Line Networks with Erasure Codes and Network Coding

by

Yang Song

B.Sc., Beijing University of Posts and Telecommunications, 2009

A Thesis Submitted in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

Line Networks with Erasure Codes and Network Coding

by

Yang Song

B.Sc., Beijing University of Posts and Telecommunications, 2009

Supervisory Committee

Dr. Xiaodai Dong, Supervisor

(Department of Electrical and Computer Engineering)

Dr. Lin Cai, Departmental Member

(Department of Electrical and Computer Engineering)

**Supervisory Committee**

Dr. Xiaodai Dong, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Lin Cai, Departmental Member
(Department of Electrical and Computer Engineering)

## ABSTRACT

Wireless sensor network plays a significant role in the design of future Smart Grid, mainly for the purpose of environment monitoring, data acquisition and remote control. Sensors deployed on the utility poles on the power transmission line are used to collect environment information and send them to the substations for analysis and management. However, the transmission is suffered from erasures and errors along the transmission channels. In this thesis, we consider a line network model proposed in [1] and [2]. We first analyze several different erasure codes in terms of overhead and encoding/decoding costs, followed by proposing two different coding schemes for our line network. To deal with both erasures and errors, we combine the erasure codes and the traditional error control codes, where an RS code is used as an outer codes in addition to the erasure codes. Furthermore, an adaptive RS coding scheme is proposed to improve the overall coding efficiency over all SNR regions. In the end, we apply network coding with error correction of network errors and erasures and examine our model from the mathematical perspective.

# Contents

# List of Figures

# List of Abbreviations

| | |
|---|---|
| 3GPP | 3rd Generation Partnership Project |
| AR | autoregressive |
| AWGN | additive white Gaussian noise |
| BEC | binary erasure channel |
| BER | bit error rate |
| CSI | channel state information |
| CSIT | CSI at the transmitter |
| CRC | cyclic redundancy check |
| FEC | forward error corrction |
| LDPC | low density parity check |
| LT | Luby transform |
| PDF | probability density function |
| RLNC | random linear network codes |
| RLFC | random linear foutain codes |
| RS | Reed-Solomon |
| SNR | signal to noise rate |

# Acknowledgement

First and foremost, I would like to show my deepest gratitude to my supervisor Dr. Xiaodai Dong, who has provided me with valuable guidance in my thesis as well as my life during the whole master study. Her constant motivation, ample support and impressive patience have been and will always be enlightening me not only in this thesis but also in my future career.

I would extend my appreciation to my committee member Dr. Lin Cai for her insightful guidance and constructive comments, and to Dr. Yang Shi for being as the external examiner. My sincere appreciation will also go to Dr. Wu-Sheng Lu, Dr. Andreas Antoniou, Dr. Kui Wu for their guidance and help through my graduate study.

I shall express my deepest thanks to Moyuan Chen, my husband. Moyuan has been accompanied me through my entire college life, with his endless love, kindness and support. Besides, this work would not have been possible without my parents. Together or apart, they are always providing their love, patience, encouragement, understanding and care.

Last but not least, I would like to thank my colleagues and friends, Chenyuan Wang, Binyan Zhao, Zhuangzhuang Tian, Lebing Liu, Ted Liu, Yi Shi, Yuzhe Yao, Youjun Fan, Congzhi Liu, Guowei Zhang, Xue Dong, Teng Ge, Ping Li, Yijia Xu, for their presences and support in both study and life.

# Dedication

*To my dearest parents*

# Chapter 1

# Introduction

In the design of future Smart Grid, wireless sensor network plays an important role in various parts for the purpose of environment monitoring, data acquisition and remote control. Sensors are deployed on or around the utility poles to monitor the temperature, humidity and other elements. With wireless communication technologies such as ZigBee, Bluetooth, etc., these data are collected and sent to the substations to be analyzed and managed.

References [1] and [2] proposed a wireless line network model to support the transmission lines monitoring applications. In this model, sensors deployed on one pole can only apply a short-range communication, i.e., they can only send messages to the relay node on the same pole; the relay nodes can apply long-range communications to forward the collected messages to the substation hop by hop. Each pole needs to transmit its own messages collected from the sensors deployed on this pole, as well as the messages it received from the previous pole. Thus the messages to be transmitted are accumulating after passing each pole, that is, the closer to the substation the pole is, the heavier traffic load it will carry. In order to increase the traffic efficiency, different technologies are suggested to use. For instance, [4] makes a slight improvement

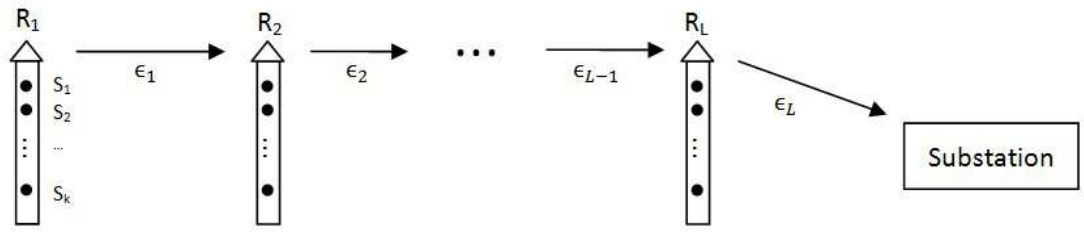


Figure 1.1: System Model

on the model to use cellular network to take responsibility for delivering information to the substations efficiently and effectively; [5] utilizes systematic random linear network coding and compares its performance with the performance of using an uncoded automatic repeat request reference scheme.

## 1.1 System Model

In this thesis, we consider a line network model proposed in [1] and [2]. As shown in Figure 1.1, a total number of $L$ poles are deployed along the power transmission line and $K$ sensors are deployed on each pole to monitor the power transmission. Sensors on the poles can collect the required information for the power grid, such as current, temperature, etc. On each pole, there is one sensor working as a relay node, which is responsible for collecting messages from itself and other sensors on the same pole, and for relaying messages from relays coming before it in the line network to the relays after it. The communication range of each relay node is within one hop, i.e. it will only receive messages from the pole before it and transmit messages to the pole next to it. The other sensors on the same pole work under short-range communication, i.e. they only communicate with their own relay, but not with nodes on other poles. However, sensors are designed in the way that they can switch from the short transmission range mode to the one-hop transmission range mode, so that

if the relay node of one pole is down, one of the other sensors can serve as a new relay node by switching the transmission mode.

The sensors will attempt to send reports periodically to a substation at the end of the transmission line. Considering the communication range, the reports are sent in a hop-by-hop manner. In each time period, we suppose that on each pole $R_l$ there are $K$ message packets to be sent to the substation, denoted by $\mathbf{s}_l = [\mathbf{s}_{l1}\mathbf{s}_{l2}\cdots\mathbf{s}_{lK}]$.

The wireless links between the nodes suffers from mainly two types of problems: erasures and errors. Channels with erasures for transmitting one bit are called binary erasure channel (BEC), first introduced by Elias [6] in 1955. One common method for communicating over BEC is to employ a feedback channel from receiver to sender that is used to control the retransmission of erased packets. For instance, the receiver sends back an acknowledgment to identify the missing packets. By receiving such acknowledgment, the sender is able to retransmit the indicated missing packets. Such feedback schemes embrace the advantage of simple complexity and can work without the knowledge of the erasure probability $\epsilon$. However, these schemes are in some sense a waste of capacity according to Shannon theory. If the erasure probability $\epsilon$ is high, the number of acknowledgment messages will be very large. Moreover, in the case of a broadcast channel with erasures, each receiver will receive a random fraction $(1-\epsilon)$ of the packets, and will probably be different fractions of the packets. The feedback scheme will cause huge redundancy since the sender has to retransmit every packet that is missed by one or more receivers.

To deal with the problem, tremendous efforts have been made to study and design a specific set of codes which are capable of correcting erasures and require no feedback or least feedback, namely erasure codes. The classic block codes for erasure correction are called Reed-Solomon (RS) codes [7]. An $(N, K)$ RS code can recover the original $K$ source symbols if any $K$ of the $N$ transmitted symbols are received. However,

such RS code suffers from the encoding and decoding cost of order $K(N-K)\log N$ packets operations. A better type of codes, Luby Transform codes (LT codes) is first proposed by Luby [8]. And a more recent code, Raptor codes, which is based on LT codes, has gained much attention in literature for its linear encoding and decoding cost. Details of these codes will be discussed in Chapter 2 where we investigate the possible erasure codes for our line network model.

Besides erasures, errors induced by radio channel propagation also impair the wireless transmission. One error in one link could cause the collapse of the entire network through error propagation. Therefore, we need a solution which can correct not only erasures but also errors. One simple way is to use the cyclic redundancy check (CRC) scheme to detect the errors. Moreover, an forward-error-correction (FEC) code could be used as an outer code in addition to the erasure code to correct errors.

Furthermore, network coding based on the traditional point-to-point error correcting codes is a feasible option. Network coding with proper design is capable of detecting and correcting erasures and errors. The reason why it has the error-correction function in network scale lies in its relationship with the traditional point-to-point error-correction codes, i.e., algebraic coding. In fact, algebraic coding can be viewed as an instance of network coding in one link scenario. In this thesis, we will investigate the details of how to implement network coding with error-correction in our line network model.

## 1.2 Thesis Outline

The remainder of the thesis is organized as follows.

In Chapter 2, we review and compare the performance of several existing codes for

erasure attacks on the line network model, including non-systematic random linear fountain codes, systematic random linear fountain codes, LT codes and Raptor codes. Their overhead and encoding/decoding costs when applied to our line network model will be analyzed. Then we propose two general packet processing strategies at multi-hop relays: complete encoding and decoding and decode-at-destination. Different strategies suit different systems in terms of node capacity, delay and memory space constraints. We investigate these performances and compare the two strategies.

In Chapter 3, we further add error detection and correction mechanism in the line network besides erasure handling. We use cyclic redundancy check to detect the errors and apply a type of RS code as an outer code to correct the errors. The packet loss rate and the code efficiency of the combined coding scheme are analyzed and simulation results are provided. Furthermore, we propose an adaptive RS coding scheme which can improve the overall coding efficiency over all SNR regions by selecting adjustable RS code rate.

In Chapter 4, we investigate network coding with error correction of network errors and erasures at random locations in two scenarios from the theoretical perspective. First, the known results in the single-source multicast scenario is reviewed. Then, we expand the model to the multiple-source scenario, which is exactly our line network model. We analyze and present results on the capacity region and error performance for the specific line network model.

Chapter 5 concludes the thesis and states possible future works.

