Analysis of A Loss-Resilient Proactive Data Transmission Protocol in Wireless Sensor Networks

Yingqi Xu*, Wang-Chien Lee*
Pennsylvania State University, USA.
{yixu, wlee}@cse.psu.edu

Jianliang Xu†
Hong Kong Baptist University, Hong Kong.
xujl@comp.hkbu.edu.hk

***Abstract*— Many of sensor network applications require reliable data communication such that data packets can be delivered to the destination without loss. However, existing reliable transmission techniques either are too costly for resource-constrained sensor networks or have limited capabilities for achieving desirable reliability. In this paper, an effective coding scheme that exploits the tradeoff between redundant data transmission and encoding/decoding complexity is proposed, with an in-depth study on two key design parameters, the *degree of repair packets* and the *number of repair packets*. Furthermore, the expected probability of a destination obtaining all data packets under recoverable and permanent failure model for proactive transmission is analyzed, respectively. Simulations have been conducted to verify our theoretical results. The simulation results reveal profound insights in achieving high communication reliability in wireless sensor networks.**

I. Introduction

Among a wide range of networked sensing applications, many of them (e.g., disaster forecast, structural condition assessment) require reliable data communication, such that a target destination can obtain all data packets with a high probability. However, sensor networks are unreliable in nature, and sensor nodes are constrained by energy, computation power and storage. Existing reliable transmission techniques designed for Internet and ad hoc networks are not effective (if not infeasible) for sensor networks. In this paper, we study reliable communication techniques for wireless sensor networks.

There are two categories of approaches for improving communication reliability, i.e., *reactive retransmission* and *proactive transmission*. In reactive retransmissions [5], [10], [19], [21], the source node is notified to retransmit a lost packet until all data packets are correctly received at the destination. However, retransmission is triggered by packet losses, which prolongs the communication delay and significantly incurs the network traffic (due to NACK/ACK messages and data retransmissions). To remedy these deficiencies, proactive approaches have been recently proposed [7], [27], in which by transmitting redundant coded packets to the destination, lost packets can be reconstructed as long as the destination receives a sufficient number of redundant coded packets.

Redundant coding (or *coding* for short) is crucial for proactive reliable transmissions and has been studied in various fields [2], [18], [20], [24]. However, with different performance requirement, the existing coding schemes are not feasible for wireless sensor networks due to either their very high computation overhead or very high coding redundancy. In this paper, inspired by existing coding schemes [17], [18], [25], we craft a coding scheme for wireless sensor networks by taking into consideration its scarce bandwidth, limited energy, and constrained computation and storage capacity.

To optimize the performance of the proposed coding scheme in wireless sensor networks, we focus on exploring the *tradeoff* between the coding computation complexity and coding redundancy, which are controlled by two critical parameters, the *degree of repair packets (γ)* and the *number of repair packets (k)*, respectively. Moreover, we examine the proposed coding scheme under two different network failure models, i.e., *recoverable failure model* and *permanent failure model*. The proposed schemes significantly increase the probability of a destination obtaining all data packets. Their computation complexity and communication overhead are shown to be suitable for sensor networks. Our contributions can be summarized as follows:

- We develop an efficient coding scheme for use in proactive transmissions in sensor networks. Parameters critical to our coding scheme, i.e., *the degree of repair packets* and the *number of repair packets*, are mathematically derived to minimize the communication cost and the computation complexity, while ensuring that the lost packets can be recovered with a high probability.
- We mathematically analyze the recoverability of our coding scheme. The analysis provides a guidance for trading off the reliability with the communication overhead. The computation complexity of the coding scheme is examined and compared with other representative coding schemes.
- We analyze the expected performance of proposed techniques under recoverable and permanent failure model, respectively.
- We conduct simulations to validate our theoretical analysis. The simulation results show that proposed schemes enable the destination to obtain all data packet with

*Wang-Chien Lee and Yingqi Xu were supported in part by National Science Foundation grant CNS-0626709 and IIS-0328881.

†Jianliang Xu's work was supported by grants from the Research Grants Council, Hong Kong SAR, China (Projects No. HKBU 211505 and FRG/05-06/II-65).

a higher probability, yet incurring less communication overhead than existing approaches.

The remainder of the paper proceeds as follows. The related work is reviewed in Section II. The detailed designs of the proposed coding scheme and the proactive reliable transmission techniques are presented in Section III and Section IV, respectively. A performance evaluation is provided in Section V. Finally, we conclude our work and discuss future work in Section VI.

II. Related Work

Coding schemes have been widely adapted for fault tolerant computing in various forms of digital data communication. For reliable data communication, a source node, based on a coding scheme, encodes the data packets into *repair packets*, from which a destination is able to recover the lost packets. Thus, coding schemes improve the reliability at the cost of transmitting additional repair packets. In the following, we briefly examine several representative coding schemes.

Forward Error Correction (FEC) [2] is a type of error control code that uses redundancy (extra information) to detect and correct errors caused by the *noisy* channel in a communication system. The two main categories of FEC are *block codes* and *convolutional codes*. There are many types of block codes, including Hamming code [12], BCH code [24] and Reed Solomon code (RS) [25]. The most widely used by far is RS code due to its nearly *optimal* ability of error correction. Generally, RS encodes the block's message as points in a polynomial plotted over a finite field. The coefficients of the polynomial are the data symbols of the block. Convolutional code [8], [9], [28] is another type of error-correcting code in which an n-bit message to be encoded is transformed into an m-bit symbol, where m/n is the code rate ($m \geq n$). The encoding is a function of the last k information symbols. To ensure the error correction ability, convolutional code requires at least a constraint length k of 7 (usually less than 9) and a code rate m/n of 1/2. Compared with RS code, convolutional code has much less computation complexity, but incur higher coding redundancy. The same problem exists in the Turbo code [1]. Moreover, Hamming code, BCH code and convolutional code, focusing on correcting errors in bit-level, usually are not used for recovering lost data packets (which is the focus of our study), due to its quickly increasing requirement of memory space for encoding and decoding with the increasing amount of information.

