Bulk Data Dissemination in Wireless Sensor Networks: Analysis, Implications and Improvement

Xiaolong Zheng, *Member, IEEE,* Jiliang Wang, *Member, IEEE,* Wei Dong, *Member, IEEE,* Yuan He, *Member, IEEE,* and Yunhao Liu, *Fellow, IEEE*

Abstract—To guarantee reliability, bulk data dissemination relies on the negotiation scheme in which senders and receivers negotiate transmission schedule through a three-way handshake procedure. However, we find negotiation incurs a long dissemination time and seriously defers the network-wide convergence. On the other hand, the flooding approach, which is conventionally considered inefficient and energy-consuming, can facilitate bulk data dissemination if appropriately incorporated. This motivates us to pursue a delicate tradeoff between negotiation and flooding in the bulk data dissemination. We propose SurF (Survival of the Fittest), a bulk data dissemination protocol which adaptively adopts negotiation and leverages flooding opportunistically. SurF incorporates a time-reliability model to estimate the time efficiencies (flooding vs. negotiation) and dynamically selects the fittest one to facilitate the dissemination process. We implement SurF in TinyOS 2.1.1 and evaluate its performance with 40 TelosB nodes. The results show that SurF, while retaining the dissemination reliability, reduces the dissemination time by 40% in average, compared with the state-of-the-art protocols.

Index Terms—Wireless sensor networks, protocol design, bulk data dissemination

1 Introduction

Wireless sensor networks (WSNs) [1] have been applied in a variety of application areas such as environmental monitoring [2], structural protection [3], and military surveillance and so on. Most WSNs, once deployed, are intended to operate unattended for a long period. During the lifetime of a WSN, it is often necessary to fix bugs, reconfigure system parameters, and upgrade the software. Bulk data dissemination is one of the building blocks of WSNs that enable the above-mentioned important tasks.

In this paper, we focus on the bulk data dissemination from the sink to the nodes, which is usually used for over-the-air firmware updates and reprogramming. The updates are not uncommon. For example, in the decade of TinyOS development, 11 major versions are released [4]. Each version contains several major changes and numerous bug fixes. Bulk data dissemination is then necessary to fulfill the demands. We envision that the operating system of WSN will have even more frequent updates as the operating system of smart phones such as Android. Besides the operating system updates, we need to fix the bugs in the application software which makes the bulk data dissemination more frequent.

Generally, bulk data dissemination in WSNs must meet two requirements. First, it should be *reliable* despite of the unreliable wireless links in the network. Second, they should be *time-efficient* to converge for the entire network. A long dissemination time means sustained interruptions in the normal network operations, which is not desired. It is therefore significant to shorten the dissemination process. Most of the existing bulk data dissemination protocols disable duty cycling and keep the radio awake during the dissemination process to restore to normal system functions as soon as possible. Hence, a short dissemination time usually provides good energy efficiency because radio activity accounts for most of the energy consumption on sensor nodes.

A number of protocols have been proposed in recent years, such as, Deluge [5] which adopts the negotiation scheme proposed in [6] to guarantee reliability and reduce redundant transmissions. Three types of messages are defined in Deluge. ADV messages are advertisement messages that a node uses to announce the version of the data it possesses. REQ messages are request messages that a node uses to request its interested data after receiving the ADV messages. The requested data will be packaged into DATA messages. Every Deluge node periodically broadcasts ADV messages to announce its own latest version of data. Neighboring nodes hear the ADV messages and send REQ messages to the ADV sender if a newer version is found. After receiving the REQ messages, the node starts sending DATA messages.

We notice that the negotiation scheme, although effective for ensuring the reliability of data delivery, incurs a large overhead in terms of dissemination time. In a typical experiment with two TelosB nodes transmitting 10KB data using Deluge, the time spent on negotiation comprises 71% of the total dissemination time, which is far beyond the usual expectation.

The analysis and experiment results motivate us to *selectively* use the negotiation scheme *only when absolutely necessary* throughout the entire dissemination process, so as to improve the dissemination efficiency while retaining reliability. On

- *Xiaolong Zheng, Jiliang Wang, Yuan He, and Yunhao Liu are with TNLIST and School of Software, Tsinghua University. E-mail: {xiaolong, jiliang, he, yunhao}@greenorbs.com*
- *Wei Dong is with the College of Computer Science, Zhejiang University. E-mail: dongw.cs@gmail.com*
- *Jiliang Wang is the corresponding author.*

the other hand, dissemination without negotiation (so-called *flooding*) makes each node probabilistically broadcast a packet n times.

We observe that (1) for a certain success ratio of dissemination, flooding often has a much shorter dissemination time. This is because it can quickly increase reliability in the initial phase since most of the nodes do not have the latest data. (2) On the other hand, for a higher dissemination success ratio, flooding becomes inefficient because blind flooding without feedback tends to result in large amounts of redundancy and unsatisfactory results. In contrast, the use of negotiation in that phase may effectively avoid redundancy by explicitly requesting the missing packets.

In this paper, we propose **SurF** (**Sur**vival of the **F**ittest), a bulk data dissemination protocol which *selectively* utilizes negotiation to improve efficiency. Flooding is adopted as a substitute for negotiation *opportunistically*. SurF adaptively decides the best strategy and switches between flooding and negotiation to achieve improved dissemination efficiency while remaining reliable.

A key issue in SurF's design is to determine when and how nodes transit between the two schemes (flooding vs. negotiation). Bad transition timing may result in a longer dissemination time. SurF incorporates a time-reliability model to predict the time efficiency of the two schemes. Based on that model, each SurF node estimates the potential benefit brought by either of the two schemes, respectively and makes a decision dynamically about the most appropriate dissemination scheme in a distributed manner.

We implement SurF based on TinyOS 2.1.1 and evaluate its performance on a 40-nodes testbed. The evaluation results demonstrate that (1) the model within SurF can accurately predict the completion time of two schemes. (2) SurF reduces the dissemination time by 40%, compared to Deluge.

The contributions of this paper are summarized as follows.

- We find that the selective use of negotiation and opportunistic leveraging of flooding will improve the dissemination time without harming reliability.
- We adopt an accurate time-reliability model to estimate and predict the performance of different schemes. Through such model, our method can capture the opportunities of selective negotiation to improve the time efficiency.
- We implement SurF and evaluate its performance through experiments on real testbeds. The results demonstrate the advantages of SurF in terms of dissemination time, compared with Deluge.

The rest of this paper is organized as follows. Section 2 analyzes the two kinds of existing methods, flooding and negotiation-based methods. Section 3 describes the analytical model for estimating the dissemination performance. Section 4 elaborates on the design of SurF. Section 5 presents the evaluation results. Section 6 discusses the related work, and Section 7 concludes this paper.

2 Analysis

In this section, we show the analysis of two kinds of existing methods, flooding and negotiation-based methods. Firstly, we summarize and analyze their merits and deficiencies. Then we use experiments to test and verify the analysis of these two kinds of methods.

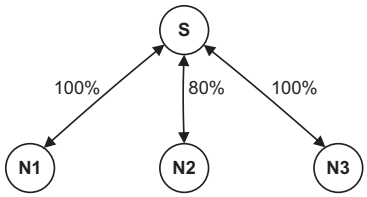

Fig. 1. Example of unnecessary negotiation

2.1 Intuitive analysis

Flooding A straightforward way to carry out bulk data dissemination is flooding. In flooding, each node immediately rebroadcasts the received packets, hence, the flooding process can be quite quick. However, since the data are transmitted by broadcasting, which has no ACK or NACK, the reliability cannot be guaranteed. Considering the unreliable wireless links, the reliability can be very lower, so to improve the reliability, retransmission is needed. However, without the help of ACK or NACK, the senders have no idea of which packets are missing. Hence, all the packets have to be retransmitted, resulting in the so-called blind retransmission problem. The blind retransmission problem incurs unnecessary retransmissions and prolongs the completion time of achieving high reliability. No guarantee of reliability is the biggest obstacle for flooding to be adopted in bulk data dissemination.

