CORD: Energy-efficient Reliable Bulk Data Dissemination in Sensor Networks

Leijun Huang Sanjeev Setia
Computer Science Department
George Mason University
Fairfax VA 22030

***Abstract*—This paper presents CORD, a reliable bulk data dissemination protocol for propagating a large data object to all the nodes in a large scale sensor network. Unlike well-known reliable data dissemination protocols such as Deluge whose primary design criterion is to reduce the latency of object propagation, CORD's primary goal is to minimize energy consumption. To achieve its goals CORD employs a two phase approach in which the object is delivered to a subset of nodes in the network that form a connected dominating set in the first phase, and to the remaining nodes in the second phase. Further, CORD installs a coordinated sleep schedule on the nodes in the network whereby nodes that are not involved in receiving or transmitting data can turn off their radios to reduce their energy consumption. We evaluated the performance of CORD experimentally on both an indoor and outdoor sensor network testbed and via extensive simulations. Our results show that in comparison to Deluge (the de facto network reprogramming protocol for TinyOS) CORD significantly reduces the energy consumption for reliable data dissemination while achieving a comparable latency.**

I. Introduction

Many sensor networks are deployed in environments where physically collecting previously deployed nodes for the purposes of re-programming them, i.e., updating their software, is either very difficult or infeasible. This gives rise to the need for "over-the-air" network re-programming for updating sensor nodes in place. The critical service required to enable multi-hop network re-programming is a *reliable bulk data dissemination* protocol.

Several protocols have been designed for reliable bulk data dissemination in sensor networks, such as MOAP [1], Deluge [2], MNP [3] and Sprinkler [4]. Notably, Deluge [2] is the *de facto* network reprogramming protocol for TinyOS[5]. In this paper, we present CORD (COre based Reliable Dissemination), an energy-efficient reliable bulk data dissemination protocol for large scale sensor networks.

Protocols for reliable bulk data dissemination can be divided into two categories based on the manner in which the object is propagated within the network. The first category includes Deluge, MOAP, and MNP. A common characteristic of these protocols is that the data object is propagated from the sink to the rest of the network in a neighborhood by neighborhood fashion, i.e., nodes that receive the object (or part of the object) become sources for distributing the object to their neighbors who are further downstream in relation to the sink. In these protocols, the data propagation wave moves from one neighborhood to the next, after the object has been disseminated to *all the nodes* in the first neighborhood.

In contrast, in the second category of protocols that includes our protocol CORD and Sprinkler, the object dissemination is divided into two distinct phases. Before the data dissemination commences, a subset of nodes in the network that have reliable links and that form an approximate minimum dominating set [6] are selected as *core nodes*. After this core construction step, in the first phase of the data dissemination protocol, the object is reliably propagated from the sink to the core nodes. After the entire object has been propagated to the core nodes, the second phase commences in which the core nodes disseminate the object to their neighboring non-core nodes in parallel.

The core-based two-phase approach used by CORD is motivated by the goal of reducing the energy consumption for disseminating the object within the network. By constructing a core for data dissemination, the protocol implicitly selects the set of nodes that are responsible for disseminating the object to their neighbors. This reduces the number of control messages that need to be exchanged between neighboring nodes in comparison to protocols such as Deluge and MNP which use a three-message (advertisement-request-data) handshake between nodes to select senders and disseminate data. Reducing the number of control messages and the number of nodes within a neighborhood competing to transmit the object also results in a reduction in the number of message collisions, which is an important consideration, especially in dense sensor networks.

Second, the two-phase core-based approach is also suitable for heterogeneous networks, where a subset of nodes in a network are more powerful than the others. Such networks are likely to be increasingly common in the near future. Selecting the more powerful nodes as core nodes and using CORD to disseminate large data objects saves energy at the more power-constrained non-core nodes, prolonging the life time of the whole network.

A distinctive feature of CORD is that in addition to adopting a two-phase approach, it aggressively uses sleep scheduling in order to further reduce energy consumption for large object dissemination. At the time of core construction, CORD installs a coordinated sleep schedule on the nodes in the network whereby nodes that are not involved in receiving or transmitting data can turn off their radios to reduce their energy

consumption. In CORD, both the core and non-core nodes adhere to this sleep schedule during the data dissemination. However, only the core nodes actively participate in the first phase of the protocol, whereas the non-core nodes participate passively by listening to the transmissions of core nodes.

A. Our Contributions

- Our protocol CORD uses a novel approach that involves the use of sleep scheduling in conjunction with a two phase approach for minimizing the energy consumed for the reliable dissemination of a large data object. Although sleep scheduling is widely used in sensor networks, and other data dissemination protocols such as Sprinkler have used a two-phase approach, our contribution is to demonstrate the feasibility and benefits of using coordinated sleep scheduling in conjunction with a two-phase approach for reliable bulk data distribution in sensor networks.
- We evaluate the energy and latency tradeoffs between CORD and single-phase approaches, specifically Deluge and MNP, via indoor and outdoor experiments and extensive simulations. To our knowledge, this is the first detailed evaluation of the energy tradeoffs between single-phase and two-phase protocols for reliable bulk data dissemination in sensor networks. In Section V, we present empirical results and simulation results that show that the energy consumption of CORD *is 30-60% of that of Deluge*, while the object dissemination latency of CORD *is comparable to that of Deluge for most scenarios.*