There is another class of coding schemes, usually called *erasure code*, were designed for information recovery in packet-level (i.e., to recover lost data packets). RS code can be employed as erasure code [26]. More specifically, each encoded packet (i.e., repair packet in the paper) is generated from a polynomial calculation over n data packets ($d_1, d_2, \cdots, d_n$), i.e., $F(\alpha) = d_0 + d_1\alpha + \cdots + d_n\alpha^{n-1}$. The repair packets are $\{F(1), F(\alpha), F(\alpha^2), \cdots\}$. By solving a set of *linearly independent* equations, which are represented by the repair packets, the destination is able to recover the lost packets. In RS code, α is usually large to minimize the number of repair packets needed (i.e., only m repair packets are needed for recovery of m lost packets in the optimal case). Moreover, its encoding and decoding is performed in time $\theta(n^2)$ and $\theta(n^3)$, respectively and the requirement for memory space is noticeable due to the polynomial operations.

Tornado code [18] is another type of erasure code. By using simple XOR operations for encoding and decoding, it significantly reduces the encoding and decoding complexity at the cost of transmitting more encoded packets (i.e., $n/(1-\beta)$, where β is usually $1/2$). However, Tornado codes require a *prior construction* of a cascading sequence of bipartite graphs between several layers of packets at encoder and decoder, which is cumbersome in practice [17]. To solve the above problem, Luby Transform (LT) code generates each encoded packet by applying XOR operations over γ original data packets, where the γ ($1 \leq \gamma \leq n$) is randomly selected from some distribution. By successively sending the encoded packets, LT code is able to decode all data packets as long as an enough number of packets are received (regardless of their order). For decoding, all encoded packets with $\gamma = 1$ are first decoded, since they are data packets themselves. At each subsequent step, a randomly selected, already decoded data packet is removed from all encoded packets that have this packet encoded. This process stops when all original data packets are decoded from the encoded packets. LT code performs encoding and decoding with a much lower computation complexity than RS code, which is very attractive for resource-constrained wireless sensor networks. However, LT code ensures the decoding success at the cost of a large number of encoded packets (i.e., n original data packets can be decoded from $n + O(\sqrt{n}\ ln^2(n/\delta)$ encoded packets with a probability $1-\delta$), which is an adversary for performing energy optimization in wireless sensor networks. Moreover, LT code does not take into consideration the condition of the communication path (i.e., the expected packet losses), which however could be used to optimize the coding performance in terms of the computation complexity and the cost of transmitting the encoded packets. By considering the expected lost rate of the communication path, we take a different angle from LT code to analyze the degree of repair packets (γ) and the number of repair packets (k). Our designs aim at minimizing both communication cost and computation complexity to achieve a satisfactory communication reliability.

III. Design of a Coding Scheme

In this section, we discuss the detailed design of our proposed coding scheme. The basic idea of our proposal is presented in Section III-A. The mathematical analysis of the tradeoff between the *degree of repair packets* and the *number of repair packets* is presented in Section III-B. Section III-C derives the expected performance of proposed coding scheme. Finally, Section III-D analyzes the complexity of the proposed coding schemes.

A. Encoding and Decoding

Without causing confusion, we call the encoded packets as repair packets that are used for recovering the data packets

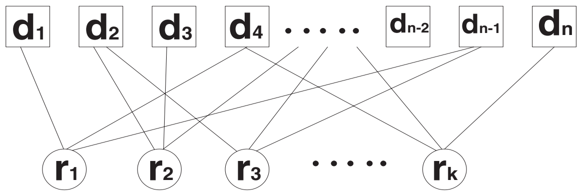


Fig. 1. Encoding $\gamma = 3$

lost during the transmission. The process of a source node generating a repair packet is conceptually very easy to describe. We define the degree of a repair packet as the number of data packets used to generate a repair packet, denoted by γ. Encoding involves the following two steps:

- randomly choose γ distinct original data packets, which are called *inputs* of the repair packet;
- generate a repair packet by XOR($\oplus$) its inputs

Figure 1 illustrates the encoding process, where each repair packet (denoted by r_i) is produced from $\gamma = 3$ randomly chosen data packets out of $d_1, ..., d_n$. Our coding scheme, similar to some existing schemes [3], [17], employs XOR operations such that the computation complexity for both encoding and decoding (described below) is minimized. The source node generates k repair packets, and sends them along with n data packets together to the destination. We assume the source node and the destination employ the same pseudo random number generator (e.g., linear congruential generator). Thus, the source only need to send the *seed* of the random generator with the repair packet. The seed together with packet sequence id is used for generating/re-generating the ids of data packets (i.e., inputs) used for encoding a repair packet.

The decoding process consists of two steps. The first step is similar to the decoding process used by LT code [17]. More specifically, the destination node divides the received packets into three sets: *unprocessed* set, *processed* set, and repair packet set. Initially, all received data packets (excluding repair packets) are in the unprocessed set and the processed set is empty. At each subsequent process, the destination randomly picks a data packet d_i from the unprocessed set and scans the repair packet set. d_i is removed from the repair packet j that encodes d_i and γ_j decreases by 1. When $\gamma_j = 1$, this repair packet is completely decoded and moved to the unprocessed set. After scanning all repair packets, d_i is moved to the processed set. The above procedure stops when the unprocessed set is empty, or all lost data packets are recovered. [17] has shown that to recover all packets, a large number of repair packets are needed. However, this is against our goal of designing a low-overhead technique. Therefore, we design the second decoding step, which takes place if the first step stops without recovering all data packets, but leaving some of repair packets un-decoded.

In the second step, the destination collects the remaining repair packets that have not been completely decoded by the first step, and views each repair packet as an equation with a number of variables (i.e., the lost packets not recovered yet). Therefore, decoding the remaining repair packets is a process of solving a set of equations and the solutions are the lost packets. In fact, this step can also be independently used for decoding without the first step. However, there are two reasons for us not to do so. First, the complexity of solving a set of equations is a quadratic growth in the number of equations, which is much more significant than the decoding complexity of the first step. Moreover, solely using this method, *none* of the lost packets can be recovered if not enough number of repair packets are received. In contrast, with the first decoding step, the destination is still likely to recover some of lost packets, even when the number of repair packets received is less than the number of data packets lost. Therefore, our design takes advantages of both schemes by incorporating them into a two-step decoding process.

Comparing with RS code, our coding scheme, involving XOR operations only, is obviously more attractive for sensor nodes which have constrained computation and storage capabilities. However, this simplicity raises critical research challenges on other performance aspects, i.e., the robustness and the communication cost. In this coding scheme, γ and k are critical parameters for the proposed coding scheme. Determining the number of data packets encoded into each repair packet, γ has an important impact on the ability of recovering the lost packets and the computation complexity of the encoding and decoding process. On the other hand, k, representing the coding redundancy, also affects the robustness of the coding scheme. Therefore, with a satisfactory probability of recovering lost data packets, we aim at deriving the value of k such that the communication cost is minimized, while keeping the computation complexity as low as possible as well. In the following, by deriving the key design parameter γ and k of proposed coding scheme, analyzing its expected performance, and examining its computation complexities, we carefully examine our design choices.