When applying flooding in dense networks, there is another problem known as the broadcast storm problem. The broadcast storm problem has three chief phenomena: redundancy, contention and collision [7]. The broadcast storm problem may result in a long completion time.

Negotiation-based methods Negotiation is proposed to settle the problems that flooding suffers from. Negotiation adopts control messages acting as the NACK to avoid the blind retransmission problem and guarantee the reliability. It also selects only one forwarder in a local area to alleviate the broadcast storm problem.

Even though the negotiation scheme is successful in overcoming the shortcomings of flooding, it is prone to have a prolonged completion time since the additional control process postpones data transmissions. In addition, during negotiations, the nodes have to turn the radio on to overhear the control messages to finish the negotiation process. This process does not disseminate any data packets, however it consumes the same energy as transceiving.

At first glance, negotiation seems to be a must for bulk data dissemination. However, we surprisingly find that negotiation is *not always* needed during the *whole* dissemination process. Actually, negotiation may be unnecessary in some cases. Take the topology in Fig. 1 as an example. Node S has three neighbors N1, N2 and N3. The numbers near the link are link qualities, measured by Packet Reception Ratio (PRR). Suppose node S has a shorter distance to the sink node than the other three nodes. As a consequence, when S receives

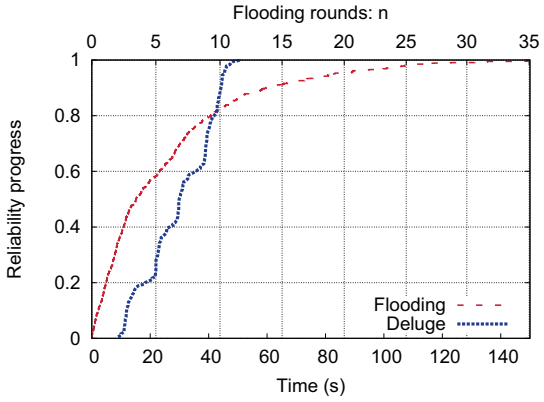

Fig. 2. Reliability progress of flooding and Deluge

a batch of new data from upstream nodes, there is a high probability that N1, N2 and N3 do not have the new data. In such a case, node S can predict that the result of negotiation will be that the data has a high probability of being required by other nodes. Node S can therefore directly broadcast the new data without negotiation. In this way, that particular round of negotiation is not necessary and can be eliminated to improve efficiency. Otherwise, N1, N2 and N3 may not receive the new data until the negotiation is finished, prolonging the dissemination time. However, if we eliminate negotiation totally, blind retransmissions may occur due to the unreliable link between S and N2. Through this example, we can see that negotiation should be adopted but not during the whole process as there are unnecessary negotiations that prolong the completion time meaninglessly.

2.2 Validation experiments

To validate our analysis, we conduct the validation experiments. We use an indoor testbed which consists of 40 TelosB nodes to study the performance of these two kinds of methods. The network is formed as a grid topology with the sink placed at the bottom left corner. Distance between two adjacent nodes on the grid is 20cm. We set the radio power level to 1 to emulate multi-hop transmissions. Under such settings, the average PRR of all the links is 0.8, measured by 4-bit link estimator from [8].

- Flooding. Each node performs probabilistic flooding with probability p [7], that is, upon receiving a new packet, the receiver rebroadcasts the packet once with probability p. For the sink node, $p = 1$, and for other nodes $p = 0.9$. We let the sink node incessantly repeats flooding all the packets sequentially. We call it one flooding round when the sink node finishes broadcasting all the packets once.
- Negotiation-based dissemination. We use the default Deluge protocol with a page size of 48 packets.

We disseminate five pages in the network. We define the *node reliability* at time t as the success ratio of dissemination on this node, that is, the ratio of the number of unique received packets before time t to the total number of necessary packets (i.e. 48×5). We define the *network reliability* as the average of all node's reliability. When discussing the reliability progress in the remainder of this paper, it refers to the success ratio of the dissemination.

We record the receiving events with a timestamp. We also keep a record of the completion time of each round of flooding. Fig. 2 presents the time-reliability curve obtained from our experiments. We can see that (1) flooding can achieve a certain level of reliability quickly. For example, flooding is faster than the negotiation scheme before the reliability reaches a certain threshold (e.g., 80% in Fig. 2). (2) However, negotiation is more efficient in making up the remaining reliability after a certain point of high reliability. For example, after getting 80% of reliability, to make up the remaining 20% of reliability, the negotiation scheme takes only around 10 seconds while flooding takes more than 100 seconds. (3) A combination of flooding and negotiation may improve the time efficiency. We want the dissemination to start with fast flooding to get to a certain level of reliability quickly and then turn to negotiation to perform retransmissions for a guarantee of reliability. If we can put flooding and negotiation into use at the right time, we can improve the time efficiency. (4) We need an analytical model to decide how to integrate flooding and negotiation schemes. A naive combination of the two schemes with fixed flooding rounds cannot work well. Turning to negotiation too early cannot exploit the full advantages of flooding while turning to it too late may also degrade the performance.

3 Analytical Models

In this section, we present the analytical model for SurF, which seizes the most opportune moment to put negotiation to use, so as to minimize the completion time.

To allow distributed computation at each node, we model SurF's completion time in a neighborhood. In Section 4.6, we also analyze the multi-hop performance improvement. We further show that SurF's local optimality often leads to considerable network performance improvement through the experiments, presented in Section 5. Here we simply regard the local optimality as the objective of optimization. Therefore, we estimate $T_{ij}^{H}(n,\phi,q_{ij})$, the completion time at node i with SurF, given the number of flooding n, the reliability requirement ϕ, and the worst link quality q_{ij} from i to one of its neighbors j. SurF minimizes the completion time by finding the optimal transition point between two schemes, based on the analytical model.

Some notations used in our design are listed below.

- N, the number of packets in one page.
- p_i, the rebroadcasting probability of node i in flooding.
- T_{pkt}, the average transmission time per packet.
- T_{back}, the expected back-off time before sending out a packet.
- R_j^0, the initial node reliability at j.
- N_{supp}, the expected number of suppressed ADVs due to the suppression scheme used in negotiation.
- τ_l, the expected time between two successive ADVs.
- τ_r, the expected time between two successive REQs.
- $R_j(n,R_j^0)$ denotes the expected reliability that node j obtains after node i flooding the data n times, given that

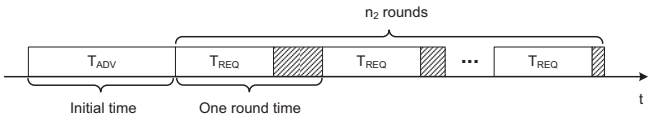

Fig. 3. Composition of the completion time of the negotiation scheme

node j already has a reliability of R_j^0, which is:

$$R_j(n,R_j^0) = 1-(1-R_j^0)\cdot(1-p_iq_{ij})^n \tag{1}$$

For all the nodes in the negotiation and sink in the flooding, $p_i = 1$. For other nodes in the flooding, $p_i \in (0,1]$.

- $n(\phi,R_j^0)$, the number of transmission rounds. We define one transmission round as one transmission of a page, including the corresponding control process. Given the required reliability ϕ, n can be calculated by letting $R_j(n,R_j^0) \geq \phi$. Based on Eq. 1, it is:

$$n(\phi,R_j^0) = \left\lceil \frac{\log(1-\phi)-\log(1-R_j^0)}{\log(1-p_iq_{ij})} \right\rceil \tag{2}$$

- $n_F(\phi,R_j^0)$, the transmission rounds to achieve the required reliability ϕ by only flooding, given the initial node reliability R_j^0.
- $T_{ij}^F(n)$, the time needed by flooding from i to j, with n rounds.

$$T_{ij}^F(n) = n\cdot N\cdot(T_{pkt}+T_{back}) \tag{3}$$

- $T_{ij}^N(\phi,R_j,q_{ij})$, the time needed by negotiation from i to j, given the reliability requirement of ϕ, the already achieved reliability R_j, and the link quality q_{ij}.

