Protocol Design and Optimization for Delay/Fault-Tolerant Mobile Sensor Networks

Yu Wang, Hongyi Wu, Feng Lin, and Nian-Feng Tzeng
The Center for Advanced Computer Studies
University of Louisiana at Lafayette
P.O. Box 44330, Lafayette, LA 70504

Abstract

While extensive studies have been carried out in the past several years for many sensor applications, they cannot be applied to the network with extremely low and intermittent connectivity, dubbed the Delay/Fault-Tolerant Mobile Sensor Network (DFT-MSN). Without end-to-end connections due to sparse network density and sensor node mobility, routing in DFT-MSN becomes localized and ties closely to medium access control, which naturally calls for merging Layer 3 and Layer 2 protocols in order to reduce overhead and improve network efficiency. DFT-MSN is fundamentally an opportunistic network, where the communication links exist only with certain probabilities and become the scarcest resource. At the same time, the sensor nodes in DFT-MSN have very limited battery power like those in other sensor networks. Clearly, there is a tradeoff between link utilization and energy efficiency. To address this tradeoff, we develop a cross-layer data delivery protocol for DFT-MSN, which includes two phases, i.e., the asynchronous phase and the synchronous phase. In the first phase, the sender contacts its neighbors to identify a set of appropriate receivers. Since no central control exists, the communication in the first phase is contention-based. In the second phase, the sender gains channel control and multicasts its data message to the receivers. Furthermore, several optimization issues in these two phases are identified, with solutions provided to reduce the collision probability and to balance between link utilization and energy efficiency. Our results show that the proposed cross-layer data delivery protocol for DFT-MSN achieves a high message delivery ratio with low energy consumption and an acceptable delay.

1 Introduction

The wireless sensor network has been extensively studied in the past several years, with numerous approaches proposed for routing, medium access control, data aggregation, topology control, power management, etc. While these mainstream approaches in the literature are well suitable for many sensor applications, they cannot be applied to the scenarios with extremely low and intermittent connectivity due to sparse network density and sensor node mobility. Two typical examples are pervasive air quality monitoring, where the goal is to track the average toxic gas inbreathed by human beings everyday, and flu virus tracking, which aims to collect data of flu virus (or any epidemic disease in general) in order to monitor and prevent the outbreak of devastating flu. Note that, while samples can be collected at strategic locations for flu virus tracking or air quality monitoring, the most accurate and effective measurement shall be taken at the people, naturally calling for an approach of deploying wearable sensors that closely adapt to human activities. As a result, the connectivity among the *mobile* sensors is poor, and thus it is difficult to form a well connected mesh network for transmitting data through end-to-end connections from the sensor nodes to the sinks.

In order to address this problem, the Delay/Fault-Tolerant Mobile Sensor Network (DFT-MSN) has been introduced recently for pervasive data acquisition [1]. DFT-MSN aims to gather a massive amount of information from a statistic perspective and to update the information base periodically. Thus, data transmission delay and faults are usually tolerable in such application scenarios. A DFT-MSN under our consideration consists of two types of nodes, the wearable sensor nodes and the high-end sink nodes. The former are attached to people (or other mobile objects), gathering target information and forming a loosely connected mobile sensor network for information delivery. With a short sensor transmission range and nodal mobility, the connectivity of DFT-MSN is very low, where a sensor connects to other sensors only occasionally. A number of high-end nodes (e.g., mobile phones or personal digital assistants with sensor interfaces) are either deployed at strategic locations with high visiting probability or carried by a subset of people, serving as the *sinks* to receive data from

wearable sensors and forward them to access points of the backbone network.

Without end-to-end connections, routing in DFT-MSN becomes localized and ties closely to Layer 2 protocols, which naturally calls for merging Layer 3 and Layer 2 in order to reduce overhead and improve network efficiency. In this research, we develop and evaluate a cross-layer data delivery protocol for DFT-MSN. Note that, while cross-layer design has been discussed extensively in the past several years, this work distinguishes itself from others by considering the unique characteristics of the sparsely-connected, delay-tolerant mobile sensor network. The goal of conventional sensor network protocols (e.g., [2–4]) is to optimize energy consumption with a given delay or throughput requirement, with the sensor nodes usually enjoying stable connectivity and ample channel bandwidth. DFT-MSN, however, is fundamentally an opportunistic network, where the communication links exist only with certain probabilities and are deemed the scarcest resource. A naive approach is to let sensors work aggressively in order to catch every possible opportunity for data transmission. Under such an approach, however, the sensors are likely to drain off their battery quickly, resulting in poor performance of the overall network. Clearly, there is a tradeoff between link utilization and energy efficiency. None of the existing sensor network protocols have considered such a unique network environment and performance tradeoff.

Our goal is to make efficient use of the transmission opportunities whenever they are available, while keeping the energy consumption at the lowest possible level. To address this tradeoff between data delivery ratio/delay and overhead/energy, we design the data delivery protocol with two phases, i.e., the asynchronous phase and the synchronous phase. In the first phase, a sender contacts its neighbors to identify a set of appropriate receivers. Since no central control exists, the communication in the first phase is contention-based. In the second phase, the sender gains channel control and multicasts its data message to the receivers. Furthermore, several optimization issues in these two phases are identified, with possible solutions proposed to reduce the collision probability and improve energy efficiency. Extensive simulations are carried out for performance evaluation. Our results show that the proposed cross-layer data delivery protocol for DFT-MSN achieves a high message delivery ratio with low energy consumption and an acceptable delay.

