Loss-Resilient Proactive Data Transmission in Wireless Sensor Networks

Yingqi Xu† Jianliang Xu‡ Wang-Chien Lee†
†Pennsylvania State University, USA. {yixu, wlee}@cse.psu.edu
‡ Hong Kong Baptist University, Hong Kong. xujl@comp.hkbu.edu.hk

***Abstract*— Among a wide range of sensor network applications, many of them require reliable data communications 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 reliable data communication. In this paper, an effective coding scheme that exploits the tradeoff between redundant data transmission and encoding/decoding complexity is proposed. Two key design parameters of the proposed scheme, the *degree of repair packets* and the *the number of repair packets*, are derived to achieve a high data recovery probability with minimum coding redundancy and computation overhead. Furthermore, the proposed scheme is leveraged under recoverable and permanent failure models for proactive transmission. Accordingly, the expected probability of a destination obtaining all data packets is analyzed. Simulations have been conducted to verify our theoretical results. The simulation results reveal profound insights in support of the proactive transmission paradigm to achieve high communication reliability in wireless sensor networks.**

***Index Terms*— wireless sensor networks, data communication, reliability, erasure coding**

I. Introduction

RECENT advances in hardware have made possible the use of inexpensive, low-power miniature sensors in in-situ sensing applications. These sensor nodes can be deployed throughout a physical space and organized as a wireless network to provide high-resolution measurements of physical phenomena (such as temperature, humidity, and light). Among a wide range of networked sensing applications, many of them (e.g., disaster forecast, structural condition assessment) require reliable data communications, such that a target destination can obtain all data packets with a high probability. However, sensor networks are unreliable in nature due to fragile sensor nodes, error-prone wireless communications, and possibly hostile deployment environments. Moreover, 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 [6], [11], [23], [25], the source node is notified to retransmit a lost packet until all data packets are correctly received at the destination. This reactive approach has several disadvantages. First, 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). Second, it requires the source to maintain the transmission state and buffer all unacknowledged packets, which entails a large buffer size at the source. To remedy these deficiencies, proactive approaches have been recently proposed for highly unreliable networks (e.g., wireless sensor and ad hoc networks) [8], [34]. By transmitting redundant coded packets to the destination, proactive reliable transmission is designed to facilitate recovery from possible packet losses. 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. Many coding schemes have been proposed for various fields, such as forwarding error correction coding [3], [30], erasure channel [22] and digital fountain [24]. However, with different performance requirement, these coding schemes are not feasible for wireless sensor networks due to either their very complicated computation in encoding and/or decoding processes or very high coding redundancy. In this paper, inspired by existing coding schemes [21], [22], [31], 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 *trade-off* 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. Our research shows that these two key parameters have major impacts on the ability of the proposed coding scheme of recovering the lost data packets. We aim at minimizing both coding redundancy and computation complexity. Thus, the proposed coding scheme is able to recover the lost data packets with a high probability. 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 for a destination to obtain all data packets. Their computation complexity and communication overhead are shown to be suitable for wireless 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 identify two different network failure models, i.e., recoverable failure model and permanent failure model, and craft two reliable transmission techniques correspondingly. We analyze the expected performance of proposed techniques under both models.
- We conduct simulations to validate our theoretical analysis. The simulation results show that proposed coding scheme and proactive reliable transmission techniques enable the destination to obtain all data packet with 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 some future work directions in Section VI.

II. Related Work

The dynamic and lossy nature of wireless communications, hostile deployment environments, and vulnerable hardware components on sensor nodes all pose challenges for reliable network communications in wireless sensor networks [37], [38]. Although existing reliable transmission techniques are inefficient for wireless sensor networks, our work is informed and inspired by a number of research studies. In the following, we briefly survey the related work to our proposal. Reliable transmission techniques are reviewed in Section II-A and coding schemes are covered in Section II-B.

A. Reliable Transmission Techniques

The simplest reliable transmission technique is reactive end-to-end retransmission. However, under this technique, all data packets have to be buffered at the source node before being acknowledged by the destination node. A large buffer size is not desirable for resource-scare wireless sensor networks. Moreover, since the retransmission is triggered by NACK or ACK messages, both communication delay and communication overhead are substantial for highly unreliable networks. Reactive retransmission at link-layer is widely employed by highly unreliable wireless networks [36]. However, when a set of adjacent relay nodes on transmission path fail at the same time, link-layer retransmission fails to achieve transmission reliability.