From these notations, $T_{ij}^H(n,\phi,q_{ij})$ can be estimated as:

$$T_{ij}^H(n,\phi,q_{ij}) = T_{ij}^F(n)+T_{ij}^N(\phi,R_j(n,0),q_{ij}) \tag{4}$$

where $T_{ij}^F(n)$ is given in Eq. 3 and $T_{ij}^N(\phi,R_j(n,0),q_{ij})$ is analyzed as follows. In the following analysis, we omit the parameters for simplicity when no ambiguity occur.

As shown in Fig. 3, the time of the negotiation scheme T_{ij}^N comprises three parts: (1) $T_{ij}^{ADV}(q_{ij})$, the time of the initial ADV from a sender; (2) $n_2T_{ij}^{REQ}(q_{ij})$, the time of n_2 rounds REQ transmission, taking $T_{ij}^{REQ}(q_{ij})$ time for each round; (3) $T_{ij}^{DATA}(R_j(n,0),q_{ij})$, the time of DATA transmission. That is:

$$T_{ij}^N(\phi,R_j(n,0),q_{ij}) = T_{ij}^{ADV}+n_2\cdot T_{ij}^{REQ}+T_{ij}^{DATA} \tag{5}$$

where n_2 is the transmission rounds of the negotiation phase to meet the reliability of ϕ, given the reliability already achieved by flooding with n rounds, $R_j(n,0)$. That is:

$$n_2\big(\phi,R_j(n,0)\big) = \left\lceil \frac{\log(1-\phi)-\log(1-R_j(n,0))}{\log(1-p_i\cdot q_{ij})} \right\rceil \tag{6}$$

The time of the initial ADV from node i to j is the expected time that j receives an ADV from i, which is given by

$$T_{ij}^{ADV}(q_{ij}) = \tau_l\cdot\left(\frac{1}{q_{ij}}-1+N_{supp}\right) \tag{7}$$

The time of the REQ per round is the expected time node i receives a REQ after j receives an ADV:

$$T_{ij}^{REQ}(q_{ij}) = \tau_r\cdot\left(\frac{1}{q_{ij}}-1\right)+E[N_{ADV}]\cdot T_{ij}^{ADV}(q_{ij}) \tag{8}$$

where $E[N_{ADV}]$ is the expected number of additional ADVs needed in one round. Note that nodes cannot send unlimited REQ in one round and stops trying after λ times, the maximum number one can try before the next ADV is heard. Suppose X is a random variable which represents the number of REQs transmitted for one page. Thus, $X \sim G(p_{ij})$. Then the expected number of additional ADVs during one page's dissemination is:

$$\begin{aligned} E[N_{ADV}] &= \sum_{k=0}^{\infty} k\cdot P(N_{ADV}=k) \\ &= \sum_{k=0}^{\infty} k\cdot P(k\lambda < X \leq (k+1)\lambda) \\ &= \frac{(1-q_{ij})^\lambda}{(1-(1-q_{ij})^\lambda)} \end{aligned} \tag{9}$$

Even though the DATA transmission scatters in different rounds, shown as the dashed areas in Fig. 3, the time can be measured by the expected number of transmitted packets for this page. Given the reliability that has already been achieved by flooding n times, $R_j(n,0)$, and the link quality between i and j, the time of the DATA is the expected time of transmission for this page, which can be written as

$$T_{ij}^{DATA}(R_j(n,0),q_{ij}) = \frac{\big(1-R_j(n,0)\big)}{q_{ij}}N(T_{back}+T_{pkt}) \tag{10}$$

Note that Eq. 5 gives the completion time of SurF in the case where $n_2 \neq 0$. We can re-express the completion time defined in Eq. 4 in a more general form. That is:

$$T_{ij}^H(n,\phi,q_{ij}) = \begin{cases} T_{ij}^N(\phi,0,q_{ij}), & n=0 \\ T_{ij}^F(n)+T_{ij}^N(\phi,R_j(n,0),q_{ij}), & 0<n<n_F \\ T_{ij}^F(n_F), & n=n_F \end{cases}$$

N_F is the upper bound of flooding rounds, n, since the reliability requirement can be fulfilled by flooding alone. When $n=0$, the integration becomes retrograde in negotiation-based methods. When $0<n<n_F$, the integration leverages two schemes. When $n=n_F$, the integration turns into flooding. However, since the required reliability in reliable bulk data dissemination is 100%, then $n_F \to \infty$, that is, the negotiation is necessary to guarantee reliability. Therefore, $n \in [0,n_F)$. We prove that given ϕ and q_{ij}, $T_{ij}^H(n,\phi,q_{ij})$ has the following properties.

Property 1: When n is in the continuous space [0, $+\infty$), there exists such a n^* that: $T_{ij}^H(n^*,\phi,q_{ij})$ is the minimum value; when $n \in [0,n^*)$, $T_{ij}^H(n,\phi,q_{ij})$ is monotonically decreasing; when $n \in (n^*,+\infty)$, $T_{ij}^H(n,\phi,q_{ij})$ is monotonically increasing.

Property 2: When $n \in Z$, $\lceil n^* \rceil$ and/or $\lfloor n^* \rfloor$ make(s) $T_{ij}^H(n,\phi,q_{ij})$ minimum, where n^* satisfies *Property 1*.

Property 3: There is one and only one n^* that satisfies *Property 1*. It is also the only one that satisfies $\frac{dT_{ij}^H}{dn}=0$. The

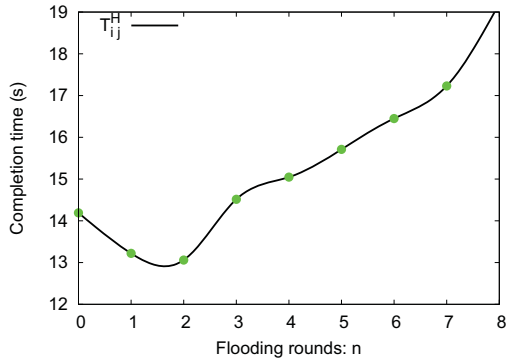

Fig. 4. Completion time vs. flooding rounds

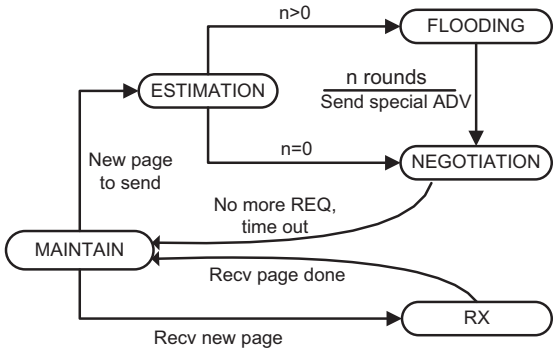

Fig. 5. State transition diagram of SurF

close form expression of n^* is as follows.

$$n^* = \frac{\ln \frac{q_{ij}\cdot(T_{ij}^{REQ}(q_{ij}) - N\cdot(T_{pkt}+T_{back}))}{N\cdot(T_{pkt}+T_{back})\cdot\ln(1-p_i\cdot q_{ij})}}{\ln(1-p_i\cdot q_{ij})} \tag{11}$$

These properties can easily be proved by analyzing the function, $T_{ij}^H(n^*,\phi,q_{ij})$. Hence, we omit the proofs. Instead, we give a concrete example to show the properties of T_{ij}^H. Consider the dissemination between i and j. The parameters in the negotiation scheme are: $\tau_l = 2s$, $\tau_r = 0.5s$, $\lambda = 2$, and $N_{supp} = 1$, which are all consistent with previous work [5]. Other parameters are: $T_{back} = 19ms$, $T_{pkt} = 1ms$, $N = 100$, and $p_i = 0.9$. Given that $q_{ij} = 0.5$, $\phi = 99\%$, we can get $n_F = 8$ in this case. The function curves of $T_{ij}^H(n, 99\%, 0.5)$ is shown in Fig. 4. The function curves reveals the properties of $T_{ij}^H(n,\phi,q_{ij})$. In this case, $n^* = 1.6$ and $\lceil n^* \rceil = 2$ satisfy *Property 1* and *Property 2*.