# Chapter 2

# Erasure Codes for Multi-Source Line Networks in Erasure Channels

As introduced in the previous chapter, to deal with erasure channels, a specific type of code, erasure codes, is widely used. The most simple erasure code is random linear fountain codes. It simply generates combinations of randomly selected transmitted messages and then transmits the encoded messages. The random linear fountain codes (RLFC) can be easily implemented, but it incurs large decoding cost of $K^3$, where $K$ is the number of source packets. A better type of codes, Luby Transform codes (LT codes) are first proposed by Luby [8]. It achieves better decoding cost which scales with $K \ln K$. And a more recent code, Raptor codes [9], which is an improvement of LT codes, has gained much attention in the literature for its linear encoding and decoding cost.

In this chapter, we first review the above mentioned erasure codes, including: RLFC, LT codes, Raptor codes, etc. The existing literature studies the performance of these codes in one link scenario. We will implement these codes in our wireless line network with multiple hops and multiple sources and calculate the actual encoding

and decoding cost and the overhead to ensure the successful decoding. Finally, we propose packet processing strategies at multi-hop relays: complete decoding and re-encoding, decode-at-destination, and compare their performances in terms of delay and memory requirement.

## 2.1 Erasure Codes and Its Application to Line Networks

### 2.1.1 Random Linear Fountain Codes

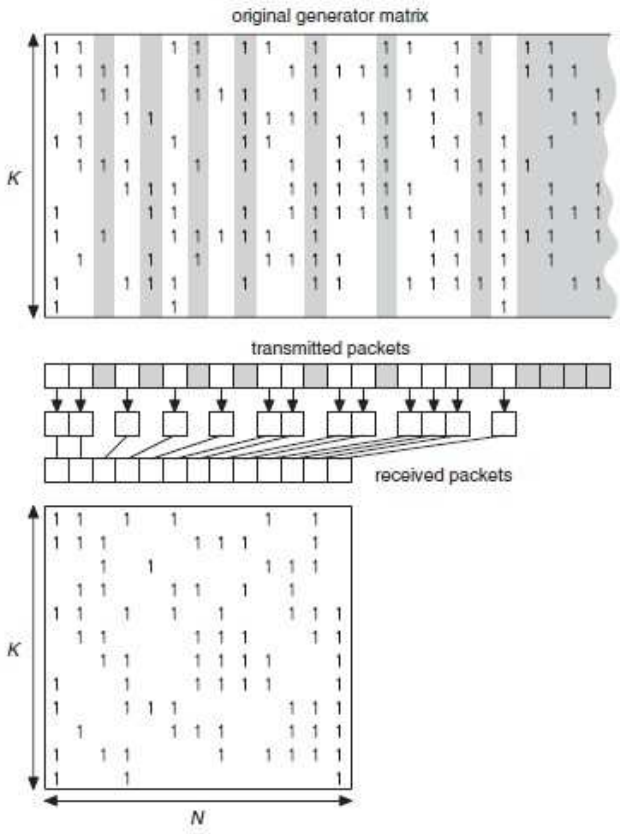


Figure 2.1: The generator matrix of a random linear code [3]

In coding theory, fountain codes are a class of erasure codes that can potentially generate limitless groups of encoding symbols from a given set of source symbols. The "fountain" is a metaphor to the encoded codes of the source codes because of the limitless property. In this way, we focus on making sure that enough packets are received to recover the original packets, regardless of how many encoded packets have been transmitted. Thus, the fountain codes are also named as rateless codes.

The RLFC are the simplest fountain codes. Figure 2.1 shows how the encoding matrix is generated. Consider a total number of $K$ packets to be sent, each of which consists $b$ information bits. Due to the property of an erasure channel, a packet is the elementary unit which is either perfectly received or erased. Here, we define a time slot as the duration of an original packet to be encoded and transmitted. In the n-th time slot, the encoder generates a $K$-bit encoding vector $\mathbf{g}_n$ consisting of bit 0 and 1 with equal possibilities. Information of $\mathbf{g}_n$ can be attached in the header of the transmitted packet. We denote $\mathbf{G} = [\mathbf{g}_1\mathbf{g}_2 \cdots \mathbf{g}_n \cdots]$ as the encoding matrix and $G_{kn}$ as its $k$-th element at the $n$-th time slot. Thus, the transmitted packet at each time slot is formed as the bit-sum of the original packets with $G_{kn} = 1$. One encoded packet to be transmitted is $b + K$ bits long including its header. We assume the encoded packets are sent out at the rate of 1 packet/time-slot. Therefore, the transmitted packets $\mathbf{t}_n$ at time slot $n$ is calculated as

$$\mathbf{t}_n = \sum_{k=1}^{K} \mathbf{s}_k G_{kn}. \tag{2.1}$$

Denote $\mathbf{S}_{b\times K} = [\mathbf{s}_1\mathbf{s}_2 \cdots \mathbf{s}_K]$ as the source $K$-packet matrix and $\mathbf{T}_{b\times n} = [\mathbf{t}_1\mathbf{t}_2 \cdots \mathbf{t}_n]$ as the transmitted packet matrix. According to (2.1), the transmitted packet matrix $\mathbf{T}$ is calculated as

$$\mathbf{T} = \mathbf{S} \cdot \mathbf{G}. \tag{2.2}$$

In GF(2), the additive operations are implemented as the XOR operations. The encoder generates the encoded packets continuously until the receiver receives enough packets and successfully decodes the source packets. To recover the original packets, the decoder performs the inverse operation of the encoding matrix, and obtains the original packets as

$$\mathbf{S} = \mathbf{T} \cdot \mathbf{G}^{-1}. \tag{2.3}$$

The decoding process applies Gaussian elimination. It can be easily seen that the probability that the original packets can be successfully recovered is the probability that the encoding matrix $\mathbf{G}$ is invertible, i.e., it is of full rank. Given the assumption that bit 0 and 1 are generated with equal probability in the generator matrix, we can now compute the decoding probability when $N_R = K$, where $N_R$ is the total number of received packets. It is obvious that the $K \times K$ matrix $\mathbf{G}$ is invertible when each of the columns in $\mathbf{G}$ is linearly independent of the preceding columns. Therefore, the decoding probability, equivalent to the probability of invertibility, is a product of $K$ probabilities, $(1 - 2^{-K})(1 - 2^{-(K-1)}) \ldots (1 - \frac{1}{8})(1 - \frac{1}{4})(1 - \frac{1}{2})$. For $K$ larger than 10, the product is 0.289, which is not promising.

What if we are now allowing the receiver to accept more than $K$ packets, i.e., $N_R > K$? If $N_R = K + E$, where $E$ is the small number of excess received packets at the decoding side, the original packets can be recovered with a relative low failure probability [3] of

$$\delta(E) \leq 2^{-E}, \tag{2.4}$$

where $\delta$ is the probability that the receiver will not be able to decode the original packets and it is a function of $E$. Figure 2.2 [3] shows a typical relationship between

the actual probability of failure and its upper bound. The failure probability $\delta$ is plotted against $E$ for the case $K = 100$.

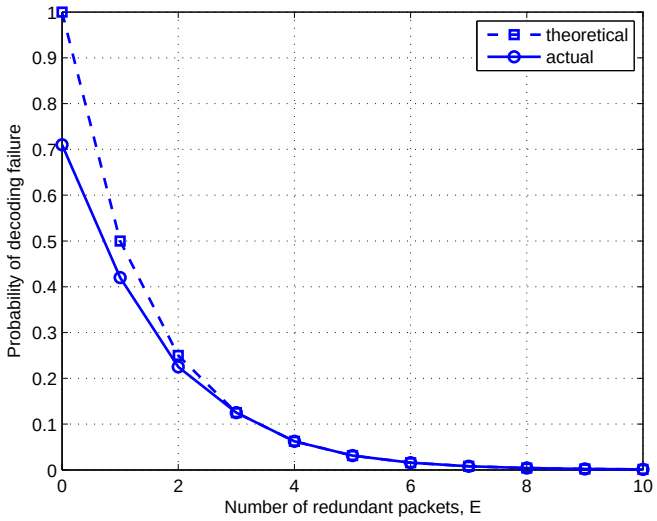


Figure 2.2: Probability of decoding failure versus the number of redundant packets $E$ [3]

.

Fountain codes can be divided into two categories: non-systematic and systematic. Systematic codes contain the input symbols in the output symbols and simply append the parity data to the source block. Conversely, non-systematic codes do not contain the source symbol block. In the following two subsections, we describe and compare the non-systematic and systematic RLFC codes adopted by our line network. We denote the erasure rates between every two poles by $\epsilon_1, \epsilon_2, \cdots, \epsilon_L$. Let the transmit packets after encoding at each relay be represented as $\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_L$.

### 2.1.2 Non-Systematic Random Linear Fountain Codes

In this coding scheme, the first relay $R_1$ collects information from sensors on the same pole and implements non-systematic random linear fountain codes to encode the packets to be transmitted. The encoded packets are expressed as $\mathbf{t}_{1n} = \sum_{k=1}^{K} \mathbf{s}_{1k} G_{kn}$, where $\mathbf{s}_{1k}$ is the $k$-th packet of the original message from $R_1$. Thus, when $R_2$ receives enough packets to recover the packets from $R_1$, it sends an acknowledgment to $R_1$ to inform $R_1$ to stop transmitting packets and the receiving packets are decoded as $\mathbf{s}_{1k} = \sum_{n=1}^{N} \mathbf{t}_{1n} (\mathbf{G}^{-1})_{kn}$. Then $R_2$ combines the decoded packets and its own packets together. Therefore, the packets to be transmitting at $R_2$ are $\mathbf{s}_2 = [\mathbf{s}_{11}, \mathbf{s}_{12}, \cdots, \mathbf{s}_{1K}, \mathbf{s}_{21}, \mathbf{s}_{22}, \cdots, \mathbf{s}_{2K}]$. Again, $R_2$ performs non-systematic RLFC and transmits the encoded packets to $R_3$. Hence, there are totally $L \times K$ packets to be sent from $R_L$ to the substation.

For the convenience of analyzing the codes, we first define the following performance parameters, which are in the same spirit of those defined in [9].

- *Overhead:* The overhead is a function of the decoding algorithm used, and is defined as the number of output packets that the decoder needs to collect in order to recover the input packets with high probability, minus the number of input symbols, which is exactly $E = N_R - K$ as in the previous section.

- *Cost:* The cost of the encoding and the decoding process. Normally, the cost are in terms of the number of binary operations and packet operations.

At $R_1$, there are totally $K$ packets to be transmitted in each time period. As we discussed before, at the receiver side, for high decodability, it needs more than $K$ received packets to decode. The excess part of the packets are considered as overhead at the decoding side. The expected overhead $\mathbf{E}[\mathcal{O}_{NS}(K)]$ for non-systematic RLFC of length $K$ is given by [10]

$$\mathbf{E}[\mathcal{O}_{NS}(K)] = \sum_{i=1}^{K} \frac{1}{q^i - 1}, \tag{2.5}$$

where $q$ is the size of Galois Field. [10] also gives an upper bound for the overhead $\mathbf{E}[\mathcal{O}_{NS}(K)]$.

$$\mathbf{E}[\mathcal{O}_{NS}(K)] = \sum_{i=1}^{K} \frac{1}{q^i - 1} \leq \frac{q^2 - q + 1}{(q-1)^3}. \tag{2.6}$$

For $q = 2$, we get

$$\mathbf{E}[\mathcal{O}_{NS}(K)] = \sum_{i=1}^{K} \frac{1}{2^i - 1} \leq 3, \tag{2.7}$$

which is very low.