Considering the limitations of link-layer retransmission, researchers have proposed that the source node retransmits once notified of the broken transmission path (and packet loss) by any intermediate relay node (c.f., the destination node in end-to-end retransmission) [15], [28], [29]. For retransmission, the source node establishes a new path. However, this scheme still suffers from a long communication delay since the new path is built on-the-fly after being notified of the packet loss. To reduce the communication delay of the above scheme, a class of reactive multipath schemes were proposed and studied by [6], [11], [23], [25]. In such an approach, a set of *alternative paths* are established and maintained during the period when the data packet is transmitted along a *primary path*. Upon detecting a failure on the primary path, the source node switches to one alternative path immediately for retransmission. However, the reduced delay is achieved at the cost of communication overhead caused by the maintenance of alternative paths. In contrast, the proactive multipath paradigm employs multiple paths for simultaneous data transmissions [20], [26], [27]. However, without mechanisms for retransmission or recovery of lost packets, this paradigm is limited to reduce the packet loss ratio only. High transmission reliability is not achieved. Our work combines redundant coding with proactive multipath routing to address this deficiency.

B. Coding Schemes

Coding schemes have been widely adapted for fault tolerant computing for various forms of digital data communications. For reliable data communications, 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. A detailed analysis of the computation complexity for some of these schemes is provided in Section III-D.

Forward Error Correction (FEC) [3] 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*. Block codes work on fixed-size blocks of bits or symbols of predetermined size, while convolutional codes work on bit or symbol streams of arbitrary length. There are many types of block codes, including Hamming code [13], BCH code [30] and Reed Solomon code (RS) [31]. 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. In addition to correct errors in bit-level (in a information stream or a data packet), RS codes are also used in packet-level (i.e., to recover lost data packets) [32]. 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. In this paper, we focus on the transmission reliability in packet-level. The proposed coding scheme employs much simpler XOR operations for encoding and decoding. Hence, the computation complexity is significantly reduced. Moreover, we minimize the coding redundancies by deriving the optimum degree of repair packets, a critical parameter affecting both of the computation complexity and the coding redundancies.

Convolutional code [9], [10], [35] is a type of error-correcting code in which an n-bit message to be encoded is transformed into a m-bit symbol, where m/n is the code rate ($m \geq n$). The encoding is a function of the last k information symbols and k is the constraint length of the code. Longer constraint lengths produce more powerful codes in terms of error correction, but the complexity of the coding scheme increases exponentially with constraint lengths. The Viterbi algorithm, which employs maximum likelihood estimation approach to make inferences about the underlying probability distribution of the given received bits, is universally used as decoding algorithm for convolutional codes. 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 (i.e., the coding redundancies is the same as the original data). Compared with RS codes, convolutional codes has much less computation complexity, but incur higher coding redundancies and works well with the communication channel with lower signal-to-noise (SNR) rate. The same problem exists in the Turbo code [2]. Moreover, event though being applicable, convolutional nodes 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.

Another class of coding schemes [4], [21], [22] were proposed for the *digital fountain* paradigm [1], [24], which is designed for delivery of a large amount of data over high bandwidth, high latency Internet links. These schemes, by using simple XOR operations for encoding, reduce the computation complexity at the cost of transmitting more encoded packets. For instance, in Luby Transform (LT) code, each encoded packet is generated 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 ensures that no matter when the destination starts to receive the packets at whatever rate, it 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, as LT code only sends part of the original data packets (i.e., $c\ln(n/\delta)$ out of n original data packets for decoding with a probability $1-\delta$; c is some suitable constant $c > 0$), it focuses on the distribution of γ such that there is always at least one decoded data packet that has not been removed from the encoded packets. As such, the decoding process can continue. However, this requirement incurs 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 $1-\delta$) during decoding process, which is an adversary for performing energy optimization in wireless sensor networks. Moreover, LT code, not designed for recovering lost packets, does not take into consideration the the condition of the communication channel (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. Thus, inspired by RS code and LT code, our approach sends both original data packets and repair packets, and designs a two-step decoding process, which efficiently decreases the number of packets required for decoding the packets. Moreover, by considering the expected lost rate of the communication channel, 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 aims 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 encoding and decoding is presented in Section III-A. The mathematical analysis of the tradeoff between the *degree of repair packets* and the *number of repair packets*, which are critical parameters for determining the computation complexity and the coding redundances 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 and compares it with RS code and LT code in terms of the number of arithmetic operations.