II. Related Work

A. Data Dissemination in Sensor Networks

Protocols for reliable bulk data dissemination in sensor networks that use a single-phase approach include MOAP [1], Deluge [2] and MNP [3]. All these protocols share some basic characteristics. First, these protocols were developed for supporting network reprogramming in multi-hop networks, and are used for entire code image delivery as opposed to difference-based application adjustment [7]. Second, these protocols extend the three-phase handshaking protocol used in SPIN-BL [8] for handling large data objects. Third, these protocols all borrow ideas such as the use of selective NACKs and hop-by-hop error recovery from prior work in reliable transport protocols [9] [10]. Deluge and MNP differ from MOAP in that a node does not need to receive the entire data object before it can start retransmitting it. By breaking the object up into pages, and allowing pipelined page delivery, Deluge and MNP take advantage of spatial multiplexing to reduce the latency of network reprogramming. MNP differs from Deluge in its approach for sender selection and in allowing radios to be turned off to avoid unnecessary packet receptions. CORD also divides objects into pages and uses pipelining; however, it differs from these protocols in using a core-based two-phase approach for data dissemination.

Sprinkler [4] is a reliable bulk data dissemination protocol that uses a two-phase approach. Sprinkler assumes the existence of a localization service at each node, and uses the latter to construct a connected dominating set (CDS). Sprinkler uses packet-level pipelining during the first phase of object dissemination (whereas CORD uses page-level pipelining). Further, Sprinkler uses TDMA to schedule packet transmissions among the CDS nodes. While Sprinkler and CORD both attempt to reduce energy consumption by minimizing packet transmissions, CORD introduces sleep scheduling at each node to further save energy. In Sprinkler, a core node forwards every newly received data packet and piggybacks the negative acknowledgment (NACK) for the lost data packets and its parent ID while transmitting a data packet. Therefore, a core node cannot sleep when its child core nodes are transmitting in their TDMA slots, since it needs to collect the NACKs.

CORD's core-based two-phase approach is similar to the approach used in GARUDA [11], which is a reliable data delivery protocol for sensor networks. GARUDA is designed to reliably deliver both small (single-packet) messages as well as larger messages. However, unlike CORD its design is not optimized for very large messages and therefore it does not use features such as pipelining which are critical for reduced data propagation latency in large networks. Another difference between CORD and GARUDA is that CORD uses sleep scheduling to minimize energy consumption during dissemination.

B. Connected Dominating Set (CDS)

The problem of CDS computation is widely explored in wireless routing and clustering protocols, where a subset of nodes in a network are selected as a backbone for routing, or as cluster-heads for data aggregation and forwarding. Many CDS computation algorithms have been proposed, such as the centralized algorithms proposed by Guha and Khuller [6], and distributed algorithms proposed by Das and Bharghavan [12], Wan and Alzoubi [13], Cheng et al. [14] and Wu et al. [15], [16]. CORD adapts Cheng's single leader algorithm [14] for core construction due to the similarity between the traffic pattern of the dissemination and the CDS structure rooted at the sink (the leader).

III. Protocol Overview & Design Tradeoffs

CORD first selects a subset of the nodes in the network as core nodes before data dissemination. Following the core construction step, each node in the network is either a core node, or is an immediate neighbor of a core node.

After the core is set up, the data object is propagated from the sink (the only node that initially possesses the data object to be distributed) to the core nodes in the first phase of dissemination by reliable hop-by-hop forwarding. In this phase, the non-core nodes passively participate in the protocol by listening to communications between core nodes. Consequently, at the end of the first phase, a non-core node may already possess a significant fraction of the object being disseminated. After the core nodes have received the entire

object, the protocol enters the second phase of the protocol. In this phase, each non-core node requests and receives the missing portions of the object from its neighboring core node.

A. Design Tradeoffs

The core-based approach used in CORD reflects a different design choice than protocols such as Deluge and MNP. By establishing a core, we are implicitly selecting the nodes that will be responsible for transmitting packets to their neighbors *for all the pages* of an object. In contrast, Deluge selects a sender separately *for each page* using a three-message handshake, and thus requires more control messages. The designers of Deluge motivate their approach by arguing that core-based approaches are relatively inflexible to variations in connectivity between nodes [2].

Our approach is motivated by the fact that most sensor networks are stationary, and by empirical evidence from extensive indoor and outdoor experiments that suggests that while the links between nodes are highly variable in their quality, it is possible to identify links that have low loss rates, and that the quality of such links is relatively stable *at the time scale of an object propagation.* The results presented in Section V demonstrate the advantages of our approach in reducing the number of control messages, which in turn reduces the energy consumption of the protocol.

More importantly, a core-based approach makes it possible to use coordinated sleep schedules to reduce the energy consumption of the protocol. In contrast, single-phase protocols such as Deluge rely on the use of passive listening by nodes for dynamically adjusting the rate of advertisements and for request suppression; these techniques only work well when nodes are awake most of the time, thus precluding the use of aggressive sleep scheduling. The results in Section V demonstrate that sleep scheduling is the most effective technique in reducing energy consumption. On the other hand, an open question that needs to be addressed is whether it is possible to use sleep scheduling to reduce energy consumption while at the same time achieving an object dissemination latency that is comparable to that of protocols such as Deluge.

IV. CORD Design

In this section, we describe the main components of the CORD protocol. For a more detailed description of the protocol including the packet formats and the state transition diagram, we refer the reader to [17].