At $R_L$, there are totally $L \times K$ packets to be transmitted. Thus, the expected overhead at the substation can be calculated by (2.7). For simplicity we assume that the erasure rate $\epsilon_1 = \epsilon_2 = \cdots = \epsilon_L = \epsilon$. Therefore the total number of transmitted packets from all the relays needed to recover conveyed information is obtained as

$$n_{total} = ((1 + 2 + \cdots + L) \cdot K + \sum_{l=1}^{L} \mathbf{E}[\mathcal{O}_{NS}(l \cdot K)]) \cdot \frac{1}{1-\epsilon} \leq (\frac{(1+L)L}{2} K + 3L) \cdot \frac{1}{1-\epsilon}. \tag{2.8}$$

For non-systematic RLFC, note that the encoding matrix generates 1 and 0 with equal probability, so on average half of the packets are added up (a packet operation is the exclusive-or of two packets). Hence the expected encoding complexity is $K/2$ packet operations per packet. On the other hand, the decoding complexity is quite high due to the cost of matrix inversion, which is $K^3$ binary operations and $K^2/2$ packet operations [3].

The simulation parameters are as follows: the erasure rate $\epsilon = 0.01$, the number

of poles is from $L = 1$ to $L = 10$ and each pole has its own $K = 50$ packets to be transmitted. Figure 2.3 shows the total number of transmitted packets as a function of the number of poles. We can see that the actual total number of transmitted packets is well bounded by the upper bound in (2.7) and the gap between the two is increasing as the total number of transmitted packets increases.

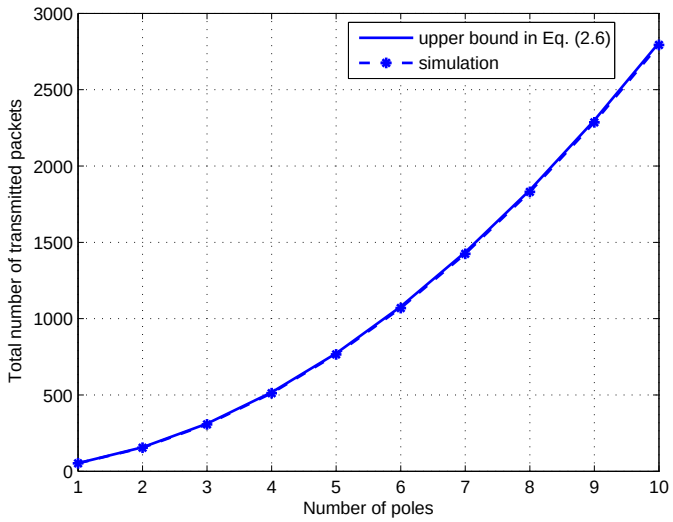


Figure 2.3: The total number of transmitted packets versus the number of the poles using non-systematic RLFC.

Figure 2.4 shows the total number of binary operations for decoding at each pole with $L = 10$. Although the number of binary operations are not as high as $K^3$, it is at the order of $K^3$. It can be seen that the non-systematic RLFC incurs huge decoding complexity. When the total number of poles is increasing, the total number of packets the last pole needs to transmit is increasing in the order of $(LK)^3$.

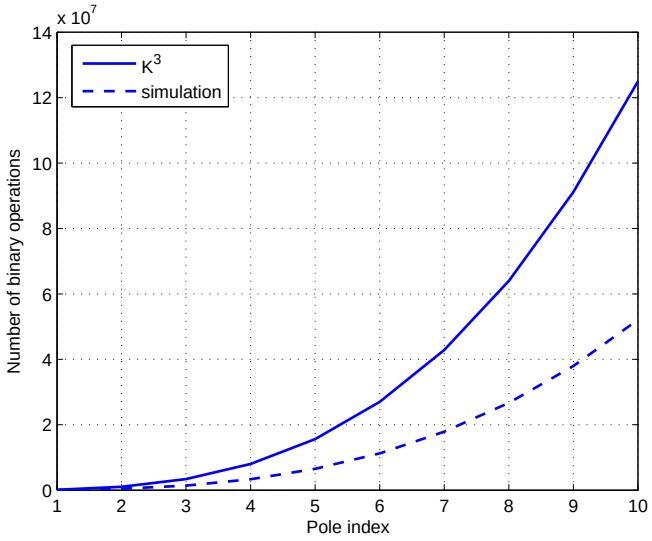


Figure 2.4: Decoding cost for non-systematic RLFC: total number of binary operations in decoding versus pole index.

### 2.1.3 Systematic Random Linear Fountain Codes

In systematic RLFC, at each relay the original packets are first sent without encoding, which can be considered as being encoded with an identity encoding matrix. Then a linear combination of the packets are sent following the original packets. Except for the encoding matrix, the systematic RLFC is deployed the same way as non-systematic RLFC. The encoding matrix can be written as

$$\mathbf{G}_{K\times n} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & \cdots \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots \\ 0 & 0 & 1 & 0 & \cdots & 0 & 1 & 1 & \cdots \\ \multicolumn{9}{c}{\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots} \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 & 1 & \cdots \end{bmatrix}. \tag{2.9}$$

The expected overhead $\mathbf{E}[\mathcal{O}_{SYS}(K,\epsilon)]$ can be derived from the overhead of non-

systematic RLFC and its upper bound is given by [10]

$$\mathbf{E}[\mathcal{O}_{SYS}(K,\epsilon)] = \sum_{i=1}^{K} \mathcal{O}_{NS}(i)\binom{K}{i}\epsilon^i(1-\epsilon)^{K-i} - K\epsilon \leq \frac{q^2-q+1}{(q-1)^3}(1-(1-\epsilon)^K) - K\epsilon, \tag{2.10}$$

where $\epsilon$ is the erasure rate. For $q = 2$,

$$\mathbf{E}[\mathcal{O}_{SYS}(K,\epsilon)] \leq 3(1-(1-\epsilon)^K) - K\epsilon, \tag{2.11}$$

which is even smaller than 3.

For our line network model, we can also calculate the total number of transmitted packets for completely recovery of the original packets as

$$n_{total} \leq \frac{(1+L)L}{2}K + [3L - 3(1-\epsilon)^K - 3(1-\epsilon)^{2K} - \cdots - 3(1-\epsilon)^{LK} - \frac{(1+L)L}{2}K\epsilon] \cdot \frac{1}{1-\epsilon}, \tag{2.12}$$

where the erasure rate $\epsilon_1 = \epsilon_2 = \cdots = \epsilon_L = \epsilon$.

For systematic RLFC, the decoding complexity is $(K\epsilon)^2 \log(K\epsilon)$. Compared with non-systematic RLFC, the decoding complexity is lower. Since the original packets are sent first, the decoding of linear combination part will be reduced as to invert a sparse $K\epsilon \times K\epsilon$ for each decoding cycle with $K$ original packets, which will cost $\mathcal{O}((K\epsilon)^2 \log(K\epsilon))$ binary operations [5].

The result is simulated with number of poles from $L = 1$ to $L = 10$ and $K = 50$ and each pole has its own 50 packets to send, as show in Figure 2.5. For simplicity we have $\epsilon_l = \epsilon = 0.01$ for $l = 1, 2, \ldots, 10$.

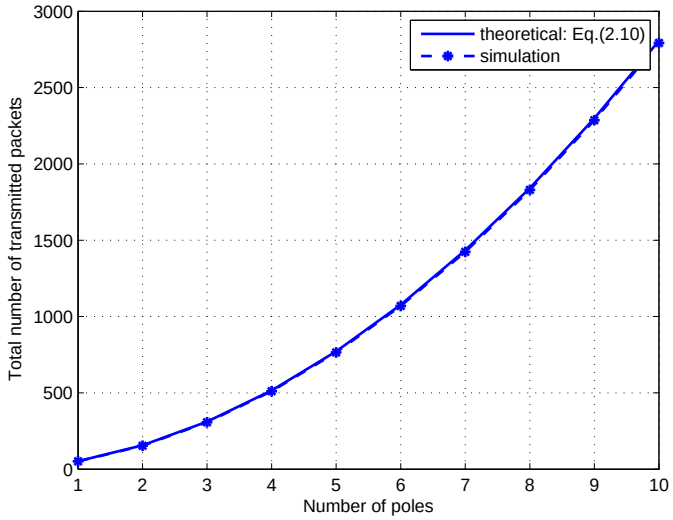


Figure 2.5: Total number of transmitted packets at each pole versus the number of the pole for non-systematic RLFC.

Figure 2.6 shows the total number of binary operations for decoding at each pole for systematic RLFC. Note that although it seems there is a large gap between the actual number of binary operations and $K^2$, it is at the magnitude of $10^5$, and thus still at the order of $K^2$.

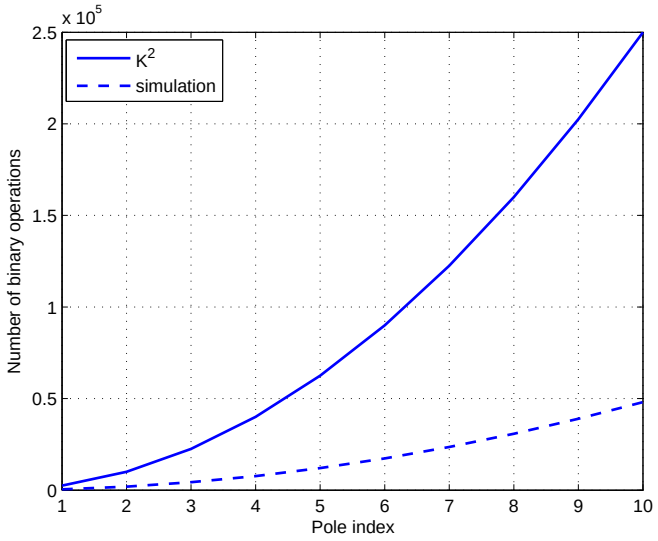


Figure 2.6: Decoding cost for systematic RLFC: total number of binary operations in decoding versus pole index.

As discussed above, systematic RLFC outperforms non-systematic RLFC from the perspective of transmit overhead and the algorithm complexity. On the other hand, non-systematic RLFC performs better in terms of undetected decoding error probability when the minimum free distance of the code is larger [7].

While random linear codes as we described above may not be in the technical sense of a 'perfect' code, they actually perform quite well in terms of overhead and the algorithm complexity. An excess of $E$ packets increases the probability of success to at least $(1-\delta)$, where $\delta = 2^{-E}$ [3]. Thus, as $K$ increases, RLFC can get arbitrarily close to Shannon limit. However, they suffer from their encoding and decoding costs, which are at least quadratic in the number of packets encoded. Although this scaling is not important if $K$ is small, in our line network model, the number of packets sent from the sensors is increasing as they are closer to the destination. Therefore, we still need to find a solution with lower computational cost.