A. Encoding and Decoding

In this section, we describe the encoding and decoding process of proposed coding scheme. Without causing confusion, we call the encoded packets as repair packets that are used for recovering the data packets 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 the repair packet by exclusive-oring ($\oplus$) 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 [4], [21], employs XOR operations such that the computation complexity for both encoding and decoding (described below) is minimized.

When using the repair packets to recover the original data packets, the destination needs to know the degree of repair

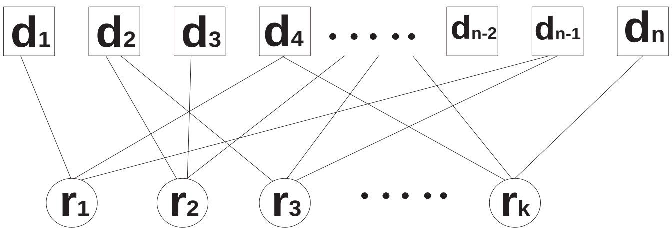


Fig. 1. Encoding $\gamma = 3$

packets γ and the inputs of each repair packet. The simplest way for conveying this information is to include it into each repair packet. However, the communication overhead is proportional to γ. To minimize this overhead and hence conserve energy, we let 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.

After identifying the inputs for each repair packet, the next step is decoding these repair packets to reconstruct the lost data packets. The decoding process consists of two steps. The first step is similar to the decoding process used by LT code [21]. More specifically, the destination node divides the received packets into three sets. The first set consists of *unprocessed* data packets, the second consists of *processed* data packets, and the last set holds all the repair packets. Initially, all received data packets are in the unprocessed set and the processed set is empty. For instance, if m data packets are lost, the unprocessed set has $n - m$ packets. Moreover, at the beginning of the decoding, all repair packets have the same degree (i.e., $\gamma_1 = \gamma_2 = ... = \gamma$). 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 is 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. [21] has shown that to recover all packets with this approach, a large number of repair packets are needed. However, this is against our goal of designing a low-overhead reliable communication 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 redundancies, 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) are 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 *optimal* γ, such that the number of repair packets is minimized without dramatically 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* [16], 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

$$\mathbb{E}(\mathbf{X}) = \sum_{i=1}^{n} \frac{1}{1-(i-1)/n} = n\sum_{i=1}^{n}\frac{1}{i} = nH_n$$

As the harmonic series $H_n = \ln n + r + O(\frac{1}{n})$, we approximate $\mathbb{E}(\mathbf{X})$ as $\mathbb{E}(\mathbf{X}) = n\ln n$

Since k denotes the number of repair packets collected and each repair packet represents γ balls, we must have

$$k \cdot \gamma \geq n \ln n \tag{1}$$

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 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, 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:

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

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

$$\sum_{i=1}^{m} a_{1i} \geq 1$$

For simplicity, we assume that a packet has an equal probability of being selected and not being selected by a repair packet. Thus, the probability of $A(0,0)$ transiting to $A(1,1)$ after adding $a_1^{(m)}$ is $1-\frac{1}{2^m}$, and the probability of $A(0,0)$ transiting to $A(1,0)$ is $\frac{1}{2^m}$. 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 $1-\frac{2^h}{2^m}$.

Figure 2 depicts the probability transition of the rank of repair matrix increasing from 0 up to m. Let $S(i)$ denote the expected number of m-tuples needed for reaching rank m from

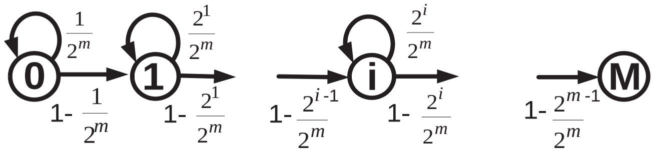


Fig. 2. State Transition Diagram for Repair Matrix

state i, we obtain the following equations:

$$\begin{cases} S(0) = \left(\frac{1}{2^m}\right)(S(0)+1) + \left(1-\frac{1}{2^m}\right)(S(1)+1) \\ S(1) = \left(\frac{2^1}{2^m}\right)(S(1)+1) + \left(1-\frac{2^1}{2^m}\right)(S(2)+1) \\ \ldots \\ S(i) = \left(\frac{2^i}{2^m}\right)(S(i)+1) + \left(1-\frac{2^i}{2^m}\right)(S(i+1)+1) \end{cases}$$

By recursively applying the above equations, we obtain the expected number of repair packets (denoted as $\mathbb{E}(k)$) for recovering m lost packets as follows

$$\mathbb{E}(k) = S(0) = \left\lceil m + \sum_{i=1}^{m} \frac{1}{2^i - 1} \right\rceil \tag{2}$$

