Ad Hoc Networks 37 (2016) 404–417

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

CoCo+: Exploiting correlated core for energy efficient dissemination in wireless sensor networks

Zhiwei Zhao [a], Jiajun Bu [a], Wei Dong [a,*], Tao Gu [b], Xianghua Xu [c]

[a] College of Computer Science, Zhejiang University, China #38, Zheda Road, Hangzhou, China
[b] School of CS and IT, RMIT University, Australia
[c] School of Computer Science, Hangzhou Dianzi University, China

ARTICLE INFO

Article history:
Received 4 February 2015
Revised 28 August 2015
Accepted 7 September 2015
Available online 24 September 2015

Keywords:
Wireless sensor networks
Energy efficiency
Dissemination
Link correlation
Adaptive mechanism

ABSTRACT

Bulk data dissemination is a basic building block for enabling a variety of applications in wireless sensor networks such as software update, network bug fixing, surveillance video distribution, etc. 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 (called CoCo+). CoCo+ has three salient features. First, CoCo+ is based on an efficient node selection algorithm for constructing the core structure by exploiting link correlation (called Correlated Core). Second, CoCo+ employs a novel consecutive transmission mechanism, which allows out-of-order transmissions and reduces propagation delay. Third, CoCo+ uses a novel lightweight negotiation mechanism that greatly reduces negotiation overhead as compared to the existing structured protocols. We implement CoCo+ with TinyOS/TelosB and conduct both simulation and testbed experiments. Results show that our proposed solution outperforms the state-of-the-art by 52.3 and 49.6% in terms of the number of transmissions and the completion time, respectively.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

A wireless sensor network (WSN) is composed of a large number of small, inexpensive sensor nodes that integrate sensing, computation, and wireless communication capabilities [1]. Bulk data dissemination is used to distribute a large data object reliably to all network nodes in a multi-hop manner. It is one of the key enabling techniques for many WSN applications (e.g., software deployment, reprogramming, surveillance video distribution, etc. [2] [3]) and has attracted much research attention [4–11]. Dissemination has the following requirements: (1) full reliability. Due to that data dissemination is often used for software update, command distribution, etc., each node is required to receive the data object in its entirety. (2) energy efficiency. We regularly face the requirement for software update and maintenance. For example as reported in [12], the software version increases from 158 to 285 during Dec. 2010–April 2011. Therefore, the energy efficiency of dissemination has a large impact on the network lifetime. (3) low latency. Since dissemination is often used for software update, during which the network would be temporarily down, the dissemination should be done as soon as possible. Due to the above requirements, most existing dissemination protocols segment a large data object into several pages for a page-by-page, pipelined transmission. A three-way handshake (ADV-REQ-DATA) protocol is typically used to ensure data consistency.

According to the propagation manner, existing work can be basically divided into two categories: structureless and

* Corresponding author. Tel.: +86 057187953955.
E-mail addresses: zhaozw@zju.edu.cn (Z. Zhao), bjj@zju.edu.cn (J. Bu), dongw@zju.edu.cn (W. Dong), tao.gu@rmit.edu.au (T. Gu), xhxu@hdu.edu.cn (X. Xu).

http://dx.doi.org/10.1016/j.adhoc.2015.09.003
1570-8705/© 2015 Elsevier B.V. All rights reserved.

structured protocols. Structureless protocols [5,7] use simple flooding for data propagation from the sink to all network nodes. Differently, structured protocols [9,10] establish a network core structure before data dissemination. Such structure can be, for example, Connected Dominating Set (CDS) employed in CORD [10]. 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 for forwarding the data object to the rest of the network, and data propagation can be done in a more efficient way by appropriate transmission and sleep scheduling. They introduce less broadcast overhead as compared to structureless protocols that are prone to the broadcast storm problem. Hence, it offers a good solution for dense and low-power wireless sensor networks.

In nutshell, core structure construction, data transmission and negotiation are three fundamental building blocks of structured dissemination. However, a number of limitations exist in all the three building blocks in the existing structured protocols. First, the core constructions fail to consider link correlation. Link correlation is often identified as the correlation of packet receptions on different links [13]. For example, if two links are strongly correlated, we can infer that when one link's receiver receives/loses a packet, the other link's receiver is much likely to receive/lose that packet. Link correlation has shown a great impact on the efficiency of broadcast, i.e., transmissions are more efficient when link correlation is stronger [13–15]. Existing structured protocols ignore link correlation when selecting the core nodes. As a result, the selected core nodes may have poor link correlation, which can seriously affect the transmission efficiency. While link correlation has been effective to achieving more efficient flooding [14] [15], no existing work has been done to exploit link correlation for structured dissemination in wireless sensor networks.

Second, the existing page-by-page reliable transmission mechanism is not efficient. In CORD, the state-of-art work, a sender has to deliver an entire page to its receivers before transmitting the next page. As wireless links are usually unreliable, a page may need multiple transmission rounds to be fully delivered. For example, a sender tends to transmit 10 pages (each page consists of 10 packets) to a receiver with link quality of 90%. The first round transmits 10 packets among which one packet is lost. In the second round, it only re-transmits the missing packet. As a result, it takes two rounds to transmit a page. In total, there will be 20 transmission rounds, with 10 rounds transmitting 10 packets each and another 10 rounds retransmitting only one packet each. The problem lies in retransmission which is less efficient. Obviously, it needs more transmission rounds which (1) increase the propagation delay, (2) and also incur more negotiation overheads. Ideally, if we can combine the re-transmissions for the missing 10 packets into one transmission round, it would be much efficient.

Third, the three-way handshake mechanism (ADV-REQ-DATA) may incur considerable message overhead in structured protocols. (i) the ADV messages are originally designed for discovering neighbors and data packets when there are no underlying structures. It becomes necessary as in structured dissemination, each node has a fixed parent and child nodes and always receives data packets from its parent. (ii) when a node receives ADVs and identifies useful data, it will prepare an REQ message. Before the REQ transmission, a back-off timer is used to avoid possible collisions. However, existing REQ back-off timer is set to be constant, e.g., 256 ms for CORD. It may incur large delay overhead in sparse networks and severe REQ collision in dense networks.

