Exploiting Link Correlation for Core-based Dissemination in Wireless Sensor Networks

Zhiwei Zhao[1], Wei Dong[1], Jiajun Bu[1], Tao Gu[2], Chun Chen[1], Xianghua Xu[3] and Shiliang Pu[4]
[1]College of Computer Science, Zhejiang University, China
[2]School of Computer Science and IT, RMIT University, Australia
[3]School of Computer Science, Hangzhou Dianzi University, China
[4]Hangzhou Hikvision Digital Technology Co., Ltd
[1]{zhaozw, dongw, bjj, chenc}@zju.edu.cn [2]tao.gu@rmit.edu.au [3]xuxhcs@zju.edu.cn [4]pusi@hikvision.com

***Abstract*—Bulk data dissemination is a basic building block for enabling software update and reprogramming in wireless sensor networks. The recent structure based approach looks promising for efficient dissemination since it facilitates transmission and sleep scheduling. However, a number of limitations exist in existing structured protocols. In this paper, we propose a correlated core based solution for efficient bulk data dissemination in wireless sensor networks. We propose an efficient backbone node selection algorithm to construct the core structure by exploiting link correlation. We also design a novel negotiation mechanism which greatly reduces the control message overhead as compared to the existing structured protocols. We conduct both simulation and testbed experiments, and the results show that our proposed solution outperforms the state-of-the-art in terms of both the number of transmissions and the completion time.**

I. Introduction

Bulk data dissemination is used for reliably disseminating a large data object to all nodes in a network in a multihop manner. It is one of the key enabling technologies for software update, network reprogramming, and surveillance video distribution in wireless ad hoc networks [1]. It has attracted many research efforts recently [2] [3] [4] [5]. Due to the size and reliability requirements, a large data object has to be segmented into several pages for a page-by-page, pipelined transmission and a three-way handshake (ADV-REQ-DATA) protocol is typically used to ensure data consistency. Existing work can be basically divided into two categories according to their propagation methods used: structureless and structured protocols. In structureless protocols (including Deluge [2] and MNP [5] and ECD [4]), each node can potentially be a forwarder for data propagation from the sink to all network nodes. In structured protocols (including Sprinkler [6] and CORD [7]), a network core structure is constructed before data dissemination (e.g., CORD [7] employs a Connected Dominating Set (CDS) structure). Data dissemination is first done by propagating the data object to all the core nodes. Each core node then disseminates the object to their neighboring nodes. Structured protocols explicitly select the set of core nodes which are responsible of disseminating the object to the rest nodes. This facilitates the transmission and sleep scheduling for more efficient data propagation. Structured protocols have less broadcast overheads as compared to structureless protocols which are prone to the broadcast storm problem, and hence, offer a good solution for dense and low-power wireless sensor networks. However, a number of limitations exist in the existing structured protocols.

Recent study has shown that link correlation has a large impact on the transmission efficiency of dissemination protocols, i.e., transmissions are more efficient when link correlation is stronger [8]. Existing structured protocols fail to capture link correlation. As a result, many of the selected core nodes may have poor link correlations which seriously affect the number of data transmissions. While link correlation has been used in [9] [10] [11] to achieve more efficient flooding, as far as we know, no existing works have been done to exploit link correlation for structured dissemination in wireless sensor networks.

On the other hand, the three-way handshake mechanism (ADV-REQ-DATA) is originally used in structureless protocols for ensuring full reliability, but it may incur unnecessary message overhead when applied in structured protocols. The ADV messages are designated for discovering neighbors and data pages. It is not necessary in structured dissemination since each node has a fixed parent and child nodes. In addition, in dense sensor networks, severe REQ collisions may occur due to the limitation of the existing REQ back-off timer design.

Aiming to address the limitations of the existing structured protocols, we propose an Correlated Core based solution (CoCo) for efficient bulk data dissemination in wireless sensor networks. First, we exploit link correlation to optimize core node selection in the core structure. Nodes with strong link correlation and better link quality will be more likely to be selected as the core nodes. As a result, the expected number of transmissions (ETX) can be reduced. Second, we optimize the negotiation mechanism by eliminating ADV messages and employing an adaptive timer to reduce REQ collisions with different network densities.

We conduct extensive large-scale simulations in TOSSIM [12], and also evaluate CoCo in our TelosB [13] node testbed. The results show that (i) by introducing link correlation in the core construction, CoCo reduces the number of transmissions by 26.2% and 48.5% compared to CORD

978-1-4799-4657-0/14/$31.00 © 2014 IEEE

and Deluge, respectively; (ii) by applying the optimized negotiation mechanism, CoCo reduces the negotiation message overhead more than 50%, and the REQ timer in CoCo is scalable to dense sensor networks. It is worth noting that CoCo also reserves energy efficiency since it incorporates the coordinated schedules in CORD.

In summary, the paper makes the following contributions.

1) We exploit both link correlation and link quality to design a core construction algorithm for more efficient dissemination.
2) We optimize the three-way handshake mechanism used in structured protocols, aiming to reduce the negotiation message overhead and reduce REQ collisions with different network densities.
3) We implement CoCo in TinyOS, and evaluate its performance in both simulation and testbed experiments. The results show that CoCo outperforms the state-of-the-art in terms of the completion time and number of transmissions.

The rest of the paper is organized as follows: Section II introduces the related work. Section III describes the motivation of this work by two examples. Section IV describes the core node selection algorithm. Section V gives the detailed design of the CoCo protocol. Section VI evaluates CoCo's performance by comparing its performance with both CORD and Deluge. Section VIII concludes our work.

II. Related work

We discuss the related work in this section. Existing bulk data dissemination protocols can be mainly divided into two categories: structureless protocols and structured protocols.