Given the sum of the degree of repair packets (in Equation 1) $k\gamma \geq n \ln n$, we derive the average degree for each repair packet is

$$\mathbb{E}(\gamma) \geq MIN \left\{ n, \quad \left\lceil \frac{n}{m + \sum_{i=1}^{m} \frac{1}{2^i-1}} \ln n \right\rceil \right\} \tag{3}$$

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 all 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,

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

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

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

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 shown in Figure 2. 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:

$$P(m,m) = \frac{2^m - 1}{2^m} \cdot \frac{2^{m-1}-1}{2^{m-1}} \cdot \ldots \cdot \frac{1}{2}$$

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

$$P(m+1,m) = P(m,m)\left(\frac{1}{2^m} + \frac{1}{2^{m-1}} + \ldots + \frac{1}{2}\right)$$

Denoted by $\Upsilon(m+1,m)$, the probability of m lost packets being recovered from $m+1$ repair packets as follows:

$$\begin{aligned} \Upsilon(m+1,m) &= P(m,m) + P(m+1,m) \\ &= P(m,m)\left(1 + \frac{1}{2^m} + \frac{1}{2^{m-1}} + \ldots + \frac{1}{2}\right) \\ &= \frac{2^{m+1}-1}{2^{m+1}} \cdot \frac{2^m-1}{2^m} \cdot \ldots \cdot \frac{2^2-1}{2^2} \end{aligned}$$

Similarly, we derive

$$\Upsilon(m+2,m) = \frac{2^{m+2}-1}{2^{m+2}} \cdot \frac{2^{m+1}-1}{2^{m+1}} \cdot \ldots \cdot \frac{2^3-1}{2^3}$$

$$\ldots$$

$$\Upsilon(k,m) = \frac{2^k-1}{2^k} \cdot \frac{2^{k-1}-1}{2^{k-1}} \cdot \ldots \cdot \frac{2^{k-m+1}-1}{2^{k-m+1}}$$

Hence, given k repair packets and γ, the probability of m

$$\mathbb{P}\left\{\bigcup_{i=1}^{m} B_i^C\right\} \cdot \Upsilon(k,m) \tag{4}$$

D. Analysis of Computation Complexity

The recoverability analysis of the proposed coding scheme in the previous sections offers guidance for designing a robust coding scheme at a low cost. 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 [12] 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 [14] even does not support division operations, which requires the compiler to transform the division into other operations, which further complicates the overall computation.

	XOR	ADD	MUL	DIV
Our Coding Scheme	$2n \ln n \sim n \ln n + n^3$	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

The impact of the computation complexity on energy and storage cost is easy to understand. The latency of an operation may also have an impact on the data transmission rate. More specifically, when an encoding process is long (i.e., taking more clock cycles), it is more likely that the data transmission rate is constrained by the rate of generating repair packets at the source node. In this section, we estimate the approximate numbers of different types of arithmetic operations in both encoding and decoding processes for the proposed coding scheme, RS code, and LT code, which gives us a general idea for the complexity of these coding schemes.

First, for RS code, each repair packet is created by $F(\alpha) = d_0 + d_1\alpha + \cdots + d_n\alpha^{n-1}$. This polynomial equation can be calculated by $(((d_n\ \alpha + d_n - 1)\ \alpha + d_{n-2})\ \alpha + \cdots) + d_0$. Therefore, encoding each packet requires n multiplication operations and n addition operations. For m repair packets, a total of nm multiplication and nm addition operations is used. We assume decoding uses the Gaussian Elimination, the total computation complexity is $O(n^3)$, which includes approximately n^3 multiplication, n^3 division, and n^3 subtraction operations [33].

For LT code, [21] shows that each repair packet encodes $\ln n$ data packets on average, and at least $(n + \sqrt{n} \ln^2 n)$ repair packets needed for decoding n data packets (since LT codes only sends the repair packets, not the data packets). Therefore, the source needs at least $\ln n(n + \sqrt{n} \ln^2 n)$ XOR operations for encoding. The decoding process needs $n \ln n$ XOR operations for each repair packet, thus a total of $\left(n \ln n \left(n + \sqrt{n} \ln^2 n\right)\right)$ XOR operations is consumed for decoding all packets.

Now turn to our proposed coding scheme. Since each repair packet encodes γ original data packets, the total number of operations involved in generating $k = m + \sum_{i=1}^{m} 1/(2^i - 1)$ repair packets is $k \cdot \gamma = n \ln n$ XOR operations. For decoding, in one extreme case, all repair packets can be decoded by using the first-step decoding process, which requires $n \ln n$ XOR operations. On the other hand, it is possible the first decoding step does not recover any lost packet. In this case, the decoding requires approximately n^3 XOR operations in total. Table I summaries the breakdown of different arithmetic operations needed for each coding scheme.