Note that when SurF decides negotiation is not efficient, it leverages flooding as a substitute. Hence, to decide the optimal transition point for a minimal completion time is equivalent to deciding the optimal flooding n. From the models and analysis, the benefits of SurF are presented and the optimization demand of flooding rounds n is revealed. However, how to exploit the benefits and find the optimal n in a distributed manner needs to be addressed.

4 Design

Based on the analytical results, we present the design of SurF. SurF adopts several design principles: (1) exploiting the model to select the dissemination strategy; (2) adapting to the dynamic networks; (3) segmentation and pipelining for scalability.

4.1 Overview

Fig. 5 shows the state transition diagram of SurF. At the beginning, each node stays in a *maintain* state. When a node gets an updated page, it estimates the benefit of different strategies by our analysis model. According to the estimation result, the node will switch to the *negotiation* state or *flooding* state. If a node switches to the *flooding* state, the node firstly floods the page for n rounds, where n is the estimated optimal rounds. Afterwards, the node turns into the *negotiation* state to finish the dissemination by the negotiation scheme to guarantee the reliability. If a node in *negotiation* does not receive any REQ message for a certain time period, it returns to the *maintain* state and sends out the periodical ADV messages. When a SurF node receives packets of new page, it transits into *rx* state and will not return to the *maintain* state until it receives the whole page. Based on the main working flow, SurF has three key components: (1) parameter estimation component; (2) the strategy selection component; (3) the state switching component. In the following subsections, we present SurF's detailed design.

4.2 Parameter estimation

There are two parameters to estimate in our design. We need to estimate link qualities and the number of suppressed ADVs.

4.2.1 Link qualities

The link quality is the key parameter in our design to select the fittest strategy. SurF incorporates the LEEP link estimation protocol [9] to estimate the link qualities. The LEEP header contains a sequence number to help the receiver estimate the inbound link quality from one neighbor by counting the successfully received packets among all the packets that neighbor transmits. Outbound link qualities can be obtained by the advertisements from its neighbors, which announce its inbound link qualities. We attach the LEEP header to all the messages in SurF, including flooding data packets, ADV, REQ, and DATA messages. The plentitude of data packets can effectively estimate the inbound link qualities. SurF integrates the outbound link quality information into ADV messages instead of extra advertisement packets. The neighbors can then learn its outbound link qualities by its neighbors' periodical ADV messages.

4.2.2 Number of suppressed ADVs

In the design of the negotiation scheme, the sender suppresses the ADV packet if similar ADV packets are overheard in its neighborhood. As a knock-on effect, T_{ADV} may be delayed since each suppression results in one additional ADV waiting time. In previous work such as [5], linear topology is assumed and N_{supp} is assumed to be 1. Therefore, the accurate estimation of N_{supp} in a general topology is crucial to estimate

the time of the negotiation scheme. SurF measures N_{supp} by measuring the expected number of upstream neighbors who have the chance to suppress the ADVs, taking link qualities into consideration. The expected number of suppressed ADVs of node i is measured as:

$$N_{supp} = \sum_{j \in M_i^{UP}} 1 \cdot q_{ji} \tag{12}$$

where M_i^{UP} is the set of upstream neighbors.

4.3 Strategy selection

The decentralized strategy selection component decides the fittest strategy to minimize the completion time in a single hop. Based on *Properties 1, 2 and 3* and the estimated parameters, we can calculate and select the best strategy, that is, the flooding rounds n. The overall operations on each node for strategy selection are as follows.

Step 1: Get the link qualities and N_{supp} from the parameters estimation component;

Step 2: Calculate n based on *Property 1-3*;

Step 3: If $n > 0$, switch to $flooding$; If $n = 0$, switch to *negotiation*;

Step 4: If a new page is received, go to *Step 1*.

There are two problems to address while selecting the strategy in SurF. First, the diversity of link qualities should be considered. Normally, the completion time of bulk data dissemination is decided by the completion time of the last node. However, using the worst link quality to select a strategy is not appropriate since this incurs too many redundancy flooding rounds. On the other hand, selecting a very high link quality can lead to too many negotiations without fully utilizing flooding. To address this problem, we use the median of the link qualities for all neighbors to decide n. Through this approach, the flooding approach can achieve a quick progress and not be influenced by some extremely low link qualities.

Second, the diversity of the node statuses should be taken into consideration. The downstream neighbors of a sender may be able to receive the data packets earlier than the sender from other paths. The sender should not consider covering those nodes. In our design, we obtain the neighbors' statuses by overhearing. When a downstream neighbor is sending the same page as the sender, the sender will omit this node from the neighbor set when calculating the flooding rounds n for this page. Due to the status diversity of the receivers, if the newly estimated n is smaller than the actual flooding rounds already conducted by the sender, the sender switches to the *negotiation* state.

4.4 State switching

Through strategy selection, each node can select the best strategy during the dissemination process and switch among those strategies. Switching among different strategies should be carefully designed to fulfill the following requirements.

First, the switching should be efficient. When nodes transit from the $flooding$ state to the *negotiation* state, the sender periodically sends out ADVs to set up the negotiation with the receivers. However, this introduces significant overhead since the initial ADV negotiation time is quite long, especially when link qualities are poor or N_{supp} is large. To improve the efficiency, we subtlety use the data packets in the flooding to serve as the initial ADVs. This significantly reduces the overhead of ADVs.

Second, the switching should be error-resilient. As soon as a node transits to the *negotiation* state, it broadcasts multiple special ADVs to notify its neighbors about the transition. The receivers then send REQs to request the missing packets. After multiple special ADVs, the sender will periodically send out the ADV messages to remain consistent. Hence, even if the receiver lost all the special ADVs, it still can acquire the missing packets by periodical ADVs.

This way, SurF can significantly reduce the negotiation time while still guaranteeing delivery of the ADVs to the receivers.

4.5 Negotiation-based and flooding methods in SurF

SurF can be incorporated to any negotiation-based and flooding protocols. In current SurF design, we use an improved version of Deluge, the standard dissemination protocol of TinyOS. We improve it by reducing the initial ADV packets in Deluge by leveraging data packets in the flooding.

We use the probabilistic flooding [7] as the dissemination scheme in the flooding phase. Probabilistic flooding is a light-weight broadcast scheme which alleviates the broadcast storm problem from two aspects: (1) mitigating collision by the random back-off scheme; (2) reducing the redundancy by probabilistic rebroadcasting. The random backs-off time used in SurF is 10-25 ms. In SurF, the rebroadcasting probability should be adaptive to network conditions due to pivotal nodes. Pivotal nodes are the nodes whose children can only get packets from them [10]. Without the pivotal nodes broadcasting, some of the children cannot receive any packet. Therefore, the rebroadcasting probability of the pivotal node should be adaptively adjusted. In SurF, we use the REQ messages to detect the pivotal nodes. In each page's dissemination, the sender keeps the record of the flooded packets. By receiving REQ from the receiver, the sender can check the dependency of the receiver to it.