B. Analysis of γ and k

A necessary condition for recovery of all lost data packets is that the repair packets received at the destination have each lost data packet encoded at least once. Otherwise, the recovery has no way to succeed. The degree of repair packets γ is a key design parameter for this condition. As one can expect that if γ is small, each repair packet encodes a small number of data packets, which leads to a low probability that a lost packet is covered by the repair packets. Thus, more repair packets, i.e., larger k (reflecting the communication overhead) is needed to achieve a high recovery probability. On the other hand, if γ is very large, the computation complexity of both encoding and decoding processes increases at least linearly. More importantly, it becomes harder to decode all repair packets. Considering the first decoding step with a large γ, it is less likely that γ reduces to 1 after removing all received data packets from the repair packets, which results in a decoding failure. For the second decoding step, when γ is very large, it becomes more difficult to form *linearly independent* equations. The impact of γ on the coding performance will be further studied by simulation experiments in Section V-A. In the following, we derive the expected number of repair packets required for recovering lost data packets and an γ, such that the number of repair packets is minimized without increasing the computation complexity or jeopardizing the effectiveness of the decoding process.

Since a source node, at the time of encoding, does not have the knowledge about which data packets would be lost during transmission, the above requirement of covering all lost data packets by the repair packets becomes that each data packet has to be encoded (or covered) by at least one repair packet. The question is how many repair packets are needed to satisfy this requirement. This is similar to the classical *balls and bins process* [14], which states that a number of balls are thrown to a collection of n bins, and each ball goes into a random bin. In order to have at least one ball in each bin, how many balls are needed. Considering our problem, the n bins are analogy to the n data packets, and each repair packet represents γ balls. Let X_i be the number of balls that are thrown such that a new bin is hit given $i-1$ bins already contain balls. After $i-1$ bins contain balls, a new ball has a $\frac{i-1}{n}$ chance of hitting the $i-1$ bins, and a $1-\frac{i-1}{n}$ chance of hitting a new bin. Thus, X_i follows a Geometric distribution, $X_i \sim$ Geometric $(1-\frac{i-1}{n})$. Let $\mathbf{X}$ denote the number of balls thrown before all n bins are non-empty, $\mathbf{X} = X_1 + X_2 + ... X_n$. Given $\mathbb{E}(X_i) = \frac{1}{1-(i-1)/n}$, we obtain $\mathbb{E}(\mathbf{X}) \approx n \ln n$. Since k denotes the number of repair packets collected and each repair packet represents γ balls, we must have

$$k \cdot \gamma \geq n \ln n \quad (1)$$

We now analyze the expected number of repair packets ($\mathbb{E}(k)$) needed to recover m lost packets. To simplify our discussion, we does not consider the impact of the first decoding step since it only helps to reduce decoding complexity. Hence, the problem now becomes solving m variables from k equations. Based on the encoding process discussed in Section III-A, all packets, including data packets and repair packets generated by the source node, have the following matrix relation:

$$\begin{bmatrix} I_n \\ ... \\ a_1 \\ a_2 \\ ... \\ a_k \end{bmatrix} \bullet \begin{bmatrix} d_1 \\ d_2 \\ ... \\ d_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ ... \\ d_n \\ a_{11}d_1 \oplus a_{12}d_2 \oplus ... \oplus d_{1n}d_n \\ a_{21}d_1 \oplus a_{22}d_2 \oplus ... \oplus a_{2n}d_n \\ ... \\ a_{k1}d_1 \oplus a_{k2}d_2 \oplus ... \oplus a_{kn}d_n \end{bmatrix},$$

where we use XOR summation and I_n is a diagonal matrix with all diagonal elements equal to 1. Each vector a_i, formed randomly by the source node, consists of a random combination of 0's and 1's (i.e., $a_{ij} = 0$ or 1), such that:

$$\sum_{j=1}^{j=n} a_{ij} = \gamma, \text{where } 1 \leq j \leq n \text{ and } \ 1 \leq i \leq k$$

For the matrix at the right-hand side of the equation, the first n rows are the original data packets and are always sent as they are, and the remaining rows are the repair packets, which are also sent to the destination for possible packet recovery.

To solve m variables from k equations[1], at least m linearly independent equations are needed. Without loss of generality, we assume the m lost packets are $d_1, \cdots, d_m$. Thus, the packets $d_{m+1}, \cdots, d_n$ are correctly received. Let $r_1, \cdots, r_k$ be the k repair packets received. Thus, the destination node can obtain the following k equations from k repair packets:

$$\begin{cases} a_{11}d_1 \oplus a_{12}d_2 \oplus ... \oplus a_{1m}d_m = r_1 \oplus \sum_{i=m+1}^{\oplus\, n} a_{1i}d_i \\ a_{21}d_1 \oplus a_{22}d_2 \oplus ... \oplus a_{2m}d_m = r_2 \oplus \sum_{i=m+1}^{\oplus\, n} a_{2i}d_i \\ ... \\ a_{km}d_1 \oplus a_{k2}d_2 \oplus ... \oplus a_{km}d_m = r_k \oplus \sum_{i=m+1}^{\oplus\, n} a_{ki}d_i \end{cases}$$

where $\sum_i^{\oplus}$ denotes XOR summation. We define a *repair matrix* $A(l, h)$,

$$A(l,h) = \begin{pmatrix} a_{11}, a_{12}, \ ..., \ a_{1m} \\ a_{21}, a_{22}, \ ..., \ a_{2m} \\ ... \\ a_{l1}, a_{l2}, \ ..., \ a_{lm} \end{pmatrix} = \begin{pmatrix} a_1^{(m)} \\ a_2^{(m)} \\ ... \\ a_l^{(m)} \end{pmatrix},$$

where l denotes the number of rows in a repair matrix, and h denotes the matrix rank (i.e., the number of linearly independent rows). The destination node is able to build a repair matrix based on received repair packets. When its repair matrix $A(l, h)$ satisfies that $h = m$, all m lost packets can be recovered and $l = k$ is the number of repair packets needed. We analyze $\mathbb{E}(k)$ in the following.

