and passive contention. (1) In active contention, potential senders exchange control messages to inform each other about the impact values. A node fails the contention if it detects a larger impact, and starts data transmission otherwise. Active contention requires multiple rounds of control message exchanges, which incurs considerable transmission and delay overhead. (2) In passive contention, each potential sender starts a back-off timer according to its own impact. A node starts transmission if its timer fires. If a node receives data packets before the back-off timer fires, it fails the contention and stops its own timer. The benefit is that it incurs no control message exchanges. The drawback is that it may fail to select the most effective sender if the back-off is not carefully designed.

Table 1: Series of dissemination protocols using native packets

estimation / contention	random	MNP's	ECD's	γ
None	Deluge [8]			
active		MNP [9]		
passive			ECD [10]	$\gamma-Deluge$

Table 2: Series of dissemination protocols using network coding (RD:Rateless Deluge, γ-RD:γ-Rateless Deluge)

estimation / contention	random	ECD's	γ
None	RD [11],Synapse [12]		
active		UFlood	
passive			γ-RD

We propose to use a passive approach, employing an impact-based back-off mechanism. Intuitively, in order to prioritize the transmissions of the potential senders with larger γ, we need to assign a short back-off time for a node with a large γ and a long back-off time for a node with small γ. There are several ways to correlate the back-off time with γ. Here, we simply use the reciprocal, $T_{backoff}(u) = \frac{C}{\gamma} + \Delta$, where C is a constant value and Δ is a small random value to differentiate the back-off time when γ values are the same. In our experiments, we found the back-off design is effective (see Section VI.C). With this back-off time design, the largest impact node is most likely to transmit first. Other nodes will cancel the data transmission. It is worth noting that compared to traditional sender selection approaches, γ brings only computational overhead (metric calculation). Considering that in low power network, radio operations (transmitting, receiving, idle listening) are the dominating source of energy consumption [29], [30], the overhead is negligible compared to the reduction of transmissions (see Section VI).

C. The γ-component

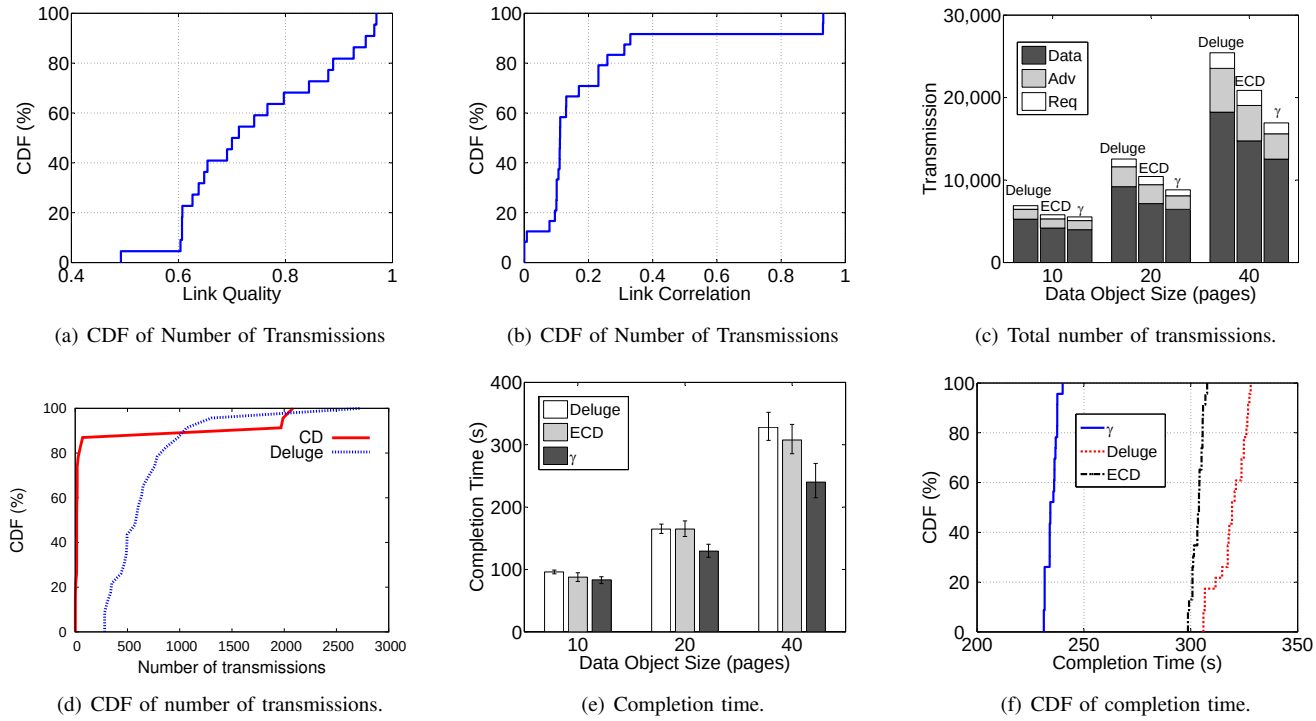
We abstract γ-component, a general sender selection scheme based on γ. Estimation and contention are two key interfaces of γ-component. The estimation interface estimates γ, and it depends on the LEEP component for estimating link quality. The contention interface prioritizes transmission according to the γ value. Figure 3 illustrates the relationship between these two interfaces.