### 2.1.4 LT codes

Luby transform codes (LT codes), named after its inventor [8], are the first class of practical fountain codes that are near optimal erasure correcting codes. LT codes retain the good overhead performance of the random linear fountain codes, while reducing the encoding and decoding cost dramatically.

*Encoding Algorithm [8]:*

1. Randomly choose the degree[1] $d$ of the encoding symbol from a degree distribution $\rho(d)$. Different designs of degree distribution are given in details later.

2. Choose uniformly at random $d$ distinct input symbols as neighbors of the encoding symbol.

3. The value of the encoding symbol is the exclusive-or of the $d$ neighbors.

At the receiver side, the decoder requires the information of the degree and the set of neighbors of each encoding symbol in order to recover the original input symbols. This information are delivered to the decoder in different ways in practice. For instance, the degree and a list of neighbor indices may be given explicitly to the decoder for each encoding symbol. Another example would be using a key to associate with each encoding symbol and then both encoder and decoder apply the same function to the key to produce the degree and the set of neighbors of the encoding symbol. The latter method is more favorable in our network model since it reduces communication overhead, which is important for the low data rate wireless relay nodes.

The decoding process of LT codes employs a mathematical graph called decoding graph. It is defined in [9]: the decoding graph of an algorithm of length $N$ is a bipartite graph with $K$ nodes on the one side denoting the input symbols, and $N$

[1]The degree of an encoding symbol is defined as follows: if the encoding symbol is the result of the XOR operation of $n$ source symbols, the degree of this encoding symbol is $n$.

nodes on the other denoting the output symbols. There is an edge between an input symbol and an output symbol if the input symbol contributes to the value of the output symbol.

Now we can describe the decoding process of LT codes as follows. *Decoding Algorithm [8]:*

1. Find a check node $\mathbf{t}_n$ that is connected to only one source packet $\mathbf{s}_k$.

   (a) Set $\mathbf{s}_k = \mathbf{t}_n$.

   (b) Add $\mathbf{s}_k$ to all checks $\mathbf{t}_{n'}$ that are connected to $\mathbf{s}_k$:

   $$\mathbf{t}_{n'} := \mathbf{t}_{n'} + \mathbf{s}_k \text{ for all } n' \text{ such that } G_{n'k} = 1.$$

   (c) Remove all the edges connected to the source packet $\mathbf{s}_k$.

2. Repeat (1) until all $\mathbf{s}_k$'s are determined.

From the encoding and decoding process, we can see that the probability distribution $\rho(d)$ of the degree is a very crucial. Two requirements should be satisfied: 1) each source packet must be connected to at least one output packet; 2) there exist some output packets that are connected to only one source packet each, so that the decoding process can get started. Luby has provided three basic distribution functions in [8].

**Definition 1.** *All-at-Once distribution:* $\rho(1) = 1$

Applying this all-at-once distribution corresponds to the situation where each encoding symbol is generated by selecting a random input symbol and copying its value to the encoding symbol. It has been proven in [8] that $K \ln(K/\delta)$ encoding

symbols will be generated for all $k$ input symbols. Therefore, although the all-at-once distribution enjoys the simplest encoding complexity, the number of encoding symbols is unacceptably large.

**Definition 2.** *Ideal soliton distribution: the ideal soliton distribution is* $\rho(1), \ldots, \rho(K)$, *where*

- $\rho(1) = \frac{1}{K}$
- *For all* $i = 2, \ldots, K, \rho(i) = \frac{1}{i(i-1)}$.

As the name suggests, ideal soliton distribution results in one check node which has degree one at each decoding iteration. Ideally, at each iteration, when this check node is processed, the degrees in the decoding graph are reduced in such a way that one new degree-one check node appears. The expected degree under this distribution is $\ln K$. One might imagine this way of decoding is perfect since it avoids redundancy. However, in practice, this degree distribution performs poorly, because it is very likely that at some iteration, there will be no degree-one check nodes or a few source symbols will lose connections with the output symbols. To deal with the problem, the robust soliton distribution is proposed in [8].

**Definition 3.** *Robust soliton distribution: The robust soliton distribution is* $\mu(\cdot)$ *defined as follows: Let* $S = c\ln(\frac{K}{\delta})\sqrt{K}$ *for some suitable constant* $c > 0$ *where* $\delta$ *is the allowable failure probability of the decoder to recover the data for a given number of* $K$ *encoding packets. Define*

$$\tau(i) = \begin{cases} \frac{S}{iK} & \text{for } i = 1, \ldots, \frac{K}{S} - 1 \\ \frac{S}{K}\ln(\frac{S}{\delta}) & \text{for } i = \frac{K}{S} \\ 0 & \text{for } i = \frac{K}{S} + 1, \ldots, K \end{cases} . \tag{2.13}$$

*Add the ideal soliton distribution* $\rho(\cdot)$ *to* $\tau(\cdot)$ *and normalize to obtain* $\mu(\cdot)$*:*

- $\beta = \sum_{i=1}^{K} \rho(i) + \tau(i)$.
- *For all* $i = 1, \ldots, K$, $\mu(i) = (\rho(i) + \tau(i))/\beta$.

The idea behind robust soliton distribution is that by adding the function $\tau(\cdot)$, it ensures that the expected number of degree-one checks at each iteration of the decoding process is

$$S = c \ln(\frac{K}{\delta})\sqrt{K}, \tag{2.14}$$

rather than 1. Furthermore, it is shown in [8] that receiving $N_R = K + 2\ln(S/\delta)S$ packets ensures that the original packets can be recovered with probability at least $1 - \delta$.

Now we calculate the total number of transmitted packets of all relays for the successful decoding at the substation $n_{total}$. According to $N_R = K + 2\ln(S/\delta)S$ and (2.14), the number of transmitted packets at relay $R_l$ is

$$n_{R_l} = lK + 2c \ln\left(\frac{c\ln(lK/\delta)\sqrt{lK}}{\delta}\right) \ln(\frac{lK}{\delta})\sqrt{lK}. \tag{2.15}$$

Therefore, $n_{total}$ is obtained as

$$\begin{aligned} n_{total} &= \left(\frac{(1+L)L}{2}K + 2c\sum_{l=1}^{L}[\ln\left(\frac{c\ln(lK/\delta)\sqrt{lK}}{\delta}\right) \cdot \ln\left(\frac{lK}{\delta}\right)\sqrt{lK}]\right) \cdot \frac{1}{1-\epsilon} \\ &< \frac{(1+L)L}{2}K + \frac{2c^{\frac{3}{2}}}{\delta^{\frac{5}{4}}}\sum_{l=1}^{L} l^{\frac{3}{2}}K^{\frac{3}{2}}. \end{aligned} \tag{2.16}$$

The last inequality follows the fact $\ln(x) < \sqrt{x}$ for all $x > 0$.

The encoding and decoding cost of LT codes is analyzed in [9]: a random LT-code with $K$ input symbols has encoding cost $K/2$, and ML decoding is a reliable decoding algorithm for this code of overhead $O(K\ln(K))$.

We apply LT codes in our line network model, with parameters defined after. Figure 2.7 shows the total number of transmitted packets at all relays against number of poles from $L = 1$ to $L = 10$, with erasure rates $\epsilon_1 = \epsilon_2 = \cdots = \epsilon_L = \epsilon$. Figure 2.8 shows the decoding cost. Simulations are done with $L = 10$ and $c = 0.01$. Compared to non-systematic and systematic RLFC, LT codes made a considerable improvement in reducing the decoding cost.

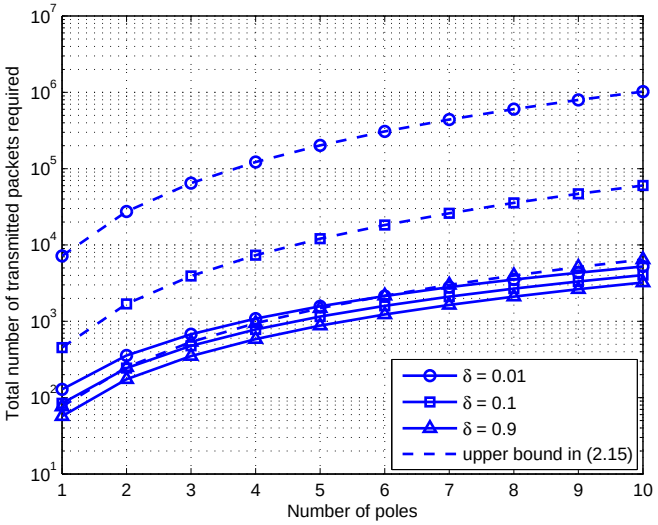


Figure 2.7: Total number of transmitted packets to ensure the successful recovery probability at least $1 - \delta$ versus the number of poles $L$.

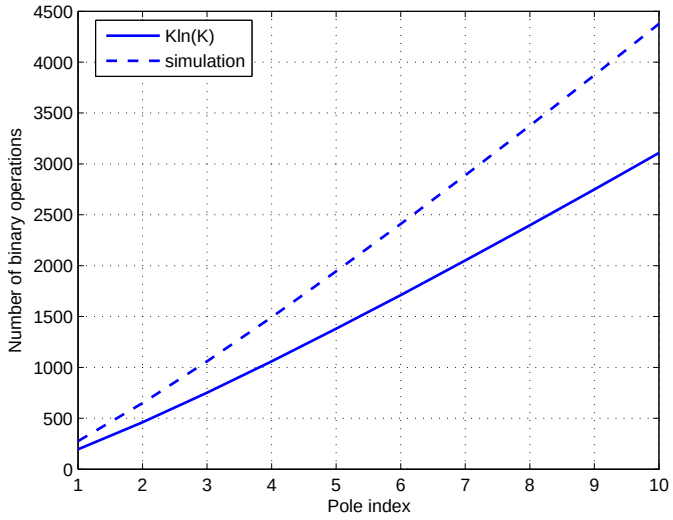


Figure 2.8: Decoding cost of LT codes versus pole index.

### 2.1.5 Raptor codes

Raptor codes [9] enjoys linear encoding and decoding costs, which outperforms LT codes that has $K \ln K$ encoding and decoding costs.

It is well understood that the overall encoding and decoding complexity of LT codes scales as $K \ln K$ because the average degree of the packets in the decoding graph was $\ln K$. Raptor codes employs an LT code with a roughly constant average degree $\bar{d} = 3$. This constant average degree will in return result in the linear encoding and decoding cost. However, a fraction of the source packets will not be connected to the graph if $K$ source packets are to be transmitted. The fraction, denoted by $\tilde{f}$, for $\bar{d} = 3$ is determined to be roughly 5%. How to deal with this fraction of packets are key of Raptor codes. The encoding and decoding procedure are described as follows.