Structured protocols including Sprinkler [6] and CORD [7] typically build a topology structure such as Connected Dominating Set before data dissemination, in which all nodes are divided into two categories: core nodes and non-core nodes. Each non-core node is associated with a core node. Data dissemination is done in two phases. First, the sink transmits the data object to all the core nodes; then each core node disseminates data to all its neighboring core nodes.

Sprinkler requires geography information and tends to establish a minimum connected dominating set (MCDS). The rational is that by minimizing the number of core nodes (forwarding nodes), the number of transmissions can also be minimized. But this may not hold true in the case of unreliable wireless links. Sprinkler uses TDMA [14] for packet level pipelining in the two-phase dissemination, which requires each packet to be separately acknowledged.

CORD follows the sample principle as Sprinkler, but improves Sprinkler in two ways. First, CORD considers link quality when constructing the core structure. It first eliminates the poor quality links, and then selects the node with the most neighboring nodes in a neighborhood as a core node. Second, CORD enables coordinated schedules by employing object segmentation, page-by-page transmission and three-way handshaking. Coordinated schedules divide time into three fixed-size slots: P, C and Q, for transmitting, receiving and sleeping, respectively, In slot P, a node acts as a parent, broadcasting ADV messages to inform downstream nodes of its received pages, and transmits data packets within certain page when REQ messages received. In slot C, a node acts as a child, transmitting REQ messages when receiving ADV messages that contain more pages, and then receives packets from its parent node. In slot Q, a node turns off its radio until the slot ends to save energy consumption. Note that the three slots have an equal length.

We aim to address the limitations of CORD in the following ways. 1) We exploit link correlation to further improve the selection of core nodes by estimating the expected number of transmissions (ETX) of a candidate. The node with less ETX to downstream nodes are more likely to be selected as core nodes. 2) We incorporate a density-aware negotiation mechanism to reduce the delay and control messages.

There is also several structureless protocols [2], [5], [4], [15], [16], [17] These works follow a similar principle for page-by-page transmission and three-way handshaking. But, they rely on passive listening for continuous link measurement and neighbor discovery, which precludes sleep scheduling and introduces more negotiation overhead. Different from these work, we use structure based dissemination, which facilitate sleep scheduling and saves negotiation overhead.

Splash [18] exploits constructive interference [19] to improve the performance of bulk data dissemination. The use of constructive interference, however, requires strict timing during both packet transmission and reception. SYREN [20] exploits the synergy among link correlation and network coding. Similar with Collective Flooding [9], it uses an ACK message to infer other nodes' ACKs with respect to their link correlations. However, SYREN does not yield a very good performance when 100% reliability is required, which is a crucial issue in bulk data dissemination (a node can securely extracts the data object only when it receives all the data packets). In contrast, CoCo efficiently ensures 100% reliability.

III. Motivation

A. CDS construction

Figure 1 shows a simple example where node 1 is the sink, The arrow line indicates a directed link. The percentage on each edge indicates the link quality of the link. We define the correlation between link 1→2 and 1→3 as the probability that when node 1 broadcasts a packet, node 2 loses a packet given that node 3 loses the same packet. The link correlation between link 2→4 and 2→5 is 0, which means that when node 2 transmits a packet, node 5 will not lose (i.e., will receive) the packet given that node 4 loses the packet. The link correlation between link 3→4 and 3→5 is 1, which means that when node 3 transmits a packet, node 5 will lose the packet given that node 4 loses the same packet.

Node 1 is the sink and also the first core node. It prepares to transmit 10 packets to all other nodes. We call the nodes that compete to be the core nodes in a neighborhood as core candidates, for example, nodes 2 and 3 are two core

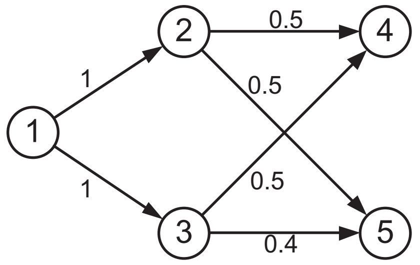


Fig. 1. An example for constructing CDS structure. Node 1 is the first core node, node 2 and node 3 contend to be the next core node. Figures next to the edges indicate the corresponding link qualities (packet reception ratio).

candidates. The CDS construction is done when one of them is selected as a core node,

The key problem in the existing CDS construction is the selection of the core nodes. CORD first eliminates the poor quality link nodes with a threshold, and then the node with most downstream neighboring nodes will be selected as a core node.

In this example, with a different threshold, CORD may either select node 2 as a core node or randomly select a node. If CORD's threshold is set below 0.4 or above 0.5, no link is eliminated. Both nodes 2 and 3 have two downstream neighboring nodes. In this case, CORD is not able to decide which node is better, as a result, it may randomly select a core node between them. If the threshold is between 0.4 and 0.5, link 3→5 is first eliminated. Node 2 has two downstream neighboring nodes (nodes 4 and 5) while node 3 has only one downstream neighboring nodes (node 4). Hence, node 2 is selected as a core node. In this way, the core is constructed with nodes 0 and 2 as core nodes.

Next, we study if the selected core node can facilitate more efficient transmission than other nodes. The number of transmissions of node 2 to cover nodes 4 and 5 can be calculated as $N \times (\frac{1}{q_{24}} + \frac{1}{q_{25}} - \frac{1}{1-(1-q_{24})\times c_{45}^2})$, where N is the number of packets to send, q_{24} and q_{25} is the link quality of links 2→4 and 2→5, respectively, c_{45}^2 is the link correlation between links 2→4 and 2→5 (The detailed derivation can be found in section IV). The number of transmissions is then 30. Considering that node 1 should transmit 10 packets to cover node 2, the total number of transmissions is 30+10=40.