The benefit of γ-component is two-fold. On one hand, we can easily incorporate γ into existing data dissemination protocols such as Deluge, MNP, ECD, and Rateless Deluge. On the other hand, we can also implement other sender selection algorithms such as in MNP or ECD for the same interface in order to compare the effect of different sender selection algorithms for the same dissemination protocol.

The γ-component abstraction also allows us to generate a series of different dissemination protocols as illustrated in Tables 1 and 2.

D. γ-Deluge and γ-Rateless Deluge

Figure 3 shows how we can incorporate γ-component with Deluge into γ-Deluge. The dark arrows indicate the information exchange of Deluge and γ-component.



(a) CDF of Number of Transmissions (b) CDF of Number of Transmissions (c) Total number of transmissions.

(d) CDF of number of transmissions. (e) Completion time. (f) CDF of completion time.

Fig. 5: Performance comparison of Deluge, ECD and γ-Deluge.

γ-Deluge works as follows: When receiving an REQ message, Deluge invokes the estimation interface. Combining the REQ from Deluge and link quality information from LEEP, the estimation interface updates the γ value. When the estimation period ends, the updated γ value is delivered to the contention interface. Based on the γ value, the contention interface starts a back-off timer. When the timer fires, the contention interface informs Deluge to start data transmission. If Deluge receives data messages during the back-off timer period, the contention fails and the node does not start transmissions.

Similarly, we incorporate γ-component with Rateless Deluge into γ-Rateless Deluge. The difference lies in that Rateless Deluge invokes a decoding component when receiving an entire page.

VI. Evaluation

We now move to performance evaluation by implementing γ-Deluge and γ-Rateless Deluge in TinyOS 2.1.1 [21]. We first conduct testbed experiments to compare existing protocols using two key performance metrics, i.e., the number of transmissions and the completion time of the dissemination. We then evaluate γ through large scale simulation in TOSSIM [31] to gain more insights by analyzing the sender selection behaviors and the impact of link correlation.

A. Methodology

We built a 8×10 testbed with TelosB [32] nodes to form a multi-hop low power network (shown in Figure 4). We first measure the link characteristics of the network.