The rest of this paper is organized as follows: Sec. 2 discusses related work. Sec. 3 presents the proposed cross-layer data delivery protocol for DFT-MSN. Sec. 4 discusses protocol optimization. Sec. 5 presents simulation results. Finally, Sec. 6 concludes the paper.

2 Related Work

Research on Delay/Fault-Tolerant Mobile Sensor Network (DFT-MSN) is motivated mainly by the Delay-Tolerant Network (DTN) and its applications in sensor networks and mobile ad hoc networks. An overview of DFT-MSN is presented in [1], outlining its application scenarios and unique characteristics, such as sensor mobility, loose connectivity, fault tolerability, delay tolerability, and buffer limit. Following that, an in-depth study of DFT-MSN is given in [5], where two basic data delivery approaches in DFT-MSN, namely, direct transmission and flooding, are discussed with their performance analyzed by using queuing models. Based on the analytic results, a message Fault-tolerance-based Adaptive Delivery (FAD) scheme is proposed, comprising two key components respectively for data transmission and queue management.

DTN technology has been recently introduced into wireless sensor networks. Its pertinent work can be classified into the following three categories, according to their differences in nodal mobility. (1) *Network with Static Sensors.* The first type of DTN-based sensor networks are static. Due to the limited transmission range and battery power, the sensors are loosely connected to each other and may be isolated from the network frequently. For example, the Ad hoc Seismic Array developed at the Center for Embedded Networked Sensing (CENS) employs seismic stations (i.e., sensors) with large storage space and enables store and forward of bundles with custody transfer between intermediate hops [6]. In [7], wireless sensor networks are deployed for habitat monitoring, where the sensor network is accessible and controllable by the users through the Internet. The SeNDT (Sensor Networking with Delay Tolerance) project targets at developing a proof-of-concept sensor network for lake water quality monitoring, where the radio connecting sensors are mostly turned off to save power, thus forming a loosely connected DTN network [8]. DTN/SN focuses on the deployment of sensor networks that are inter-operable with the Internet protocols [9]. [10] proposes to employ the DTN architecture for mitigating communication interruptions, so as to provide reliable data communication across heterogeneous, failure-prone networks. (2) *Network with Managed Mobile Nodes.* In the second category, mobility is introduced to a few special nodes to improve network connectivity. For example, the Data Mule approach is proposed in [11] to collect sensor data in sparse sensor networks, where a mobile entity called data mule receives data from the nearby sensors, temporarily store them, and drops off the data to the access points. This approach can substantially save the energy consumption of the sensors as they only transmit over a short range, and at the same time, enlarge the service area of the sensor network. (3) *Network with Mobile Sensors.* While all of the above delay-

COMPUTER SOCIETY

tolerant sensor networks center at static sensor nodes, there are several examples besides DFT-MSN that are based on mobile sensors. ZebraNet [12] employs the mobile sensors to support wildlife tracking for biology research. The ZebraNet project targets at building a position-aware and power-aware wireless communication system. A history-based approach is proposed for routing, where the routing decision is made according to the node's past success rate of transmitting data packets to the base station directly. When a sensor meets another sensor, the former transmits data packets to the latter if the latter has a higher success rate. The Shared Wireless Info-Station (SWIM) system is proposed in [13, 14] for gathering biological information of radio-tagged whales. It is assumed in SWIM that the sensor nodes move randomly and thus every node has the same chance to meet the sink. A sensor node distributes a number of copies of a data packet to other nodes so as to reach the desired data delivery probability.

3 Proposed Cross-Layer Data Delivery Protocol for DFT-MSN

In this section, we propose the data delivery protocol for DFT-MSN by taking a cross-layer approach, aiming to strike the balance between link utilization and energy efficiency. By considering the unique characteristics of DFT-MSN where the sensors are sparsely-connected and usually experience long transmission delay, this work distinguishes itself from other cross-layer approaches that have been discussed for conventional sensor networks. In the rest of this section, we first introduce two important protocol parameters and then present our proposed cross-layer protocol design.

3.1 Protocol Parameters

The proposed data delivery protocol is based on two important parameters, namely, the nodal delivery probability and the message fault tolerance, as discussed below separately.