Looking closely, node 3 should be selected as a core node. As the link correlation between links 3→4 and 3→5 is 1, the number of transmissions can be calculated as $10 \times (\frac{1}{0.4} + \frac{1}{0.5} - \frac{1}{1-(1-0.5)\times 1}) = 25$. Considering that node 1's 10 transmissions to cover node 3, the total number of transmissions is 25+10 = 35 $<$ 40.

From this example, we observe that with link quality and link correlation we can improve the CDS construction in CORD to select the more appropriate core nodes.

B. The negotiation mechanism

The three-way handshake mechanism (ADV-REQ-DATA) is originally designed for structure-less protocols. It works as follows: Each node i periodically broadcasts ADV messages for data advertisement and neighbor discovery. When another node j receives the ADV of i, it sends a REQ message to i to request the data packets. In order to avoid REQ collisions, the REQ message is sent in a random interval. Upon receiving the REQ, node i transmits DATA packets to node j.

This mechanism, when applied to structured protocols, may incur extra overhead. First, as there is an underlying structure, a receiver node R is aware of its parent node P and node P also knows how many packets R has received via REQ message, thus ADV messages are unnecessary as it is originally employed for neighbor discovery. In addition, the REQ timer is not designed properly. As mentioned above, before sending the REQ messages, existing designs employ a random timer between a fixed interval ([16,256] ms). There exists a tradeoff between the REQ collision resolution and the transmission delay: When there are more receivers, the REQ timer should be set to a larger value such that it can effectively resolve the REQ contention; While when there are fewer receivers, the REQ timer should be set to a smaller value to reduce the REQ delay. However in CORD, the REQ timer is set between 16-256ms disregarding the node density.

IV. The core node selection metric

When constructing the underlying core structure, the key issue is to define an appropriate metric for evaluating core node candidates. The number of downstream neighboring nodes has been used as a metric in both Sprinkler and CORD. The only difference is that CORD eliminates the poor-link nodes in advance. The rational of using this metric is that a node with more downstream neighboring nodes is more effective. However, as discussed in the example in section III-A, such a metric fails to select the most effective nodes.

To evaluate a core node more accurately, we calculate the effective coverage of k (i.e., considering link loss). Instead of using link quality or correlation to indicate node k's coverage, we directly calculate the ETX of node k to cover its downstream neighboring nodes with one packet. However, ETX is not good enough as if a node has more receivers, its ETX is expected to be larger while it may be more effective than another node with only one receiver. Therefore, to avoid the bias, we define the benefit/cost ratio (number of nodes covered by one transmission) as the metric m_k for node k:

$$m_k = \frac{N_k}{ETX_k} \tag{1}$$

where N_k is the number of node k's downstream nodes, and ETX_k is the expected number of transmissions for node k to cover k's downstream neighboring nodes.

When node k has a large N_k, it has more downstream nodes and m_k is larger. When node k has a small ETX_k, it covers the downstream nodes with less transmissions and

m_k is larger. Therefore, the larger the metric value m_k of node k, the more effective node k is.

Before we present the calculation of m_k, we introduce the following notations.

- S denotes the set of downstream nodes of k: $\forall j \in S$, $h_j = h_k + 1$, where h_j is j's hop count.
- n denotes the number of downstream nodes, i.e., $n = |S|$.
- q_n^k denotes the link quality of link $k \rightarrow n$.
- $P_n^k\{X = j\}$ denotes the probability that k needs to transmit j packets to cover n receivers.
- $P^k_{S_{(n-1)}/n}$ denotes the probability that at least one node in the remaining $n-1$ nodes loses a packet from k, given that the n-th node loses the same packet.

Next, we present the calculation of ETX_k. We first calculate the probability that j transmissions cover all downstream nodes. We then can accumulate the probabilities to get ETX_k.

1) The probability that j transmissions cover all n nodes, $P_n^k(X = j)$.
We first calculate the probability that number of transmissions are more than j times, $P_n(X > j)$, which equals to the probability that j transmissions could not cover all the n nodes.

$$\begin{aligned} P_n^k(X > j) = (1 - q_n^k)^j + P_{n-1}^k(X > j) - \\ ((1 - q_n^k) \times P^k_{S_{(n-1)}/n})^j \end{aligned} \tag{2}$$

where $(1 - q_{kn})^j$ denotes the probability that j transmissions cannot cover the n-th node, $P_{n-1}^k(X > j)$ denotes the probability that j transmissions cannot cover the remaining $n - 1$ nodes, i.e., there is at least one node which cannot be covered by j transmissions, and $((1 - q_{kn}) \times P^k_{n-1/n})^j$ denotes the probability that j transmissions cannot cover the n-th node and at least one node in the remaining $n - 1$ nodes.
We can calculate $P_n^k(X > j)$ recursively, starting from $P_1^k(X > j) = (1 - q_1^k)^j$, as shown below.

$$\begin{aligned} & P_n^k(X > j) = P_n^k(X > j) - P_{n-1}^k(X > j) + \\ & \quad P_{n-1}^k(X > j) - P_{n-2}^k(X > j) + \\ & \quad ... + P_2^k(X > j) - P_1^k(X > j) + \\ & \quad P_1^k(X > j) \\ & = (1 - q_n^k)^j - ((1 - q_n^k) \times P^k_{S_{(n-1)}/n})^j + \\ & \quad (1 - q_{n-1}^k)^j - ((1 - q_{(n-1)}^k) \times P^k_{S_{(n-2)}/n-1})^j + \\ & \quad ... + (1 - q_2^k)^j - ((1 - q_2^k) \times P^k_{1/2})^j + \\ & \quad (1 - q_1^k)^j \\ & = \sum_{m=1}^{n} ((1 - q_m^k)^j - ((1 - q_m^k) \times P^k_{S_{(m-1)}/m})^j) \end{aligned} \tag{3}$$

We note that $P^k_{0/1} = 0$ based on the definition.
Therefore, the probability that the expected number of transmissions is j can be calculated as:

$$P_n^k(X = j) = P_n^k(X > j - 1) - P_n^k(X > j) \tag{4}$$

2) ETX_k. To cover all n nodes, the expected number of transmissions of a single packet can then be calculated as follows.

$$ETX_k = \sum_{j=1}^{+\infty} j P_n^k(X = j) \tag{5}$$

Then, we can get m_k as Eq. (1). m is essentially the benefit/cost ratio. A core candidate with large m value means its transmissions can cover more receivers and should be more likely to be selected as a sender.