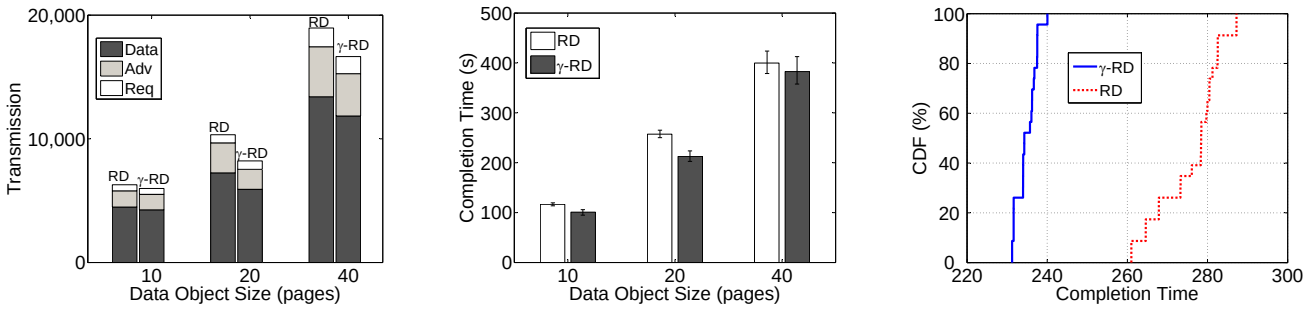
Figure 5(a) shows the CDF of pair-wise link quality. With the power setting, different links have different link quality. Figure 5(b) shows the CDF of average outbound link correlation for each node. Both good link correlation and poor link correlation exist.

In γ-Deluge, we set the page size and packet payload size using Deluge's default settings (1034K bytes/page and 23 bytes/packet). In γ-Rateless Deluge, we set the page size and packet payload size using Rateless Deluge's default settings (460K bytes/page and 23 bytes/packet).

We place a sniffer node near the testbed for listening reports from each node. At the beginning, the sink node broadcasts a `start` message at the maximum radio power. Upon receiving the `start` message, the sniffer node records the start time of dissemination. Each node broadcasts a `report` message once it has received the whole data object, also in the maximum radio power. When `report` messages from all nodes are collected at the sniffer, we can get the performance metrics from the sniffer. We also use local logging to record the interested events at each node in external flash.

We use two key metrics for comparison:

1) Completion time. It is the time from the start of dissemination to the end of dissemination at each individual node. The network completion time is the maximum completion time among all nodes.
2) Number of transmissions. The number of transmissions include data packet transmissions and control packet transmissions.



(a) Total Number of Transmissions (b) Completion Time (c) Completion Time

Fig. 6: Performance comparison of Rateless Deluge and γ-Rateless Deluge.

B. Dissemination Performance

We first compare the overall performance of dissemination protocols with γ-component and their counterparts (both protocols using native packets and network coding).

1) Comparison with dissemination protocols using native packets: We compare γ-Deluge with Deluge and ECD in terms of the number of transmissions and the completion time.

Figure 5(c) shows the performances of γ-Deluge, Deluge and ECD in terms of the number of transmissions. The data object size ranges from 10, 20 to 40 pages (each page consisting of 1K bytes). For typical low power wireless applications based on TelosB motes, few data objects are larger than 40K bytes. We classify three kinds of transmissions according to the message types (i.e., ADV, REQ, and DATA). Both ADV and REQ are control-plane messages. (1) For ADV, γ-Deluge reduces 8.6% and 1.5% transmissions compared to Deluge and ECD, respectively. The reason is that ADV is broadcasted periodically by each node, as γ-Deluge has shortest completion time, it has fewest ADV transmissions. (2) For REQ, γ-Deluge also reduces transmissions by 6% and 3% compared to Deluge and ECD, respectively. Although Deluge employs an REQ message suppression, it generally postpones the REQ transmission instead of cancelling it (unless it receives all the missing packets during the postponed period). Hence, its suppression does not greatly reduce the REQ transmissions. As good links are selected in γ and ECD, γ and ECD reduce the number of REQ transmissions. γ also reduces the REQ transmissions compared with ECD. The reason is that γ-Deluge considers the reception statuses, which enables the sender's data transmissions in γ-Deluge effectively cover more receivers than that in ECD. More covered receivers produce fewer REQ transmissions. (3) For DATA, γ-Deluge reduces transmissions by 20.4% and 16.5% compared to Deluge and ECD, respectively. The reason is that the selected senders' data transmissions in γ-Deluge have more receivers. We can also see that, the reduction in data transmissions is larger than control transmissions, and the reduction increases when the page size increases.