- First pre-encode the $K$ source packets into $\tilde{K} = K/(1 - \tilde{f})$ packets using a traditional code that can correct erasures with $\tilde{f}$ erasure rate, such as LDPC

codes or Hamming codes, then apply LT codes to transmit these intermediate encoded symbols.

- At the receiver, more than $K$ received packets can recover $(1-\tilde{f})\tilde{K}$ of the intermediate packets, which is about $K$ packets. Then the same code employed in the precoding will now be used to decode and recover the input source symbols.

Figure 2.9 shows the encoding and decoding costs of Raptor codes. Here we use the LDPC codes adopted in [9] and $c = \delta = 0.01$ for the LT codes used in the second encoding stage. We still simulate our line network model with $L = 10$ and $\epsilon = 0.01$ as in last section. It is clear that the encoding and decoding costs are linear to the number of original packets $K$.

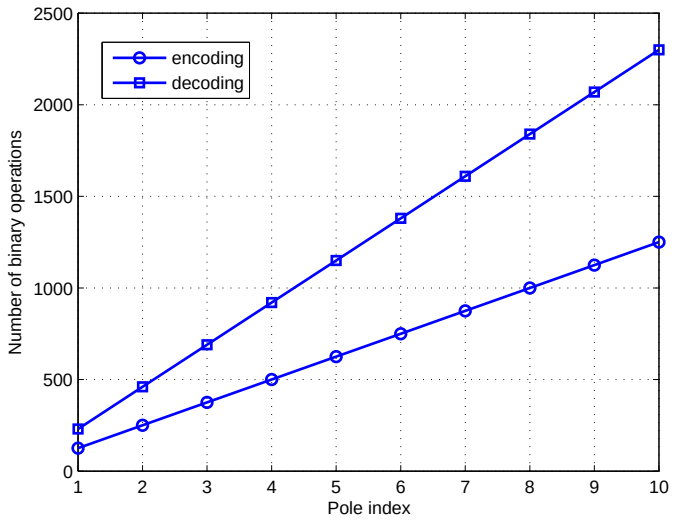


Figure 2.9: Encoding and decoding costs of Raptor codes.

## 2.2 Delay and Memory Requirement of Erasure Coding Schemes for Line Network

Section 2.1 has reviewed several erasure codes and simulation results have been provided for their overhead and encoding/decoding performance while being simply implemented in our proposed line network model, all of which deploy decoding and re-encoding at each relay.

In this section, we propose and compare two general packet processing strategies for line network, complete decoding and re-encoding (as performed in Section 2.1) and decode-at-destination. For both strategies, we investigate the delay and memory requirement and discuss how to apply different schemes according to different system requirements and nodes constraints.

The line network model we described in last section can be considered as a single line graph consisting of $L$ source nodes and a destination. The $L$ edges between the nodes corresponds to independent memoryless erasure channels with erasure probability over the $i$th link being $\epsilon_i$.

We ease the analysis by considering the only case when $L = 2$ in the same spirit of the analysis in [5], which is depicted in Figure 2.10. The generalization to cases with larger $L$ will be discussed. The source node $A$ encodes $K$ packets to create $N_1$ encoded output packets using a code $\mathcal{C}_1$ and sends them over the channel $A-B$. Node $B$ will receive on average $N_1(1-\epsilon_1)$ coded symbols over $N_1$ time slots. Here, we still assume that one packet is transmitted in one time slot. Then node $B$ will send $N_2$ packets, using a code $\mathcal{C}_2$. Note that the packets sent from node $B$ contains both the packets from $A$ and its own source packets, that is, node B functions as a source node as well as a relay node to relay the information from node $A$. Denote $N_T$ as the exact number of transmitted packets when all source packets can be recovered. If node

B finishes transmitting at time $N_T$, then node $C$ will receive on average $N_2(1-\epsilon_2)$ packets after $N_T$ time slots. Thus, from the perspective of node $A$, its own source packets will experience a delay of $N_T - 2K/C_{link}$ (assume $C_{link} = 1$) [5], where $C_{link}$ is the link capacity of one wireless channel.

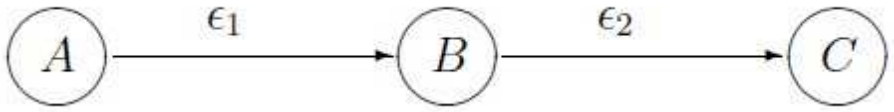


Figure 2.10: Line network model with three nodes

### 2.2.1 Simple Feedback Scheme

We first analyze the optimal delay and memory requirements relying on the simple feedback scheme with perfect feedbacks. In this scheme, no erasure coding is employed and the transmitted packets are the raw information packets. In particular, each node repeats transmission of each packet until it is successfully received at the destination and perfect feedback (zero error and zero time slot occupied) is assumed. It is easy to see that no coding scheme without using feedback can transmit in less time, i.e. the minimal delay occurs when we apply the simple feedback scheme without coding. This will be considered as the theoretical benchmark and be used to evaluate the coding schemes that we propose.

Note that our system differs from which in [5] where for our model, each node in the line network is generating information traffic, while [5] investigates single source case. However, we can follow the similar analysis approach in [5] and extend the results to the multi-source case. Note that in our line network, node $B$ (see Figure 2.10) is also a source node. For simplicity, we have $\epsilon_1 = \epsilon_2 = \epsilon$. Using the simple feedback scheme described before, node $A$ will roughly finish its transmission in $N_T = K/(1-\epsilon)$

(assuming $C_{link} = 1$). Define $x_i \in 0, 1, 2, \cdots$ as the number of received packets that still need to be sent at a time slot at node $B$. We can use a Markov chain with states $x_i$ to describe the situation of node $B$. At each time slot $i$, node $B$ requires $x_i$ storage space in addition to $K$ packets of its own. At each time slot, the state is either increased or decreased by 1 with equal probability $\epsilon(1-\epsilon)$ and remain unchanged with probability $1 - 2\epsilon(1-\epsilon)$. Applying the knowledge of random walk and Markov chain, after $N$ time slots, the system can be interpreted of a random walk over $N' = 2\epsilon(1-\epsilon)N$ steps. Thus the expected value of $x_N$ is the expected value of the absolute value of a random walk after $N'$ steps, which is $\mathbf{E}[x_N] = O(\sqrt{N'}) = O(\sqrt{\epsilon K})$ [5]. Then node $B$ will transmit the remaining $x_N + K$ packets in a time $N'_T = (x_N + K)/(1-\epsilon)$. Therefore, for this feedback scheme, from the perspective of node $B$ the expected memory requirement is $O(\sqrt{\epsilon K} + K)$ and from the perspective of node $A$ the 'delay' is $O((\sqrt{\epsilon K} + K)/(1-\epsilon))$ [5].

### 2.2.2 Packet Processing Strategies at Relay Nodes

In this section we describe and compare two possible packet processing strategies at relay nodes for a line network. Naturally, a complete decoding and re-encoding scheme can be applied, where each node, except for the first one, will completely decode the message from its previous node and then re-encode these packets together with its own packets. Conversely, another scheme would be each node, except for the first one, simply encodes what it received and its own message together and transmits these encoded packets to the next node, leaving all the decoding procedure to the destination node. We now will describe each in detail and their corresponding delay and memory requirements, and discuss when to use which kind of these two strategies according to the node capability and system constraints.

**Complete Decoding and Re-encoding**

This scheme uses a separate code for each of the $L$ links and has each node except for the first node completely decode and re-encode the incoming packets. However, from the perspective of the each node, its own packets will suffer an additional delay of roughly $N-K = K\epsilon/(1-\epsilon)$ time slots after each node it passes, where $N \approx K/(1-\epsilon)$ since we have proven that the required received packets are slightly more than $K$ in last section.

Therefore, when message of the first pole $R_1$ arrives at $R_2$, it needs $N = K/(1-\epsilon)$ time slots for successful decoding at $R_2$. Thus, the delay for $R_1$'s packets at $R_2$ is $N-K$. At $R_2$, the node decodes and re-encodes messages from both $R_1$ and $R_2$ and sends them to the next hop. It needs $2K/(1-\epsilon)$ time slots. When they reach $R_3$, for message from $R_1$, there is another delay of $2K/(1-\epsilon) - K$... In total, the delay for message from the first pole at the substation is:

$$D_1 = \left(\frac{K}{1-\epsilon} - K\right) + \left(\frac{2K}{1-\epsilon} - K\right) + \cdots + \left(\frac{LK}{1-\epsilon} - K\right) = \frac{L-1+2\epsilon}{2(1-\epsilon)} LK. \quad (2.17)$$

As such, the total delay for a message from $R_l$ $(l = 1, 2, \cdots, L)$ at the substation can be derived as

$$\begin{aligned} D_l &= \left(\frac{lK}{1-\epsilon} - K\right) + \left(\frac{(l+1)K}{1-\epsilon} - K\right) + \cdots + \left(\frac{LK}{1-\epsilon} - K\right) \\ &= \frac{2(L-l+1)K\epsilon + (L+l-2)(L-l+1)K}{2(1-\epsilon)}. \end{aligned} \quad (2.18)$$

Furthermore, each node will require a storage of the messages from preceding nodes and the storage space increases linearly to the index of the node. For example,

node $l$ will need an extra memory storage of $lK$ packets, which is larger than that if simple feedback scheme is used in Section 2.2.1.

Indeed, all the simulation results on the line network model for using different erasure coding schemes in previous sections are based on complete decoding and re-encoding protocol. Again, we apply systematic RLFC code for encoding and decoding to simulate the delay performance. Figure 2.11 shows the actual delay of the packets for each pole.

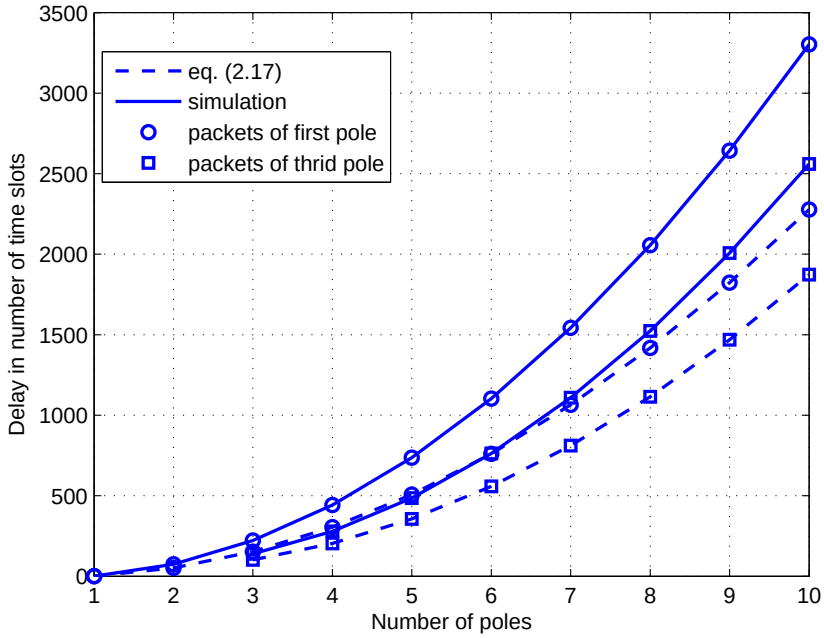


Figure 2.11: Delay performance of complete decoding and re-encoding scheme

The complete decoding and re-encoding scheme distributes the decoding tasks and decoding computational complexity to all the nodes in the line network (except for the first one). And it also imposes additional memory storage burden to these nodes. Therefore, it can be concluded that this scheme is suitable for the line network with relatively high computational capacity and storage space nodes.