3.1.1 Nodal Delivery Probability

The decision on data transmission is made based on the *delivery probability*, which indicates the likelihood that a sensor can deliver data messages to the sink. Let ξ_i denote the delivery probability of sensor i. ξ_i is initialized with zero and updated upon an event of either message transmission or timer expiration. More specifically, the sensor maintains a timer. If there is no message transmission within an interval of Δ, the timer expires, generating a timeout event. The timer expiration indicates that the sensor couldn't transmit

any data messages during Δ, and thus its delivery probability should be reduced. On the other hand, whenever sensor i transmits a data message to another node k, ξ_i should be updated to reflect its current ability in delivering data messages to the sinks. Note that since end-to-end acknowledgement is not employed in DFT-MSN due to its low connectivity, sensor i doesn't know whether the message transmitted to node k will eventually reach the sink or not. Therefore, it estimates the probability of delivering the message to the sink by the delivery probability of node k, i.e., ξ_k. More specifically, ξ_i is updated as follows,

$$\xi_i = \begin{cases} (1-\alpha)[\xi_i] + \alpha\xi_k, & Transmission \\ (1-\alpha)[\xi_i], & Timeout, \end{cases} \quad (1)$$

where $[\xi_i]$ is the delivery probability of sensor i before it is updated, and $0 \le \alpha \le 1$ is a constant employed to keep partial memory of the historic status. If k is the sink, $\xi_k = 1$, because the message is already delivered to the sink successfully. Otherwise, $\xi_k < 1$. Clearly, ξ_i is always between 0 and 1.

3.1.2 Message Fault Tolerance

Each sensor has a data queue that contains data messages ready for transmission. The data messages of a sensor come from three sources: (a) After the sensor acquires data from its sensing unit, it creates a data message, which is inserted into its data queue; (b) When the sensor receives a data message from another sensor, it inserts the message into its data queue; (c) After the sensor sends out a data message to a non-sink sensor node, it may also insert the message into its own data queue again, because the message is not guaranteed to be delivered to the sink. In DFT-MSN, multiple copies of the message may be created and maintained by different sensors in the network, resulting in redundancy. The fault tolerance degree (FTD) is introduced to represent the amount of redundancy and to indicate the importance of a given message. More specifically, each message copy is associated with an FTD, which is defined as the probability that at least one copy of this message is delivered to the sink by other sensors in the network. Let $\mathcal{F}_i^M$ denote the FTD of message M in the queue of sensor i. $\mathcal{F}_i^M$ is determined when the message is inserted into the queue. For a new message generated from the sensing unit, $\mathcal{F}_i^M = 0$, i.e., with the highest importance. During data transmission, the message FTD will be updated. Without loss of generality, let's consider a sensor i, which is multicasting data message M to a set of nearby sensors denoted by Φ. The message copy transmitted to neighbor node j is associated with an FTD of $\mathcal{F}_j^M$,

$$\mathcal{F}_j^M = 1 - (1 - [\mathcal{F}_i^M])(1 - \xi_i) \prod_{m \in \Phi, \, m \ne j} (1 - \xi_m), \quad (2)$$

and the FTD of the message at sensor i is updated as

$$\mathcal{F}_i^M = 1 - (1 - [\mathcal{F}_i^M]) \prod_{m \in \Phi} (1 - \xi_{\psi_m}), \qquad (3)$$

where $[\mathcal{F}_i^M]$ is the FTD of message M at sensor i before multicasting.

The data queue management is to appropriately sort the data messages in the queue, to determine which data message is to be sent when the sensor meets another sensor, and to determine which data message is to be dropped when the queue is full. Our proposed queue management scheme is based on the FTD. More specifically, the message with a smaller FTD is more important and should be transmitted with a higher priority. This is done by sorting the messages in the queue with an increasing order of their FTD's. Message with the smallest FTD is always at the top of the queue and transmitted first. A message is dropped at the following two occasions. First, if the queue is full when a message arrives, the new message is inserted into the queue at an appropriate position according to its FTD, while the message at the end of the queue is dropped. Second, if the fault tolerance of a message is larger than a threshold, the message is dropped, even if the queue is not full. This is to reduce transmission overhead, given that the message will be delivered to the sinks with a high probability by other sensors in the network. A special example is the message which has been transmitted to the sink. It will be dropped immediately because it has the highest FTD of 1.

3.2 Cross-Layer Data Delivery Protocol for DFT-MSN

In the proposed data delivery protocol, each sensor has a working cycle that consists of two modes, the sleep mode and the work mode. The length of the working cycle is dynamic, as will be discussed in Sec. 4. Without loss of generality, we consider a sensor i with a message M at the top of its data queue. If it does not serve as either a sender or a receiver in the past L transmissions as discussed below, sensor i enters sleep mode for a period of T_i. The optimal value of T_i will be discussed later in Sec. 4. Upon waking up, sensor i goes through two phases for possible data transmission to nearby sensors, namely, asynchronous phase and synchronous phase, as elaborated below.

3.2.1 Asynchronous Phase

The asynchronous phase starts after the node wakes up from sleeping. Since no central control exists during this phase, all communication is contention-based. More specifically, sensor i turns on its radio and listens for a period of τ_i (see Fig. 1). If the channel is idle, it transmits a preamble to occupy the channel and inform its neighbors to prepare for

receiving its RTS (Request-To-Send) packet. If collision happens (i.e., receiving preamble from other nodes), it gives up its attempt for further transmission and restarts the asynchronous phase again.

After the preamble, sensor i sends an RTS packet that contains its nodal delivery probability (i.e., ξ_i), the FTD of message M (i.e., $\mathcal{F}_i^M$), and the length of the contention window (denoted by W). The contention window is the period to allow the neighboring sensors to reply.