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 for the complexity of our scheme, RS code, and LT code in terms of computation, storage, and latency. More detailed evaluation of the computation cost will be our future work.

To implement our proposal on real sensor nodes, the proposed coding scheme needs to be customized according to the hardware features of current sensor nodes (e.g., 4Mhz 8-bit ATMega128 CPU from Atmel, 4KB RAM, 128KB program flash, 512KB data flash [14]). Similar to the implementation of RS code [17], each data packet is divided into small sub-data packets (e.g., 8 bits) and encoding is applied over a set of sub- data packets. Each repair packet then consists of a set of sub- repair packets and the destination node each time decodes a sub- repair packet.

IV. ANALYSIS OF 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, the path failure is temporary. More precisely, 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 [5], [26]. As we pointed out in Section II-A, both approaches involve significant overhead, due to control messages in reactive approach and duplicate data packets in proactive approach. Moreover, end-to-end retransmission suffers from prolonged communication delay.

Here we propose a redundant-coding based proactive transmission approach to overcome the recoverable network failures, with the objective of minimizing both communication delay and 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 which are generated based on the encoding scheme discussed in Section III-A. 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,

$$\mathbb{P}\{Z = K_r\} = \binom{K_s}{K_r} (1-p)^{K_r} p^{K_s - K_r},$$

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 4, 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

$$\mathbb{P}\{Z \geq \mathbb{E}(k)\} = 1 - \mathbb{P}\{Z \leq \mathbb{E}(k) - \frac{1}{2}\} \quad (5)$$

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 5 can be 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 (e.g., fire and flooding). To overcome permanent failures, most existing studies [11], [23], [25] 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 propose 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 [19], [27]. 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 in the sense that success or failure of one path does not imply the state of another. Moreover, the estimation of the number of lost packets m has been studied under various network conditions in the literature [7], [18]. 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 4, 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,

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

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,

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

Similarly, the cumulative density function for variable Y that follows a standard normal distribution is

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

Thus, Equation 6 is rewritten as

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

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

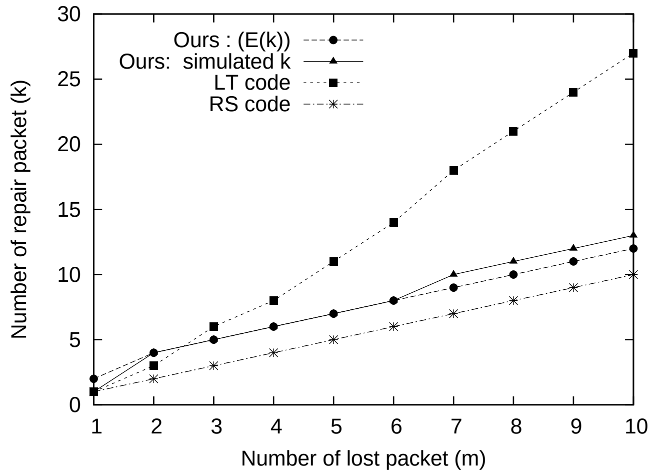
In this section, we present our simulation results, from which we expect to further understand the performance trade-off between the communication reliability and overhead for the proposed coding scheme under both recoverable and permanent failure models. Moreover, we intend to, via simulation, validate the theoretical results obtained in Section III and Section IV, which provide important guidance for selecting the coding parameter to achieve reliable communication at a low cost. In the following, 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-B) in Section V-A. Section V-B and Section V-C evaluate the strategies which leverage the proposed coding scheme in proactive transmission paradigm for recoverable and permanent failure models, respectively. In the simulation, we do not assume a specific data routing algorithm, rather compare the proposed techniques with existing reliable transmission techniques including end-to-end retransmissions, reactive multipath transmissions and proactive multipath transmissions. We implement all the schemes under comparison in MATLAB and C++. The performance results are obtained by averaging the results over 100 runs for each scheme.