A. Core Construction

We used Cheng's degree-aware single leader algorithm [14] as the basis for CORD's core construction approach. Note that a core is set up separately for each object dissemination, which is expected to be a relatively infrequent event. In Cheng's algorithm, initially all nodes are labeled as white and non-active. During the core construction, nodes are marked either black or gray. Black nodes become dominators (core nodes), while the gray ones become dominatees (non-core nodes). The leader initiates the core construction by marking itself black (and thus becoming a dominator). A white node marks itself gray and becomes a dominatee if one of its neighbors becomes a dominator. A non-active white node changes to active if one of its neighbors becomes a dominatee. Active nodes compete to be a dominator. The node with the maximum effective degree (number of white neighbors) wins the competition, and its gray parent also becomes a dominator to make the CDS connected.

a) Link Quality: We extend Cheng's algorithm in several ways for bulk data dissemination. First, we extend the algorithm to take link quality into account while forming the core. Two nodes are considered connected only when the link quality between them is above a threshold, Q_{th}. We assume that a sensor node maintains statistical information for the packet loss rate of its links to its neighbors. If the sensor network has been deployed for sufficient time, a node can estimate the quality of the links to its neighbors [18] based on packet loss rates. Alternatively, for motes using the more recent CC2420 radio such as MicaZ and TelosB, we can use the CC2420's Link Quality Indicator (LQI) as a metric of the link quality [19]. This requires only a single packet reception instead of relatively long-term link quality estimation. In our protocol, the link quality between u and v, Q_{uv}, is a metric that combines packet receive rates in both directions, $Q_{u\rightarrow v}$ and $Q_{v\rightarrow u}$, by $Q_{uv} = min(Q_{u\rightarrow v}, Q_{v\rightarrow u})$. Consequently, asymmetric links are not included in the core.

b) Establishing Coordinated Schedules: We further extend Cheng's algorithm to facilitate the establishment of coordinated schedules at each node (described in more detail in IV-B). The schedule consists of a repeating series of fixed-length slots in which a node either acts as a parent node in the data dissemination, or as a child node, or is quiescent. The schedules of neighboring nodes are coordinated to reflect their role in the data dissemination. The role of each slot is assigned after the dissemination begins; however, the schedule is installed at the time of core construction, as discussed below.

We modified Cheng's algorithm to integrate core construction with the establishment of coordinated node schedules. Specifically, the sink initiates core construction by starting the schedule and sending a CLAIM message, announcing itself as a core node. The message contains the time offset from the beginning of its first time slot to when the message was sent, as well as a list of neighbors to which the sink has good links. Nodes at level one that receive the CLAIM message (i.e., nodes that are one hop away from the sink) update their effective degrees by counting their neighbors that are not included in the sink's neighbor list. If a node that receives the CLAIM message has a good link with the sink, it selects the sink as its parent in the core, and initiates its own repeating schedule (based on the information in the message), such that the start time of the slots in its schedule coincides with the slots of the sink.

The nodes that have selected the sink as their parent then broadcast their effective degrees in COMPETE messages. Nodes at level 2 that receive one or more COMPETE messages respond with a SUBSCRIBE message to the competitor with

the maximum effective degrees among the competitors it hears (the node ID is used to break the tie). A node that receives SUBSCRIBE messages become a core node, otherwise it becomes a non-core node. Each core node at level one then broadcasts a CLAIM message announcing that it is a core node. This process continues recursively until every node in the network becomes either a core node or a non-core node.

While Cheng's algorithm first selects the core nodes at even-numbered levels and then chooses the core nodes at odd-numbered levels to make the CDS connected, in our modified algorithm, the core nodes are selected sequentially at each level. The modification guarantees that every node that wins the competition and becomes a core node already knows its parent and is synchronized with the parent.

B. Coordinated Node Sleep Scheduling

The coordinated schedule adopted by nodes in CORD is motivated by the observation that in protocols that use a pipelined data dissemination approach, nodes that transmit data simultaneously should be at least three hops apart to ensure that transfers of different pages do not interfere with each other. For example, consider the linear network shown in Figure 1. In this network, nodes A and D can simultaneously send different pages to their downstream neighbors without interference. Thus, after delivering page 0 to node B, node A has to wait until page 0 has been delivered by B to C and by C to D, before it starts transmitting page 1 to B. Consequently, protocols such as Deluge are designed to ensure the three hop separation between senders [2].

CORD exploits this enforced delay between transmissions of consecutive pages by allowing nodes that are idle to go to sleep to conserve energy. For example, when node A is transmitting page 0 to node B, node C is idle and can go to sleep. When node B is transmitting page 0 to node C, node A can go to sleep. Finally, when node C is transmitting page 0 to node D, node B can go to sleep.

Based upon this simple idea, a coordinated schedule is established at each node (during core construction) that will be used during data dissemination in the first phase of the protocol. For core nodes, the schedule consists of repeating fixed-length slots of time L in which a node takes turns acting as a parent in the data dissemination, acting as a child in the dissemination, and sleeping. We refer to these consecutive slots in a node's schedule as a P-slot, C-slot, and Q-slot respectively (where the P, C, and Q denote parent, child, and quiescent respectively). For non-core nodes, the schedule consists of a C-slot followed by two Q-slots.

As discussed in Section IV-A, each node synchronizes the boundaries of its time slots with its parent in the core at the time of core construction. However, it is also necessary to coordinate the schedules of neighboring nodes so that a node's C-slot coincides with the P-slot of its parent in the core. This coordination is performed at the time of data dissemination as follows. The sink marks its first slot as a P-slot and initiates the data dissemination by broadcasting advertisement packets. Nodes that receive advertisements or data from their parents

Fig. 1. Pipelined data dissemination

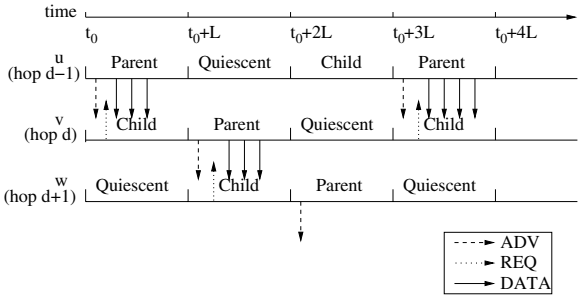

Fig. 2. Coordinated schedules of adjacent core nodes in the first phase of dissemination

assign the current slot to be a C-slot. Subsequently, each core node follows the repeating C-P-Q schedule, whereas each non-core node follows the C-Q-Q schedule. Figure 2 illustrates the coordinated schedule of three nodes u, v, and w at adjacent levels of the core.