Initially, the destination has an empty repair matrix $A(0, 0)$. Adding $a_1^{(m)}$ (obtained from the first repair packet) to repair matrix $A(0, 0)$, the destination gets $A(1, 1)$, if and only if $\sum_{i=1}^{m} a_{1i} \geq 1$.

The probability of $A(0, 0)$ transiting to $A(1, 1)$ after adding $a_1^{(m)}$ is $1 - \binom{n-m}{\gamma} / \binom{n}{\gamma}$, and the probability of $A(0, 0)$ transiting to $A(1, 0)$ is $\binom{n-m}{\gamma} / \binom{n}{\gamma}$ [2]. Given $A(1, 1)$, a newly added m-tuple $a_2^{(m)}$ is able to increase the rank of repair matrix (i.e., $A(2, 2)$), if and only if the new tuple is not all 0's and not equal to $a_1^{(m)}$. More generally, given a repair matrix A(l, h), the matrix rank is increased by one after adding a new m-tuple, if and only if the newly added tuple is not the linear combination of the previous l rows. In other words, $A(l, h)$ becomes $A(l + 1, h + 1)$ when

$$c_1\, a_1^{(m)} + c_2\, a_2^{(m)} + ... + c_h\, a_h^{(m)} \neq a_{l+1}^{(m)},$$

where $c_1, c_2, \cdots, c_h$ are constant values equal to 0 or 1 owing to its binary linear combination. There are totally 2^h cases that adding a_{l+1} does not increase $A(h, l)$'s rank. Thus, the probability that adding a m-tuple to $A(l, h)$ results in $A(l + 1, h + 1)$ is $1 - \left[\binom{n-m}{\gamma} + 2^h - 1\right] / \left[\binom{n}{\gamma}\right]$.

Let $S(i)$ denote the expected number of m-tuples needed for reaching rank m from state i, we obtain the following equations:

[1]Here, we assume all repair packets (k) are successfully received by the destination. The case where repair packets are lost during transmission is studied shortly in Section IV.

[2]For simplicity, we set $\binom{a}{b} = 0$, when $a < b$.

$$\begin{cases} S(0) = \frac{\binom{n-m}{\gamma}}{\binom{n}{\gamma}}(S(0)+1) + \Big(1 - \frac{\binom{n-m}{\gamma}}{\binom{n}{\gamma}}\Big)(S(1)+1) \\ S(1) = \frac{\binom{n-m}{\gamma}+2^1-1}{\binom{n}{\gamma}}(S(1)+1) + \Big(1 - \frac{\binom{n-m}{\gamma}+2^1-1}{\binom{n}{\gamma}}\Big)(S(2)+1) \\ \qquad \dots \\ S(i) = \frac{\binom{n-m}{\gamma}+2^i-1}{\binom{n}{\gamma}}(S(i)+1) + \Big(1 - \frac{\binom{n-m}{\gamma}+2^i-1}{\binom{n}{\gamma}}\Big)(S(i+1)+1) \end{cases}$$

By recursively applying the above equations, we obtain the average number of repair packets for recovering m lost packets, given the degree of a repair packet γ as follows

$$k(\gamma) = S(0) = \left\lceil m + \sum_{i=1}^{m} \frac{\binom{n-m}{\gamma} + 2^i - 1}{\binom{n}{\gamma} - \Big(\binom{n-m}{\gamma} + 2^i - 1\Big)} \right\rceil \qquad (2)$$

Given Equation (1), the average degree for each repair packet $\mathbb{E}(\gamma)$ and the average number of repair packets $\mathbb{E}(k)$ are

$$(\mathbb{E}(k),\, \mathbb{E}(\gamma)) = \Big\{ \, (\lceil k(\gamma) \rceil,\, \lceil \gamma \rceil) \;\Big|\; k(\gamma) \cdot \gamma = n \ln n \Big\}$$

where $1 \leq \gamma < n$. The above $\mathbb{E}(k)$ and $\mathbb{E}(\gamma)$ are derived by aiming at minimizing the number of repair packets needed for recovering all lost packets with the minimum computation complexity, such that the data communication reliability is achieved at the minimum cost. In the next section, we further analyze and verify our results.

C. Analysis of the recoverability

As we pointed out, to fully recover m lost data packets, two requirements have to meet. First, all data packets have to be encoded by at least one repair packet. Second, given the second decoding step, at least m linearly independent equations are formed out of k repair packets. In this section, we deepen our study and investigate the probability of satisfying the above two requirements given γ and k.

First, we consider the probability of m lost data packets being encoded by at least one repair packet given k repair packets. Let B_i be the event that a lost data packet i ($1 \leq i \leq m$) is not covered by any repair packet,

$$\mathbb{P}\{B_i\} = \Big(1 - \frac{\gamma}{n}\Big)^k$$

Thus, the probability of all m lost packets being covered by at least one repair packet is

$$\mathbb{P}\left\{\bigcup_{i=1}^{m} B_i^C\right\} = \Big(1 - \mathbb{P}\{B_i\}\Big)^m = \Big(1 - \Big(1 - \frac{\gamma}{n}\Big)^k\Big)^m$$

Now we study the second requirement: given k repair packets, what is the probability of all m lost packets being recovered? We denote this probability by $\Upsilon(k,m)$. To simplify the analysis, we assume the rank transition of repair matrix $A(l,h)$ follows the state transitions. Let $P(m,m)$ denote the probability that m repair packets can recover m lost packets, $P(m,m)$ (equal to $\Upsilon(m,m)$), is given by:

$$P(m,m) = \Big(1 - \frac{\binom{n-m}{\gamma}}{\binom{n}{\gamma}}\Big) \cdot \Big(1 - \frac{\binom{n-m}{\gamma} + 2^1 - 1}{\binom{n}{\gamma}}\Big) \cdot \dots \cdot \Big(1 - \frac{\binom{n-m}{\gamma} + 2^{m-1} - 1}{\binom{n}{\gamma}}\Big)$$

Similarly, let $P(m+1,m)$ denote the probability that *exact* $m+1$ repair packets can recover m lost packets. $P(m+1,m)$ is derived as:

$$P(m+1,m) = P(m,m)\Big(\frac{\binom{n-m}{\gamma}}{\binom{n}{\gamma}} + \frac{\binom{n-m}{\gamma} + 2^1 - 1}{\binom{n}{\gamma}} + \dots + \frac{\binom{n-m}{\gamma} + 2^{m-1} - 1}{\binom{n}{\gamma}}\Big)$$

Denoted by $\Upsilon(m+1,m)$, the probability of m lost packets being recovered from $m+1$ repair packets as $\Upsilon(m+1,m) = P(m,m) + P(m+1,m)$. More generally, given $i \geq 1$ and $k \geq m$,

$$\Upsilon(k,m) = P(m,m) \sum_{i=0}^{k-m} \Big(\frac{\binom{n-m}{\gamma}}{\binom{n}{\gamma}} + \frac{\binom{n-m}{\gamma} + 2^1 - 1}{\binom{n}{\gamma}} + \dots + \frac{\binom{n-m}{\gamma} + 2^{m-1} - 1}{\binom{n}{\gamma}}\Big)^i$$

Hence, given k repair packets and γ, the probability of m lost packets being recovered is

$$\mathbb{P}\left\{\bigcup_{i=1}^{m} B_i^C\right\} \cdot \Upsilon(k,m) \qquad (3)$$

D. Analysis of Computation Complexity

In this section, we further study the computation complexity of our coding scheme and compare it with RS code and LT code.