Figure 5(d) shows the CDF of the number of transmissions for each node in γ-Deluge, Deluge and ECD, respectively. We can see that γ-Deluge has the lowest transmission overhead. This is because γ jointly considers link quality and reception statuses, and thus selects more efficient senders. We can also see that there are several γ-Deluge nodes that have more transmissions than the other two protocols. The reason is that with γ-component, the nodes with good links are almost selected all the time. As a result, there are fewer senders selected in γ-Deluge, and some senders transmit more in γ-Deluge than in Deluge.

Figure 5(e) shows the result for the completion time. ECD has shorter completion time compared to Deluge for its link quality aware sender selection. γ-Deluge further reduces the completion time by 20.5% compared to Deluge and 16.1% compared to ECD, respectively. The reason is two-fold: First, γ-Deluge exploits the reception statuses, which provides fine-grained information about link correlation and missing packets. Thus γ can accurately estimate sender impact and select senders with lower ETX. Second, the transmission contention does not incur message exchange, which also reduces the contention time.

Figure 5(f) shows the CDF of the completion time for each node in γ-Deluge, Deluge, and ECD, respectively. The result shows that for each individual node, γ-Deluge has the shortest completion time. Although the contention module in γ-component incurs an extra back-off timer compared to Deluge, it greatly reduces the number of transmissions. As a result, it still reduces the completion time compared to Deluge.

2) Comparison with dissemination protocols using network coding: We compare γ-Rateless Deluge with Rateless Deluge in terms of transmissions and the completion time.

Figure 6(a) shows the performances of γ-Rateless Deluge and Rateless Deluge in terms of the number of transmissions. The result shows that (1) γ-Rateless Deluge reduces the total number of transmissions by 10.5% compared to Rateless Deluge. This is because γ-Rateless Deluge employs sender selection which favors good link quality. (2) The transmission reduction of γ-Rateless Deluge to Rateless Deluge is less than that of γ-Deluge to Deluge. Apparently, it is due to the impact of network coding. By using network coding, a packet transmission is useful for more receivers. Hence, Rateless Deluge greatly reduces the number of transmissions than Deluge. However, although γ-Deluge does not use network coding, it considers the expected reception number of a transmission. In γ-Rateless Deluge, a packet can also be received by more receivers than that in γ-Deluge, but the

improvement is not as significant as that of Rateless Deluge to Deluge. As a result, the reduction of γ-Rateless Deluge to Rateless Deluge is less than that of γ-Deluge to Deluge.

Figure 6(b) shows the result of the completion time. γ-Rateless Deluge reduces the completion time by 11.9% compared to Rateless Deluge. The reduction is less than that of γ-Deluge to Deluge (γ-Deluge reduces the completion time by 20.5% compared to Deluge). This is because the decoding delay in rateless protocols occupies a large fraction of the completion time. As reported in [16], Rateless Deluge outperforms Deluge mainly in networks with high link loss. We envision that the improvement of γ-Rateless Deluge will be significant in large and lossy networks.

Figure 6(c) shows the CDF of the completion time for each node in γ-Rateless Deluge and Rateless Deluge, respectively. We can see that (1) for all nodes, the completion time of γ-Rateless Deluge is less than that of Rateless Deluge. (2) The result in γ-Rateless Deluge falls into a small range (i.e., 230-240), while the result in Rateless Deluge is distributed in a large range (i.e., 260-290). This may be due to that γ-Deluge favors the sender with strong outbound link correlation. Hence, receivers tend to receive the object at the same time. Rateless Deluge employs random sender selection, and the completion time is sparsely distributed.

C. System Insights

We conduct both testbed experiments and TOSSIM simulations [31] to gain more system insights of the γ factor. We first validate whether our metric accurately selects the appropriate sender for actual data transmission. We then perform correlation studies to examine the characteristics of most selected senders. Finally, we conduct experiments to explore how link correlation affects protocol performance.

1) Sender selection accuracy: We perform simulation studies on γ-Deluge, Deluge, MNP and ECD in TOSSIM.

We use the LossyBuilder provided in TinyOS to generate a topo file containing pair-wise link quality. The topology is a 10×10 grid with 5 feet inter-node spacing. Link quality and pair-wise link correlation are both randomly set and distributed. The sink node starts the dissemination of 10 pages. We collect over 2000 times of sender selections by repeating the simulation.