Once a neighbor node receives the RTS packet, it checks if it can serve as a qualified receiver. A qualified receiver of node i must satisfy two conditions. First, it must have higher delivery probability than node i. Second, it should have available buffer space for messages with FTD of $\mathcal{F}_i^M$. Each of the qualified receivers then sends back a CTS (Clear-To-Send) packet, which includes its nodal delivery probability and available buffer space, to node i at a random time point during the period of W. Obviously, since there is no central control, the transmission of CTS packet is also contention-based. If more than one CTS arrive at node i at the same time, all of them are lost.

Note that, although the above RTS/CTS handshaking mechanism appears to be similar to that of the IEEE 802.11 protocol (which is the reason we use the notion of RTS/CTS), information delivered through RTS/CTS in these two protocols are different. In DFT-MSN, the two control packets exchange nodal delivery probability and available buffer space between a sender and its potential receivers, and they are crucial for the nodes to make efficient decisions on data transmission.

Node i keeps listening on the channel and collects the CTS packets, which are used to construct the neighbor table. After that, node i can easily make central arrangement for next data transmissions. Hereafter, the communication enters the synchronous phase. In addition, similar to IEEE 802.11 DCF function, the network allocation vector (NAV) mechanism can be employed to minimize overhearing and address the hidden station problem. Thus, the neighboring nodes of each receiver update their NAVs upon receiving the CTS. The details of updating NAV is omitted here.

In the above asynchronous phase, contention mainly happens in two situations. First, multiple nodes may try to grasp the channel by transmitting a preamble simultaneously. Second, multiple qualified neighbor nodes may reply with CTS packets simultaneously. For both situations, we have designed simple and effective schemes to address the tradeoff between the collision probability and bandwidth utilization, as to be discussed in Sec. 4.

3.2.2 Synchronous Phase

In this phase, all the data transmissions are synchronized, and thus contention-free. After obtaining information from
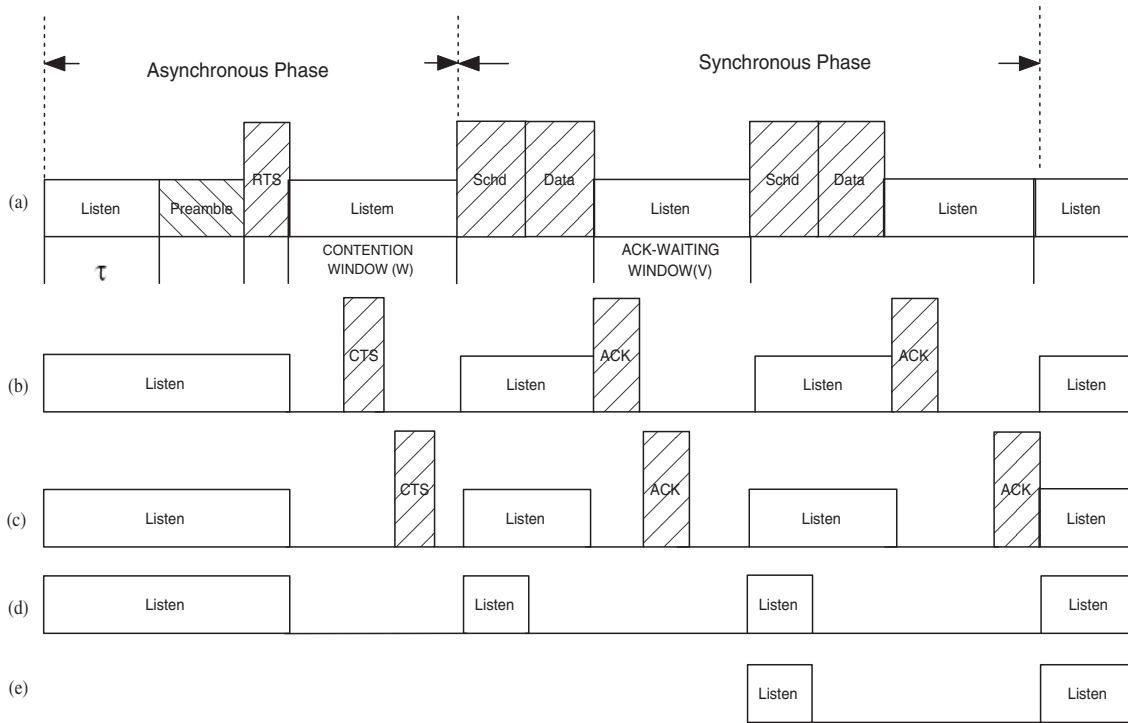
Figure 1. Proposed cross-layer data delivery protocol: (a) Sender; (b) and (c) Qualified neighbors; (d) Unqualified neighbors; (e) Possible new arrivals.

the qualified receivers, node i decides which of them are to be selected for data forwarding, according to the FTD of the outgoing message M and the receivers' delivery probabilities. Let $\Xi = \{\psi_z \mid 1 \le z \le Z\}$ denote the Z qualified receivers, sorted by a decreasing order of their delivery probabilities. In order to reduce unnecessary transmission overhead, only a subset Φ of them are selected for transmission so that the total delivery probability of message M is just enough to reach a predefined threshold $\Re$. The procedure for determining Φ is as follows:

$\Phi = \emptyset$.
for $z = 1 : Z$ **do**
  if $\xi_i < \xi_{\psi_z}$ AND $B_{\psi_z}(\mathcal{F}_i^M) > 0$ **then**
    $\Phi = \Phi \cup \psi_z$.
  end if
  if $1 - (1 - \mathcal{F}_i^M) \prod_{m \in \Phi}(1 - \xi_m) > \Re$ **then**
    Break.
  end if
end for

where $B_{\psi_z}(\mathcal{F}_i^M)$ is the available buffer space at node ψ_z for any message with FTD equal to $\mathcal{F}_i^M$, i.e., the total number of buffer slots in the queue of node ψ_z that are either empty or occupied by the messages with FTD greater than $\mathcal{F}_i^M$.

Node i then constructs and sends out a SCHEDULE packet which includes the list of IDs of the selected receivers and the corresponding FTD of the message copy to be received by each receiver (which is calculated according to Eq.(2)). The ID list also implicitly determines the order of ACK packets to be transmitted by the receivers. Following that, node i sends out the data message and waits for the acknowledgements from the receivers.

If a node j finds its ID in the SCHEDULE packet, it accepts the following message M, which is then inserted into its data queue and attached with the FTD indicated in the SCHEDULE packet. Then, node j replies with an ACK packet at a specific time slot according to the receiver list in the SCHEDULE packet. For example, if node j is the k-th receiver in the list, it transmits its ACK at $k \times t_{ack}$ after receiving the message, where t_{ack} is a constant period for the ACK transmission.

If any ACK packet is lost, node i assumes that the corresponding neighbor is invalid and removes it from Φ. After receiving the ACK packets, node i recalculates the FTD of its local copy of message M by using Eq. (3) and updates the data queue, as discussed in Sec. 3.1.

Each sensor repeats the above two-phase process as long as it is in the work mode.

4 Protocol Optimizations

Communication link and battery power are the two scarcest resources in DFT-MSN, directly dictating its data delivery performance. Given their very low nodal density, the mobile sensors are intermittently connected, thus calling for the needs of making the utmost use of the temporarily available communication links. A naive approach is to let sensors stay in the work mode continuously, so as to catch every possible opportunity for data transmission. Under such an approach, however, the sensors may drain off their battery quickly, resulting in poor performance of the overall network.

Clearly, there is a tradeoff between link utilization and energy efficiency. In this section, we address this tradeoff from the following three perspectives.

4.1 Periodic Sleeping

The radio transceiver can be in one of the four possible states: transmitting, receiving, listening, and sleeping, which correspond to four power levels. For short range wireless communication as in sensor networks, power consumed is about the same for listening to idle channels as for reception or transmission of useful data. Worse yet, due to the inherent nature of sparse connectivity of DFT-MSN, the sensor nodes are mostly in the idle listening state if they do not turn off their radios. Hence periodic sleeping is clearly necessary for prolonging the lifetime of individual sensors and accordingly the entire DFT-MSN. On the other hand, it is obvious that sleeping will degrade the link utilization, thus lowering the protocol performance in terms of the delivery ratio and delay.

In order to deal with this tradeoff, a simple and effective scheme is proposed to determine when and how long a node should go to sleep. As we have discussed in Sec. 3, a node i turns off its radio for a period of T_i if it does not act as either a sender or a receiver in the past L transmissions. The sleeping period T_i is determined by two factors. The first one is how likely the node can perform transmissions if the radio is turned on (i.e., it meets another node with a higher delivery probability). To facilitate our discussion, we introduce a parameter ρ_i defined as

$$\rho_i = \begin{cases} s_i/S, & s_i \neq 0 \\ 1/S, & s_i = 0, \end{cases} \quad (4)$$

where s_i is the number of working cycles in which node i has done transmission successfully in the past S consecutive cycles. If ρ_i is large, the sleeping period T_i ought to be shortened to enable more transmissions. On the other hand, if ρ_i is small, the node may employ a large T_i so as to reduce unnecessary power consumption.

The second one is related to node's available message buffer. When the message buffer is likely to be full, a short sleeping period should be used since the sensor needs to transmit whenever there is a chance in order to avoid dropping the important messages. To this end, we define

$$\alpha_i = \frac{K_i^{\mathcal{F}}}{K}, \quad (5)$$

where $K_i^{\mathcal{F}}$ is the number of messages with FTD smaller than $\mathcal{F}$, while K is total buffer space in terms of the number of messages.