We note that since CORD only requires slot coordination between neighboring nodes, it does not require an explicit network-wide time synchronization service. Further, as discussed below, these slots are relatively long, so even neighboring nodes do not need tight clock synchronization. The time interval corresponding to a slot, L, is set to be long enough so that a core node will be able to reliably deliver all the K packets in a page to its downstream children in the core. Thus, it should be larger than the time taken to deliver K packets plus the expected time for retransmitting packets that were not received in the previous round of transmissions. Unlike most TDMA approaches where a node is assigned a time slot only long enough to send one packet, in CORD, we use much longer time slots due to two reasons. First, by having longer time slots, the schedule coordination in CORD is more robust to clock drifts. Second, for most sensor nodes, switching the radio on and off consumes extra energy and time. With longer time slots, the overhead of switching the radio on and off is reduced.

C. Two-phase Data Dissemination

In CORD, the two-phase data dissemination follows the core construction step. In the first phase, pages of the object are propagated through the core in a pipelined fashion. A core node that has received one or more pages of the object broadcasts an advertisement with this information. Its child core node sends a request to the parent for the desired packets. The parent then broadcasts the requested data packets. Requests are suppressed if another request is being sent for the same or lower numbered page. This process is repeated until the entire page has been received by the downstream nodes in the core. In the second phase, non-core nodes make requests to their local core nodes for any missing data packets.

Unlike Deluge where all nodes broadcast advertisements, and each of them has a possibility to become a data sender, in CORD only core nodes are allowed to send advertisements and data packets. Moreover, all nodes stick to a fixed schedule. The schedules are synchronized such that when a parent node enters a P-slot, its child nodes enter a C-slot, and when the child nodes enter the following P-slot, the parent node enters a Q-slot and turns its radio off. Generally, nodes send advertisements and data packets in P-slots, and send requests and receive data packets in C-slots. However, a node may sleep in P or C-slots if there is no need to send or receive packets, e.g., if it (and its children) has already received the entire object.

V. Protocol Evaluation

We implemented CORD using the nesC programming language on the TinyOS platform, and evaluated the relative performance and energy consumption of CORD and Deluge through experiments on indoor and outdoor sensor network testbeds consisting of TelosB motes. We also used simulations to evaluate the protocols for variety of scenarios including larger network configurations than our testbeds.

A. Evaluation Metrics & Methodology

For evaluating the time taken by a protocol to reliably disseminate bulk data to the network, we measured the latency for delivering the entire object to all the nodes in the network. We refer to this latency as the object delivery latency. For CORD, this latency also includes the time for core construction. For all the protocols, we also report the individual object delivery latency for each node. For CORD, we also report the time spent in core construction and in the first and second phases of the protocol.

To compare the energy consumption of the protocols, we determined the average energy consumption per node during the time the data object is being disseminated. Due to the lack of a mechanism for directly measuring the residual energy level of the battery in a TelosB mote, we used an indirect mechanism for determining the energy consumption of the protocol. In our experiments, we logged 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 as well the contributions of different operations – specifically radio transmissions and reception, external EEPROM reads and writes, and CPU – to the total energy consumed by a node. The energy consumption of an operation was based on the TelosB current specification as shown in Table I [20] [21].

CPU current in active state	1.8mA
CPU current in sleep state	5.1uA
Radio current in receive state	23mA
Radio current in transmit state	21mA
Radio current in sleep state	1uA
External EEPROM current in write state	20mA
External EEPROM current in read state	4mA
External EEPROM current in sleep state	2uA

TABLE I
TelosB current specification

B. Testbed Description and Results

We used two different testbeds for our experimental evaluation. The indoor testbed consisted of 20 TelosB motes located in various offices on the same floor of our building (Figure 3). We ran multiple experiments on the testbed over a period of 32 hours commencing on a Sunday afternoon, covering times when there was little human activity in the building as well as times with both heavy human activity and wireless LAN traffic in the building. An additional node was programmed such that it issued a command to the network once every four hours to initiate the dissemination of an object of the same size.

The outdoor testbed consisted of 33 TelosB motes placed in a 3x11 grid on the floor on the top of a garage (Figure 4). The spacing between two adjacent nodes in the same row was 2 meters, whereas the spacing between adjacent nodes in the same column was 2.6 meters. Figure 5 shows the statistical packet reception rate as a function of distance between nodes observed in our experiments. We note that the TelosB motes usually have a larger radio range than shown in the figure when they are placed at an elevation. However, in our experiments, the nodes were placed on the floor in order to form a multi-hop network in a reasonable area. The outdoor experiments were conducted when there was little to no human activity near the testbed.

In both the indoor and outdoor experiments, an object of 960 packets, divided into 20 pages ($K = 48$), was disseminated from the sink (Node 0) to the network. Each packet has a 23 byte payload. For the CORD experiments, the packet reception rate between a node and each of its neighbors is estimated by exchanging HELLO messages before each run. This information is used by CORD during the core construction.