We review the example shown in Figure 1, according to the above equation, the expected numbers of transmissions ETX_2 and ETX_3 are 3 and 2.5, respectively. Hence, we get the metrics as follows: $m_2 = 2/3 = 0.67$ and $m_3 = 2/2.5 = 0.8 > m_2$. We then select node 3 as a core node, which has been verified to be a more effective node.

V. The CoCo Protocol

In this section, we present the CoCo protocol that incorporates the proposed core node selection selection.

In high level, CoCo consists of three phases, CDS construction, propagation phase and recovery phase. During the CDS construction, we apply our new metric to the core node selection. During the other two phases, we apply the negotiation mechanism, which will be presented in detail in section V-C1. We also incorporate the coordinated schedules in CORD, where time is divided into three recursive slots (P slot for transmitting, Q slot for sleep and C slot for receiving). The schedules are planned at the same time with the core node selection.

When the network is structured with a core and each node obtains its own schedule, the two-phase dissemination starts. The propagation phase disseminates the data object to all the core nodes. Each core node updates its parameters and status for recovering the non-core nodes, and then enters into the recovery phase. In the recovery phase, each core node transmits the missing packets to its neighboring nodes (i.e., non-core). The core node selection is used in the CDS construction. The optimized negotiation mechanism are employed in both the propagation and recovery phase.

A. Link Measurement

As discussed in section IV, to calculate a node k's metric m_k, we need node k's link quality and correlation of the outbound links to its downstream nodes. We also need the link correlation between a downstream node n and the other downstream nodes $P^k_{(n-1)/n}$. We apply a method proposed in [10]. Each node in the network periodically broadcasts a hello message at an adaptive time interval, which is adjusted based on the link's stability. A hello message contains a node ID and a sequence number. Each node maintains recent reception bit vectors of hello messages for each neighboring node, e.g., the latest 10 hello messages and periodically share the vectors with neighboring nodes. Note that a "0" denotes a lost packet and a "1" denotes a received packet.

With these vectors, a node k can get the inbound link quality from node i as $q_k^i = \frac{\sum_{m=1}^{m=|B_{ik}|} B_{ik}(m)}{|B_{ik}|}$, where B_{ik} is k's bit vector for node i's hello message receptions, $|B_{ik}|$ is the

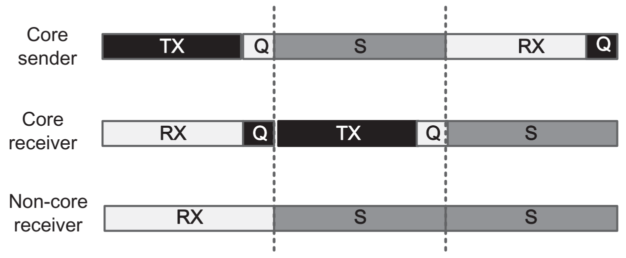


Fig. 2. The coordinated scheduling.

vector length and $B_{ik}(m)$ is the mth bit in B_{ik}. Similarly, the outbound link quality of node k to node i can be calculated with B_{ki}.

For link correlation $P^k_{S_{(n-1)}/n}$, as it denotes the probability that at least one node in the remaining $n-1$ nodes loses a packet from k, given that the n-th node loses the same packet. We then can obtain $P^k_{S_{(n-1)}/n}$ as follows.

$$P^k_{S_{(n-1)}/n} = p_k(S_{n-1}|n) = \frac{\sum_{m=1}^{m=|B_{kn}|} B_{kn}(m) \& B_{kS_{n-1}}(m)}{\sum_{m=1}^{m=|B_{kn}|} B_{kn}(m)} \quad (6)$$

where S denotes the set of links and $B_{kS} = \bigcup_{j \in S} B_{kj}$. With the above information, a node is able to obtain its own metric for core node selection.

B. CDS Construction

In the CDS construction, we construct a CDS structure as well as coordinate link schedules, similar to [7]. We apply the degree-aware single leader algorithm [21] as the basis for CoCo's CDS construction. The sink first announces itself as a core node by broadcasting a CLAIM message. The CLAIM message contains the time offset from the beginning of its first time slot to the time when the message is sent. When receiving the CLAIM message, each node k that is one hop away from the sink calculates its metric according to Eq. (1). Each node selects the sink as its parent node, and obtains its own repeating schedules according to the sink's schedule (when the sink is in P/Q/C slot, the obtained slot is C/P/Q). In this way, node k's schedules can coincide with the sink's schedules.

The node that selects the sink as its parent will then compete to be a core node. They broadcast CANDIDATE messages that contain their metric values. Nodes at next hop that receive one or more CANDIDATE messages respond with a SUB message to the candidate node with the largest metric value. A node that receives the SUB message will be a core node, otherwise, it becomes a non-core node. Each node that has been a core node will then broadcast a CLAIM message to announce itself as a core node. The receivers obtain their own schedules according to the CLAIM message. The process continues recursively until each node is decided to be a core node or a non-core node.