Instead of only considering the number of operations with traditional Big-O notations, we look deeper into the approximate number of arithmetic operations for each coding scheme. This is because, given the extremely constrained resource on each sensor node, executing different arithmetic operations have very different requirements for memory space and clock cycles, which in turn determines the time and energy cost of an operation. Taking sensor nodes in [11] as an example, one addition, subtraction or XOR operation takes one clock cycle only, one multiplication operation takes 6 clock cycles, while one division operation takes up to 37 cycles. Moreover, sensor motes [13] does not support division operations, which requires the compiler to transform the division into other operations, which further complicates the overall computation. Table I shows the breakdown of different arithmetic operations needed for each coding scheme. A more detailed analysis of computation complexity for the comparing coding schemes can be found in [29].

As we can see that compared with RS code, our coding scheme requires a much less number of operations and involves simple XOR operations only. Our scheme in the best case has less computation complexity than LT code, while even in the worst case, the computation complexity is still competitive, considering a much less number of packets needed for recovering all lost packets. The above discussions give us some insights about the complexity of our scheme, RS code, and LT code. More detailed evaluation of the computation cost will be our future work.

	XOR	ADD	MUL	DIV
Proposal	$2n\ln n \sim n\ln n + n^3$	0	0	0
RS codes	0	$nm+n^3$	$nm+n^3$	n^3
LT codes	$(n+1)(n\ln n + \sqrt{n}\ln^3 n)$	0	0	0

TABLE I

BREAKDOWN OF ARITHMETIC OPERATIONS FOR THREE CODING SCHEMES

IV. PROACTIVE RELIABLE COMMUNICATION

In this section, we apply the proposed coding scheme to proactive transmission under two representative network failure models, i.e., *recoverable failure model* and *permanent failure model*, and analyze the communication performance. To be focused, the failure models exclude the network failures that can be recovered by the link-layer retransmissions, although our proposal can be used together with link-layer retransmissions to improve the communication reliability.

Recoverable Failure Model. In this model, each packet forwarded along a path has a probability of failure, which is however independent from that of other packets forwarded along the same path. Recoverable failures could be caused by short-period adversary conditions, e.g., radio interference, communication collision, and network congestion. To overcome the recoverable failures, the existing studies adopted either reactive end-to-end retransmission or proactive transmission that sends several copies of packets along different paths [4], [22]. However, both approaches involve overhead, due to control messages in reactive approach and duplicate data packets in proactive approach.

Here we study a redundant-coding based proactive transmission approach to overcome the recoverable network failures, with the objective of minimizing communication overhead. More specifically, given an estimate of packet loss rate, a source node sends all data packets and a reasonable number of repair packets. The destination, by decoding the received repair packets, is able to recover the lost data packets. In the following, we first analyze the recovery probability given that a total of K_s repair packets is sent by the source node. Given a packet loss probability p, the total number of repair packets received at the destination node (denoted by a random variable Z) follows a Binomial distribution, as each packet has an independent probability of failure. Let K_r be the total number of repair packets received at the destination node. Thus,

$$\mathbb{P}\{Z = K_r\} = \left(\begin{array}{c} K_s \\ K_r \end{array}\right)(1-p)^{K_r} p^{K_s - K_r}$$

which can be approximated by a normal distribution with expectation $K_s(1-p)$ and variance $\sqrt{K_s(1-p)p}$.

Hence, combining with Equation 3, the probability that, given K_s, the destination is able to obtain all packets under recoverable failure model is

$$\sum_{K_r=m}^{K_s} \mathbb{P}\{Z = K_r\} \cdot \mathbb{P}\left\{\bigcup_{j=1}^{m} B_i^C\right\} \cdot \Upsilon(K_r, m)$$

Recall that in Section III-B, the parameter of the proposed coding scheme $\mathbb{E}(k)$ is derived without assuming any packet losses. Now we derive the number of repair packets needed to be sent by the source node, $\mathbb{E}_s(k)$, such that at least $\mathbb{E}(k)$ (Equation 2) repair packets are received at the destination. We have

$$\mathbb{P}\{Z \geq \mathbb{E}(k)\} = 1 - \mathbb{P}\{Z \leq \mathbb{E}(k) - \frac{1}{2}\} \quad (4)$$

Given the cumulative density function for Z which follows a normal distribution,

$$\Phi(Z) = \mathbb{P}\{Z \leq z\} = \frac{1}{2}\left[1 + \text{erf}\left(\frac{z}{\sqrt{2}}\right)\right]$$

where $\text{erf}(\cdot)$ is the error function and $\text{erf}(z) = \int_0^z e^{-t^2} dt$. Since $erf(-z) = erf(z)$, Equation 4 is rewritten as

$$\mathbb{P}\{Z \geq \mathbb{E}(k)\} = \frac{1}{2} + \frac{1}{2} erf\left(\frac{\mathbb{E}_s(k)(1-p) - \mathbb{E}(k) + \frac{1}{2}}{\sqrt{2\mathbb{E}_s(k)(1-p)p}}\right)$$

Thus, the solution to $\mathbb{E}_s(k) \cdot \mathbb{P}\{Z \geq \mathbb{E}(k)\} = \mathbb{E}(k)$, which can be numerically solved, is the desired number of repair packets sent by the source node with the proposed coding scheme.