1) Empirical Results: Although the indoor experiments spanned periods with both light and heavy human activity, we found that the performance of both CORD and Deluge was relatively insensitive to the time at which the experiment was conducted. For example, Figure 6 plots the object delivery latency and average node uptime for each experiment in which CORD was used to disseminate the object. We observe that the object delivery latency for CORD varies between 200 seconds to 270 seconds depending on the time of day.

For CORD, the size of the core structure remains almost constant, consisting of four to six core nodes, although the core nodes may be different in different experiments. Figure 3 shows the core structure in one experiment. Nodes with circles are core nodes, and lines show the parent-child relationship between two nodes.

Table II shows the object delivery latency, average node uptime, average number of packet transmissions per node and average energy consumption per node averaged over

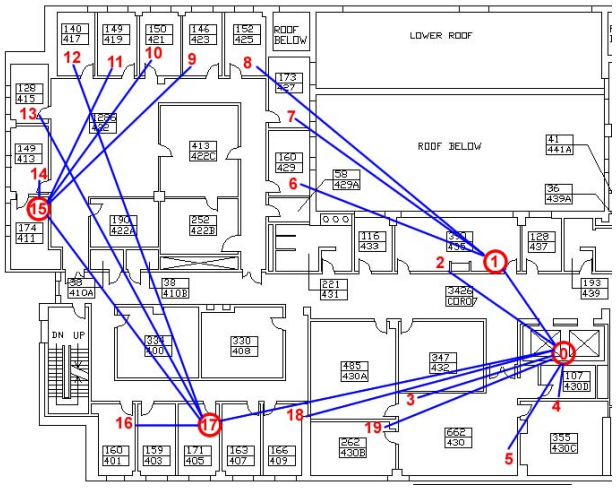

Fig. 3. Indoor TelosB network testbed including the core structure from one experiment (nodes in circles are core nodes)

Fig. 4. Outdoor 3x11 TelosB testbed

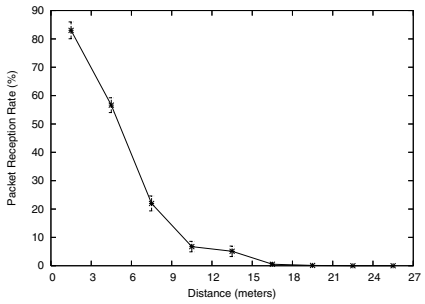

Fig. 5. Mean packet reception rate as a function of distance observed for outdoor 3x11 TelosB network (confidence intervals shown at 90% confidence level)

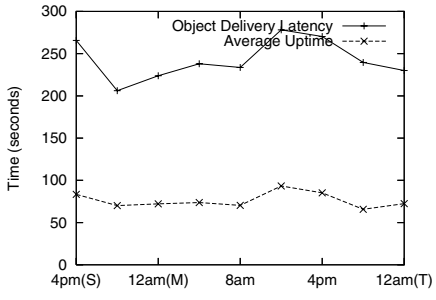

Fig. 6. Object delivery latency and average node uptime at different times of day for CORD on the indoor testbed

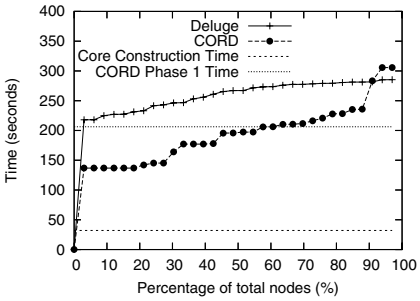

Fig. 7. Individual object delivery latency for CORD in one experiment on outdoor 3x11 TelosB network

the experiments. Note that the average uptime is equal to the object delivery latency for Deluge since nodes do not sleep in Deluge. The latency and uptime for CORD includes the time for core construction, and the number of packet transmissions for CORD includes the messages exchanged during core construction. The energy expenditure per node is computed based on TelosB current specification as shown in Table I [20] [21]. The result shows that on average CORD consumes 67% less energy than Deluge. This is because for CORD, on average, a node sleeps for 69% of the time during the dissemination. Further, a node transmits 20% fewer packets in CORD than in Deluge. The object delivery latency for CORD is slightly higher than for Deluge. Note, however, that the network has only three hops, therefore pipelining in Deluge or CORD is not effective in these indoor experiments.

In experiments conducted on the outdoor testbed, we found the diameter of the network was 4-5 hops, thus making pipelining more useful. In the CORD experiments, we found that 9 to 11 of the 33 nodes acted as core nodes while the remaining were non-core nodes. Table III shows the object delivery latency, average node uptime, average number of packet transmissions and energy consumption per node. Our results show that on average CORD consumes 69% less energy than Deluge in disseminating a 960-packet object to the 33-node network, while achieving a comparable object delivery latency.