A. Study of Proposed Coding Scheme

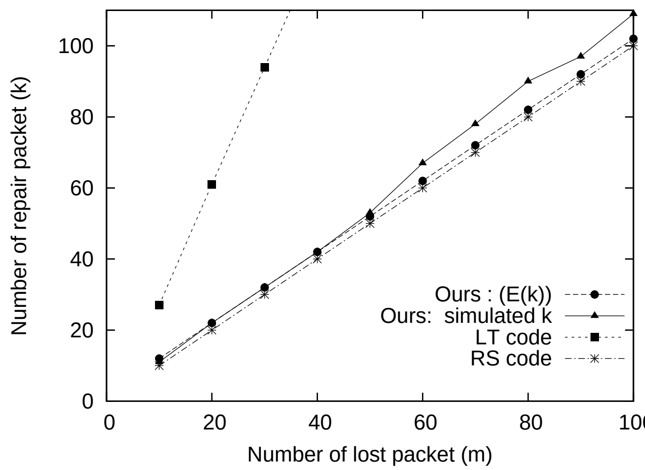
This section evaluates the effectiveness of proposed coding scheme (Section III), which is expected to recover the data packets lost during the transmission.

Figures 3(a) and 3(b) plots the number of repair packets (k) needed to recover m lost packets for a variety of values of m, given n of 10 and 100, respectively. We compare the simulation result of our coding scheme with those of LT code and of RS code. Moreover, in order to evaluate the performance of the expected number of repair packets ($\mathbb{E}(k)$) derived from Equation 2, we also plot the theoretical result of $\mathbb{E}(k)$. Both Figures 3(a) and 3(b) show that for our coding scheme, 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. It is observed that the simulation results are consistent for $n = 10$ and $n = 100$. Therefore, in the following, we only present the results with $n = 100$ to save space.

Next, we study the impact of the degree of repair packets (γ) on the performance of proposed code. As we have argued in Section III-B, with a larger γ, more computation is involved since more data packets are encoded and decoded for each repair packet, but a less number of repair packets are expected to cover any lost packet. Figure 4 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 (from Equation 3) 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.



(a) n = 10



(b) n = 100

Fig. 3. Proposed Coding Scheme

The above performance study has shown that by carefully selecting the degree of repair packets (γ) based on our analytical result, the proposed code is able to fully recover the lost packets with the minimum number of repair packets and the minimum computation overhead. In the following two sections, we investigate how the proposed coding scheme

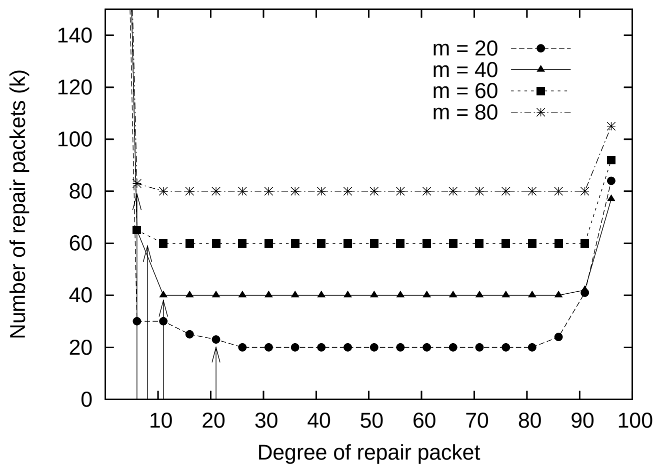


Fig. 4. Degree of Repair Packets γ (n = 100)

improves communication reliability under recoverable failure model and permanent failure model, respectively.

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 proposed in Section IV. We define *communication reliability* as the probability of a destination node obtaining *all* data packets (with possible recovery). The metric *communication cost* is defined as the total number of packets involved in communications, including the original data packets, redundant repair packets, and NACK/ACK control messages.

In Figure 5(a), we study the communication reliability of proposed proactive multipath transmission scheme on Z-axis, by varying p on X-axis from 0.1 to 0.5 and varying the communication cost (i.e., $n+K_s$) on Y-axis 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. More specifically, before reaching a certain threshold, increasing number of (repair) packets sent by a source node almost does not improve the reliability probability. 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.

To further evaluate the performance of proposed proactive reliable transmission under recoverable failure model, 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 a significant overhead for NACK messages with a short communication delay, while the aggregated NACK could suffer from a long communication delay with much less NACK overhead. In the following, we compare our approach with both of the NACK schemes.