We output each node's reception statuses using the `dbg` statements so that we know the network statuses at any instant. We define a selection event when a node (is selected and) starts transmitting data packets. Upon each event, we judge whether it is the right sender with the smallest ETX according to the link condition and the reception statuses of all neighboring nodes. We define the accuracy of sender selection as the number of correct decisions divided by the total number of selection events. Correspondingly, we define the error of sender selection as the number of wrong decisions divided by the total number of selection events.

Table 3 shows the selection errors of γ-Deluge, ECD, MNP and Deluge, respectively. We further classify the errors based on the causes. (1) Estimation error which means the sender with the highest estimated metric is not the real best sender. This may due to insufficient requests are collected or the request messages are simply lost. (2) Contention error which means that the sender with the highest estimated metric is the right one but it loses the contention in the contention phase.

Table 3: Sender selection accuracy.

Protocols	Selection err.	Estimation err.	Contention err.
γ-Deluge	10.9%	2.1%	8.8%
ECD	29.5%	20.4%	9.1%
MNP	33.2%	21.1%	12.1%
Deluge	64.3%	-	-

Table 4: The Pearson's correlation between protocol performance and various factors.

Factors / Protocols	link correlation	link quality	hop count
Rateless Deluge	0.104352	0.131334	-0.428381
γ-Rateless Deluge	0.383020	0.266300	-0.401995
Deluge	0.087548	0.171700	-0.589562
ECD	0.281017	0.238520	-0.559540
$\gamma-Deluge$	0.453888	0.234081	-0.558477

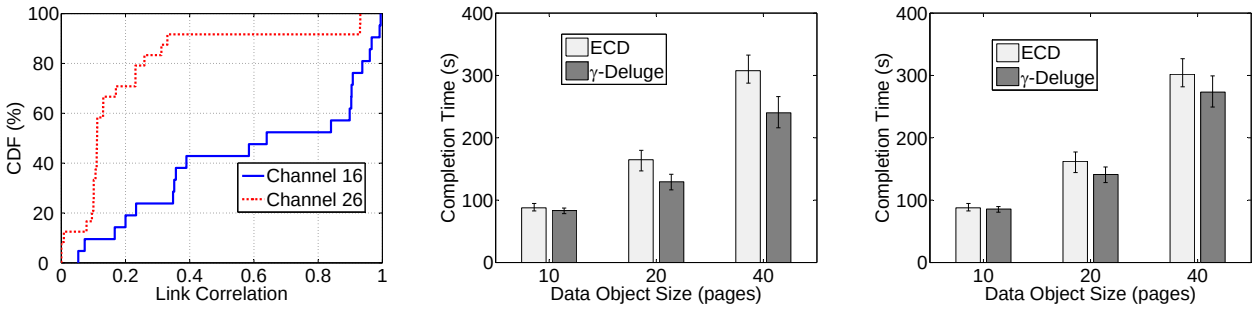
From the results in Table 3, the γ factor achieves the smallest overall selection error. It yields more accurate sender selection by 155.2%, 36.1% and 29.2% compared with the metrics used in Deluge, MNP, and ECD. The estimation error is much smaller than the contention error. The large contention error is caused by the small fraction of randomness in the back-off timer based contention design.

For other protocols, the estimation errors are much larger. The sender selection error rates of ECD, MNP and Deluge are 29.5%, 33.2% and 64.3%, respectively. This is because they do not take into account the joint impact of link quality and reception statuses. As Deluge uses a random sender selection scheme, its expected selection error should be $1-\frac{1}{N}$, where N is the average number of the potential senders (neighboring nodes which receive the requests and stores the page to send). The selection error of Deluge (64.3%) is consistent with the above expected value with N=3 in our simulation.

2) Characteristics of sender selection: We now investigate which senders are more likely to transmit more packets. We perform correlation analysis between the number of data transmissions and three impact factors, i.e., average link quality to neighboring nodes, average link correlation value in the neighborhood, and the hop count from the sink. We use Pearson correlation coefficients to explore the correlation between the number of transmissions for each node and various factors that impact the dissemination process.