Based on ρ_i and α_i, the sleeping period of node i, T_i, can then be calculated as follows:

$$T_i = Max(T_{min}, T_{min} \times \lceil \frac{1}{\rho_i} \times \frac{1}{(1 - H + \alpha_i)} \rceil), \quad (6)$$

where T_{min} is the minimum sleeping period and H is a predetermined threshold. If $\alpha_i \geq H$, the sleeping period is shortened; otherwise, a larger sleeping period is employed. Assume the power consumption of turning on/off the radio is P_{change}, while power consumption for the radio in the idle state and the sleep state equals P_{idle} and P_{sleep}, respectively. To ensure a net power saving, T_{min} should be

$$T_{min} \geq \frac{2P_{change}}{P_{idle} - P_{sleep}}. \quad (7)$$

In addition, since the minimum value of ρ_i is $\frac{1}{S}$, the maximum sleeping period, i.e., T_{max}, is

$$T_{max} = \frac{S}{1 - H} \times T_{min}. \quad (8)$$

4.2 Collision Avoidance During RTS Transmission

Collision happens when multiple neighboring nodes try to transmit simultaneously. As a result, the colliding messages are lost, and both energy and bandwidth consumed for the communication are then wasted. In the proposed protocol, collisions mainly happen in the asynchronous phase, where nodes contend for channel access, due to two reasons. First, multiple nodes may try to grasp the channel by transmitting their preambles simultaneously. Second, if a single preamble is successfully transmitted followed by an RTS, multiple qualified neighbor nodes may reply with CTS simultaneously, which again results in collisions. In this subsection, we discuss the approach for minimizing the collision probability of transmitting preambles and RTS, while the problem involved in CTS transmission will be discussed in the next subsection.

As discussed in Sec. 3, when a node i wakes up, it listens to the channel for a period of τ_i before initiating any

data transmission. A carefully designed collision avoidance scheme is needed here, in order to reduce the overall collision probability, while allowing the node with a lower delivery probability to have a better chance to grasp the channel. This is important for achieving high channel efficiency, because, once winning channel contention, the node with lower delivery probability is more likely to identify receivers, given a node always transmits data to other nodes with higher delivery probabilities.

Here, a simple and effective scheme is proposed for a node to select its listening period adaptively. Let τ_{max} denote the maximum listening period. Each time when a node i starts listening to the channel, it dynamically selects a listening period, which is uniformly distributed between 1 and σ_i. σ_i is determined by the following equation,

$$\sigma_i = \xi_i \times \tau_{max}. \qquad (9)$$

Next, the probability for a node to grasp the channel is estimated. For simplicity, we consider an independent cell, where all nodes inside can communicate with each other, but none of them can contact any node outside the cell. Since we are considering a sparse network, the interference from the nodes outside the cell can be ignored. Assume there are m nodes inside the cell and every node knows the delivery probability of others by referring to its neighbor table. The probability that an arbitrary node i can grasp the channel is approximated as

$$P_i = \sum_{\tau_i=1}^{\sigma_i} \frac{1}{\sigma_i} \times \prod_{j=1}^{m} \frac{\theta_{ij}}{\sigma_j}, \qquad (10)$$

where

$$\theta_{ij} = \begin{cases} \sigma_j - \tau_i, & \sigma_j > \tau_i, \\ 0, & \sigma_j \le \tau_i. \end{cases} \qquad (11)$$

Thus, the probability that no one can grasp the channel (probability of collisions), i.e., γ, is

$$\gamma = 1 - \sum_{i=1}^{m} P_i. \qquad (12)$$

Obviously, the larger τ_{max} is, the less likely the collision will happen. However, with a large τ_{max}, the sensor nodes need to listen for a long time, resulting in low channel efficiency and high energy consumption. Our goal is to find a minimum τ_{max} which keeps the collision probability under a predefined threshold H, i.e.,

$$\min(\tau_{max} | \gamma \le H). \qquad (13)$$

This can be done by individual sensors using typical optimization schemes.

4.3 Collision Avoidance During CTS Transmission

Each RTS packet contains a field of contention window to indicate the period during which the sender waits for CTS packets. All qualified neighbors should reply with CTS packets within this period. If multiple nodes answer simultaneously, collision happens and all colliding CTS packets are lost.

In order to reduce the collision probability, the contention window is split into a number of small time slots. Each slot equals the time to transmit a CTS packet by the receiver, plus the time for the sender to process the CTS packet. While it is desirable to have each qualified neighbor reply CTS in a unique time slot, this, however, is not realistic since the nodes are not synchronized at this moment. An intuitive approach is to let a qualified receiver randomly select a time slot. In this simple approach, the length of the contention window, i.e., W, is a crucial parameter. More qualified neighboring nodes need a larger contention window.

With these consideration in minde, we propose a simple schemes to determine the optimal value of contention window W, in order to achieve the desired collision probability and at the same time minimize the signaling overhead. Assume node i has sent RTS successfully to a set of its neighbors, denoted by ϕ_i. Let $|\phi_i|$ be the number of nodes in ϕ_i.

Let every qualified neighbor node randomly select a time slot between 1 and W for sending its CTS packet. Then, the probability that every CTS is transmitted in a collision-free unique time slot is $\binom{W}{|\phi_i|} \times |\phi_i|! \times (\frac{1}{W})^{|\phi_i|}$, and accordingly the overall collision probability equals

$$\gamma_o = 1 - \binom{W}{|\phi_i|} \times |\phi_i|! \times (\frac{1}{W})^{|\phi_i|}. \qquad (14)$$

A minimum W can thus be chosen by a linear search, to ensure γ_o lower than an expected value.