To address the above limitations, in this paper, we propose CoCo$^+$, an energy efficient bulk data dissemination protocol built on CoCo [16], that efficiently cope with the above limitations using three separate improving approaches. First, inspired by recent works on link correlation [15,17], we formally model the relationship between expected number of transmissions (ETX) and link correlation when establishing the core structure. Nodes with strong link correlation and better link quality are more likely to be selected. As a result, the transmission overhead can be reduced. Second, we propose a novel transmission mechanism to reduce both propagation delay and redundant transmissions. For each transmission round, we allow out of order transmissions. Instead of sending only the missing packets from last page, we top up a full page-size batch with more packets from the next pages for transmission. In this way, we make use of each transmission round more efficient. Third, we optimize the three-way handshake by eliminating ADVs and employ an adaptive back-off timer to avoid REQ collisions as well as reduce delay under different network densities. It is worth noting that CoCo$^+$ reserves the coordinated transmission/sleep scheduling in CORD [10]. We implement CoCo$^+$ in TinyOS and evaluate its performance by extensive experiments and simulation. The results show that CoCo$^+$ outperforms the state-of-the-art work by 52.3 and 49.6% in terms of the number of transmissions and the completion time, respectively.

In summary, the paper makes the following contributions.

1. We design a core construction algorithm for more efficient transmission, which exploits link correlation to accurately estimate a potential sender's expected number of transmissions (ETX), and select the nodes with small ETX as core nodes.
2. We propose a novel consecutive transmission mechanism, which can transmit packets in an out of order fashion, and effectively reduces the propagation delay.
3. We optimize the three-way handshake mechanism in structured protocols, reducing the negotiation message overhead and REQ collisions with different network densities.

The latter two contributions are the main differences of CoCo$^+$ from the conference version CoCo [16]. The rest of the paper is organized as follows: Section 2 introduces the related work. Section 3 describes the motivation of this work by two examples. Section 4 describes the key improving mechanisms in CoCo$^+$. Section 5 gives the detailed design of CoCo$^+$. Section 6 reports the evaluations by comparing its performance with both CORD and Deluge. Section 8 concludes our work.

2. 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 [9] and CORD [10] 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 rationale is that by minimizing the number of core nodes (forwarding nodes), the number of transmissions can also be minimized. However, this may not hold true in the case of unreliable wireless links. Sprinkler uses TDMA [18] for packet level pipelining in the two-phase dissemination, which requires each packet to be separately acknowledged.

CORD follows the same 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. It is worth noting that the coordinated sleep scheduling is not the key contribution of this paper, and could be possibly replaced by better scheduling algorithms such as [19].

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 is more likely to be selected as core nodes. (2) We design a novel transmission mechanism, which allow us to utilize transmission slots more efficiently, i.e., transmit more packets in a transmission round compared to the existing page-by-page mechanism. (3) We incorporate a density-aware negotiation mechanism to reduce the delay and control messages. The latter two features are the main differences between CoCo$^+$ and CoCo.

There are also several structureless protocols [5,7,11]. These works follow a similar principle for page-by-page transmission and three-way handshaking. However, 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.

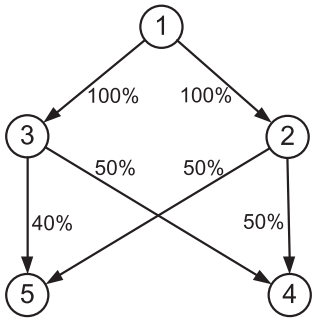


Fig. 1. An example for constructing CDS structure.

Splash [20] exploits constructive interference [21] to improve the performance of bulk data dissemination. The use of constructive interference, however, requires strict timing during both packet transmission and reception. SYREN [22] exploits the synergy among link correlation and network coding. Similar with Collective Flooding [14], it uses an ACK message to infer other nodes' ACKs with respect to their link correlations. However, SYREN does not guarantee 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.

GARUDA [23] is an inspiring work that establishes out-of-sequence transmissions and achieves impressively efficient results for reliable downstream point-to-multipoint data delivery. Our work differs from GARUDA mainly in the following aspects. First, instead of single or a few packets, we aim to disseminate a large data object with the page-level pipelining. Second, we employ an underlying structure that exploits link quality and link correlation. Nodes with smaller ETX values are selected as core nodes. Third, we incorporate coordinated scheduling, which greatly reduces the radio-on time of network nodes.

3. Motivation

In this section, we demonstrate and analyze the limitations of existing works with two detailed assisting examples.

3.1. Core structure construction

Fig. 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 $1 \rightarrow 2$ and $1 \rightarrow 3$ 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 $2 \rightarrow 4$ and $2 \rightarrow 5$ 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 $3 \rightarrow 4$ and $3 \rightarrow 5$ 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 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 $\rightarrow$ 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 $2 \rightarrow 4$ and $2 \rightarrow 5$, respectively, c_{45}^2 is the link correlation between links $2 \rightarrow 4$ and $2 \rightarrow 5$ (The detailed calculation can be found in Section 4.1). 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 $3 \rightarrow 4$ and $3 \rightarrow 5$ 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 obverse that with link quality and link correlation we can improve the CDS construction in CORD to select the more appropriate core nodes.

3.2. Data transmission

Fig. 2 shows an example of the propagation process in CORD. The sender S prepares to deliver a two-page data object to receiver R, with four packets in a page. Sender S needs multiple transmission rounds to fully deliver the two pages. The blocks right beside S and R denote the packet transmissions and receptions for each transmission round. The colors of the blocks denote the status of each packet: a dark block at sender S (or receiver R) indicates a packet being transmitted (or received); a grey block at S (or R) indicates a packet has been acknowledged (or received); a white block at S (or R) indicates a packet to be sent (or received). For example, when transmitting page 1 in the second round, node S has two packets acknowledged and is transmitting the last two packets in page 1. Node R has two packets received and

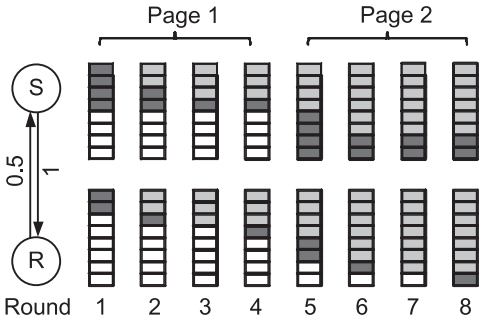


Fig. 2. The page-by-page transmission mechanism.

receives only one packet in this round due to link quality (i.e., 50%).

Sender S first transmits page 1 (four packets) to R, and R receives only two packets. S then broadcasts ADV messages and R replies an REQ for the missing packets in page 1. Upon receiving R's REQ, S transmits the two missing packets and R receives one of them. Then, S re-transmits the last missing packet twice to one reaches R (as the link quality is 50%). The same process repeats for the transmission of page 2.