	Latency (sec)	Node Uptime (sec)	Number Packet Transmissions	Node Energy(mAh)
CORD	243 ± 14.7	76.3 ± 5.55	261 ± 32.6	0.52
Deluge	226 ± 17.3	226 ± 17.3	331 ± 21.5	1.56

TABLE II
AVERAGE OBJECT DELIVERY LATENCY AND ENERGY EXPENDITURE PER NODE FOR INDOOR EXPERIMENTS (CONFIDENCE INTERVALS ARE SHOWN WITH 90% CONFIDENCE LEVEL)

	Latency (sec)	Node Uptime (sec)	Number Packet Transmissions	Node Energy(mAh)
CORD	301 ± 10.0	95.9 ± 5.56	398 ± 78.7	0.66
Deluge	313 ± 10.1	313 ± 10.1	483 ± 5.91	2.15

TABLE III
AVERAGE OBJECT DELIVERY LATENCY AND ENERGY EXPENDITURE PER NODE FOR OUTDOOR EXPERIMENTS (CONFIDENCE INTERVALS ARE SHOWN WITH 90% CONFIDENCE LEVEL)

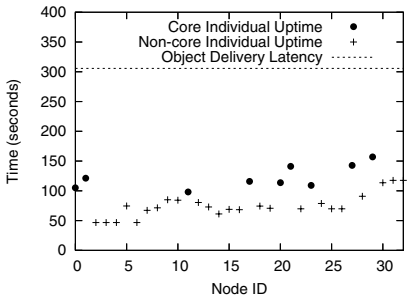

Fig. 8. Individual node uptime for CORD in one experiment on outdoor 3x11 TelosB network

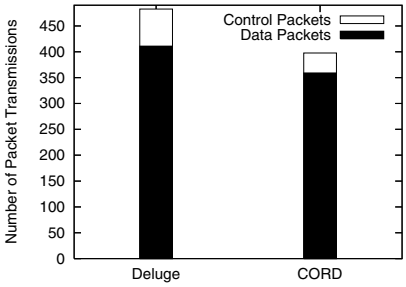

Fig. 9. Average number of packet transmissions per node on outdoor 3x11 TelosB network

To better understand the dynamics of CORD and Deluge, we now focus on the observations from single experiments. Figure 7 shows the distribution of individual object delivery latencies observed in one experiment for CORD and Deluge respectively. To deliver the 960 packet object, divided into 20 pages to all the nodes in the network, CORD uses slightly more time than Deluge. However, in the CORD experiment, most nodes receive the object in less time than in the Deluge experiment. Further investigation shows the last three nodes receiving the object in CORD are those in the farthest row (Nodes 30, 31 and 32) from the sink in the grid. Being at the edge of the network, they only receive a small fraction of the object by passive listening during the first phase of CORD. Meanwhile, as non-core nodes, they have to wait until the second phase to make requests for missing packets in the object.

In the same figure, we also plot the core construction time for CORD, which is about 30 seconds in this experiment, and the time for the first phase of the data dissemination. We see that by the end of the first phase around 60% of the nodes have received the entire object, although there were only 9 core nodes (27%) in the network in this experiment (see Figure 8). All of the non-core nodes finishing in the first phase receive the object by passive listening, without sending a packet.

Figure 8 shows the individual node uptime during the dissemination from the time when the sink initiates core construction to the time when the last node receives the object. While the core nodes generally have larger uptimes than non-core nodes, on average, a node is up for only one-third of the time during the dissemination.

Finally, Figure 9 shows the number of packet transmissions per node averaged over multiple experiments. We see that overall CORD needs fewer packet transmissions to deliver the same object. The relative saving in control packets is more pronounced, illustrating the effectiveness of the core-based approach used in CORD. However, the vast majority of the packets are data packets. Thus, the major savings in energy consumption in CORD relative to Deluge are achieved via the use of sleep scheduling, and not through a reduction in packet transmissions.

C. Simulation Results

In order to compare CORD with Deluge for larger network sizes, object sizes, and topologies, we used the TOSSIM [22] and PowerTOSSIM [23] simulation tools to evaluate the relative performance and energy consumption of CORD, Deluge and MNP for a variety of scenarios. The simulation code for Deluge and MNP was based on the version included in the TinyOS distribution[5]. We used an empirical packet loss model in our simulations that was obtained from indoor packet reception experiments using MICA2 motes. This model captures the radio irregularities and link asymmetry that have been reported by several researchers [18], [24], [25]. Figure 10 plots the mean packet reception rate as a function of the distance between the nodes and the transmission power level of the nodes.

We considered both a grid topology in which the nodes are organized in a rectangular grid and a random topology where nodes are uniformly distributed over a given area. However, following the trend established in previous simulation studies [2], [3], the majority of the simulation results in this paper are for grid networks. In the following discussion, a grid network is denoted by mxn-s, where m and n are the dimensions of the rectangular grid topology, and s is the distance between a node and its closest neighbor measured in meters. Starting from a default scenario with the parameters listed in Table IV, we varied the network size, network density and object size one at a time, and evaluated the effect of these factors on the performance and energy consumption of the protocols.

For all scenarios, only the sink (Node 0), located at a corner, has the complete data object at the beginning of the simulation. Unless stated otherwise, all of the results reported in Section V-C are mean values obtained from multiple simulation runs. All results have 90% confidence intervals with width less than 5% of the mean.

1) Effect of Network Size: Simulations of CORD on 5xn-9 (n=10,20 and 30) MICA2 networks show the diameter of the network increases when n increases from 10 to 30. The diameter of the network is 6 hops for the 5x10-9 network and 17 hops for the 5x30-9 network. However, the percentage of core nodes in the network is relatively constant, at 34%-39%, reflecting a constant network density.