Table 4 shows some facts. First, hop count has strong negative correlation with the transmission overhead. This is easy to understand since the sink starts the dissemination process. Second, in both ECD and γ based protocols (i.e., γ-Deluge and γ-Rateless Deluge), link quality has positive correlation with the transmission overhead as all the protocols take link quality into account. Most importantly, in γ based protocols, link correlation also has a stronger positive correlation with the transmission overhead than other protocols.

3) Impact of link correlation: Next, we explore the performance of γ in different conditions with different levels of link correlation. We conduct experiments in our lab with



(a) CDF of link correlation. (b) Comparing completion time in channel 16. (c) Comparing completion time in channel 26

Fig. 7: Comparison of channel 16 and channel 26.

channels 26 and 16, respectively. Channel 16 is overlapped with the WiFi channel used in the testbed room. As reported in [16] and our measurement, communications over this channel is supposed to be interfered with WiFi signals and experience high link correlation. On the other hand, channel 26 is known to be non-overlapped with any WiFi channels. Hence, communications over channel 26 will experience lower link correlation.

Figure 7(a) shows the CDF of link correlation for all link pairs of a common sender. As expected, channel 16 experiences a higher link correlation than channel 26.

Figure 7(b) shows the completion time of γ-Deluge and ECD in channel 16. Figure 7(c) shows the completion time of γ-Deluge and ECD in channel 26. γ-Deluge outperforms ECD in both channels. However, γ-Deluge's performance improvement to ECD at channel 16 is 16.14% while the performance improvement at channel 26 is only 6.74%. This can be explained as follows. As analyzed in Section IV.C, when link quality is similar for the two channels, link correlation essentially reshapes the missing packets of each node and affects the number of common missing packets. At channel 16, there are more cases that receivers have similar missing packets. As a result, γ-Deluge has more opportunity to select senders with both good outbound link quality as well as strong link correlation in the neighborhood. While in channel 26, the link correlation is inherently weak, and the receivers tend to have random or different missing packets. There are less opportunity for γ-Deluge to find neighborhood with strong link correlation. Hence, γ-Deluge's performance in channel 16 is better than channel 26. In contrast, ECD has similar performances in channel 16 and channel 26. Consequently, the performance improvement of γ-Deluge to ECD in channel 16 is larger than that in channel 26.

4) Impact of density aware REQ mechanism: The accuracy of γ with static REQ timer decreases along with the network density. Comparatively, γ with density aware REQ timer consistently achieves accurate sender selection. The accuracy will start to decrease when the network density is above a certain threshold. The reason is that in such situation, the optimal REQ timer has exceeded Thr_{req} and the timer is set as Thr_{req} when network density is above the threshold. Experimental results on the impact of density aware REQ mechanism can be found in our technical report.

The discussions on the impact of time-varying link characteristics, the comparison between γ and κ/μ, the situations where γ is the most benefical and the application of γ to routing protocols and be found in our technical report at *http://www.emnets.org/dongw/pub/TON-gamma.pdf*.

VII. Conclusion

In this paper, we first identify that link quality alone is far from enough for accurate sender selection. Then we propose γ, an accurate sender selection metric that takes both link quality and link correlation to accurately calculate the *gain/cost ratio* for evaluating each sender's effectiveness. We further present γ-component, a generic sender selection scheme based on γ, which can be easily adopted by most dissemination protocols. We incorporate γ-component with existing dissemination into two novel protocols: γ-Deluge and γ-Rateless Deluge.

We implement γ-Deluge and γ-Rateless Deluge in TinyOS on a TelosB motes testbed. We conduct comprehensive experiments and large scale TOSSIM simulations. Results show that: (1) γ achieves more accurate sender estimation compared to the metrics in MNP and ECD/UFlood. (2) Both γ-Deluge and γ-Rateless Deluge outperform existing bulk data dissemination protocols. By large scale simulations, we confirm that γ favors senders with good link quality and strong link correlation for both native packets and network coding based dissemination protocols.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks*, vol. 38, pp. 393–422, 2002.

[2] Y. Gao, W. Dong, K. Guo, X. Liu, Y. Chen, X. Liu, J. Bu, and C. Chen, "Mosaic: A low-cost mobile sensing system for urban air quality monitoring," in *Proc. of IEEE INFOCOM*, 2016, pp. 1–9.