Permanent Failure Model. This model assumes that a network failure is permanent or longer than the maximum delay that one data communication can perceive. Thus, when failure happens, all packets routed along the path are lost. This kind of failures is usually caused by the network dynamics, malfunctions of sensor nodes, or adversary environmental conditions. To overcome permanent failures, most existing studies [10], [19], [21] employed a reactive multipath scheme to retransmit the lost packets along a new path or a proactive multipath scheme to send different packets along different paths. The reactive multipath scheme has negative impacts on communication delay and energy efficiency. On the other hand, the proactive multipath scheme without recovery incorporated, has limited capabilities of improving communication reliability. Therefore, we consider to combine our coding scheme with proactive multipath transmission to increase packet recovery rate.

We assume that all paths are mutually disjoint, i.e., the paths do not share the same nodes, which can be formed by the algorithms proposed by [16], [23]. A path, indexed as i ($i = 1, \cdots, s$), is assigned a failure probability p_i, where s is the total number of paths used for communication. Since there are no common nodes among the paths, the failure probabilities of paths are independent. Moreover, the estimation of the number of lost packets m has been studied under various network conditions in the literature [6], [15]. Let variable y_i denote the event that all repair packets forwarded along path i ($1 \leq i \leq s$) can be received by the destination. If path i fails, all the repair packets (and the data packets as well) sent along this path are lost such that $y_i = 0$ and $\mathbb{P}\{y_i = 0\} = p_i$, otherwise $y_i = 1$ and $\mathbb{P}\{y_i = 1\} = 1 - p_i$. y_i follows a Bernoulli distribution. Let Y denote the total number of paths that successfully forward the packets, i.e., $Y = \sum_{i=1}^{s} y_i$.

In the following, again, we first analyze the recovery probability given that a total of K_s repair packets is sent by the source node. Let K_r denote the total number of repair packets received at the destination. Thus, the total number of paths that successfully forward the packets is $\lceil s\ K_r/K_s \rceil$. Combining with Equation 3, the probability that, given K_s, the destination obtains all data packets under permanent failure model is

$$\sum_{K_r=m}^{K_s} \mathbb{P}\left\{ Y = \left\lceil \frac{s\ K_r}{K_s} \right\rceil \right\} \cdot \mathbb{P}\left\{ \bigcup_{j=1}^{m} B_i^C \right\} \cdot \Upsilon(K_r, m)$$

When all paths have the same probability of failure p,

$$\mathbb{P}\left\{ Y = \left\lceil \frac{s\ K_r}{K_s} \right\rceil \right\} = \binom{\left\lceil \frac{s\ K_r}{K_s} \right\rceil}{s} (1-p)^{\lceil \frac{s\ K_r}{K_s} \rceil} p^{s - \lceil \frac{s\ K_r}{K_s} \rceil}$$

We next derive number of repair packets needed to be sent by the source node, $\mathbb{E}_s(k)$, such that at least $\mathbb{E}(k)$ (Equation 2) repair packets are received at the destination. Based on the *Central Limit Theorem*, we have

$$Y_{norm} = \frac{\sum_{i=1}^{s} y_i - \sum_{i=1}^{s}(1-p_i)}{\sqrt{\sum_{i=1}^{s} p_i(1-p_i)}} \sim N(0,1)$$

Hence,

$$\begin{aligned} & \mathbb{P}\left\{ Y_{norm} \geq \frac{\lceil \frac{s\,\mathbb{E}(k)}{\mathbb{E}_s(k)} \rceil - \sum_{i=1}^{s}(1-p_i)}{\sqrt{\sum_{i=1}^{s} p_i(1-p_i)}} \right\} \\ = \quad & 1 - \mathbb{P}\left\{ Y_{norm} \leq \frac{\lceil \frac{s\,\mathbb{E}(k)}{\mathbb{E}_s(k)} \rceil - \frac{1}{2} - \sum_{i=1}^{s}(1-p_i)}{\sqrt{\sum_{i=1}^{s} p_i(1-p_i)}} \right\} \qquad (5) \end{aligned}$$

Thus, Equation 5 is rewritten as

$$\begin{aligned} & \mathbb{P}\left\{ Y_{norm} > \frac{\lceil \frac{s\,\mathbb{E}(k)}{\mathbb{E}_s(k)} \rceil - \sum_{i=1}^{s}(1-p_i)}{\sqrt{\sum_{i=1}^{s} p_i(1-p_i)}} \right\} \\ = \quad & \frac{1}{2} + \frac{1}{2}\,\mathrm{erf}\left(\frac{\sum_{i=1}^{s}(1-p_i) - \lceil \frac{s\,\mathbb{E}(k)}{\mathbb{E}_s(k)} \rceil + \frac{1}{2}}{\sqrt{2\sum_{i=1}^{s} p_i(1-p_i)}} \right) \qquad (6) \end{aligned}$$

The solution to

$$\mathbb{E}_s(k) \cdot \mathbb{P}\left\{ Y_{norm} > \frac{\lceil \frac{s\,\mathbb{E}(k)}{\mathbb{E}_s(k)} \rceil - \sum_{i=1}^{s}(1-p_i)}{\sqrt{\sum_{i=1}^{s} p_i(1-p_i)}} \right\} = \mathbb{E}(k)$$

is the desired number of repair packets sent by the source node with the proposed coding scheme.

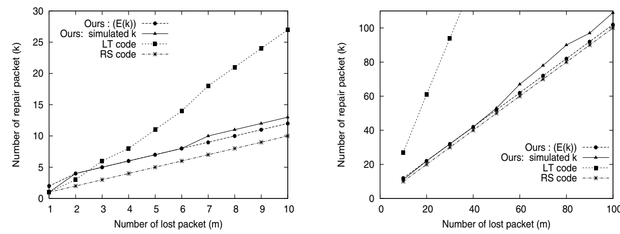
V. Experimental Results

In this section, we examine the performance of our proposed coding scheme by comparing with two state-of-the-art coding schemes (i.e., RS code and LT code, which are described in Section II) In the simulation, we do not assume a specific data routing algorithm, rather compare the proposed techniques with end-to-end retransmissions, reactive multipath transmissions and proactive multipath transmissions. We implement all the schemes under comparison in MATLAB and C++. The results are obtained by averaging the results over 100 runs for each scheme.

A. Study of Proposed Coding Scheme

This section evaluates the effectiveness of proposed coding scheme (Section III), which is expected to recover the data packets lost during the transmission.