In this example, the total transmission delay is $16 \times t_{pkt} + 8 \times t_{ng}$, where t_{pkt} is the time for transmitting a single packet and t_{ng} is the inter-page negotiation time. The three-way handshake mechanism uses an ADV timer to control the ADV message broadcast rate [24] and uses a random REQ timer to avoid wireless collisions. Using the default settings in CORD, the ADV timer is initially set to 2 ms, and increased exponentially when no REQ message received, and the REQ timer is set to a random value between [16, 256]. The expected t_{ng} is $2 + 4 + \frac{16+256}{2} = 142$ ms. The time for transmitting a data packet t_{pkt} is about 8 ms on the cc2420 radio. The negotiation time is about $142 \times 8 = 1136$ ms while the data transmission time is about 8×16=128 ms. While the transmission delay of data packets is inevitable, the inter-round negotiation delay is unnecessarily large and should be minimized.

We analyze the reason for the long negotiation delay which are two-fold. First, the page-by-page transmission mechanism is used to reduce the overhead for ensuring reliability, but it incurs redundant transmission rounds. As a receiver request missing packets via REQ messages, the REQ messages should carry the information of which packets are missing. In order to avoid that REQ messages carrying a very long bit-vector to indicate the whole data object, CORD employs the page-by-page transmission mechanism such that it is enough for each REQ message to carry only the bit-vector for the receiving page. However, such transmission mechanism significantly degrades transmission efficiency. We can see from the above example that in rounds 2–4, S only transmits 2, 1 and 1 packets, respectively while S has many more packets to send (the packets in page 2). Intuitively, if we can transmit more packets, transmission efficiency can be improved.

3.3. Negotiation

In the example of Fig. 2, we can see that the delay between ADV and REQ is considerably large. In a structured network,

R is aware of its neighboring node S and S also knows how many packets R has received via REQ messages, thus ADV messages are unnecessary as it is used for neighbor discovery. In addition, the REQ timer is not designed properly. The REQ timer is set for resolving multi-request collisions. We can infer that when there are more receivers, the REQ timer should be set to a larger value while when there are fewer receivers, the REQ timer should be set to a smaller value. However in CORD, when there is only one contending receiver, the REQ timer is still 16–256 ms which will result in large delay overhead. Moreover, in a dense network, the REQ timer may be not long enough to resolve REQ collisions.

4. Key mechanisms in CoCo$^+$

In this section, we propose several solutions to address the above limitations. (1) We design an efficient novel core node selection metric which considers link correlation and quality to reduce the number of transmissions during the CDS construction. (2) We design a novel transmission mechanism which always ensures a full page transmission to reduce the propagation delay, (3) We propose an optimized negotiation mechanism for structured protocols, aiming to minimize the negotiation overhead. In the rest of this section, we will discuss our proposed mechanisms in detail.

4.1. Core node selection

In core node selection, the key issue is to define an appropriate metric for evaluating core node candidates [25]. 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 rationale of using this metric is that a node with more downstream neighboring nodes is more effective. However, as discussed in the example in Section 3A, 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 fewer 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.

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 the number of transmissions is larger than j, $P_n(X > j)$, which equals to the probability that j transmissions could not cover all the n nodes.

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

   where $(1 - q_{kn})^j$ denotes the probability that j transmissions cannot cover the nth 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 $((1 - q_{kn}) \times P_{n-1/n}^k)^j$ denotes the probability that j transmissions cannot cover the nth 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.

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

   We note that $P_{0/1}^k = 0$ 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.

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

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 Fig. 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.

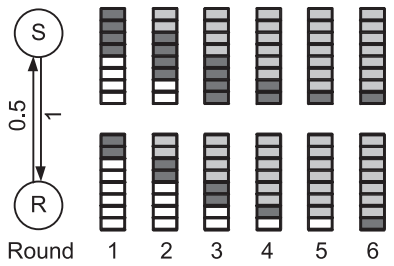


Fig. 3. An illustration of the proposed transmission mechanism.

4.2. Transmission mechanism

The proposed transmission mechanism does not segment a data object into multiple pages. To reserve the multihop pipelining, we also transmit a page-sized batch of packets in a transmission round. The difference with the page-by-page transmission is that we always transmit a page-sized batch of packets as long as there are enough unacknowledged packets, and these packets may be from different pages. When there are not enough packets for a batch, the sender transmits as many packets as possible.

We revisit the example in Section 3.2, and Fig. 3 shows our proposed transmission process. Node S transmits a full page size of packets when it has enough packets to send. When there are not enough packets for a page size, node S transmits as many packets as possible. Node R receives 4 packets in only two rounds, while in the example shown in Fig. 2, node R receives 4 packets in four rounds. In total, node R receives all 8 packets in six rounds, reducing R negotiation rounds ($142 \times 2 = 284$ ms) when comparing to CORD.

It is worth noting that when the data object is larger, we save more transmission delays. When there are n pages (4 packets in a page), CORD needs $4n$ transmission rounds to finish the dissemination as each page requires four rounds (as depicted in Fig. 2). When using our transmission mechanism, for the first $4 \times (n-1)$ packets, each 4 (page sized) packets requires two rounds to be fully delivered. Hence, there are $2 \times (n-1)$ rounds for the first $4 \times (n-1)$ packets. For the last 4 packets, 4 rounds are needed as depicted in Fig. 3. In total, only $2 \times (n-1) + 4 = 2n + 2$ rounds are needed. Our mechanism saves $\frac{2n-2}{4n}$ rounds as compared to CORD. When n is very large, we save about 50% negotiation rounds.

4.2.1. REQ message design

One of the key issues in designing our transmission mechanism is how to ensure 100% reliability without incurring extra overhead. In CORD, as each transmission is restricted within one page, an REQ message carries a page number and the bit-vector indicating the missing packets within that page. The overall reliability is achieved by per-page reliability. If we do not follow the page-by-page transmission, intuitively, the REQ should carry a very long bit vector to indicate the missing packets of the entire object. As a data object is typically large in size (it may contain hundreds or thousands of packets, the long bit vector transmission is unacceptable). We solve this problem simply as follows: each receiver aggressively sends REQ messages to the sender after each transmission round. An REQ message contains the position of the first missing packet and a batch sized bit-vector starting from that position. Upon receiving multiple REQ messages, the sender combines the bit-vectors and then sends the first batch-size unreceived packets.

4.2.2. Deal with REQ lost

When an REQ is lost, the sender is unaware of whether the transmissions are successfully received by the receiver. In such case, the sender continues to transmit the packets after the last transmitted page. For example, after page 1 is transmitted and no REQ messages are received. The sender continues to transmit data packets of page 2. If there are no more pages for transmission, the sender retransmits the unacknowledged data pages. To guarantee the reliability, the sender stops transmission until all packets have been acknowledged.

4.3. 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.