Different from CORD's algorithm, we use the new metric to evaluate a core candidate, and we select the core nodes which use less transmissions.

C. Propagation Phase

When the core is constructed and the coordinated link scheduling is established, the propagation phase starts from the sink node.

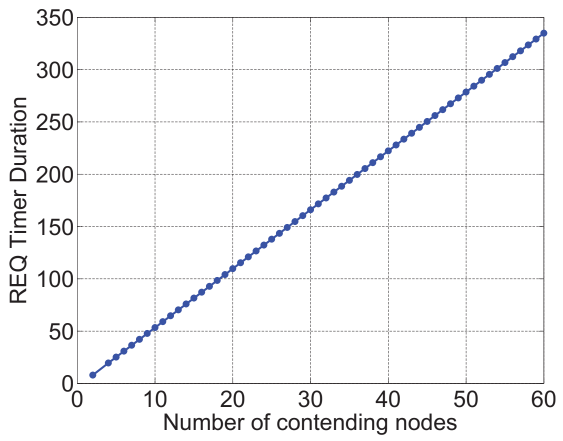
For transmission, each node locally maintains a packetsToSend vector and a packetsToReceive vector with the length of the entire object (e.g., 480 bits for an object of 480 packets). For packetsToSend, "1" denotes a packet to send and "0" denotes a packet that has been acknowledged. Initially, for a non-sink node, all bits in packetsToSend are "0" as it has no packets to send. For packetsToReceive, "1" denotes a packet to receive and "0" denotes a received packet. Initially, all bits in packetsToReceive are "1" as all packets are to be received. We describe the transmission process in the following three slots.

In P slot. A node first decides how many packets should be sent in the current slot. When there are no less than the batch size s_b packets to send, the node prepares the first s_b unsent packets in the current slot, i.e., the first s_b packets marked as "1" in packetsToSend. When there are less than s_b packets to send, the node prepares as many as possible unsent packets in the current slot. **In C slot.** For a newly received data packet, a node marks the corresponding bit in packetsToReceive as "0". Also, the node marks the corresponding bit in packetsToSend as "1". After the data transmissions, *only the downstream core nodes* actively sends a REQ message to its parent. Different from the REQs in Deluge, our REQs contains a batch size bit vector starting at the position of the first missing packet. **In Q slot.** A node turns off its radio for a slot duration.

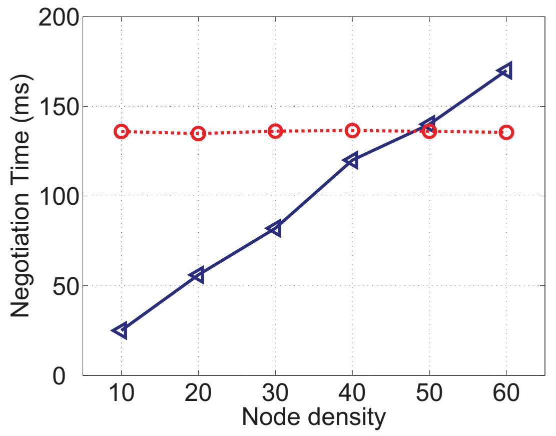
Figure 2 shows an example which consists of three nodes. One core sender and its two next hop nodes (one core node and the other non-core node). The black rectangles denote transmissions and grey rectangles denote receptions. In the P slot of the core sender, the other two slots are in C slot. The sender first broadcast data packets in P slot. After the data transmissions, the core receiver replies a REQ (denoted as the black rectangle with a Q). On the other hand, the non-core node keeps silent. After the P slot of the core sender, it enters Q slot and its next hop core node goes into P slot. The non-core node enters Q slot as it does not forward. After that, the core sender enters C slot to receive data packets from its upstream core node. The other two nodes enter Q slot and turn off their radios.

When a node's child core nodes have received the whole data object, the propagation phase ends and the recovery phase starts. It is worth noting that different core nodes enter into the recovery phase in different time.

1) Optimized Negotiation: As discussed in the previous section, the underlying core structure does not require to discover and select senders for each transmission round. Hence, we eliminate ADV messages in the negotiation and require the sender to directly start transmissions as long as it has enough new received packets.



(a) Negotiation time vs density



(b) Negotiation time vs density

Fig. 3. Density aware negotiation

The REQ timer is originally designed to avoid collisions. However, the timer is constant. When the network is sparse, the timer will incur redundant delay overhead while when the network is very dense, the timer cannot guarantee a low collision probability. We design a density-aware REQ timer for further optimizing the negotiation delay. We model the relationship among number of neighboring nodes, REQ timer duration and collision probability, then a node can decide the REQ timer according to the size of its neighborhood. Suppose there are N contending nodes in a neighborhood, and we denote the REQ timer as T_{REQ}. According to [22], 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 a REQ in the timer period,

$$P_s = NP_a(1 - P_a)^{N-1} \tag{7}$$

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

$$P_n = (1 - P_a)^N \tag{8}$$

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

$$P_c = 1 - P_s - P_n \tag{9}$$

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.

$$N \cdot \frac{2}{1+T_{REQ}} \cdot \left(1 - \frac{2}{1+T_{REQ}}\right)^{N-1} + \left(1 - \frac{2}{1+T_{REQ}}\right)^{N} = 1 - C_{thr} \tag{10}$$

When we set C_{thr} to be a constant, e.g., 0.5%, the relationship between number of contending nodes and REQ timer duration is shown in Figure 3(a). Surprisingly it approximates linear relationship. The default setting in both CORD and Deluge (256ms) can ensure a collision probability below 0.5% for a neighbor size of 49 nodes. Apparently, when there are fewer receivers, a node can set an optimal REQ timer for the receivers to keep the collision probability under C_{thr}. Our approach is to adaptively select the appropriate timer based on the neighbor size, which ensures the collision probability under a certain threshold. To reduce the computation complexity on the sensor node, we store the tuples of REQ timer duration and number of contending nodes on the external flash. An node can directly get the optimal REQ timer according to the number of receivers.