**Decode-at-destination**

The decode-at-destination scheme leaves all the decoding procedure to the destination. We again apply the systematic RLFC for encoding at each node. Suppose every node keeps transmitting its own message prior to relaying packets from other nodes.

We still assume for simple approximate analysis it needs $N \approx K/(1-\epsilon)$ packets for K original packets to be successful decoded over one hop. Thus, in the first $N$ time slots, every node has sent out their own encoded packets as systematic packets. At $R_2$, only $N(1-\epsilon)$ packets from $R_1$ are received. In addition, $R_2$ forms an average of $N\epsilon = K\epsilon/(1-\epsilon)$ extra linear combinations of the systematic packets, and all these $N$ packets are transmitted in the following $N$ time slots. Therefore, there will be a total delay of $L(N-K) = LK\epsilon/(1-\epsilon)$ for a message from $R_1$ to the substation and 0 additional memory space. For $R_l$, the total delay is given by

$$D_l = (L-l+1)(N-K) = \frac{(L-l+1)K\epsilon}{1-\epsilon}, \tag{2.19}$$

and node $l$ will also need an additional memory storage of 0.

We consider two approaches to design the systematic codes mentioned above.

*Fixed-rate Codes*: A fixed-rate random systematic code consisting of $K$ packets and $K\epsilon/(1-\epsilon)$ parity coded bits will be used. In particular, Reed-Solomon code or Tornado code can be used. These fixed-rate codes enjoy the benefit of low encoding and decoding complexities. For example, Reed-Solomon codes need $K(N-K)$ operations to encode or decode and Tornado codes need $O(N\ln(N/K))$ operations.

*Systematic Random Linear Fountain Codes*: The coding algorithm is described in Section 2.1.3. It simply transmits random linear combinations of all the received packets thus far. The systematic RLFC has the benefit of optimal delay but requires high decoding complexity which is $(K\epsilon)^2\log(K\epsilon)$ as we discussed before.

All results above are analyzed based on an average situation and can not garuantee the success of information recovery at the substation. There might be higher risks of unsuccessful decoding at the substaion for decode-at-destination than complete decoding and re-encoding scheme. Figure 2.12 shows delays for both fixed-rate codes (RS codes in our simulation) and the systematic RLFC.

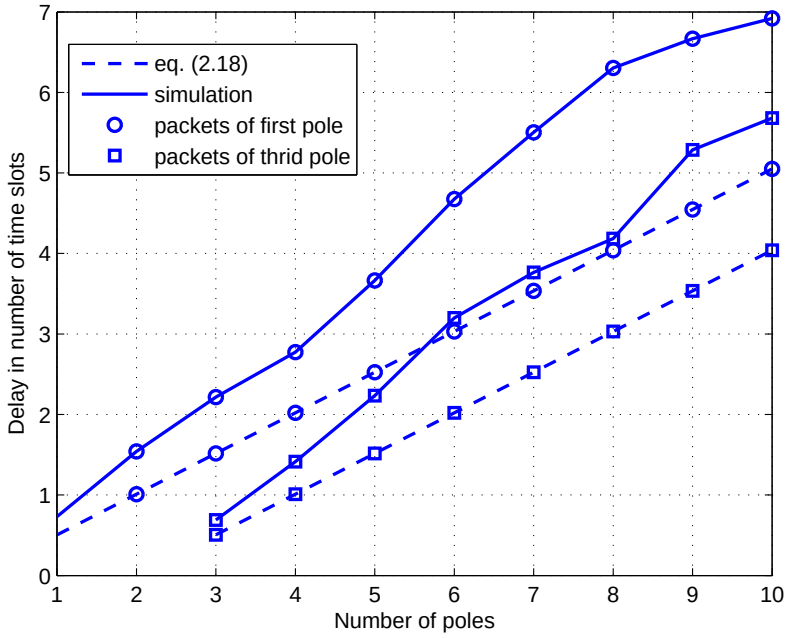


Figure 2.12: Delay performance of decode-at-destination scheme

## 2.3 Conclusions

In this chapter, we have reviewed several classical erasure codes and compared their performance on overhead and encoding/decoding cost. Analysis and simulation results have shown that systematic RLNC gives the least overhead and Raptor codes outperform other codes by its linear encoding and decoding cost.

We have also investigated possible erasure codes for the solution of our line net-

work. Two general packet processing strategies at relays have been proposed: complete encoding and decoding and decode-at-destination. Different schemes suit different systems in terms of node capacity and delay and memory space constraints. Simulation results have been included.

However, in this chapter, we have assumed the channel links only suffer from erasures, where the erasure probability $\epsilon$ was the only channel parameter. In practice, due to the fading effect, wireless channels suffer from path loss and will incur errors in the received symbols. In next chapter, we will study how to implement erasure codes in fading channels.

# Chapter 3

# Erasure Codes in Combination with Error Correction/Detection Codes for Wireless Line Network

In the previous chapter, we assume the channels are simply erasure channels. Thus, any type of erasure codes will be able to recover the original source symbols, though they may incur different overhead and encoding and decoding complexity. However, errors can occur in a wireless channel due to many phenomena. For example, electronic devices at the transceiver or amplifiers can cause thermal noise; signals transmitted in adjacent channels can cause inter-symbol interference; path-loss and shadowing effects due to radio propagation, etc. In this chapter, we consider Rayleigh fading channels and implement erasure codes in combination with cyclic redundancy check (CRC) codes to detect errors. We also propose a scheme to adopt Reed-Solomon codes as an outer code to correct bit errors. In addition, we apply an adaptive RS coding scheme to maintain good performance over all SNR regions.

## 3.1 Rayleigh Fading Channel Model

Obstacles in a propagation environment can reflect or scatter the electromagnetic waves, resulting in the multi-path effect. A signal transmitted from a transmitter could arrive at the receiver through different paths with different attenuation factors and signal delay. Due to the time-variant nature of the wireless channel, the expression of the received signal is given by [11]

$$x(t) = \sum_n \alpha(t) s(t - \tau_n(t)), \tag{3.1}$$

where $s(t)$ is the transmitted signal, $\alpha_n$ is the attenuation factor for path $n$, and $\tau_n$ is the time delay for path $n$.

Assuming $s(t) = Re[s_l(t)e^{j2\pi f_c t}]$, then we have

$$x(t) = Re\left\{\left[\int_{-\infty}^{\infty} \alpha(\tau; t) e^{-j2\pi f_c \tau} s_l(t - \tau) d\tau e^{j2\pi f_c t}\right]\right\}. \tag{3.2}$$

Let $s_l(t) = 1$ and $\theta_n(t) = -2\pi f_c \tau_n(t)$, then the received signal is obtained as

$$r_l(t) = \sum_n \alpha_n(t) e^{j\theta_n(t)}. \tag{3.3}$$

It is clear that the signals from different paths arrive at the receiver with time-variant attenuation factors and time-variant phases. If the number of paths that a signal travels through is large enough, complex Gaussian random process can be used to characterize the received signal at the receiver. The most widely adopted models are Rayleigh, Ricean and Nakagami-$m$ fading models. Ricean fading model, the envelope of which obeys Ricean distribution, is applied when there is a line of sight path between the transmitter and the receiver. Nakagami-$m$ fading model is a more general model which covers Ricean fading and Rayleigh fading as special cases.

In this chapter, we adopt the Rayleigh fading model, the envelope of which is drawn from Rayleigh distribution:

$$P_R(r) = \frac{r}{2\sigma^2} e^{\frac{-r^2}{2\sigma^2}}, \tag{3.4}$$

where $2\sigma^2$ is the average received signal power.

## 3.2 Systematic Raptor Code

The Raptor code is introduced in the previous chapter. Here we adopt a systematic version of the Raptor code, which is standardized in the 3rd generation partnership project (3GPP).

The design of systematic Raptor code is given in [9]. For simplicity, we summarize an overall encoding and decoding approach. For the details of how the encoding and decoding algorithm is designed and the theoretical proof of the design, readers are referred to Section 8 in [9]. Assume we have a Raptor code with parameter $(K, \mathcal{C}), \rho(x)$. An encoding algorithm accepts $K$ input symbols $x_1, \ldots, x_K$ and produces a set $\{i_1, \ldots, i_K\}$ of $K$ distinct indices between 1 and $K(1+\epsilon)$ and an unbounded string $z_1, z_2, \ldots$ of output symbols such that $z_{i_1} = x_1, \ldots, z_{i_K} = x_K$. The indices $i_1, \ldots, i_K$ are referred to as the systematic positions. The output symbols $z_{i_1}, \ldots, z_{i_K}$ are referred to as systematic output symbols, and the other output symbols are called non-systematic output symbols. The whole process is summarized as follows. First the systematic positions will be calculated along with an invertible binary $K \times K$ matrix $R$. The calculation process is to sample $K(1+\epsilon)$ times from the distribution $\rho(x)$ to obtain vectors $v_1, \ldots, v_{K(1+\epsilon)}$ and apply a decoding algorithm to these vectors. The matrix $R$ is the product of the matrix $A$ consisting of the rows $v_{i_1}, \ldots, v_{i_K}$ and a generator matrix of the pre-code, which is a low density parity check

(LDPC) code adopted in 3GPP. With the matrix $R$, the vectors $v_1, \ldots, v_{K(1+\epsilon)}$ and the systematic positions $i_1, \ldots, i_K$ available, we can describe the encoding algorithm as follows [9].

*Encoding Algorithm:*

Input: Input symbols $x_1, \ldots, x_K$.

Output: Output symbols $z_1, z_2, \ldots$, where the symbol $z_i$ corresponds to the vectors $v_i$ for $1 \leq i \leq K(1+\epsilon)$ and where $z_{i_j} = x_j$ for $K \geq 1$.

1. Calculate $y = (y_1, \ldots, y_K)$ given by $y^\top = R^{-1}x^\top$.
2. Encode $y$ using the generator matrix $G$ of the precode $\mathcal{C}$ to obtain $u = (u_1, \ldots, u_n)$, where $u^\top = Gy^\top$.
3. Calculate $z_i = v_i u^\top$ for $1 \leq i \leq K(1+\epsilon)$.
4. Generate the output symbols $z_{K(1+\epsilon)+1}, z_{K(1+\epsilon)+2}, \ldots$ by applying the LT code with parameter $K, \rho(x)$ to the vector $u$.