Intuitively, the REQ collision probability will increase (decrease) when the network density increases (decreases). To obtain the optimal REQ timer for given network density and collision probability requirement, we model the relationship among number of neighboring nodes, REQ timer duration and collision probability, and then calculate the optimal 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 [26], the REQ attempt probability is $P_a = \frac{2}{T_{REQ}+1}$. The probability of a successful transmission of N contending nodes in a timer duration is the probability that only one node attempts to send an REQ in the timer period,

$$P_s = NP_a(1-P_a)^{N-1} \tag{6}$$

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

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

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

$$P_c = 1 - P_s - P_n. \tag{8}$$

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

$$N \cdot \frac{2}{1+T_{REQ}} \cdot \left(1 - \frac{2}{1+T_{REQ}}\right)^{N-1} + \left(1 - \frac{2}{1+T_{REQ}}\right)^{N} = 1 - C_{thr} \tag{9}$$

When we set C_{thr} to be a constant, e.g., 0.5%. The relationship between the number of contending nodes and REQ timer duration is shown in Fig. 4. We can see that it approximates linear relationship. The default setting in both CORD and Deluge (256 ms) can ensure a collision probability below 0.5%

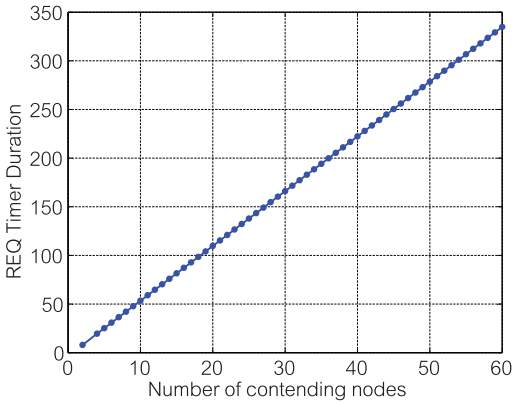


Fig. 4. The relationship between network density and the REQ timer duration while ensuring a collision probability smaller than 5%.

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. A node can directly get the optimal REQ timer according to the number of receivers.

5. The CoCo$^+$ protocol

By incorporating the above mechanisms, we present CoCo$^+$, a Correlated Core and rapid propagation mechanism based efficient bulk data dissemination protocol.

As shown in Fig. 5, CoCo$^+$ consists of three steps – CDS construction, propagation phase and recovery phase. During the CDS construction, we apply our core node selection metric to the core node selection. During the latter two phases, the novel transmission mechanism and optimized negotiation are employed. 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 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 the recovery phase. In the recovery phase, each core node transmits the missing packets to its neighboring nodes (i.e., non-core nodes).

5.1. CDS construction

In the CDS construction, we construct a CDS structure as well as coordinate link schedules, similar to [10]. We apply the degree-aware single leader algorithm [27] 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 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 fewer transmissions.

5.2. 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.

Fig. 6 shows the flowchart of the proposed transmission mechanism. The node switches among P, Q and C slots until the dissemination finishes. **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 an 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.

When a node's child core nodes have received the whole data object, the propagation phase ends and the recovery

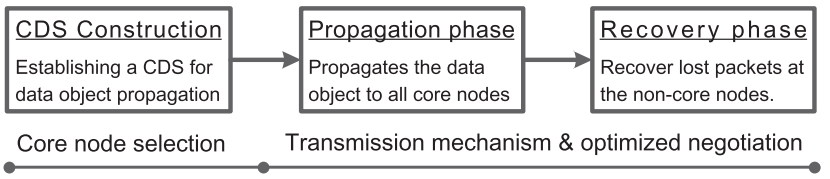


Fig. 5. Framework of CoCo+.

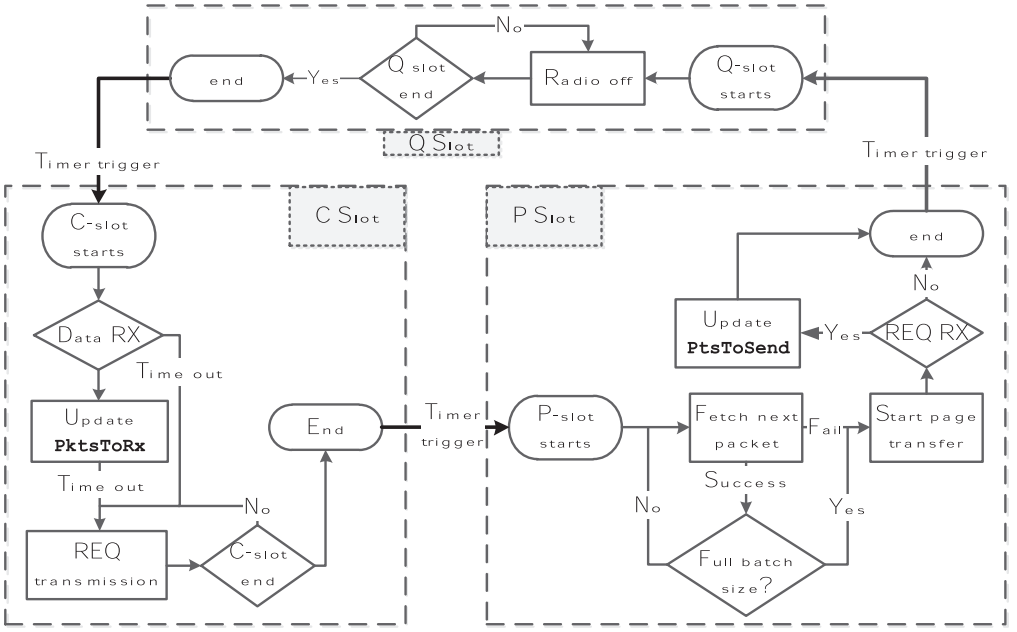


Fig. 6. An example for constructing CDS structure.

phase starts. It is worth noting that different core nodes enter into the recovery phase in different time.

5.3. 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.

6. Evaluation

We now move to evaluate CoCo+. We conduct both large-scale simulations in TOSSIM[28] and testbed experiments with TinyOS 2.1.2 [29]. In this section, we first report the testbed experimental results to evaluate the overall performance of CoCo+ as compared to both Deluge and CORD. Then we discuss the study of each proposed mechanism using large-scale simulations.

6.1. Methodology

Requirements. We first formally give the desired requirements of dissemination protocols as follows.

1. Full reliability. The data object should be received by each node in its entirety. The reason is that data dissemination is often used for software update, command distribution, etc.
2. Energy efficiency. (i) we regularly face the requirement of software upgrade for software upgrade and network maintenance. For example in a typical large scale sensor network system - CitySee, as reported in [1,2] the software version increases from version 158 to version 285 during the Dec. 2010–April 2011. The software update is performed in the frequency of near one time per day. (ii) though the dissemination frequency is lower than the collection frequency, the data size is much larger than the collected sensing data. For example, the data size of a typical update patch is about 10−40KB [3,4]. While the data collection for one entire day is about 14 KB with typical collection settings (the packet payload is 100 bytes, the collection period is 10 min, and we account the data collection for 24 h). Based on the above two observations, the energy efficiency of data dissemination, which is the basic building block of software update, is of great importance for the network energy efficiency.