[3] W. Du, Z. Xing, M. Li, B. He, L. H. C. Chua, and H. Miao, "Optimal sensor placement and measurement of wind for water quality studies in urban reservoirs," in *Proc. of IPSN*. IEEE, 2014, pp. 167–178.

[4] N. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-efficient routing protocols in wireless sensor networks: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 2, pp. 551–591, 2013.

[5] Y. Liu, G. Zhou, J. Zhao, G. Dai, X. Li, M. Gu, H. Ma, L. Mo, Y. He, J. Wang *et al.*, "Long-term large-scale sensing in the forest: recent advances and future directions of greenorbs," *Frontiers of Computer Science in China*, vol. 4, no. 3, pp. 334–338, 2010.

[6] L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X. Li, and G. Dai, "Canopy Closure Estimates with Greenorbs: Sustainable Sensing in The Forest," in *Proc. of SenSys*, 2009.

[7] W. Du, Z. Li, J. C. Liando, and M. Li, "From rateless to distanceless: Enabling sparse sensor network deployment in large areas," in *Proc. of SenSys*. ACM, 2014, pp. 134–147.

[8] J. W. Hui and D. Culler, "The Dynamic Behavior of A Data Dissemination Protocol for Network Programming At Scale," in *Proc. of SenSys*, 2004.

[9] S. Kulkarni and L. Wang, "Energy-efficient multihop reprogramming for sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 5, no. 2, p. 16, 2009.

[10] W. Dong, Y. Liu, Z. Zhao, X. Liu, C. Chen, and J. Bu, "Link Quality Aware Code Dissemination in Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1776–1786, July 2014.

[11] A. Hagedorn, D. Starobinski, and A. Trachtenberg, "Rateless Deluge: Over-the-air Programming of Wireless Sensor Networks Using Random Linear Codes," in *Proc. of IPSN*, 2008.

[12] M. Rossi, N. Bui, G. Zanca, L. Stabellini, R. Crepaldi, and M. Zorzi, "SYNAPSE++: Code Dissemination in Wireless Sensor Networks Using Fountain Codes," *IEEE Transactions on Mobile Computing*, vol. 9, no. 12, pp. 1749–1765, 2010.

[13] W. Dong, C. Chen, X. Liu, J. Bu, and Y. Gao, "A Lightweight and Density-Aware Reprogramming Protocol for Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 10, pp. 1403–1415, 2011.

[14] Z. Zhao, W. Dong, J. Bu, Y. Gu, and C. Chen, "Link-correlation-aware data dissemination in wireless sensor networks," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 9, pp. 5747–5757, 2015.

[15] J. Subramanian, R. Morris, and H. Balakrishnan, "UFlood: High-throughput flooding over wireless mesh networks," in *Proc. of INFOCOM*, 2012.

[16] K. Srinivasan, M. Jain, J. Choi, T. Azim, E. Kim, P. Levis, and B. Krishnamachari, "The κ Factor: Inferring Protocol Performance Using Inter-link Reception Correlation," in *Proc. of MobiCom*, 2010.

[17] W. Dong, Y. Liu, Y. He, T. Zhu, and C. Chen, "Measurement and analysis on the packet delivery performance in a large-scale sensor network," *IEEE/ACM Transactions on Networking*, vol. 22, no. 6, pp. 1952–1963, 2014.

[18] T. Zhu, Z. Zhong, T. He, and Z. Zhang, "Exploring Link Correlation for Efficient Flooding in Wireless Sensor Networks," in *Proc. of NSDI*, 2010.

[19] Z. Zhao, W. Dong, G. Guan, J. Bu, T. Gu, and C. Chen, "Modeling link correlation in low-power wireless networks," in *Proc. of INFOCOM*. IEEE, 2015, pp. 990–998.

[20] S. Wang, G. Tan, Y. Liu, H. Jiang, and T. He, "Coding opportunity aware backbone metrics for broadcast in wireless networks," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 275–279.