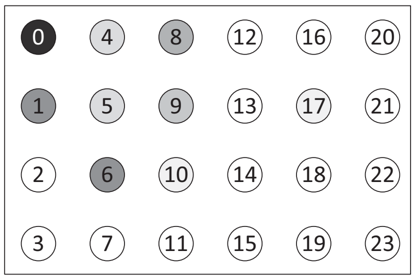
We conduct a TOSSIM simulation to study the impact of our proposed adaptive timer. We repeat the sender selection 1000 times under different network density, and compare the average REQ time in CoCo with that in CORD. Figure 3(b) shows the average REQ time with different network density (i.e., neighbor size). We can see that the average REQ time in CORD is always around 136ms with different network density. The reason is that it randomly selects a duration between [16, 256] with all different densities. In contrast, the average REQ time in CoCo increases with the network density. When the neighbor size is larger than 49 nodes, CoCo's REQ timer is larger than CORD's REQ timer, which means when network density is larger than 49, the REQ timer in CORD can no longer guarantee a collision probability below 0.5%. On the other hand, CoCo can always guarantee the low collision probability.

D. Recovery Phase

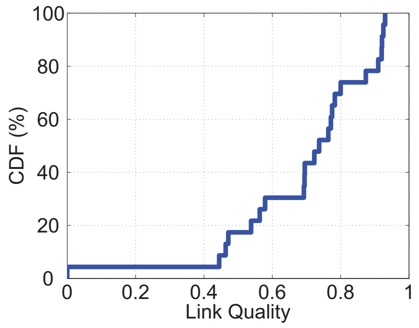
In this phase, a core node recovers the missing packets of all its non-core child nodes. Before the transmissions begin, a core node resets the REQ timer for all non-core receivers according to the number of nodes. The REQ timer information is capsuled in a `Recovery` message, which is to inform non-core nodes of entering into the recovery phase.

When receiving the `Recovery` message, a non-core node sends a long vector indicating its missing packets for the core node to update its `packetsToSend` vector.

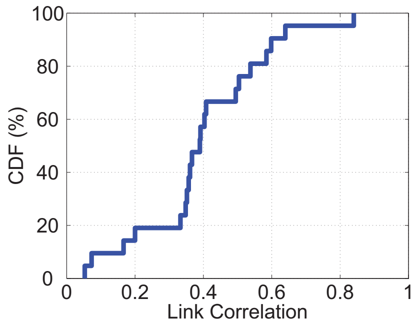
After that, the data transmission begins and the propagation and negotiation mechanisms are the same with those in the propagation phase.



(a)



(b)



(c)

Fig. 4. Wireless link behaviors of the testbed. (a)The outbound link qualities of the sink (node 0) is indicated by the grey levels: A darker color indicates a better link quality. (b) CDF of pair wise link qualities. (c) CDF of pair wise link correlations.

VI. Evaluation

We now move to evaluate CoCo. We conduct both large-scale simulations in TOSSIM [12] with TinyOS 2.1.2 [23] and testbed experiments. In this section, we first report the results from our testbed experiments to evaluate the overall performance of CoCo as compared to both Deluge and CORD, we then report the result of each of the proposed mechanisms using large-scale simulations.

VII. Methodology

A. Methodology

We set the CC2420 power to -32.5 dBm to form a multi-hop network in our testbed (Figure 4(a)). Fig. 4(b) shows the CDF of pair-wise link qualities. With the power setting, link qualities are different with different link pairs. We can see that 17.4% links are good links with PRR $>$ 90%, 13.1% links are intermediate links with PRR in the range of 10%–90%, and 69.5% links are poor links with PRR $<$ 10%. Fig. 4(c) shows the CDF of average outbound link correlation for each node. We can see that there are good link correlations as well as poor link correlations, i.e., with correlation coefficients close to 0 and 1 respectively.

In CoCo, we set the page size and packet payload size as Deluge's default settings in order for a fair comparison. The page size is 48 packets per page and packet payload size is 23 bytes. Each node updates its metric and locally changes its parent if there are upstream nodes with larger metric than its current parent.

We place a sniffer node near the testbed to monitor the performance of each node. At the beginning, the sink node broadcasts a START message in the maximum radio power. Upon receiving the START message, the sniffer node records the start 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. Each experiment is repeated 10 times.

TABLE I
Energy consumption of related operations on TelosB nodes [13].

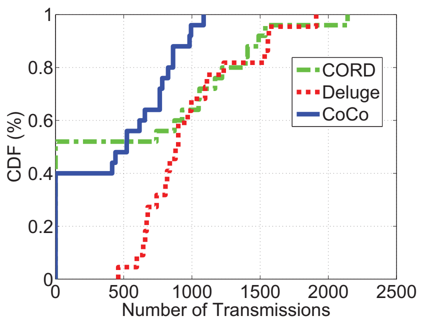
Operation	Energy(nAh)
Receive a Packet	8.000
Transmit a Packet	20.000
Idle-Listen for 1 millisecond	1.250
EEPROM Read data(per byte)	1.111
EEPROM Write/Erase data(per byte)	83.333

We use three key metrics for comparison:

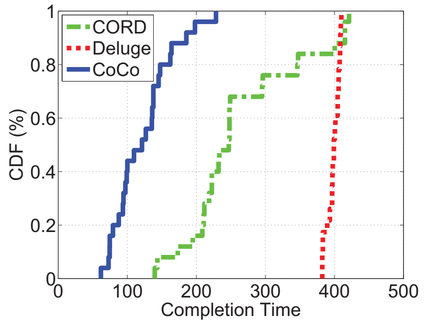
1) Completion time. It is the time from the start of dissemination to the end of dissemination at each individual node. The network completion time is the maximum completion time among all nodes.
2) Number of transmissions. The number of transmissions include data packet transmissions and control packet transmissions.
3) Energy consumption. The energy consumption during the dissemination process of each node. Due to the lack of a mechanism for directly measuring the residual energy level of the battery in TelsoB motes, we follow the method in [7] by logging each operation related with radio, CPU and external flash to calculate the energy consumption. Energy consumption for each operation are listed in Table I.