5 Simulation Results

Extensive simulations have been carried out to evaluate the performance of the proposed cross-layer data delivery protocol for DFT-MSN. In the default simulation setup, 3 sink nodes and 100 sensor nodes are randomly scattered in an area of $200 \times 200 \ m^2$. The whole area is divided into 25 non-overlapped zones, each with an area of $40 \times 40 \ m^2$. A sensor node is initially resided in its home zone. It moves with a speed randomly chosen between 0 and 5 m/s. Whenever a node reaches the boundary of its zone, it moves out with a probability of 20%, and bounces back with a probability of 80%. After entering a new zone, the sensor repeats the above process. However, if it reaches the boundary to its

home zone, it returns to its home zone with a probability of 100%. Each sensor has a maximum transmission range of 10 m and a maximum queue size of 200 messages. The data generation of each sensor follows a Poisson process with an average arrival interval of 120 s. Each data message has 1000 bits, while each control packet has 50 bits. The channel bandwidth is 10 $kbps$. The power consumption is estimated according to the transceiver used in Berkeley mote, which consumes 13.5mW, 24.75mW and 15μW, in receiving, transmitting and sleep, respectively [15]. The power consumption of idle listening is the same as that of receiving, while the power consumption of turning on/off the radio is four times of listening power consumption. Each simulation lasts 25000s. For a given simulation setup, we run the simulation multiple times and average the collected results.

In order to evaluate the performance of the proposed data delivery scheme and its optimizations, we have implemented four protocols with different optimization levels, dubbed *OPT*, *NOOPT*, *NOSLEEP*, and *ZBR*, respectively. *OPT* is the proposed protocol that employs all optimization schemes (discussed in Sec. 4 for the protocol parameters τ_{max}, W, and T_i). *NOOPT* adopts the basic protocol proposed in Sec. 3, but without optimization. Instead, fixed protocol parameters are employed. *NOSLEEP* is similar to *OPT*, except that the nodes do not perform periodic sleeping. Finally, *ZBR* differs from *OPT* only in the message transmission scheme, by replacing the message fault tolerance-based multicast scheme with the ZebraNet's history-based scheme [12], which is the most comparable scheme in the literature. Note that the data delivery scheme proposed in SWIM [13] is not simulated here, because it is designed for the network with uniform nodal mobility, and thus work ineffectively in our simulation scenarios where different sensor nodes have different delivery probabilities. Other protocols developed for mobile ad hoc networks (such as IEEE 802.11 and its enhanced versions) or conventional sensor networks (e.g., [2–4]) are not considered either, because their performance level are expected to be far lower than the proposed approach in DFT-MSN, although each of them may have been optimized for their target application scenarios.

We vary several parameters to observe their impacts on performance. Fig. 2 presents the protocol performance by varying the number of sink nodes in DFT-MSN. As shown in Fig. 2(a), with more sink nodes present, the sensors exhibit a better opportunity to reach the sink nodes, thus resulting in a higher delivery ratio. *OPT* and *NOSLEEP* outperform *NOOPT* slightly since they both employ optimized parameters to reduce the collision probability. As expected, *ZBR* has the lowest delivery ratio, especially when there are only a few sink nodes, because it employs the direct contact probability to decide message transmission. For the nodes

that never directly meet the sink nodes, the transmission becomes random, and thus less efficient.

Fig. 2(b) compares the average nodal energy consumption rate of different implementations. Obviously, with more sink nodes existing, the message can be transmitted with fewer hops, reducing energy consumption. As expected, *OPT* conserves energy effectively compared to the other approaches. Since *NOOPT* does not optimize the protocol parameters, we observe many collisions during RTS/CTS transmissions, which accordingly waste battery power. Energy consumed by *NOSLEEP* is very high (about eight times of the energy consumption of *OPT*), due to its idle listening. On the other hand, because of its inefficient transmission control, *ZBR* has higher overhead than *OPT*, although the same optimized parameters are employed.

The average message delay is compared in Fig. 2(c). The delay decreases sharply with an increase in the number of sink nodes, since the messages can then be delivered to the sinks with fewer hops. *NOSLEEP* has a shorter message delay than *OPT* and *NOOPT*, because the nodes do not sleep at all so that they can capture every possible transmission chance, resulting in faster delivery. Surprisingly, *ZBR* also has a very low delay. This delay, however, is for successfully delivered data messages only. We have observed that in *ZBR*, most of the messages delivered successfully are those generated by nodes near the sinks, thus resulting in less transmission hops and accordingly a shorter message delivery delay. Clearly, it is not meaningful to compare it with other approaches in terms of the delay metric.

Although the results are not shown here, We also investigate the impacts of node density and nodal moving speed. In all the simulations, *OPT* achieves a high delivery ratio with minimum energy consumption. When node density increases, the nodes near the sinks will carry much more messages and become the bottlenecks. Due to the limited bandwidth and buffer size, many messages are to be dropped, resulting in lower delivery ratio. On the contrary, as the nodal speed increases, the delivery ratios of all approaches rise, while the delivery delays of all approaches decrease. This is because the node with a higher speed has a better opportunity to meet other nodes and to reach the sink nodes. Thus, the messages enjoy a better chance to be delivered before they are dropped. It is also noticed that the transmission overhead of *OPT* decreases with the increase of the nodal speed, making it adaptable for networks with various nodal speeds.