Table 1
Notations.

S	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	The number of downstream nodes, i.e., $n = \|S\|$
q_n^k	The link quality of link $k \to n$
$P_n^k\{X = j\}$	The probability that k needs to transmit j packets to cover n receivers
$P^k_{S_{(n-1)}/n}$	The probability that at least one node in the remaining $n-1$ nodes loses a packet from k, given that the nth node loses the same packet

Table 2
Energy required by some operations employed in dissemination.

Operation	nAh
Receive a packet	8.000
Transmit a packet	20.000
Idle-listen for 1 ms	1.250
EEPROM read data(per byte)	1.111
EEPROM write/erase data(per byte)	83.333

3. Latency. Since dissemination is often used for distributing controlling commands or software updates, during which the network would be temporarily down, the completion time (latency) of dissemination should be minimized.

Evaluation metrics. For the end-to-end performance of data dissemination, we follow the three commonly used metrics for evaluation [5,10]:

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. This is the key evaluating metric, since during the bulk data dissemination for reprogramming or network reconfiguration, the network will be temporarily down. The completion time almost equals to the network down time.
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 [10] by logging the number of packet transmissions and receptions, radio on/off operations, and EEPROM reads and writes in the motes external EEPROM. After the experiment, each log was post-processed to compute the total energy expenditure of the node according to Table 2.

Implementation. We implement CoCo+ on real sensor node platform TelosB. TelosB nodes use the MSP430 unit and the CC2420 radio. The RAM size is only 10 KB and the external flash size is 1 MB.

Due to the limited RAM size, we need to reduce the calculation complexity of CoCo+. More specifically, the complexity of ETX estimation and the REQ timer calculation should be minimized. To this end, we calculate the ETX and optimal REQ timers for different pairs of link quality and link correlation and store the results in the external flash. When the link quality and correlation are measured, the node directly retrieves the corresponding ETX result using the binary search algorithm. The complexity is reduced to $O(\log(n))$.

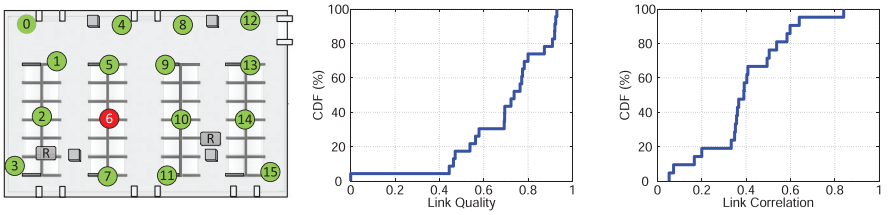
Testbed settings. The testbed consists of 16 TelosB nodes with the average node spacing of 0.6 m (as shown in Fig. 7(a)). All nodes are equipped with 802.15.4 radios (CC2420). Node 6 (placed in the middle) is the sink node. We use the power level of 3 and the 16 nodes form a 4-hop network. There is two WiFi APs near the testbed. To minimize the impact of WiFi interference, we choose channel 26 for dissemination, which does not overlap with WiFi frequencies. We will provide more details about the testbed in Section 6.2.

6.2. Testbed experiments

We build an indoor wireless sensor network testbed as shown in Fig. 7(a). The red node (Node 6) is the sink. Fig. 7(c) shows the cumulative distribution function (CDF) of link quality and Fig. 7(b) shows the CDF of link correlation. We can see that about 17% links are good links with packet reception ratio (PRR) > 90%, 14% links are intermediate links with PRR in the range of 10–90%, and 69% links are poor links with PRR < 10%. We use the conditional packet reception probability (CPRP) as the link correlation metric. About 5% links are strongly correlated (CPRP > 80%), about 60% links are weakly correlated (CPRP < 40%) and about 35% links are intermediately correlated (40% < CPRP < 80%). We can see that both the link qualities and link correlations are highly diverse, which allows for the opportunities for sender selection optimization.

The sink node starts dissemination. We place a sniffer node near the testbed for listening reports from each node. In 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 time of dissemination. Each node broadcasts a REPORT message once it has received the whole data object, also in the maximum radio power. When REPORT messages from all nodes are collected at the sniffer, we can get the completion time and number of transmissions for each node. A sniffer node is required because the network nodes are not globally synchronized, and using a sniffer node can keep the consistent time scale.

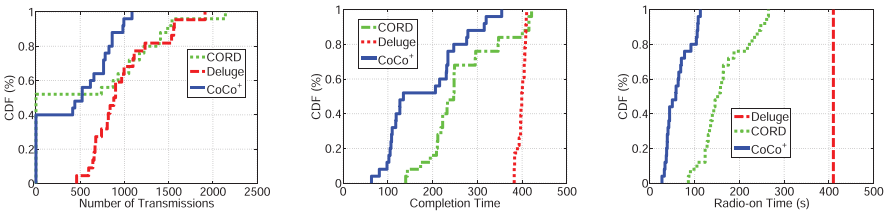
Fig. 7 (d) shows the comparison result in term of total number of transmissions. The result shows that (1) more nodes in CORD have no transmissions, which implies that fewer nodes in CORD have transmissions in the CDS. This is probably due to CORD selects the nodes with the most number of receivers as 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. 7(a), and the CDS construction in CoCo$^+$ select the nodes with strong correlations as core nodes, which can cover their downstream nodes with fewer transmissions.



(a) The topology of our indoor testbed.

(b) CDF of link quality.

(c) CDF of link correlation.



(d) CDF of transmissions number.

(e) CDF of completion time.

(f) CDF of radio-on time.

Fig. 7. Testbed results.

In contrast, both CORD and Deluge fail to consider link correlation. CoCo$^+$ reduces the overall transmission by 26.2 and 48.5% compared to CORD and Deluge, respectively.

Fig. 7 (e) 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) CoCo's transmission mechanism augments the propagation speed. Even when there is the same number of transmissions, CoCo$^+$ is expected to have a shorter completion time. (3) 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.