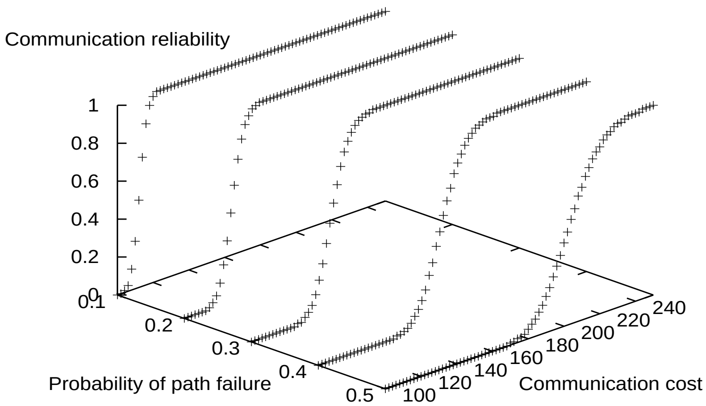
Figure 5(b) shows the communication cost for achieving a 100% communication reliability. In addition to the simulation results of proactive reliable transmission, separated NACK, and aggregated NACK schemes, we also plot the theocratical result of the communication cost for proactive multipath transmission 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.[1] 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

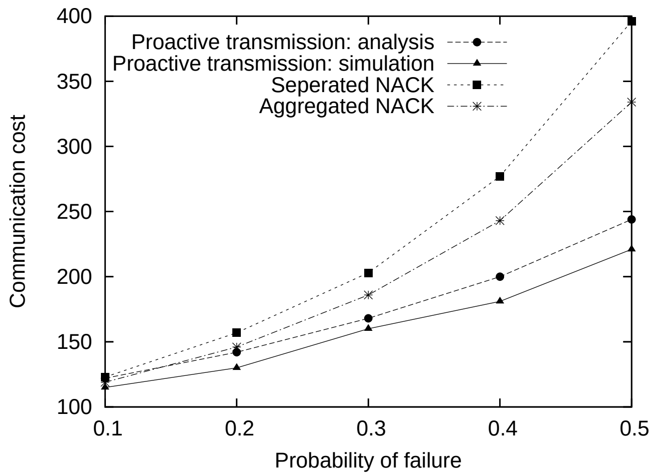
The proactive reliable transmission technique under permanent failure model takes advantage of multipath. In this section, we compare the proactive multipath transmission technique based on the proposed coding scheme against reactive multipath transmission in terms of communication reliability and communication cost defined in Section V-B.

We first study the coding based proactive multipath transmission. Figure 6(a) plots the communication reliability by varying the total number of packets sent by the source node (i.e., communication cost). Each curve in the figure represents a different number of paths (s) used for the proactive multipath transmission. 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). We use an example to explain this fact. Assuming a failure probability of $p = 0.2$ and 20 out of 100 data packets are lost. Thus, approximately 22 repair packets needs to be sent to recover the lost 20 packets. If all packets are routed along one path, there is a 80% chance that the destination receiving all packets (even without recovery). Now assuming the total number of packets are divided into two equal-size sets, which are sent along two disjoint paths each with $p = 0.2$. It only one path succeeds, 50 data packets are lost, which cannot be recovered by the 11 repair packets received from the other path. Thus, even though the probability of the destination receives nothing is dramatically reduced to $0.2 * 0.2 = 0.04$, the destination can only recover all packets when both paths