Let U denotes the set of all packets in one page, S denotes the set of packets the sender floods out, D_j denotes the set of lost packets known by the REQ from node j. We can define neighbor j's dependence indicator I_j as

$$I_j = \frac{|(U-S) \bigcap D_j|}{|U-S|} \tag{13}$$

I_j reveals the dependence between sender and receiver j. The rationale behind this is that if a receiver does not receive any packet that sender has not sent, the receiver has a high dependency on the sender since no other sender can cover this receiver with high probability. Then p_i can be revised based on the following three rules.

Rule 1: If $\exists j$, $I_j = 1$, then set $p_i = 1$;

Rule 2: If $\forall j$, $I_j = 0$, then set $p_i = 0$;

Rule 3: If $\forall j,\ I_j < 1$ and $\exists j,\ I_j > 0$, then adjust p_i by the moving average: $p_i = \alpha p_i^{'} \overline{I_j} + (1-\alpha) p_i^{'}$, where $p_i^{'}$ is the original rebroadcasting probability, $\overline{I_j}$ is the average value of all the dependence indicators, and α is the coefficient that represents the weight. In SurF, we empirically set $\alpha = 0.2$.

4.6 Discussions

4.6.1 Multi-hop performance

In our design, each node selects the dissemination strategy based on the local information. Nevertheless, we show that our approach can also lead to network-wide improvement. The completion time of the multi-hop network is determined by the last completed node. Suppose the path from the sink to the last completed node is: $PT_{st} = (r_0, r_1, \ldots, r_{|PT_{st}|})$, where r_0 is sink s and $r_{|PT_{st}|}$ is the last node t. Considering interference between different nodes, when node t receives a page, the next page is at least three hops away with pipelining. Therefore, if the $|PT_{st}| = h > 3$, then the completion time in multi-hop T_H can be depicted as:

$$T_H = \sum_{i=0}^{h} T^H_{r_{i-1}r_i} + n_{pg} \cdot \sum_{i=h-2}^{h} T^H_{r_{i-1}r_i} \tag{14}$$

where n_{pg} is the total number of pages of the code image. By Eq. 14, we can see that the local optimization can also lead to global improvement on the completion time in the network.

4.6.2 Generality of SurF

SurF can also be incorporated to other negotiation-based and flooding protocols. SurF improves the efficiency chiefly by reducing the unnecessary negotiation process, which is judged by the time-reliability model. No matter which negotiation-based method is adopted, SurF can reduce the completion time if there is an unnecessary negotiation process. Based on Eq. 4, when incorporating other new flooding or negotiation-based methods, SurF can learn the new incorporated model based on new expressions of $T^F_{ij}(n)$ and $T^N_{ij}(\phi, R_j(n,0), q_{ij})$. SurF then works according to the new time-reliability model.

For example, if coding-based flooding methods are adopted, then $T^F_{ij}(n)$ should be calculated as:

$$T^F_{ij}(n) = T_d + \sum_{i=1}^{n} \cdot N_i \cdot (T_{back} + T_{pkt}) \tag{15}$$

where T_d is the time cost of decoding the packets, N_i is the number of transmitting packets in round i.

In negotiation, MNP [11] is another popular negotiation-based method. It implicitly divides time into ADV slots. In each slot, each node estimates the benefit of its own transmission. Then the node with the highest benefit will transmit in this ADV slot and others keep silent. Based on MNP's mechanism, if an MNP-like method is adopted, then $T^N_{ij}(\phi, R_j(n,0), q_{ij})$ should be changed to:

$$T^N_{ij}(\phi, R_j(n,0), q_{ij}) = n_2 \cdot T^{ADV}_{ij} + T^{DATA}_{ij} \tag{16}$$

Fig. 6. Testbed

4.6.3 Practical issues of SurF

Computational complexity. The computational overhead mainly depends on the calculation of n based on Eq. 11. When calculating the optimal n based on Eq. 11, there is natural logarithm operation, which is not supported by MSP430F1611 on our TelosB sensor mote. However, we find that the ranges of parameters of the logarithm operation should be $(0,1]$. Therefore, in our implementation, we store a natural logarithm value table in the range $(0,1]$. By this way, we store 100 values to avoid the logarithm operation. Then the total computational complexity is O(1). In practice, the computation time of optimal value is a small constant.

Memory requirement. The extra memory requirement of SurF comes from three aspects. The first one is the memory used for storing a natural logarithm value table. Storing 100 numbers requires 400 bytes. The second one is the memory of the neighbor table that stores the neighboring nodes link qualities and other information. It costs 200 bytes for the neighbor table to store 40 entries. For other variables such as counters, it costs 96 bytes. The total extra memory requirement of SurF is then around 700 bytes. This memory consumption is affordable since the size of the RAM on one TelosB mote is 10K bytes.

5 Evaluation

We implement SurF based on TinyOS 2.1.1/TelosB. In this section, we evaluate its performance. First, we validate the accuracy of the model proposed in Section 3. Second, we compare SurF with Deluge on a real WSN testbed to show SurF's performance improvements.

5.1 Evaluation methodology

We conduct our evaluation on a testbed consisting of 5×8 TelosB nodes, as shown in Fig. 6. The network is formed as a multi-hop mesh network. The topology is a grid with the sink placed at the bottom left corner. Distance between two adjacent nodes on the grid is about 20cm. We set the radio power level to 1 to emulate multi-hop transmissions. The neighbors of a node are usually the nodes nearby in the grid. We use Deluge with its default configurations (e.g., 48 packets per page) as the

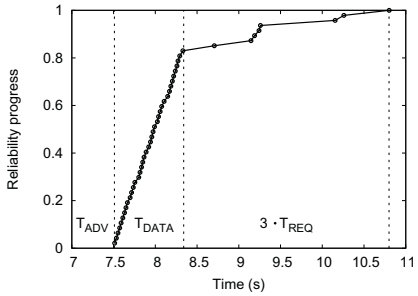

Fig. 7. Reliability progress of Node 18 when disseminating one-page data

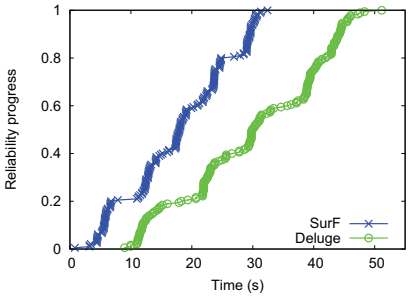

Fig. 8. Completion time of SurF and Deluge when disseminating five-page data

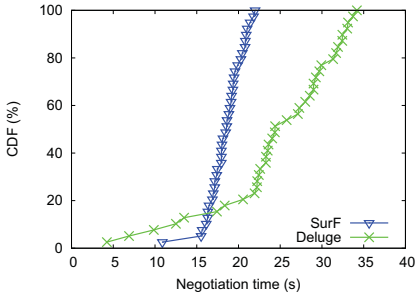

Fig. 9. CDF of negotiation time of 39 nodes when disseminating five-page data

representative of negotiation-based schemes. In the negotiation phase of SurF, the configurations are similar to Deluge.

To get the performance metrics of different protocols, we leverage the statistic reporting component in our previous work [12]. It can report counters of different events as well as each individual event. To analyze SurF, we add one bit into the packet header as the indicator of the dissemination approach. If it is 0, the packet is sent in the flooding phase. Otherwise, the packet is sent in the negotiation phase. When receiving a packet, the receiver records the receiving event with information such as timestamp and the dissemination approach for further analysis. Synchronization is performed at the start of each experiment. The sink broadcasts the time synchronization packets with the maximum power. The remaining nodes can thus synchronize to the sink after receiving the synchronization packets. After each experiment, we collect all the local logs via serial communications.

5.2 Evaluation results

5.2.1 Model validation

For the model validation, we inject a data image of one page into the sink and disseminate the data to the rest of nodes. We elect representative nodes to analyze their dissemination behaviors. Fig. 7 depicts the dissemination process on node 18. From the trace, we learn that node 18 receives the page from node 9 and the link quality between them is roughly 70%, measured by LEEP, as introduced in Section 4.2.1.