As the radio on time contributes most energy consumption in sensor nodes [30], we use radio-on time as an indication of energy consumption (like the methodology in [7,10]). Fig. 7(f) compares the energy consumption of CoCo$^+$, CoCo, CORD and Deluge (in terms of radio-on time). The result shows that (1) both CoCo$^+$ and CoCo reduce 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) CoCo$^+$ further reduces the radio on time as compared to CoCo. The reason is that CoCo$^+$ incorporates the novel transmission mechanism, which transmits more packets in each transmission round and thus results in less rounds and radio-on time. (3) 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. It is worth noting that, although CoCo$^+$ outperforms CoCo in terms of completion time and radio-on time, it does not reduce more transmissions compared with CoCo. The reason is that the reduction of transmissions mainly comes from the core node selection mechanism. While CoCo$^+$ and CoCo employ the same core node selection for core construction.

We assign each node an energy level and repeat the disseminations until 30% nodes are out of energy. Since the simulation does not represent the real-world case, we use the lifetime using Deluge as a baseline performance.

Fig. 8 compares the lifetime achieved by Deluge, CORD and CoCo$^+$. We can see that (1) CoCo$^+$ achieves the longest lifetime. The reason is that it has the least radio-on time for each dissemination round (Fig. 7(f)). (2) when the network scale increases, the improvement compared with Deluge decreases. The reason is that in CoCo$^+$, the leaf nodes are the most energy efficient since 2/3 time is in radio-off state. When network scale increases, the portion of leaf nodes decreases, thus the improvement to Deluge decreases.

6.3. System insights of the three key mechanisms

In this subsection, we evaluate each of the three key mechanisms we proposed in simulation. We use TOSSIM to generate a 100 nodes network with random topology. Both link quality and link correlation are randomly set and distributed. Note that we do not explicitly vary link quality and

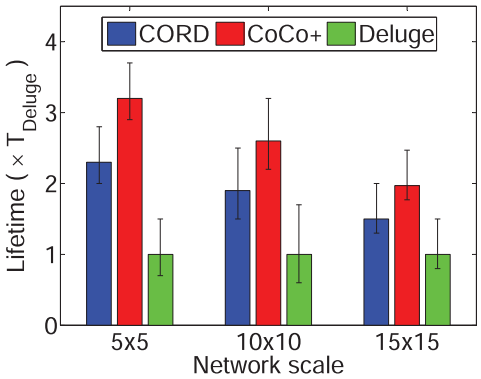


Fig. 8. Lifetime comparison.

correlation. Instead, we collect all cases with different link qualities and link correlations. Then we can obtain the average performance for each different link quality and correlation settings. To minimize performance variations, we repeat the simulations for evaluating each mechanism 1000 times and compare the average values for different settings.

6.3.1. Impact of core node selection

For fair comparison, we use the same transmission mechanism 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.

Fig. 9 (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 strongly 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 strongly 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.

Fig. 9 (b) compares the total number of transmissions under different link quality. For each pair of bars, the left one shows D-CORD while the right one shows D-CoCo$^+$. 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 Fig. 9(a) and Fig. 9(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 determines the packet reception ratio while link correlation determines which packets are expected to be received/missed.

6.3.2. Impact of the transmission mechanism

In order to isolate the impact of transmission mechanism, similarly, we use the same core structure for both CORD and CoCo$^+$. D-CORD represents the dissemination using CORD's page-by-page transmission mechanism and D-CoCo$^+$ represents the dissemination using CoCo$^+$'s transmission mechanism. Fig. 10(a) shows the comparison result in term of the transmission rounds. The result shows that (1) D-CoCo$^+$ reduces the transmission rounds as compared to D-CORD. This is because D-CoCo$^+$'s transmission mechanism assembles re-transmissions into complete pages, and thus reduces the transmission rounds. (2) when link quality becomes better, the reduction of the negotiation rounds is less. This is because when link quality becomes better, there are fewer re-transmissions, thus the optimizing space for negotiation rounds becomes less.

Fig. 10 (b) shows the comparison of the transmission delays with different average link qualities. We can see that (1) D-CoCo$^+$ reduces the transmission delay compared to D-CORD. The reason is that by eliminating the sequential order of page reception, a node in D-CoCo$^+$ transmits more packets than that in D-CORD within a transmission round. Considering that the same CDS is used, the expected number of total transmissions are the same. Therefore, the transmission delay is reduced. (2) as expected, when the average link quality increases, the reduction becomes less since the performance gain space becomes less.

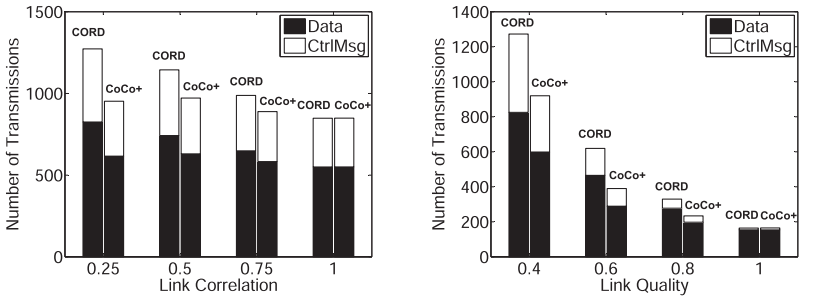
6.3.3. Impact of optimized negotiation

Fig. 11 (a) shows the comparison result in term 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 256 ms while CoCo$^+$'s nodes have different REQ timers. As discussed in Section 4.3, CoCo$^+$'s negotiation mechanism is density aware and all REQ timers are smaller than 50 ms, which corresponds to the neighbor size of 10 nodes. Fig. 11 (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 136 ms 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.

7. Discussion

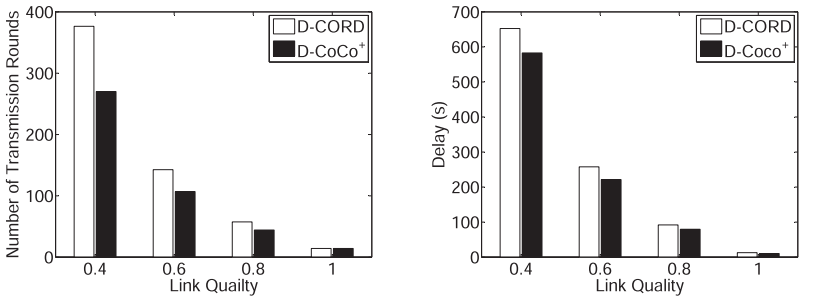
7.1. Co-existence with the data collection traffics

One typical traffic pattern in sensor networks is that all sensors transmit packets to the sink node. The traffic pattern of dissemination lies in the opposite direction.