[21] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer *et al.*, "TinyOS: An Operating System for Sensor Networks," *Ambient intelligence*, vol. 35, pp. 115–148, 2005.

[22] W. Du, J. C. Liando, H. Zhang, and M. Li, "When pipelines meet fountain: Fast data dissemination in wireless sensor networks," in *Proc. of SenSys*. ACM, 2015, pp. 365–378.

[23] Y. Wang, Y. He, X. Mao, Y. Liu, and X.-y. Li, "Exploiting constructive interference for scalable flooding in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 21, no. 6, pp. 1880–1889, 2013.

[24] K. Han, J. Luo, L. Xiang, M. Xiao, and L. Huang, "Achieving energy efficiency and reliability for data dissemination in duty-cycled wsns," *Networking, IEEE/ACM Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[25] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis, "Four-bit wireless link estimation," in *Proc. of HotNets)*, 2007.

[26] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proc. of SenSys*, 2009.

[27] 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. of SenSys*, 2009.

[28] M. Heusse, F. Rousseau, R. Guillier, and A. Duda, "Idle sense: an optimal access method for high throughput and fairness in rate diverse wireless LANs," in *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4. ACM, 2005, pp. 121–132.

[29] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, L. Mo, W. Dong, Z. Yang, M. Xi, J. Zhao, and X.-Y. Li, "Does Wireless Sensor Network Scale? A Measurement Study on GreenOrbs," in *Proc. of INFOCOM*, 2011.

[30] D. Kagaris, "Maximizing the lifetime of a wireless sensor network with fixed targets," *Ad-hoc & sensor wireless networks*, vol. 17, no. 3-4, pp. 253–268, 2013.

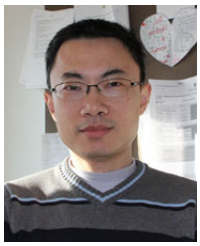
[31] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in *Proc. of SenSys*, 2003.

[32] TelosB Datasheet, "Available online: http://www. willow. co. uk," *Datasheet, TelosB/(accessed on 12 January 2010)*.

Zhiwei Zhao (S'11-M'16) received his PhD degree in computer science from Zhejiang University in 2015. He is currently an assistant professor at the College of Computer Science and Engineering in University of Electronic Science and Technology of China (UESTC). His research interests include on wireless computing, heterogeneous wireless networks, protocol design and network coding. He is a member of IEEE.

Wei Dong (S'08-M'11) received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 2005 and 2011, respectively. He is currently an Associate Professor with the College of Computer Science, Zhejiang University .His research interests include network measurement, wireless and mobile computing, and sensor networks. He is an member of IEEE and ACM, and a senior member of CCF.

Jiajun Bu (M'06) received the B.S. and Ph.D. degrees in computer science from Zhejiang University, China, in 1995 and 2000, respectively. He is a professor in College of Computer Science and the deputy dean of the Department of Digital Media and Network Technology at Zhejiang University. His research interests include embedded system, mobile multimedia, and data mining. He is a member of IEEE and ACM.



Tao Gu (S'03-M'07-SM'14) is currently an Associate Professor in Computer Science at RMIT University, Australia. He received his Bachelor degree from Huazhong University of Science and Technology, M.Sc. from Nanyang Technological University, Singapore, and Ph.D. in computer science from National University of Singapore.His current research interests include mobile computing, ubiquitous/pervasive computing, wireless sensor networks, distributed network systems, sensor data analytics, Internet of Things, and online social networks. He is a Senior Member of IEEE and a member of ACM.

Geyong Min (M'01) is the Chair Professor and Director of High Performance Computing and Networking (HPCN) Research Group at the University of Exeter, UK. He received the PhD degree in Computing Science from the University of Glasgow, UK, in 2003, and the B.Sc. degree in Computer Science from Huazhong University of Science and Technology, China, in 1995. He joined the University of Bradford as a Lecturer in 2002, became a Senior Lecturer in 2005 and a Reader in 2007, and was promoted to a Professor in Computer Science in 2012. His main research interests include Next-Generation Internet, Wireless Networks, Mobile Computing, Cloud Computing, Big Data, Multimedia Systems, Information Security, System Modelling and Performance Optimization.