Figures 2(a) and 2(b) plot the number of repair packets (k) needed to recover m lost packets for a variety of values of



(a) n = 10 (b) n = 100

Fig. 2. Proposed Coding Scheme

m, given n of 10 and 100, respectively. In order to evaluate the performance of $\mathbb{E}(k)$ derived in Section III-B, we also plot the theoretical result of $\mathbb{E}(k)$. Both Figures 2(a) and 2(b) show that the theoretical result of $\mathbb{E}(k)$ matches the simulation result of k well. Comparing with different coding schemes, as we pointed out earlier, RS code is able to minimize the number of repair packets at a high computation cost. Thus, in terms of the number of repair packets needed (transferred to communication overhead), RS represents the best case. The figures show that the performance of our proposed coding scheme is close to that of RS. On the other hand, LT code, which is not designed for transmission recovery, significantly increases the number of repair packets needed. Since the simulation results are consistent for $n = 10$ and $n = 100$, we only present the results with $n = 100$ in the following to save space.

Figure 3 shows the impact of γ on the communication overhead (i.e., the number of repair packets k) given different values of m. The simulation results verify our intuitions and provide more insights. As we expected, when γ is small, more repair packets are needed to fully recover the lost packets. However, when γ is large enough (e.g., $\gamma = 11$ for the case of $m = 40$), the coding scheme reaches the optimal number of repair packets (i.e., $k = m$); further increasing γ does not further reduces the repair overhead. More interestingly, when γ is very large (e.g., $\gamma = 82$ for $m = 20$), the number of repair packets needed increases dramatically. This is because when too many data packets are encoded for each repair packet, more repair packets are needed to construct m linearly independent equations. We marked the derived γ (derived in Section III-B) for different m's in the figure. For instance, when $m = 20$, the derived $\gamma = 21$; and when $m = 60$, $\gamma = 8$. We observe that the derived γ approaches closely to the *optimal* γ which yields the minimum number of repair packets while incurring the minimum computation overhead. This is appreciated by sensor networks with limited computation powers and energy resources.

B. Proactive Reliable Transmission under Recoverable Failure Model

First, we study the impact of communication failure probability p and total number of packets sent by the source node on the performance of proactive reliable transmission schemes studied in Section IV. We define *communication reliability* as the probability of a destination node obtaining *all* data packets (with recovery). The metric *communication cost* is defined as the total number of packets involved in communication.

In Figure 4(a), we study the communication reliability of our proposal by varying p from 0.1 to 0.5 and communication cost (i.e., $n + K_s$) from 100 to 250. When p increases, the source node has to send more packets (more K_s) to achieve the same level of reliability. Furthermore, communication reliability is not linear to the communication cost. This is because that the first decoding step has limitation for recovering lost packets, while in the second decoding step, there are no enough repair packets to form sufficient number of linearly independent equations, which results in no recovery of lost packets at all. Once the communication cost reaches a certain threshold, this reliability quickly increases to 100%. We also observe with a higher failure probability p, the communication reliability grows slower with increasing communication cost.

We compare our solution against two *NACK schemes*. In both schemes, the destination, once detecting the lost packets (e.g., by packet sequence number), sends a NACK message to the source node for the missing packets. In the first NACK scheme, called *separated NACK*, the destination sends a NACK for each lost packet to the source node for retransmission, while in the second scheme, called *aggregated NACK*, the destination waits until it receives all the non-lost packets, and sends only one aggregate NACK message for all missing packets. As one can expect, the separated NACK may incur more overhead due to NACK message with a shorter communication delay than the aggregated NACK. Figure 4(b) shows the communication cost for achieving a 100% communication reliability [3]. We plot the theocratical result of the communication cost for proactive multipath transmission with the proposed redundant coding scheme (i.e., $n + K_s$, where K_s is derived in Section IV). We observe that the simulation result for the communication cost is very close to our analytical result. Our approach constantly outperforms both NACK schemes under various values of p, i.e., 30% less packets than aggregated NACK and up to 45% less packets than separated NACK scheme.[4] Meanwhile, the communication delay in our approach is expected to be lower than both NACK schemes, since the destination does not need to send NACK for retransmissions, which causes another round trip delay. The quantitative evaluation of the communication delay is left as a future work.

C. Proactive Reliable Transmission under Permanent Failure Model

The proactive reliable transmission technique under permanent failure model takes advantage of multipath. In this section, we compare the coding based proactive multipath transmission technique against reactive multipath transmission in terms of communication reliability and communication cost.

Figure 5(a) plots the communication reliability by varying the total number of packets sent by the source node

[3]We exclude the case that the packet loss cannot be detected by NACK scheme, when all data packets are lost

[4]The gain of our approach over NACK schemes further increases when p is further increased.

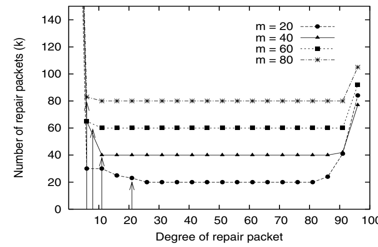
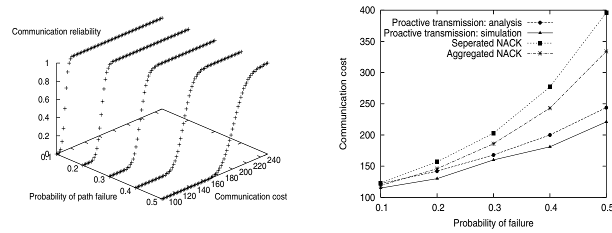


Fig. 3. Degree of Repair Packets γ (n = 100)

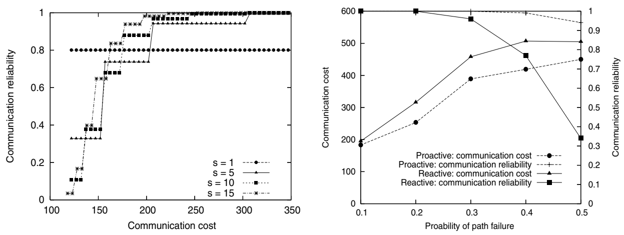


(a) Communication reliability (b) Communication cost

Fig. 4. Recoverable Failure Model

(i.e., communication cost). As we can see, the proactive transmission does not necessarily improve the probability of a destination obtaining all packets, even though the total number of packets received by the destination does increase (which is not shown in the figure). However, transmission along one path which could have a permanent failure cannot ensure the communication reliability. Figure 5(a) shows that by increasing the number of packets sent by the source node (i.e., the communication cost $n + K_s$), transmission along multiple path eventually is able to achieve a 100% reliability. Moreover, we observe that the more paths used (i.e., larger s), the less communication cost is required to achieving a 100% reliability.