At $t = 0$, N9 completes the reception of the page from its upstream nodes and it receives a REQ message from N18 and starts transmitting DATA packets at t=7.5s. After $t = 8.3s$, N9 finishes one round of transmission and starts to retransmit the requested packets at t=8.7s and t=9.1s. Since a node has a limit of 2 responsive REQs for one ADV, N18 has to wait for the next ADV from N9 at t=10.2s. Then the missing packets are requested by the following REQ. The time of the initial ADV is 7.5s and the time of DATA and 3 rounds of REQs is 3.3s. N18's completion time is thus 10.8s in total.

On the other hand, based on our analysis model, $T_{ADV} = 6.9s$. Due to the grid topology, N9 has three reliable upstream neighbors (i.e. N0, N1, and N8) who can possibly suppress the N9's ADV. Therefore, N_{supp} is 3. $T_{REQ} = 0.9s$, and the needed rounds n is 3 based on Eq. 1; $T_{DATA} = 1.4s$, where $(T_{back} + T_{pkt}) = 20ms$ in our model. The total time of one page is $11s$ based on our model. We can see that the error is only $0.2s$, which is relatively small. The results validate the accuracy of our model.

5.2.2 Protocol performance

For the protocol performance evaluation, we inject a data image of 5 pages (i.e., approximately 5KB) into the sink and disseminate the data from the sink. We record the dissemination progress of Deluge and SurF during the experiments. Fig. 8 compares the performance of SurF and Deluge in our testbed. It takes $32.4s$ for SurF and $51.2s$ for Deluge to disseminate 5 pages. In this case, SurF achieves about 40% performance improvement, compared to Deluge. SurF shortens the completion time for the following two reasons. First, SurF has shorter inter-page negotiation time (e.g., time spent in the ADV phase) compared to Deluge. This is because SurF nodes flood the new page after the reception without extra negotiation time to initiate actual data transmissions. Second, SurF has a shorter page dissemination time. Due to the flooding strategy it adopts, SurF can achieve a certain level of reliability quickly. Hence, the transmission rounds of negotiation can be greatly reduced, resulting in reduced dissemination time.

Actually, SurF shortens the completion chiefly by reducing the negotiation overhead, especially the initial ADV time. This idle-waiting time is better utilized in SurF, by useful data transmissions.

Fig. 9 plots the CDF of negotiation time of all nodes. The negotiation time is the time spent on the control process, that is, T_{ADV} and T_{REQ}. We can see that (1) in Deluge, nearly 80% of nodes use more than 23s in negotiation, (2) in SurF, all the nodes spend less than 23s in negotiation. The long negotiation time in Deluge is mainly due to losses of control messages of negotiation. On the contrary, the transmission of a batch of data packets and consecutive ADVs in SurF are more robust.

Fig. 10 depicts the reliability progress of page 4 between node 0 and node 1 where node 0 is the sink. We can see that Deluge takes three rounds of negotiation to finish the dissemination. On the other hand, in SurF, the initial flooding achieves 54% reliability in 1s, leaving 22 missing packets to be covered in the negotiation phase. SurF reduces one negotiation rounds for the transmission of page 4.

To gain more detailed insights of SurF's performance, we

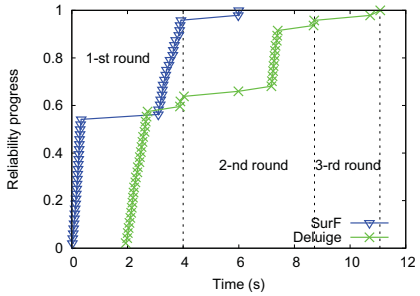

Fig. 10. Reliability progress of one page

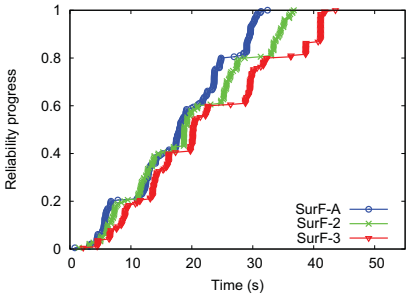

Fig. 11. Completion times of different SurF

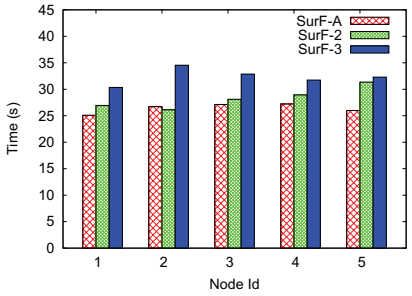

Fig. 12. Time of negotiation phase in different SurF

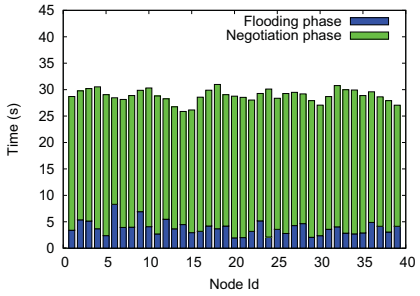

Fig. 13. Time composition of flooding phase and negotiation phase on nodes

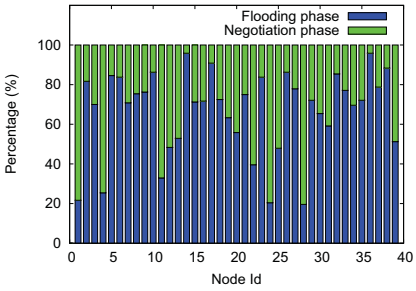

Fig. 14. Percentage of packets received in flooding phase and negotiation phase

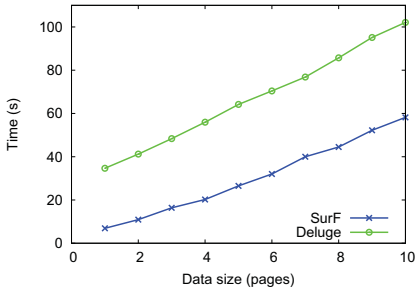

Fig. 15. Completion time for varying data sizes

evaluate SurF with different configurations: (1) SurF-A in which the value of n is adaptively estimated in a distributed manner as described in Section 4.3; (2) SurF-2 in which n is fixed to 2 for all nodes; (3) SurF-3 in which n is fixed to 3 for all nodes. Fig. 11 shows the effectiveness of our adaptive decision making algorithm. The completion times are 43.6s, 36.7s, and 32.4s for SurF-3, SurF-2, and SurF-A respectively. Compared to Deluge with a completion time of 51.2s, all three strategies reduce the completion time. However, the different strategies result in different performance gains. Compared to Deluge, the completion time is reduced by 37% for SurF-A, 28% for SurF-2, and 15% for SurF-3. The online adaptive SurF-A has a shorter completion time compared with the other two strategies. It also further reduces the completion time by 9% compared to SurF-2, and by 22% compared to SurF-3. Under the same experiment setting, these results indicate that the adaptive decision in SurF can achieve an optimized performance.

Fig. 12 depicts the times of the negotiation phase in SurF with different configurations. We use five nodes (nodes 1-5) for illustration. From the figure, we can see that the adaptive decision making algorithm can make the right decision to shorten the completion time, for individual nodes. Since nodes are in different environments in the network, the strategies for different nodes can be different. A fixed strategy is not proper due to the diversity of nodes and network dynamics. It is clear that for nodes 1, 3, 4 and 5, SurF-A has a shorter negotiation time compared to the fixed strategies, SurF-2 and SurF-3. For node 2, even though SurF-A has a longer negotiation time, it actually saves time by spending less time on flooding which is 3.6s. The results show that adaptive SurF makes individual strategy for each node and therefore achieves the shortest completion time compared to the two fixed strategies. The adaptive decision making method is effective to select the best integrating strategy to reduce the completion time.