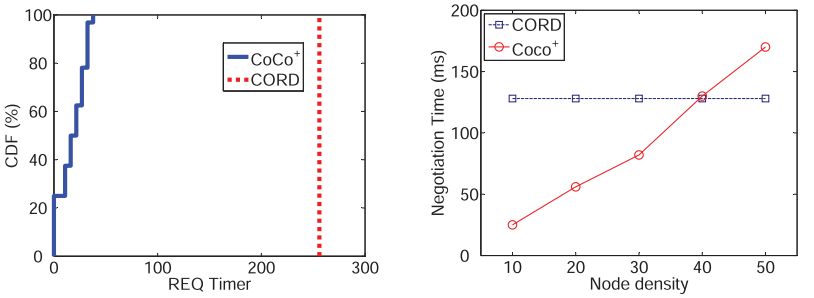
(a) Transmissions vs link correlation. (b) Transmissions vs link quality.

Fig. 9. The impact of correlation aware core node selection.



(a) Transmission rounds. (b) Transmission delay.

Fig. 10. The impact of the proposed consecutive transmission mechanism.



(a) CDF of negotiation time. (b) Negotiation time comparison.

Fig. 11. The impact of the optimized negotiation.

In this paper, we assume the dissemination traffic and the collection traffic happen at different times. Today's dissemination in wireless sensor networks is mainly aimed for software update, fixing bugs, changing sensing tasks, etc. These tasks will interrupt the network functionalities. As a result, during the process of dissemination, the data collection (from a sensor to the sink) will be suspended during dissemination until the software update or bug fixing is finished.

More general scenarios. We need to establish proper mechanisms for the co-existence of the two kinds of traffics. We might imagine that more powerful sensor nodes in the future may require application-specific bulk data dissemination, which will not interrupt the operation of the network. In such cases, the dissemination traffic and the collection traffic will co-exist in the network. Our design of the coordinated scheduling will no longer work well. We discuss two kinds of high-level solutions for the co-existence of the different traffics.

- Scheduling. We schedule the time of dissemination to avoid interrupting the data collections. This solution will work well in synchronized networks in which data collection are done in a periodic manner.
- Contention. The traffic pattern will be hard to predict if two kinds of traffic flows happen at the same time. We may need to design an efficient contention mechanism for resolving the two kinds of traffics. The mechanism should provide delay and reliability guarantee. We consider the combination of the dissemination traffic

and the collection traffic as a promising future working direction.

7.2. Page size optimization

First we explain why the dissemination should be done in a page-by-page manner. Then we give an analytical model to find the optimal page size.

7.2.1. Reasoning of page size

In data dissemination, each packet should be eventually acknowledged to ensure full reliability. Compared with packet level pipelining, page level pipelining has the following advantages: (1) the ACK overhead could be much reduced. The packet losses and receptions within one data page can be identified using only one REQ message. When using packet level pipelining, one ACK message is required for each packet transmission, incurring considerable overhead. (2) page level pipelining allows us to establish the coordinated sleep scheduling. On the other hand, for packet level pipelining, it is non-trivial to establish sleep scheduling. The reason is that the slot for one packet transmission is too short for the radio hardware turn-on and turn-off.

7.2.2. Page size optimization

We would like to model the relationship between page size, the link quality and the dissemination latency. For simplicity, we assume the link quality at each hop is the same and denoted as q. N denotes the largest hop count in the network (Table 1). We model the latency as follows. When the last hop node receives the first page, the node with hop count N-3 has received the second data page. After three hops transmission, the last hop node will receive the second page. The last hop node will receive a data page after three transmission rounds. Then the expected latency is calculated as

$$T = Nt + 3(n-1)t \tag{10}$$

where n denotes the number of data pages ($n = \frac{S}{s_p}$, where S is the data object size), and t denotes the per-hop latency, and is calculated as:

$$t = \frac{s_p}{Rq} + \frac{1}{q} - 1 \tag{11}$$

where R denotes the radio bitrate. Combining the above equations, the latency can be calculated as

$$T = (N-3)\frac{s_p}{Rq} + 3\left(\frac{1}{q} - 1\right)\frac{S}{s_p} + C \tag{12}$$

where C is a constant and equals $(N-3)(\frac{1}{q} - 1) + \frac{3S}{Rq}$. We can see that, with different link quality q, the optimal page size s_p is different. When link quality is better, the optimal page will be smaller. Otherwise, the optimal page size will be larger. When we remove the assumption of the same link quality at each hop, the problem would be much complex. We will leave the modeling without the assumption for future work.

7.3. Removal of ADV messages

The foundation of the ADV removal is that the dissemination is done over an underlying structure. When applied in mobile networks, the mechanism will not work well. For node failures, since we will update the CDS structure after each page transmission, when node failures happen, the structure update will find the node failure and select other nodes for dissemination. The reason is that failed nodes are inherent unable to broadcast CLAIM messages during the sender selection any more.

8. Conclusion

In this paper, we discover several critical limitations of existing backbone protocols and propose a correlated core based efficient bulk data dissemination in WSNs. CoCo$^+$ has three salient features: (1) it constructs a core structure considering link quality and link correlation, explicitly selects the nodes with fewer transmissions as core nodes. (2) it uses a novel consecutive transmission mechanism, which transmit packets in an out of order fashion, reducing the dissemination delay. (3) it also incorporates a novel lightweight and density aware negotiation mechanism for reducing negotiation overhead. Both simulation and testbed experiment results show that, compared to existing work, CoCo$^+$ greatly improves the dissemination performance in terms of the total number of transmissions and completion time.

Acknowledgment

We sincerely thank the reviewer and editors of Ad Hoc Networks for their valuable comments and feedback. This work is supported by the National Science Foundation of China grants no. 61202402, no. 61472360 and no. 61370087, the Fundamental Research Funds for the Central Universities, Zhejiang Provincial Platform of IoT Technology (2013E60005), Zhejiang Commonweal Project (2015C33077).