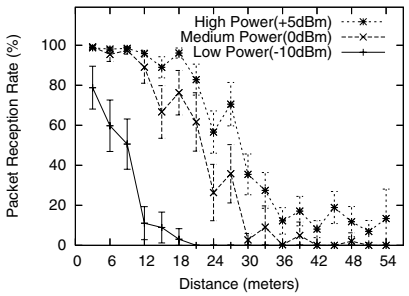

Fig. 10. Mean packet reception rate of MICA2 motes over distance on an indoor testbed (confidence intervals shown with 90% confidence level)

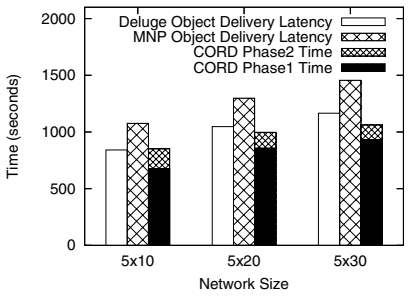

Fig. 11. Object delivery latency for various network sizes

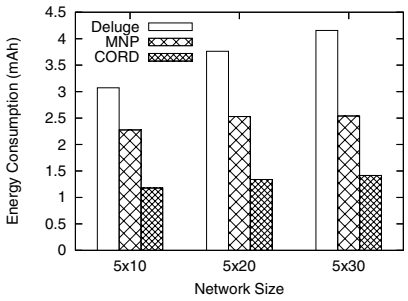

Fig. 12. Energy consumption for various network sizes

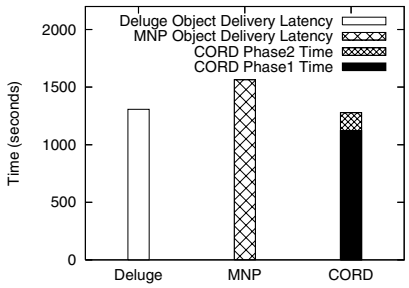

Fig. 13. Latency for random topology (200 nodes)

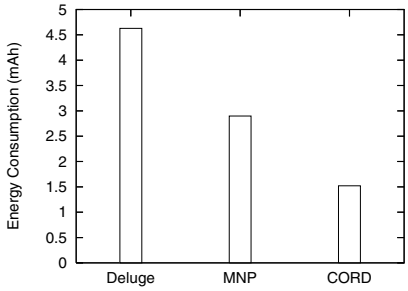

Fig. 14. Energy consumption for random topology (200 nodes)

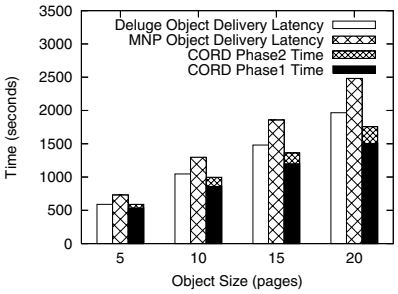

Fig. 15. Latency for various object sizes (5x20-9 network)

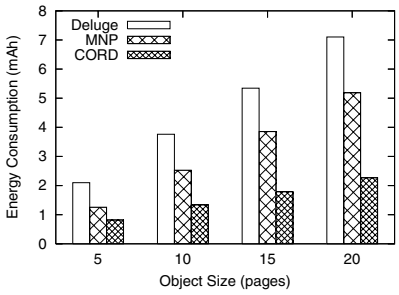

Fig. 16. Energy consumption for various object sizes (5x20-9 network)

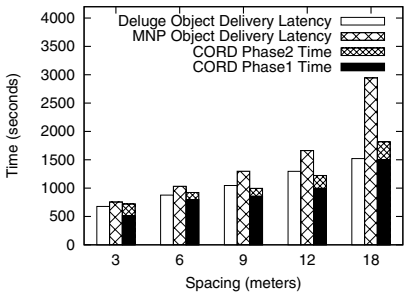

Fig. 17. Latency for various network densities (5x20 network)

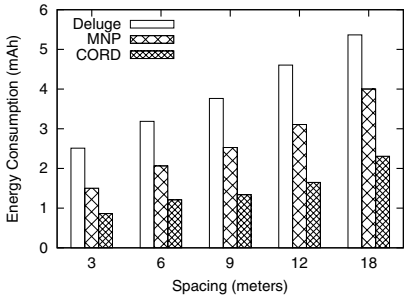

Fig. 18. Energy consumption for various network densities (5x20 network)

Network topology	5x20 Grid
Spacing	9 meters
Transmission power level	medium (0dBm)
Object size	10 pages
Page size (K)	128 packets
Packet payload size	23 bytes
Slot length (L)	6 seconds

TABLE IV
DEFAULT PARAMETER SETTINGS FOR THE SIMULATION EXPERIMENTS

Figure 11 shows the latency of delivering a 10 page object for CORD, Deluge and MNP simulations in networks with different sizes. From the figure, we can see that while the object delivery latency increases with the network size, the duration of the second phase of CORD is relatively constant, due to the approximately constant density of the given networks. CORD has a slightly lower latency than Deluge for the larger networks.

Figure 12 shows the average energy consumption at each node in the networks. For all three networks, the energy consumption of CORD is around 35% of that of Deluge, and around 50% of that of MNP.

2) Effect of Network Topology: Because of space limitations, most of the results reported in this paper are for networks in which nodes are deployed in a grid topology. However, we also examined the relative performance and energy consumption of the protocols for networks in which nodes are randomly deployed over a given area. We found that the results obtained for random topologies are similar to those for the grid topologies. For example, Figures 13 and 14 show the latency and energy consumption of Deluge, MNP and CORD for a random topology where 200 nodes are uniformly

distributed in a 50mx200m area. We refer the reader to [17] for more results for networks with random topologies.

3) Effect of Data Object Size: Figures 15 and 16 show the results for simulations in which 5, 10, 15 and 20 pages were disseminated over the default 5x20-9 MICA2 network. When the object size increases, the second phase of CORD becomes longer because there is a corresponding increase in the number of packets that a non-core node needs to obtain from its parent core node. The latency of CORD is smaller than that of Deluge when the object is larger than 5 pages. This is because CORD uses pre-selected senders and thus avoids overhead of sender selection for each page during data dissemination, which is more beneficial when disseminating larger data objects. MNP has the largest latency among the three, however, it consumes less energy than Deluge due to its energy saving optimizations such as turning the radio off when a node is neither receiving nor transmitting data. The energy consumption of CORD is 32%-39% of that of Deluge for all of the four object sizes, and the savings become larger when the object size increases.