Fig. 13 depicts the times of the flooding phase and negotiation phases on each individual node with SurF. It shows the times of 39 nodes (excluding the sink, node 0). In the experiment, flooding one page takes 0.73s on average. Hence, we can see that all SurF nodes use flooding at least once. However, the strategies on different nodes are different due to various network conditions. Among all the nodes, node 6 (N6) has the longest flooding time of 8.3s. We inspect the neighbors of N6 to find the underlying causes. It turns out that N6 has 3 upstream neighbors with good link qualities. Hence, $N_{supp} = 3$. The median link quality of its downstream neighbors is 61%. Based on the decision making algorithm, SurF decides flooding twice for each page.

Fig. 14 depicts the number of packets received during the flooding and negotiation phases. Combining Fig. 13, we see that even though the time spent in flooding is short, it is able to achieve high reliability. 28.2% of the nodes achieve more than 80% of the reliability by flooding. More than 80% of the nodes leverage short-time flooding to finish more than 50% of the dissemination. SurF selectively uses the negotiation to reduce its overhead while retaining 100% reliability.

We further conduct another experiment to study the scalability of SurF and Deluge. The experiment is repeated to present the average completion time. Fig. 15 shows the completion times varied with different code image sizes. We can see

TABLE 1
Comparison of SurF to existing protocols

Protocols	# of nodes	Data size(KB)	Reduction factor
MNP([11], 2005)	100	5.6	1.21
Rateless Deluge([14], 2008)	20	0.7	1.47
ReXOR([15], 2011)	16	4	1.53
ECD([12], 2011)	25	10	1.44
SurF	40	10	1.75

that the completion times of both Deluge and SurF show a linearly increase. SurF achieves a much better performance than Deluge.

Comparison with Existing Protocols. We adopt the reduction factor to compare SurF with other existing protocols [13]. Deluge is the default dissemination protocol in TinyOS, which is widely applied in many real WSN systems. Thus we use Deluge as the baseline of comparison. Table 1 presents the reduction factor achieved by each protocol compared to Deluge. We can see that SurF apparently outperforms other state-of-the-art protocols. Note that MNP has no report of experiment results on testbeds. The items listed in table are obtained from simulations, which do not consider practical conditions, such as collisions. Besides, it is already shown MNP is less efficient than ECD in [12]. Hence, despite that MNP used to be evaluated on a larger network, it is actually less efficient than SurF. Another fact worth noticing is that most existing protocols are compared with Deluge under TinyOS 1.x, which is actually slower than Deluge built on TinyOS 2.x. We use Deluge built on TinyOS 2.x for comparison, which means SurF outperforms the existing protocols more than the reduction factor listed in Table 1.

Energy Consumption. The majority of the energy consumption of the sensor node is caused by radio activity. It is shown in [16] that the energy consumed by idle listing or receiving signals is comparable to that consumed by transmitting signals. In other words, during the same radio-on period, transmitting more packets does not incur higher energy consumption, compared to idle listening. On the other hand, most of the existing dissemination protocols have to keep the radio awake during the dissemination process. Hence, the radio-on time can be measured roughly equal to the completion time. Since SurF has a much shorter completion time, indicating a shorter radio-on time, it is energy efficient even though more packets are transmitted. Moreover, the sleeping techniques can also be adopted in SurF to further reduce the energy consumption. Note that SurF only needs to keep the radio on in the flooding-phase, which accounts for only a small fraction of the total completion time (as shown in Fig. 13).

6 Related work

In bulk data dissemination protocols, the negotiation scheme is widely adopted to guarantee the reliability of bulk data dissemination. The negotiation-based bulk data dissemination protocols can be classified into two categories: the structure-less protocols and the structure-based protocols.

The representative of structure-less protocols is Deluge [5]. It adopts segmentation and pipelining technique for spatial multiplexing, and employs a three-way handshake negotiation scheme to guarantee the reliability of the dissemination. MNP [11] designs a sender selection scheme to pick the nodes that receive the most REQs as the next forwarder. ECD [12] is a recent work which improves the sender selection algorithm in MNP by taking link quality into consideration. It supports dynamic packet sizes to fit different PHY rate radios. SurF is similar to a structure-less protocol with regard to the ability to work well without structure construction involvement.

The other category is structure-based. CORD [17] is a representative of those protocols. CORD builds a connected dominating set (CDS) as the backbone of the network, and employs a two-phase dissemination protocol. In the first phase, the code is disseminated in along the backbone network. Then the backbone nodes forward the code image to other nodes in the second phase. SurF does not require such structures to function, avoiding any overhead to construct dissemination structures. Besides, the structure-less feature of SurF is more suitable for dynamic networks. The authors in [18] study the scaling law for multicast traffic with hierarchical cooperation to leverage long range concurrent transmissions. HRDD [19] is another structure-based method. It optimizes the backbone structure from CDS to a role-based hierarchical structure to further reduce any unnecessary transmissions. However, it does not focus on the inefficiency of the transmission process. SurF focuses on the inefficiency of the transmission process and solves it by combining flooding and negotiation-based methods wisely.

Coding is another technique that can be introduced into bulk data dissemination. Rateless Deluge [14], SYNAPSE++ [20] and ReXOR [15] are all coding-based protocols. In those protocols, the sender encodes the message using certain coding techniques and transmits the encoded packets. After receiving sufficient encoded packets, a receiver recovers the data. The authors in [21] study the impact of link qualities and sleep probability on the network coding gain to improve the coding efficiency. Hence, receivers can send NACK which simply declares the number of missing packets instead of the form of bit vectors with the specific information of the missing packets. However, a limitation of coding-based protocols is the most time consuming task in decoding which may incur a long completion time. Our work is orthogonal to those works and the coding technique can also be used in SurF to transmit the data packets.

Flooding is the representative of non-negotiation methods [7], [10], [22]–[26]. However, it is shown that blind flooding usually takes the risk of a broadcast storm [7]. In [7], the author proposes five schemes to relieve the broadcast storm problem. In [23], the authors present the adaptive forms of the five above-mentioned schemes. In Smart Gossip [10], an adaptive probabilistic flooding protocol is proposed. It automatically adjusts the rebroadcasting probability to adapt to the dynamic underlying network topology. DCB [24] adopts sender selection to avoid too much redundant broadcasting and improves the delivery ratio. Being aware of the information of neighbors in two-hop, DCB can select senders to

give every node two chances to receive the packet. In [25], the parallelization of all possible interference-free relays in broadcasting is maximized to improve the pipeline process. In [26], the authors propose to leverage beamforming to allow multiple transmitters to broadcast the same packet but with minimal transmission power to prolong the network lifetime.

The authors in [27] propose the dissemination algorithms to achieve energy efficiency in duty-cycled WSNs. Optimization algorithms are proposed in [28] to remove the hotspots and prolong the network lifetime while delay and reliability can be guaranteed. The authors in [29] propose the algorithms to disseminate data with minimum communication cost within a given delay bound, in wireless sensor and actor networks with mobile nodes. Different from above methods, the design principle of SurF is to disseminate data as quickly as possible, even with slight increase of communication cost.

SurF makes use of flooding's characteristic of fast propagation, but differs from existing flooding methods by guaranteeing the reliability.

7 Conclusion

In this paper, we present SurF, a novel bulk data dissemination protocol which selectively uses negotiation and opportunistically adopts flooding. We find that neither flooding nor negotiation is efficient when only one of them is used during the whole process. We then design SurF to effectively integrate the schemes for shorter completion times. SurF selectively uses the negotiation scheme, that is, only when necessary instead of throughout the entire dissemination process. Based on an accurate analysis model, SurF predicts the efficiency of two schemes (flooding and negotiation) and adaptively selects the fittest strategy to disseminate data. By reducing the negotiation overhead, SurF can shorten the completion time while still retaining high reliability. Moreover, SurF does not depend on special protocols and it can be incorporated with other flooding-based and negotiation-based methods. In the future, we plan to integrate other protocols into SurF and further study any potential performance improvements.