The decoding process for the systematic Raptor code consists of two steps. First the intermediate symbols $y_1, \ldots, y_K$ are obtained by decoding for the original Raptor codes. Then these intermediate symbols are transformed back to the input symbols $x_1, \ldots, x_K$ using the matrix $R$.

## 3.3 Wireless Line Network with Erasure Codes and Error Detection Codes

We apply systematic Raptor codes to the wireless line network with Rayleigh fading channels.

To deal with the fading channel, we make sure that all the received symbols used as the input of the systematic Raptor code decoder are error free. Therefore, we

use CRC in addition to the systematic Raptor code. For each Raptor coded packet, we calculate the CRC and append it to the end of the Raptor code symbols in each packet in order to detect any error in the received symbols. The widely used 32-bit CRC proposed in [12] is adopted in our model. At the receiver side, after receiving a packet, the CRC of this packet is recalculated and compared with the CRC received in the packet. If the CRC check suggests an error in the received packet, the whole packet will be discarded.

The probability that a packet is lost/discarded in a Rayleigh fading channel can be calculated. Let $\gamma_r$ be the received SNR of a channel. Assume Binary Phase Shift Keying (BPSK) modulation is used. Thus, the bit error rate (BER) is [11]

$$P_b(\gamma_r) = Q(\sqrt{2\gamma_r}). \tag{3.5}$$

Define the channel SNR $\gamma_c = E_t/N_0$, where $E_t$ is the transmitted energy of each BPSK symbol, and $N_0$ is the noisy power.

Let $\alpha$ be the path loss exponent and $d$ be the distance from the transmitter to the receiver, i.e., the distance of one hop in our model, then

$$\gamma_r = \gamma_c \times \frac{h^2}{d^\alpha}, \tag{3.6}$$

where $h$ is the channel coefficient.

We can now express the bit error rate $P_b$ as

$$P_b(h) = Q\left(\sqrt{\frac{2\gamma_c h^2}{d^\alpha}}\right). \tag{3.7}$$

Considering Rayleigh fading, the probability density function (pdf) of the channel

coefficient is [11]

$$p(h) = 2he^{-h^2}. \tag{3.8}$$

Let $s_p$ be the packet size of coded Raptor code packet in terms of bits and $s_c$ the CRC length in terms of bits. Therefore, the packet loss rate $P_{pl}$ is

$$P_{pl}(h) = 1 - (1 - P_b(h))^{s_p+s_c}. \tag{3.9}$$

Together with eq. (3.8), we have the average packet loss rate given $\gamma_c$ as

$$\bar{P}_{pl} = \int_0^\infty p(h) \left[1 - (1 - P_b(h))^{s_p+s_c}\right] dh. \tag{3.10}$$

Eq. (3.10) suggests the packet loss rate is related to both the channel SNR and the packet size. It is obvious that a smaller packet size incurs less packet loss rate. However, with the adoption of CRC, smaller packets will cause higher overhead. For practical transmission, [13] has recommended a proper set of selected packet size. Each coded systematic packet contains $T = 6$ symbols, each symbol is of 84 bytes length. Suppose a CRC code has $s_c = 32$ bits. Therefore a coded packet after CRC has $s'_p = 6 \times 84 \times 8 + 32 = 4064$ bits. With these parameters, the packet loss rate versus different channel SNR is shown in Figure 3.1. In this illustrative figure, we let $d = 1, \alpha = 2$.

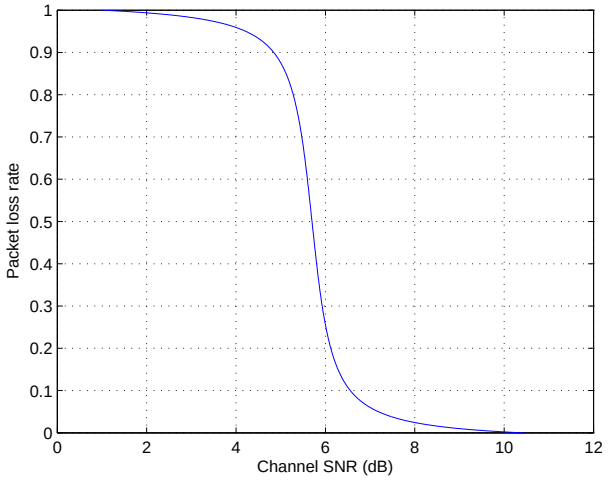


Figure 3.1: Average packet loss rate (eq. (3.10)) of systematic Raptor code under different channel SNR.

In this thesis we apply systematic Raptor codes standardized in [14, Annex B]. The decoding failure probability of standardized systematic Raptor code is investigated in [15], given by

$$P_f(E) = 0.85 \times 0.567^E, \tag{3.11}$$

where $E = N_R T - K$ and $T = 6$ *symbols/packet* as supposed, is the overhead as we defined in the previous chapter. In the current settings, $N_R$ is the number of received systematic Raptor packets and $K$ is the number of the source symbols. The transmitter keeps transmitting Raptor encoded packets until the decoder at the receiver side successfully decodes all the original source packets and sends back an acknowledgment. The probability that decoding is successful with exact $E$ overhead $P_{oh}$ can be calculated as

$$P_{oh}(E) = (1 - P_f(E))\Pi_{i=0}^{E-1} P_f(i). \tag{3.12}$$

Suppose $N_T$ packets have been transmitted and $N_R$ packets have been successfully received when all the $K$ source symbols are successfully recovered. We have $N_T \geq N_R \geq \frac{K}{T}$. We now proceed to calculate the successful decoding probability. Since the $N_T$-th transmitted packet happens to be sufficient for recovering all the source symbols, we have to make sure in the previously transmitted $N_T - 1$ packets, $N_R - 1$ packets have been successfully received. More over, the decoder is able to decode all the source symbols from the overall successfully received $N_R T$ symbols. Therefore, the successful decoding probability when $N_T$ packets have to be transmitted is

$$P_{sd}(N_T, h) = \sum_{N_R=\frac{K}{T}}^{N_T} \binom{N_T - 1}{N_R - 1} \left[(1 - P_{pl})^{N_R} P_{pl}^{N_T - N_R} P_{oh}(N_R T - K)\right], \tag{3.13}$$

where $P_{pl}$ is the packet loss rate.

We want to evaluate the performance of our scheme. However the BER is not a good system metric since in our scheme, as long as the transmitter can send enough packets, the overall source symbols can be recovered at some point. Here we use average code efficiency $\eta$ as the evaluation criteria, which is the ratio between the number of original source symbols and the number of total transmitted symbols. It is obvious that $\eta$ is related to the decoding overhead.

With eqns. (3.9), (3.11)-(3.13) available, the average code efficiency $\eta$ can be computed as

$$\eta = \int_{h=0}^{\infty} p(h) \sum_{N_T=\frac{K}{T}}^{\infty} \frac{K}{N_T T} P_{sd}(N_T, h) dh \tag{3.14}$$

Note that in the above analysis, only one hop is considered. In the previous chapter, we proposed two different ways to implement the fountain codes in multi-hop, multi-source line network: complete decoding and re-encoding; decode-at-destination. For complete decoding and re-encoding scheme the average code efficiency can be calculated for each link. For the decode-at-destination the average code efficiency

can be calculated for the last link.

## 3.4 Combination of Systematic Raptor Code and Forward Error Correcting Code

We can see from Figure 3.1 that the system performance is severely deteriorated in terms of packet loss rate when the received SNR is under 6 dB. It is because that when the received SNR is low, there is a good chance that the received packet will contain an bit error. In this case, since the CRC scheme is adopted, the whole packet will be discarded. Intuitively, we can use a forward error correcting (FEC) code as an outer code applied after the systematic Raptor code and CRC code. FEC code is generated to correct errors and consequently improves the whole system performance.

The standardized symbol size recommended by 3GPP is measured in bytes, thus a RS code based on GF(8) will be a suitable option for the outer code. Therefore, in this thesis, we use RS code as a type of FEC code combined with the systematic Raptor code in the wireless line network.

### 3.4.1 Reed-Solomon Code

We introduce fundamentals of RS code in this subsection [7]. Each symbol in a RS codeword consists of $b$ bits of source data. All the arithmetic operations of RS encoding and decoding follow the arithmetic operations defined in a specific GF depending on the number of bits in one RS symbol [7]. RS is a type of systematic codes. The parity check symbols are appended to the $K$ original information symbols to correct the erroneous or erased symbols in the codeword. Let the length of a RS codeword be $N$. The minimum distance of the RS code is $d_{min} = N - K + 1$. Then this RS code can correct up to $m = \lfloor \frac{N-K}{2} \rfloor$ erroneous symbols or detect $2m$ erroneous symbols in

a codeword. The generator polynomial of an RS code is

$$g(x) = (x - a)(x - a^2) \cdots (x - a^{2m}), \tag{3.15}$$

where $a$ is a primitive element of the GF. The codeword polynomial $c(x)$ is given by

$$c(x) = g(x) * m(x), \tag{3.16}$$

where $m(x)$ is the original message block.

The decoding procedure of an RS code consists of two steps. First the $2m$ roots of $g(x)$ are substituted into the received polynomial $r(x)$, and the $2m$ syndromes can be computed. Next the positions of the erroneous symbols in the codeword are determined. Two widely used approaches adopted here are Berlekamp-Massey and Euclid's algorithms [16]. Then the values of the erroneous symbols can be solved from the equations.

### 3.4.2 Combined Coding Scheme

In this subsection, we design the combined coding scheme in details. Consider the symbol parameters standardized in 3GPP. Each source symbol is of 84 bytes length. Therefore, we choose RS $(N, 88)$ codes which are RS $(255, 88)$ codes shortened by $255 - N$ bytes. Note that $88 = 84 + 4$ where the last 4 bytes are the 32-bit CRC. The encoding process is shown in Figure 3.2.

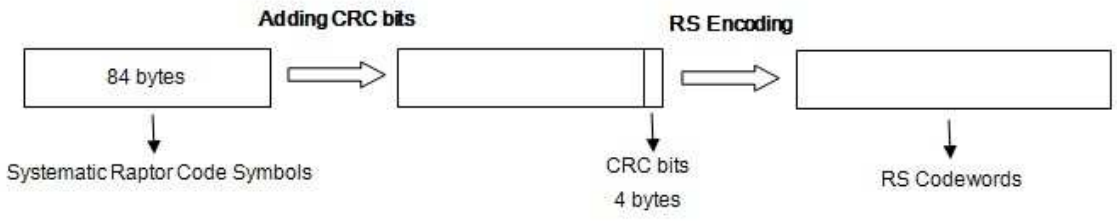


Figure 3.2: CRC and RS encoding procedure.