References

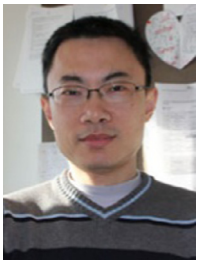
[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, Comput. Netw. 38 (2002) 393–422.
[2] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, L. Mo, W. Dong, Z. Yang, M. Xi, J. Zhao, X.-Y. Li, Does wireless sensor network scale? A measurement study on greenOrbs, in: Proceedings of INFOCOM, 2011, pp. 1983–1993.
[3] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, J. Anderson, Wireless sensor networks for habitat monitoring, in: Proceedings of WSNA, 2002, pp. 88–97.
[4] E.B. Hamida, G. Chelius, Strategies for data dissemination to mobile sinks in wireless sensor networks, Wirel. Commun. IEEE 15 (6) (2008) 31–37.
[5] J.W. Hui, D. Culler, The dynamic behavior of a data dissemination protocol for network programming at scale, in: Proceedings of SenSys, 2004, pp. 81–94.
[6] D. He, S. Chan, M. Guizani, Small data dissemination for wireless sensor networks: the security aspect, Wirel. Commun. IEEE 21 (3) (2014) 110–116.
[7] S. Kulkarni, L. Wang, Energy-efficient multihop reprogramming for sensor networks, ACM Trans. Sensor Netw. (TOSN) 5 (2) (2009) 16.
[8] X. Zheng, B. Sarikaya, Task dissemination with multicast deluge in sensor networks, IEEE Trans. Wirel. Commun. 8 (5) (2009) 2726–2734.
[9] V. Naik, A. Arora, P. Sinha, H. Zhang, Sprinkler: a reliable and energy efficient data dissemination service for wireless embedded devices, in: Proceedings of RTSS, 2005, pp. 276–286.
[10] L. Huang, S. Setia, CORD: energy-efficient reliable bulk data dissemination in sensor networks, in: Proceedings of INFOCOM, 2008, pp. 1247–1255.
[11] W. Dong, Y. Liu, C. Wang, X. Liu, C. Chen, J. Bu, Link quality aware code dissemination in wireless sensor networks, in: Proceedings of ICNP, 2011, pp. 89–98.
[12] W. Dong, B. Mo, C. Huang, Y. Liu, C. Chen, R3: optimizing relocatable code for efficient reprogramming in networked embedded systems, ACM Trans. Sensor Netw. 11 (2) (2014).

[13] K. Srinivasan, M. Jain, J. Choi, T. Azim, E. Kim, P. Levis, B. Krishnamachari, The κ factor: inferring protocol performance using inter-link reception correlation, in: Proceedings of MobiCom, 2010, pp. 317–328.
[14] T. Zhu, Z. Zhong, T. He, Z. Zhang, Exploring link correlation for efficient flooding in wireless sensor networks, in: Proceedings of NSDI, 2010, pp. 1–15.
[15] S. Guo, S. Kim, T. Zhu, Y. Gu, T. He, Correlated flooding in low-duty-cycle wireless sensor networks, in: Proceedings of ICNP, 2011, pp. 383–392.
[16] Z. Zhao, D. W., J. Bu, T. Gu, C. Chen, Exploiting link correlation for core-based dissemination in wireless sensor networks, in: Proceedings of SECON, 2014, pp. 372–380.
[17] A. Basalamah, S. Kim, S. Guo, T. He, Y. Tobe, Link correlation aware opportunistic routing, in: Proceedings of INFOCOM, 2012, pp. 3036–3040.
[18] W. Ye, J. Heidemann, D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in: Proceedings of INFOCOM, 2002, pp. 1567–1576.
[19] P. Guo, T. Jiang, Q. Zhang, K. Zhang, Sleep scheduling for critical event monitoring in wireless sensor networks, IEEE Trans. Parall. Distribut. Syst. 23 (2) (2012).
[20] M. Doddavenkatappa, M.C. Chan, B. Leong, Splash: fast data dissemination with constructive interference in wireless sensor networks, in: Proceedings of NSDI, 2013, pp. 269–282.
[21] F. Ferrari, M. Zimmerling, L. Thiele, O. Saukh, Efficient network flooding and time synchronization with Glossy, in: Proceedings of IPSN, 2011, pp. 73–84.
[22] I. Alam, S. Sultana, Y.C. Hu, S. Fahmy, Syren: Synergistic link correlation-aware and network coding-based dissemination in wireless sensor networks (2013).
[23] S.-J. Park, R. Vedantham, R. Sivakumar, I.F. Akyildiz, Garuda: achieving effective reliability for downstream communication in wireless sensor networks, IEEE Trans. Mob. Comput. 7 (2) (2008).
[24] P.A. Levis, N. Patel, D. Culler, S. Shenker, Trickle: a self regulating algorithm for code propagation and maintenance in wireless sensor networks, Comput. Sci. Div. Uni. California, 2003.
[25] W.B. Heinzelman, A.P. Chandrakasan, H. Balakrishnan, An application-specific protocol architecture for wireless microsensor networks, IEEE Trans. Wirel. Commun. 1 (4) (2002) 660–670.
[26] M. Heusse, F. Rousseau, R. Guillier, A. Duda, Idle sense: an optimal access method for high throughput and fairness in rate diverse wireless LANs, in: ACM SIGCOMM Computer Communication Review, 35, ACM, 2005, pp. 121–132.
[27] X. Cheng, M. Ding, D.H. Du, X. Jia, Virtual backbone construction in multihop ad hoc wireless networks, Wirel. Commun. Mob. Comput. 6 (2) (2006) 183–190.
[28] P. Levis, N. Lee, M. Welsh, D. Culler, TOSSIM: accurate and scalable simulation of entire TinyOS applications, in: Proceedings of SenSys, 2003, pp. 126–137.
[29] 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 Intell. 35 (2005) 115–148.
[30] R. Zheng, Asymptotic bounds of information dissemination in power-constrained wireless networks, IEEE Trans. Wirel. Commun. 7 (1) (2008) 251–259.

Zhiwei Zhao received his BS degree at the College of Electronic and Information in Xi'an Jiaotong University in 2010. He is currently a PhD student at the College of Computer Science in Zhejiang University. His research interests mainly focus on sensor networks and wireless computing. He is a student member of IEEE.

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

Wei Dong received his BS and Ph.D. degrees from the College of Computer Science at Zhejiang University in 2005 and 2011, respectively. He is currently an associate professor in the College of Computer Science in Zhejiang University. His research interests include networked embedded systems and wireless sensor networks. He is a member of IEEE.



Tao Gu is an Associate Professor in the School of Computer Science and Information Technology at RMIT University. He received his Bachelor degree from the Huazhong University of Science and Technology, and MSc from Nanyang Technological University, Singapore, and PhD in the computer science from National University of Singapore. His current research interests include mobile and pervasive computing, wireless sensor networks, distributed network systems, sensor data analytics, cyber physical system, Internet of Things, and online social networks. His publications appear in top journals such as JSAC, TMC, TC, TPDS and TKDE, and top conferences such as INFOCOM, UbiComp, ICNP, PerCom, and ICDM. He serves as TPC members for many leading conferences and Journal editorial board members. He is a senior member of the IEEE and a member of the ACM.

Xianghua Xu is now a professor in the School of Computer Science at Hangzhou Dianzi University, China. He received his Bachelor degree from the Hangzhou Institute of Electronic Engineering, and his PhD in Computer Science from the Zhejiang University, China. His current research interests include wireless sensor networks, parallel and distributed computing. He has published more than 100 peer reviewed journal and conference papers. He is the member of the IEEE, ACM. He is a recipient of the "Best Paper Award" of IISWC'2012.