Figure 5(b) compares communication cost (show by the left Y-axis) and the communication reliability (shown by the right Y-axis) of proactive multipath transmission and of reactive multipath transmission schemes. Since the underlying network issues are not the focus of this paper, we borrow existing research results for simulating the reactive multipath transmission, i.e., average maintenance overhead for each alternative path is 0.15 times of the communication cost along the primary path [10]. The reactive multipath transmission stops when both primary path and the alternative paths fail. As we observed from the figure, the proactive scheme constantly sends less number of packets than the reactive scheme for achieving the same level of reliability. More importantly, we observe a dramatic decrease of the probability of a destination obtaining all data packets in reactive multipath transmission when p increases. Yet the proactive scheme maintains a very high probability around 0.95 in all cases. Furthermore, this high reliability achieved by the proactive scheme has even less communication cost than the reactive approach which has much lower communication reliability. In summary, the proactive multipath transmission with proposed coding scheme is able to achieve significantly high communication reliability with reasonable communication costs.



(a) p = 0.2 (b) s = 10

Fig. 5. Permanent Failure Model

VI. Conclusion and Future Work

In this paper, we propose a low-computation, low-communication, loss-resilient coding scheme, suitable for resource-constrained wireless sensor networks. A two-step decoding scheme is crafted for improving its robustness with reduced communication and computation costs. Our analysis of the two key design parameters (i.e., the degree of repair packets and the number of repair packets), and of the expected recoverability of the proposed coding scheme allows the sensing applications to minimize the redundant repair packets, thus reducing the computation cost. We leveraged the proposed coding scheme upon the proactive reliable transmission with two different strategies, such that the communication reliability in recoverable and permanent failure model are both significantly improved. We have conducted simulations to evaluate the performance of our proposals and compare them against with other representative coding schemes (i.e., RS code and LT code), NACK schemes and reactive multipath schemes. The experimental results show that our proposal saves up to 45% communication overhead for achieving 100% reliability, in comparison with NACK schemes under recoverable failure model. Under permanent failure model, our proposal is up to three times more reliable than that of proactive multipath schemes with even less communication overhead.

The proposed coding scheme has shown promising features for significantly improving the communication reliability at reasonable computation and communication cost. We plan to evaluate its computation complexity and communication delay in detail by experimentations. Moreover, we plan to study the overall performance of proposed coding scheme and proactive reliable transmission with real sensor nodes.

VII. References

[1] C. Berrou, A. Glavieux, and A. Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo codes. In *Proceedings of IEEE International Conference on Communications*, pages 1064–1070, 1993.

[2] V. K. Bhargava. Forward error correction schemes for digital communications. *IEEE Commun. Mag.*, 21:11–19, Jan. 1983.

[3] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A digital fountain approach to reliable distribution of bulk data. *SIGCOMM Comput. Commun. Rev.*, 28(4):56–67, 1998.

[4] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proceedings of ICDE*, Boston, MA, Mar. 2004.

[5] S. De, C. Qiao, and H. Wu. Meshed multipath routing with selective forwarding: an efficient strategy in wireless sensor networks. *Computer Networks*, 43(4):481–497, 2003.

[6] D. S. J. De Couto, D. Aguayo, B. A. Chambers, and R. Morris. Effects of loss rate on ad hoc wireless routing. Technical Report MIT-LCS-TR-836, MIT Laboratory for Computer Science, 2002.

[7] S. Dulman, T. Nieberg, J. Wu, and P. Havinga. Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks. In *Proceedings of IEEE Wireless Communications and Networking Conference*, 2003.

[8] Jr. G. D. Forney. Convolutional codes I: Algebraic structure. *IEEE Trans. on Inform. Theory*, IT(15):720–738.

[9] Jr. G. D. Forney. Convolutional codes II: Maximum-likelihood decoding. *Information and Control*, 25:222–250, 1974.

[10] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. In *Proceedings of ACM MobiHoc*, 2001.

[11] M. Grnewald, B. Iske, and J. Klahold. Sensor systems for mobile mini-robots: Properties and resource requirements. In *Proceedings of International Symposium on Autonomous Minirobots for Research and Edutainment*, pages 209 – 218.

[12] Richard Hamming. *Coding and Information Theory*. Prentice-Hall, 1980.

[13] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *ACM SIGPLAN Notices*, 35(11):93–104, 2000.

[14] N. Johnson and S. Kotz. *URN Models and Their Application: An Approach to Modern Discrete Probability Theory.* John Wiley & Sons, 1977.

[15] S.-J. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *Proceedings of IEEE ICC*, pages 3201–3205, Jun. 2001.

[16] X. Li and L. Cuthbert. Node-disjointness-based multipath routing for mobile ad hoc networks. In *Proceedings of ACM International Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, pages 23–29.

[17] M. Luby. LT codes. In *Proceedings of the Symposium on Foundations of Computer Science*, Washington, DC, 2002.

[18] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*, 47(2):569–584, 2001.

[19] M. K. Marina and S. R. Das. Ad hoc on-demand multipath distance vector routing. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(3):92–93, 2002.

[20] M. Mitzenmacher. Digital fountains: A survey and look forward. In *Proceedings of IEEE Information Theory Workshop*, TX, 2004.

[21] A. Nasipuri and S.R. Das. On-demand multipath routing for mobile ad hoc networks. In *Proceedings of International Conference Computer Communications and Networks*, Boston, MA, Oct. 1999.

[22] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *Proceedings of ACM Sensys*, Baltimore, MD, Nov. 2004.

[23] M. R. Pearlman, Z. J. Haas, P. Sholander, and S. S. Tabrizi. On the impact of alternate path routing for load balancing in mobile ad hoc networks. In *Proceedings of ACM MobiHoc*, pages 3–10, 2000.

[24] W. W. Peterson. *Error Correcting Codes*. Cambridge, MA: MIT Press, 1961.

[25] I.S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8:300 – 304, 1960.

[26] L. Rizzo. Effective erasure codes for reliable computer communication protocols. *SIGCOMM Comput. Commun. Rev.*, 27(2):24–36, 1997.

[27] A. Tsirigos and Z.J. Haass. Multipath routing in the presence of frequent topological changes. *IEEE Communications Magazine*, 2001.

[28] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT(13):260–269, 1967.

[29] J. Xu Y. Xu and W.-C. Lee. Loss-resilient proactive data transmission in wireless sensor networks. Technical Report CSE06-005, Pennsylvania State University, 2006.