4) Effect of Network Density: In our simulations, network density is controlled by the distance between neighboring nodes in the rectangular grid. When the density decreases, neighboring nodes have fewer good links resulting in more hops needed to traverse the network. The increase in the diameter of the network and in the number of core nodes prolongs the first phase of CORD.

For the 5x20 layout of the topology, when the spacing between nodes increases from 3 meters to 18 meters, the simulations show that the diameter of the network, as observed by CORD, increases from 3 hops to 27 hops, and the percentage of core nodes also increases from 9% to 57%. Note that for the scenarios with 12 and 18 meters spacing between nodes the average packet loss rate is 17% and 25% respectively; thus these scenarios represent networks with relatively high packet loss rates. Figures 17 and 18 show the latency and energy consumption for the networks with different densities. When density decreases, all of the three protocols consume more energy. For all of the five scenarios, the energy consumption of CORD is 35%-40% of that of Deluge.

VI. Conclusions

In this paper, we presented CORD, a reliable bulk data dissemination protocol for large scale multi-hop sensor networks. CORD differs from previously proposed protocols in its aggressive use of sleep scheduling in conjunction with a two-phase core-based pipelined object propagation approach.

We evaluated the energy consumption and object delivery latency of CORD, Deluge and MNP via extensive experiments and simulations. Our results show that:

- The energy consumption for large object dissemination in CORD is 30%-60% of that of Deluge. The vast majority of the energy savings achieved in CORD is due to the aggressive use of sleep scheduling.
- The object delivery latency of Deluge and CORD are comparable, although the majority of the nodes receive the object earlier in CORD than in Deluge.
- The relative latency and energy consumption advantages of CORD over Deluge increase as the network and object sizes are increased.

References

[1] T. Stathopoulos, J. Heidemann, and D. Estrin, "A remote code update mechanism for wireless sensor networks," CENS, Tech. Rep., 2003.

[2] J. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proceedings of the 2nd ACM SenSys*, 2004.

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

[4] V. Naik, A. Arora, P. Sinha, and H. Zhang, "Sprinkler: A reliable and energy efficient data dissemination service for wireless embedded devices," in *Proceedings of 26th IEEE RTSS*, 2005.

[5] "Tinyos community forum." [Online]. Available: http://www.tinyos.net

[6] S. Guha and S. Khuller, "Approximation algorithms for connected dominating sets," in *European Symposium on Algorithms*, 1996, pp. 179–193.

[7] N. Reijers and K. Langendoen, "Efficient code distribution in wireless sensor networks," in *Proc. of the 2nd ACM WSNA Conf.*, 2003.

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

[9] F. Stann and J. Heidemann, "RMST: Reliable data transport in sensor networks," in *Proceedings of the SNPA Workshop*, 2003.

[10] C. Wan *et al.*, "PSFQ: A reliable transport protocol for wireless sensor networks," in *Proceedings of the ACM WSNA*, 2002.

[11] S. Park *et al.*, "A scalable approach for reliable downstream data delivery in wireless sensor networks," in *Proc. of ACM MobiHoc*, 2004.

[12] B. Das and V. Bharghavan, "Routing in ad-hoc networks using minimum connected dominating sets," in *Proc. of the IEEE ICC*, 1997.

[13] P. Wan, K. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," in *Proceedings of the 21st INFOCOM*, 2002.

[14] X. Cheng, M. Ding, D. Du, and X. Jia, "Virtual backbone construction in multihop ad hoc wireless networks," *Wireless Communications and Mobile Computing*, vol. 6, pp. 183–190, 2006.

[15] F. Dai and J. Wu, "Distributed dominant pruning in ad hoc networks," in *Proceedings of the IEEE ICC*, 2003.

[16] J. Wu and H. Li, "On calculating connected dominating sets for efficient routing in ad hoc wireless networks," *Telecommunication Systems, Special issue on Moible Computing and Wireless Networks*, vol. 18, no. 1/3, pp. 13–36, 2001.

[17] L. Huang, "Reliable bulk data dissemination in sensor networks," Ph.D. dissertation, Department of Computer Science, George Mason University, December 2007, Technical Report GMU-CS-TR-2008-1.

[18] A. Woo *et al.*, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *Proceedings of ACM SenSys*, 2003.

[19] B. Chen, K. Reddy, and M. Welsh, "Ad-hoc multicast routing on resource-limited sensor nodes," in *Proceedings of the 2nd International Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality*, 2006.

[20] "Telosb motes." [Online]. Available: http://www.xbow.com

[21] G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy, "Ultra-low power data storage for sensor networks," in *Information Processing in Sensor Networks (IPSN)*, 2006.

[22] P. Levis *et al.*, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in *Proceedings of the ACM SenSys*, 2003.

[23] V. Shnayder *et al.*, "Simulating the power consumption of large-scale sensor network applications," in *Proceedings of the ACM SenSys*, 2004.

[24] G. Zhou, T. He, S. Krishnamurthy, and J. Stankovic, "Impact of radio irregularity on wireless sensor networks," in *Proceedings of ACM MobiSys*, 2004.

[25] A. Cerpa *et al.*, "Statistical model of lossy links in wireless sensor networks," in *Proceedings of ACM/IEEE 4th IPSN*, 2005.