Acknowledgments

The authors would like to thank the reviewers for reading this paper and giving valuable comments to improve this paper. This work is supported in part by NSFC Major Program under grant No. 61190110, National Basic Research Program (973 program) under grant of 2014CB347800, NSFC under grants No. 61202359, No. 61202402, No. 61373166, No. 61472360, and No. 61170213, and National Science Fund for Excellent Young Scientist No. 61422207.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] X. Mao, X. Miao, Y. He, T. Zhu, J. Wang, W. Dong, X. LI, and Y. Liu, "Citysee: Urban co2 monitoring with sensors," in *Proceedings of IEEE INFOCOM*, 2012.

[3] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A wireless sensor network for structural monitoring," in *Proceedings of ACM SenSys*, 2004.

[4] Wikipedia, "Tinyos," http://en.wikipedia.org/wiki/TinyOS.

[5] J. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proceedings of the ACM SenSys*, 2004.

[6] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks," *Wireless Networks*, vol. 8, no. 2/3, pp. 169–185, 2002.

[7] S. Y. Ni, Y. C. Tseng, Y. S. Chen, and J. P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of ACM MobiCom*, 1999.

[8] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis, "Four-bit wireless link estimation," 2007.

[9] "TinyOS TEP 124: The Link Estimation Exchange Protocol (LEEP)," http://www.tinyos.net/tinyos-2.x/doc/html/tep124.html.

[10] P. Kyasanur, R. Choudhury, and I. Gupta, "Smart gossip: An adaptive gossip-based broadcasting service for sensor networks," in *Proceedings of IEEE MASS*, 2006.

[11] S. Kulkarni and L. Wang, "Mnp: Multihop network reprogramming service for sensor networks," in *Proceedings of IEEE ICDCS*, 2005.

[12] W. Dong, Y. Liu, C. Wang, X. Liu, C. Chen, and J. Bu, "Link quality aware code dissemination in wireless sensor networks," in *Proceedings of IEEE ICNP*, 2011.

[13] M. Doddavenkatappa, M. Chan, and B. Leong, "Splash: Fast data dissemination with constructive interference in wireless sensor networks," in *Proceedings of ACM/USENIX NSDI*, 2013.

[14] A. Hagedorn, D. Starobinski, and A. Trachtenberg, "Rateless deluge: Over-the-air programming of wireless sensor networks using random linear codes," in *Proceedings of ACM/IEEE IPSN*, 2008.

[15] 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.

[16] R. Fonseca, P. Dutta, P. Levis, and I. Stoica, "Quanto: Tracking energy in networked embedded systems," in *Proceedings of USENIX OSDI*, 2008.

[17] L. Huang and S. Setia, "Cord: energy-efficient reliable bulk data dissemination in sensor networks," in *Proceedings of IEEE INFOCOM*, 2008.

[18] X. Wang, L. Fu, and C. Hu, "Multicast performance with hierarchical cooperation," *IEEE/ACM Transactions on Networking*, vol. 20, no. 3, pp. 917–930, 2012.

[19] C. C. Huang, T. T. Huang, J. L. Huang, and L. Y. Yeh, "Hierarchical role-based data dissemination in wireless sensor networks," *The Journal of Supercomputing*, vol. 66, no. 1, pp. 35–56, 2013.

[20] 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.

[21] X. Wang, J. Wang, and Y. Xu, "Data dissemination in wireless sensor networks with network coding," *EURASIP Journal on Wireless Communications and Networking*, vol. 2010, pp. 2:1–2:14, Feb 2010.

[22] Q. Ren, L. Guo, J. Zhu, M. Ren, and J. Zhu, "Distributed aggregation algorithms for mobile sensor networks with group mobility model," *Tsinghua Science and Technology*, vol. 17, no. 5, pp. 512–520, 2012.

[23] Y. C. Tseng, S. Y. Ni, and E. Y. Shih, "Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network," *IEEE Transactions on Computers*, vol. 52, no. 5, pp. 545–557, 2003.

[24] W. Lou and J. Wu, "Toward broadcast reliability in mobile ad hoc networks with double coverage," *IEEE Transactions on Mobile Computing*, vol. 6, no. 2, pp. 148–163, 2007.

[25] Z. Jiang, D. Wu, M. Guo, J. Wu, R. Kline, and X. Wang, "Minimum latency broadcasting with conflict awareness in wireless sensor networks," in *Proceedings of IEEE ICPP*, 2012.

[26] J. Feng, Y. H. Lu, B. Jung, D. Peroulis, and Y. C. Hu, "Energy-efficient data dissemination using beamforming in wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 9, no. 3, p. 31, 2013.

[27] K. Han, L. Xiang, J. Luo, M. Xiao, and L. Huang, "Energy-efficient reliable data dissemination in duty-cycled wireless sensor networks," in *Proceedings of ACM MobiHoc*, 2013, pp. 287–292.

[28] A. Liu, D. Zhang, P. Zhang, G. Cui, and Z. Chen, "On mitigating hotspots to maximize network lifetime in multi-hop wireless sensor network with guaranteed transport delay and reliability," *Peer-to-Peer Networking and Applications*, vol. 7, no. 3, pp. 255–273, 2014.

[29] S. He, X. Li, J. Chen, P. Cheng, Y. Sun, and D. Simplot-Ryl, "Emd: Energy-efficient p2p message dissemination in delay-tolerant wireless sensor and actor networks." *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9-Supplement, pp. 75–84, 2013.

Xiaolong Zheng received the BE degree in the School of Software Technology from Dalian University of Technology in 2011, and the PhD degree in the Department of Computer Science and Engineering from Hong Kong University of Science and Technology in 2015. He is currently a member of Tsinghua National Lab for Information Science and Technology. His research interests include wireless sensor networks and pervasive computing. He is a member of the IEEE.

Jiliang Wang received his BE degree in computer science and technology from University of Science and Technology of China and his PhD degree in computer science and engineering from Hong Kong University of Science and Technology, in 2007 and 2011, respectively. Jiliang Wang is currently an assistant professor in School of Software and TNLIST, Tsinghua University, P.R. China. His research interests include sensor and wireless networks, network measurement and pervasive computing. He is a member of the IEEE.

Wei Dong received his BS and Ph.D. degrees from the College of Computer Science at Zhejiang University in 2005 and 2010, respectively. He was a Postdoc Fellow at the Department of Computer Science and Engineering in Hong Kong University of Science and Technology in 2011. He is currently an associate professor in the College of Computer Science in Zhejiang University. His research interests include Internet of Things and sensor networks, network measurement, wireless and mobile computing. He is a member of IEEE and ACM.

Yuan He is an associate professor in the School of Software and TNLIST of Tsinghua University. He received his BE degree in University of Science and Technology of China, his ME degree in Institute of Software, Chinese Academy of Sciences, and his PhD degree in Hong Kong University of Science and Technology. His research interests include Internet of Things, sensor networks, pervasive computing, and cloud computing. He is a member of the IEEE and ACM.

Yunhao Liu received his BS degree in Automation Department from Tsinghua University in 1995, and an MS and a Ph.D. degree in Computer Science and Engineering at Michigan State University in 2003 and 2004, respectively. Yunhao is now Chang Jiang Chair Professor and Dean of School of Software at Tsinghua University, China. Yunhao is ACM Distinguished Speaker, and now serves as the Chair of ACM China Council and also serving as the Associate Editors-in-Chief for IEEE Transactions on Parallel and Distributed Systems, and Associate Editor for IEEE/ACM Transactions on Networking and ACM Transactions on Sensor Network. He is a Fellow of IEEE.