B. Testbed Experiments

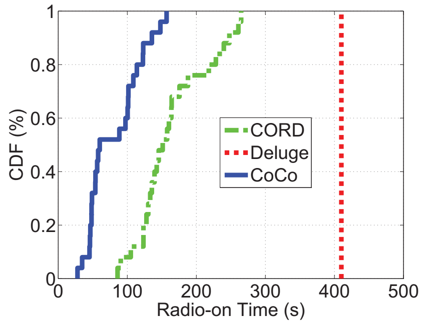
Figure 5(a) shows the comparison result in term of total number of transmissions.The result show that (1) More nodes in CORD have no transmissions, which implies that fewer nodes in CORD have been selected as core nodes. This is probably due to CORD selects the nodes with most number of receivers as core nodes, resulting fewer core nodes and more non-core nodes. (2) CoCo reduces the number of transmissions as compared to both CORD and Deluge. This is because there exists a variation in link correlation, as shown in Fig. 4(c), and the CDS construction in CoCo select the nodes with strong correlations as core nodes, which can cover their downstream nodes with fewer transmissions. In contrast, both CORD and



(a) CDF of total number of transmissions.

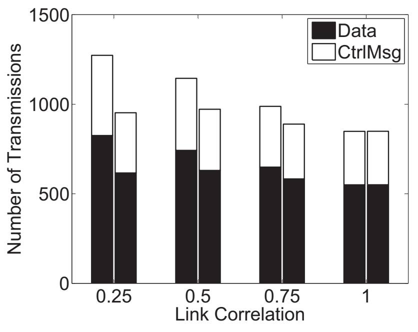


(b) CDF of completion time.

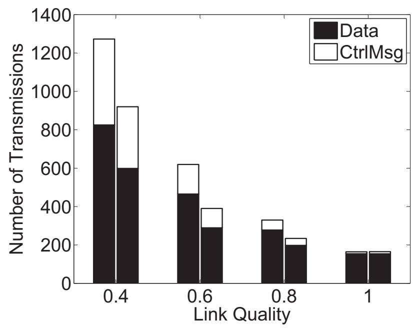


(c) CDF of radio on time.

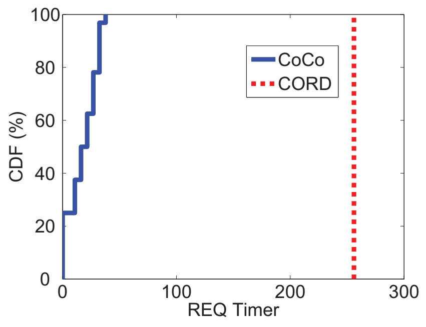
Fig. 5. Overall performance: Testbed results.



(a) Transmissions vs link correlation.



(b) Transmissions vs link quality.



(c) The CDF of REQ timers set during the dissemination process. Both CoCo and CORD have faced no REQ collisions.

Fig. 6. Evaluation of the two building blocks (The core node selection mechanism and the density aware negotiation).

Deluge fail to consider link correlation. CoCo reduces the overall transmission by 26.2% and 48.5% compared to CORD and Deluge, respectively.

Figure 5(b) shows the comparison result in term of the completion time. The result shows that CORD reduces the completion time for most of the nodes. This is because (1) CoCo has the fewest number of transmissions. (2) The negotiation mechanism in CoCo reduces the ADV messages and optimizes the REQ timer. CoCo reduces the overall completion time by 45.3% and 42.6% compared to CORD and Deluge, respectively. Note that the completion time denotes the completion time of the last node.

Figure 5(c) compares the radio on time for CoCo and CORD. The result shows that (1) CoCo reduces the radio on time compared to CORD and Delgue. CoCo uses a similar scheduling mechanism as in CORD. Considering that CoCo's completion time is shorter than CORD, CoCo's radio on time is also reduced. Deluge does not have sleep schedules, and it has the longest radio on time. (2) The reduction of radio on time to CORD is less than the reduction of completion time to CORD. This is an interesting observation, and it may be explained as follows. First, based on the coordinated scheduling, the radio on time is about 1/3 (for non-core nodes) or 2/3 (for core nodes) of the completion time. Hence, for each individual node, the reduction is also 1/3 or 2/3 of the reduction of completion time. Second, CoCo has fewer non-core nodes than CORD. As a non-core node's radio on time reduction (1-1/3=2/3) is larger than that of a core node (1-2/3=1/3), CoCo's overall reduction of radio on time should be less than CORD if they have the same completion time. However, as CoCo greatly reduces the completion time as compared to CORD, the radio on time is also reduced.

Next, we conduct TOSSIM simulations to separately evaluate CoCo's key mechanisms in large scale networks.

C. Impact of core node selection

For fair comparison, we use the same transmission and negotiation mechanisms for both CORD and CoCo. D-CORD represents the dissemination using CORD's core node selection and D-CoCo represents the dissemination using CoCo's core node selection.