In short, our simulation results demonstrate the effectiveness of the proposed cross-layer data delivery protocol with the optimization schemes, which achieves the highest data delivery ratio and the lowest energy consumption, with only marginally increased delay overhead.
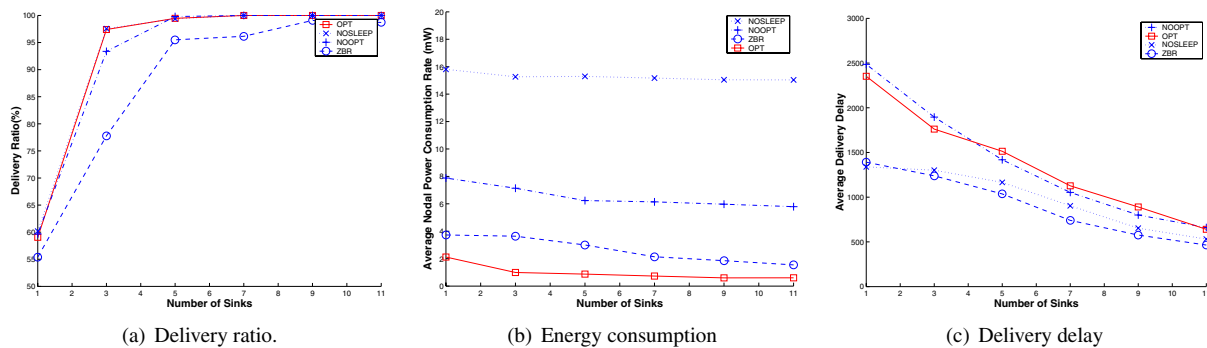
(a) Delivery ratio.　　　(b) Energy consumption　　　(c) Delivery delay

Figure 2. Impact of number of sink nodes.

6 Conclusion

This paper studies efficient data delivery in *Delay/Fault-Tolerant Mobile Sensor Networks (DFT-MSN's)*. DFT-MSN is fundamentally an opportunistic network, where the communication links exist only with certain probabilities, and thus are the most crucial resource. Without end-to-end connections, routing in DFT-MSN becomes localized and ties closely to the medium access control, naturally calling for merging layer-3 and layer-2 protocols in order to reduce overhead and improve network efficiency. To this end, we have proposed a cross-layer data delivery protocol, which consists of two phases, i.e., the asynchronous phase and the synchronous phase. In the first phase, the sender contacts its neighbors to identify a set of appropriate receivers. Since no central control exists, the communication in the first phase is contention-based. In the second phase, the sender gains channel control and multicasts its data messages to the receivers. Furthermore, we have identified several optimization issues, with solutions provided to reduce the collision probability, and to balance between link utilization and energy efficiency. Extensive simulations have been carried out for performance evaluation. Our results have demonstrated that the proposed cross-layer data delivery protocol for DFT-MSN achieves a high message delivery ratio with low energy consumption and an acceptable delay.

References

[1] Y. Wang, F. Lin, and H. Wu, "Poster: Efficient Data Transmission in Delay Fault Tolerant Mobile Sensor Networks (DFT-MSN)," in *IEEE International Conference on Network Protocols (ICNP) for poster presentation*, 2005.

[2] W. Choi, P. Shah, and S. Das, "A framework for energy-saving data gathering using two-phase clustering in wireless sensor networks," in *Proc. of The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS)*, pp. 203–212, 2004.

[3] M. Sichitiu, "Cross-layer scheduling for power efficiency in wireless sensor networks," in *Proc. of Twenty-Third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, vol. 3, pp. 1740–1750, 2004.

[4] M. Zorzi and R. R. Rao, "Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Multihop Performance," *IEEE Transactions on Mobile Computing*, vol. 2, no. 4, pp. 337–348, 2003.

[5] Y. Wang and H. Wu, "DFT-MSN: The Delay Fault Tolerant Mobile Sensor Network for Pervasive Information Gathering," in *Proc. of Twenty-Fifth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2006.

[6] http://www.cens.ucla.edu/.

[7] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring," in *Proc. of ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, pp. 88–97, 2002.

[8] http://down.dsg.cs.tcd.ie/sendt/.

[9] http://www.sics.se/cna/dtnsn/index.html.

[10] M. Ho and K. Fall, "Poster: Delay Tolerant Networking for Sensor Networks," in *Proc. of IEEE Conference on Sensor and Ad Hoc Communications and Networks*, 2004.

[11] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a Three-Tier Architecture for Sparse Sensor Networks," in *Proc. of the First International Workshop on Sensor Network Protocols and Applications*, pp. 30–41, 2003.

[12] http://www.princeton.edu/ mrm/zebranet.html.

[13] T. Small and Z. J. Haas, "The Shared Wireless Infostation Model A New Ad Hoc Networking Paradigm (or Where there is a Whale, there is a Way)," in *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, 2003.

[14] T. Small and Z. J. Haas, "Resource and Performance Trade-offs in Delay-Tolerant Wireless Networks," in *Proc. of ACM SIGCOMM05 Workshop on Delay Tolerant Networking and Related Topics (WDTN)*, 2005.

[15] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, vol. 3, pp. 1567–1576, 2002.