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

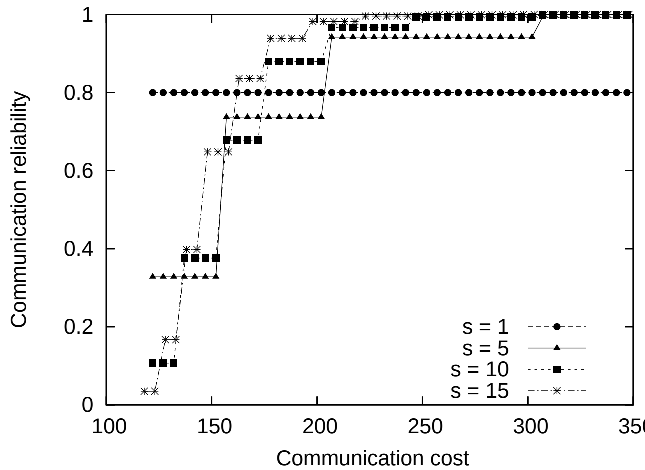


(a) Communication reliability

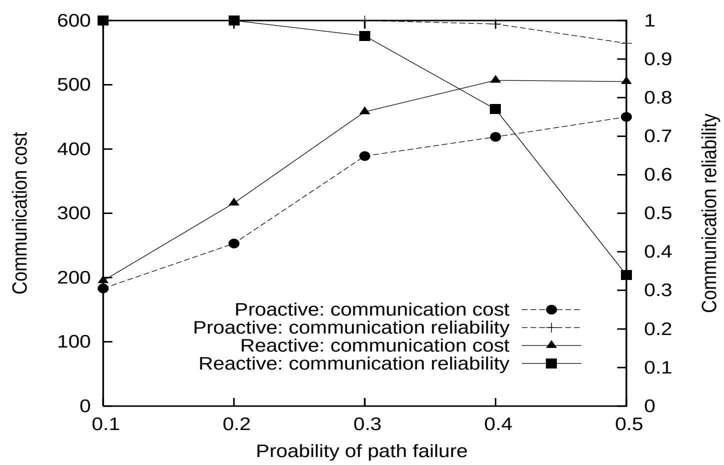


(b) Communication cost

Fig. 5. Recoverable Failure Model



(a) p = 0.2



(b) Communication cost and communication reliability

Fig. 6. Permanent Failure Model

succeed which has a probability of $0.8 * 0.8 = 0.64$, lower than that of transmission along only one path.

However, transmission along one path which could have a permanent failure cannot ensure the communication reliability. Figure 6(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 6(b) compares the performance of proactive multipath transmission combined with proposed coding scheme and of reactive multipath transmission schemes. By varying the probability of path failure from 0 to 0.5, we study both communication cost (show by the left Y-axis) and the communication reliability (shown by the right Y-axis). Since the underlying network issues are not the focus of this paper, we borrow some existing research results for simulating the reactive multipath transmission. Based on a representative reactive multipath transmission algorithm for wireless sensor networks [11], we simulated the average maintenance overhead for each alternative path disjoint from the primary path is 0.15 times of the communication cost along the primary path. Since the alternative paths are pre-established together with the primary path, the reactive multipath transmission stops when both primary path and the alternative paths fail. We assume the same number of paths used for both reactive and proactive approaches ($s = 10$). 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.

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. The proposed coding scheme simplifies the encoding and decoding process by only using XOR operations. 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 empowers the sensing applications with an ability to minimize the redundant repair packets, thus reducing the computation cost. Moreover, we look deeper into the reliable communication problem in wireless sensor networks and identify two distinct failure models, i.e., recoverable and permanent failure model. We leveraged the proposed coding scheme upon the proactive reliable transmission with two different strategies, such that the communication reliability in two failure models 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] Digital fountain. http://www.digitalfountain.com/.

[2] 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.

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

[4] 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.

[5] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proceedings of International Conference on Data Engineering*, Boston, MA, Mar. 2004.

[6] 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.

[7] 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.

[8] 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*, New Orleans, LA, Mar. 2003.

[9] Jr. G. D. Forney. Convolutional codes I: Algebraic structure. *IEEE Trans. on Inform. Theory*, IT(15):720–738, 1970.

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

[11] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking & computing*, pages 251–254, New York, NY, 2001.

[12] 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, 2003.

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

[14] 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.

[15] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*. 1996.

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

[17] S. Kim, R. Fonseca, and D. Culler. Reliable transfer on wireless sensor networks. In *Proceedings of International Conference on Sensor and Ad hoc Communications and Networks*, Oct 2004.

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

[19] 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, 2004.

[20] W. Lou, W. Liu, and Y. Fang. Spread: enhancing data confidentiality in mobile ad hoc networks. In *Proceedings of IEEE INFOCOM 2004*, March 2004.

[21] M. Luby. LT codes. In *Proceedings of the Symposium on Foundations of Computer Science*, page 271, Washington, DC, 2002.

[22] 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.

[23] 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.

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

[25] 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.

[26] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *Proceedings of ACM Sensys*, pages 250–262, Baltimore, MD, Nov. 2004.

[27] 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 International Symposium on Mobile Ad Hoc Networking & Computing*, pages 3–10, 2000.

[28] C. Perkins and E. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings of IEEE Workshop on Mobile Computer Systems and Applications*, pages 90–100, New Orleans, LA, Feb. 1999.

[29] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proceedings of the Conference on Communications Architectures, Protocols and Applications*, pages 234–244, London, United Kingdom, 1994.

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

[31] 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.

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

[33] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997.

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

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

[36] C.-Y. Wan, A. T. Campbell, and L. Krishnamurthy. PSFQ: a reliable transport protocol for wireless sensor networks. In *Proceedings of ACM International Workshop on Wireless Sensor Networks and Applications*, pages 1–11, 2002.

[37] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of ACM Sensys*, pages 14–27, 2003.

[38] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of ACM Sensys*, pages 1–13, Los Angeles, CA, Nov. 2003.