Figure 6(a) compares the total number of transmissions under different average link correlations. For each pair of bars, the left one shows D-CORD while the right one shows D-CoCo. From the result, we see that (1) compared to D-CORD, D-CoCo reduces the number of transmissions. This is because CoCo selects the nodes with strong correlated outbound links as core nodes, reducing the number of transmissions. (2)

As the link correlation becomes stronger, the reduction is less. The reason is that when link correlation increases, D-CORD has more chances to select the node with strong correlated outbound links, which reduces the number of transmissions. While D-CoCo keeps selecting the node with the least expected number of transmissions, the reduction is less. When the average link correlation is 1, D-CORD and D-CoCo transmits the same number of packets. This is because when link correlation is 1, the two structures are exactly the same.

Figure 6(b) compares the total number of transmissions under different link quality. The result shows that (1) compared to D-CORD, D-CoCo reduces the number of transmissions. The reason is that CoCo selects the nodes with better outbound links, which is expected to reduce the number of transmissions. (2) When the average link quality becomes better, the reduction of D-CoCo to D-CORD becomes less. The reason is that when link quality becomes better, D-CORD has more chances to select nodes with strong outbound links, thus the reduction is less.

Comparing Figure 6(a) and Figure 6(b), we can see that link quality is also an important factor. We can simply explain the effects of link quality and link correlation as follows. Link quality decides the packet reception ratio, while link correlation decides which packets are expected to be received/missed.

D. Impact of optimized negotiation

Figure 6(c) shows the comparison result in term of of REQ times. We use the same topology for both CoCo and CORD, where the average neighbor size is about 8. We can see that CORD's REQ timer is always 256ms while CoCo's nodes have different REQ timers. As discussed in section V-C1, CoCo's negotiation mechanism is density aware and all REQ timers are smaller than 50ms, which corresponds to the neighbor size of 10 nodes.

VIII. Conclusion

In this paper, we identify several critical limitations of existing backbone protocols and propose a correlated core based efficient bulk data dissemination protocol in WSNs. CoCo has two salient features: (1) it constructs a core structure considering both link quality and link correlation, explicitly selects the nodes with less expected transmissions as core nodes, which reduces the total transmission count and dissemination delay. (2) it also incorporates a novel lightweight and density aware negotiation mechanism for improving scalability and reducing negotiation overhead. Both simulation and testbed experiment results show that, compared to existing work, CoCo greatly improves the dissemination performance in terms of total transmission count and completion time.

Acknowledgement

This work is supported by the National Science Foundation of China Grants No. 61202402 and No. 61370087, the Fundamental Research Funds for the Central Universities, the Research Fund for the Doctoral Program of Higher Education of China (20120101120179), Demonstration of Digital Medical Service and Technology in Destined Region.

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

[3] 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.

[4] W. Dong, Y. Liu, C. Wang, X. Liu, C. Chen, and J. Bu, "Link Quality Aware Code Dissemination in Wireless Sensor Networks," in *Proc. of ICNP*, 2011.

[5] S. S. Kulkarni and L. Wang, "MNP: Multihop Network Reprogramming Service for Sensor Networks," in *Proc. of ICDCS*, 2005.

[6] V. Naik, A. Arora, P. Sinha, and H. Zhang, "Sprinkler: A Reliable and Energy Efficient Data Dissemination Service for Wireless Embedded Devices," in *Proc. of RTSS*, 2005.

[7] L. Huang and S. Setia, "CORD: Energy-efficient Reliable Bulk Data Dissemination in Sensor Networks," in *Proc. of INFOCOM*, 2008.

[8] 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.

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

[10] S. Guo, S. Kim, T. Zhu, Y. Gu, and T. He, "Correlated Flooding in Low-duty-cycle Wireless Sensor Networks," in *Proc. of ICNP*, 2011.

[11] A. Basalamah, S. Kim, S. Guo, T. He, and Y. Tobe, "Link Correlation Aware Opportunistic Routing," in *Proc. of INFOCOM*, 2012.

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

[13] T. Datasheet, "Crossbow inc."

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

[15] P. De, Y. Liu, and S. K. Das, "ReMo: An Energy Efficient Reprogramming protocol for Mobile Sensor Networks," in *Pervasive Computing and Communications, 2008. PerCom 2008. Sixth Annual IEEE International Conference on*. IEEE, 2008, pp. 60–69.

[16] Z. Zhao, Y. Wang, and J. Bu, "Integrated Mutual Selection Based Code Dissemination for Reprogramming Wireless Sensor Networks," *The Journal of China Universities of Posts and Telecommunications*, vol. 20, no. 1, pp. 79–114, 2013.

[17] Y. Gao, J. Bu, W. Dong, C. Chen, L. Rao, and X. Liu, "Exploiting Concurrency for Efficient Dissemination in Wireless sensor networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 24, no. 4, pp. 691–700, 2013.

[18] M. Doddavenkatappa, M. C. Chan, and B. Leong, "Splash: Fast Data Dissemination with Constructive Interference in Wireless Sensor Networks," in *Proc. of NSDI*, 2013.

[19] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with Glossy," in *Proc. of IPSN*, 2011.

[20] S. I. Alam, S. Sultana, Y. C. Hu, and S. Fahmy, "Syren: Synergistic link correlation-aware and network coding-based dissemination in wireless sensor networks," in *Proc. of MASCOTS*, 2013, pp. 485–494.

[21] X. Cheng, M. Ding, D. H. Du, and X. Jia, "Virtual backbone construction in multihop ad hoc wireless networks," *Wireless Communications and Mobile Computing*, vol. 6, no. 2, pp. 183–190, 2006.

[22] 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.

[23] 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.