The combined coding scheme is summarized as follows. First, we calculate the 32-bit CRC for every systematic Raptor code symbol and attach them to the end of the Raptor symbol. Then the total 88 bytes are encoded using RS $(N, 88)$ codes. Note that for each standardized source packet there will be $T = 6$ source symbols (each source symbol has 84 bytes), so after the RS coding, there will be also 6 RS codewords for each systematic Raptor code packet. These RS codewords will be sent to the receiver and the decoding process begins. First each RS codeword are decoded with the RS decoder and the 32-bit CRC for each decoded systematic Raptor code symbol will be obtainted. If there is a mismatch in the CRC check, the whole RS codeword will be discarded. The remaining systematic Raptor code symbols are therefore error-free and will be used for the systematic Raptor decoding process. After all the source symbols are decoded successfully, the receiver feeds back an acknowledgment to the transmitter. The transmitter keeps sending the information packet until it receives the acknowledgment. Figure 3.3 illustrates the whole process.

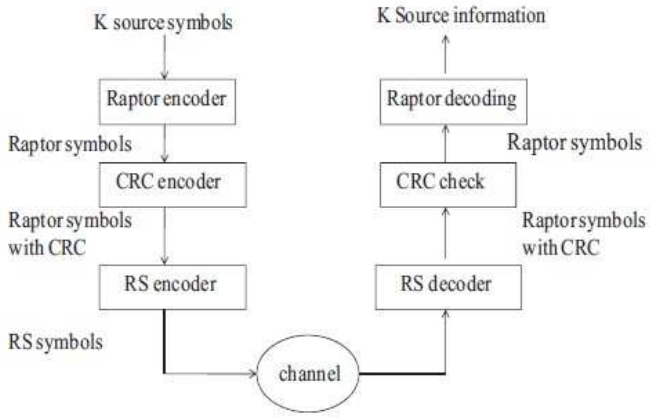


Figure 3.3: CRC and RS encoding process of the combined coding scheme.

We now analyze the packet loss rate of the coding scheme using RS code as an outer code. For the simplicity of analysis, suppose RS $(N, K)$ codes are used. Then the RS code can correct up to $m = \lfloor \frac{N-K}{2} \rfloor$ erroneous symbols. Note that the symbol we are talking about now is with respect to RS codes and therefore actually a byte. Hence the probability that there is at least one error in an RS symbol can be computed as

$$P_{rs} = 1 - (1 - P_b)^8, \tag{3.17}$$

where $P_b$ is given in eq. (3.7).

By using the RS code, up to $m$ errors can be corrected. Thus the error probability of RS decoding is[1]

$$P_e = 1 - \sum_{i=0}^{m} \left[ \binom{N}{i} P_{rs}^i (1 - P_{rs})^{N-i} \right] . \tag{3.18}$$

[1]If less than $m$ errors are received, the RS code we applied is guaranteed to correct all the errors. However, if more than $m$ errors are received, the decoding probability is in general quite difficult to calculate, or even to estimate. Readers are referred to [17] for details. Here for simplicity, we assume if more than $m$ errors are received, the codeword error occurs, which is a conservative assumption. Therefore, indeed eq. (3.18) is an upper bound of the error decoding probability. This is confirmed by the code efficiency simulation results, where for most SNR regions, the simulation results are lower bouned by the theoretical results.

There are $T = 6$ RS codewords in one packet and therefore the packet loss rate is now

$$P_{pl} = 1 - (1 - P_e)^6 \,. \tag{3.19}$$

Substituting (3.19) into (3.14) we can calculate the code efficiency of the scheme using RS code as an outer code. In next section, we present extensive simulation results for our proposed scheme.

## 3.5 Simulation Results

We assume flat slow Rayleigh fading channels, where the channel coefficient is constant over the transmission of a block of information message. For simplicity, we assume the channels for different links in the wireless line network are independent and identically distributed (i.i.d.). We adopt BPSK modulation scheme. The distance between one node and its neighbors are assumed to be $d = 1$ for all nodes in the network and the path loss exponent is assumed to be $\alpha = 2$.

### 3.5.1 Simulation Results with RS code

In the simulation, we assume the number of symbols in a block of source data is $K = 1000$. Denote, in the transmission of the $i$-th block of source data, the number of systematic Raptor parity symbols by $N_{R_i}$, and denote the number of CRC checksum symbols by $N_{C_i}$ and denote the number of RS parity symbols by $N_{RS_i}$. Thus, the total number of symbols transmitted in every block of data is given by

$$N_i = K + N_{R_i} + N_{C_i} + N_{RS_i}. \tag{3.20}$$

In each simulation 1000 blocks of data will be transmitted. Therefore, the average

code efficiency can be computed as

$$\eta = \frac{1000K}{\sum_{i=0}^{1000} N_i}. \tag{3.21}$$

We calculate the average code efficiency over 2000 simulations.

The simulation results for the code efficiency of the complete encoding Figure 3.4. We use the RS $(90, 88)$ code as the outer code. Analytical results are compared with the simulation results. It can been seen that the analytical results generally matches with the simulation results.

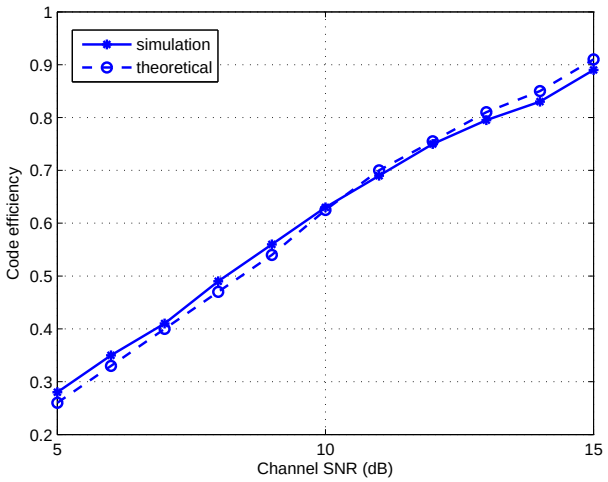


Figure 3.4: Code efficiency of systematic Raptor code with RS $(90, 88)$ code as an outer code under different channel SNRs.

We apply three different RS codes, RS $(90, 88)$, RS $(168, 88)$, RS $(208, 88)$ in Figure 3.5. Three observations are worth noting: 1) the code efficiency with the use of all three RS codes outperforms that without the use of RS codes as outer codes; 2) in the low SNR region, the lower-rate RS $(208, 88)$ codes performs the best; 3) in the high SNR region, the higher-rate RS $(90, 88)$ codes performs the best. The first observation is consistent to what we expect. The second and the third observations

are straightforward to explain. When the channel SNR increases, fewer errors occur in the received RS symbols and therefore the high-rate RS codes will be enough for error correction. The high-rate RS codes also helps to improve the overall code efficiency. On the other hand, while the channel SNR is low, high-rate RS codes are not strong enough and will incur large number of packet loss, which in return deteriorates the total code efficiency.

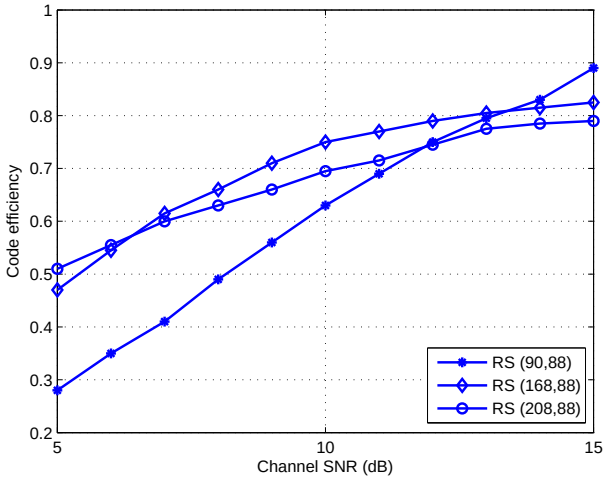


Figure 3.5: Code efficiency of three different rate RS codes as outer codes under different SNRs.

Based on the last two observations mentioned above, we will propose an adaptive RS coding scheme which will adaptively choose an RS code with a proper rate according to the channel SNR in order to achieve high code efficiency over all SNR regions.

## 3.6 Adaptive RS Coding Scheme

### 3.6.1 Adaptive RS Coding Scheme with CSI at the Transmitter (CSIT)

The combined coding scheme described in the previous section is based on the assumption that the transmitters have no knowledge of the CSI. Therefore, the chosen rate of the RS code might not perform well over all SNR regions. As in the simulation results in Figure 3.5, for different SNR regions, the best RS codes are of different rates. We propose an adaptive RS coding scheme in this section to improve the code efficiency over all SNR regions based on the assumption that the CSI is available to the transmitter.

### 3.6.2 RS Code Rate Selection

#### Perfect CSI at the Transmitter

If the transmitter has the knowledge of the perfect CSI, it can calculate the expected received SNR as

$$\gamma_r = \gamma_c \frac{h^2}{d^2}. \tag{3.22}$$

Note that eq. (3.14) computes the average code efficiency. To select an RS code with a given CSI, we pay attention to instantaneous code efficiency, that is, eq. (3.14) without averaging over all values of the channel coefficient $h$. Thus, we express the instantaneous code efficiency as follows.

$$\eta(\gamma_c, n) = \sum_{N_T=\frac{K}{T}}^{\infty} \frac{K}{N_T T} P_{sd}(N_T, h). \tag{3.23}$$

Therefore, for a given channel SNR, we select the RS $(n, 88)$ code according to

the following criteria.

$$n = \arg\max \eta(\gamma_c, n). \tag{3.24}$$

**Estimated CSI at the Transmitter**

In practice, the CSI available at the transmitter is estimated from the low-rate feedback or the transmission of the previous block of data, when the channel is experiencing slow fading. Particularly, after the transmission of the previous block of data, the receiver sends back feedback containing the channel coefficient for the previous block of data. Upon receiving the feedback, the transmitter estimates the current channel coefficient based on the CSI of the transmission of the previous block of data. There are many ways to model the two correlated fading channels. In this thesis, we apply the Autoregressive (AR) model [18]. We assume the maximum doppler frequency $f_d = 1Hz$ and the interval between the transmissions of two adjacent blocks of data is $0.1s$. Therefore, by applying eqns. (2), (4), (7) in [18], we can obtain a second-order AR model for the correlated Rayleigh fading channels as

$$X_j = 1.762X_{j-1} - 0.95X_{j-2}, \tag{3.25}$$

where the index $j$ represents the sequence of the blocks of data.

After the channel estimation, we again apply the selection criteria in eq. (3.24) to determine the RS rate.

## 3.7 Simulation Results

Figure 3.6 shows the simulation results for our model with adaptive RS coding scheme. It can be seen that the system where the transmitter with perfect CSI performs the best over all SNR regions and the simple estimation method used above also performs

well compared to other three fixed-rate RS coding schemes.

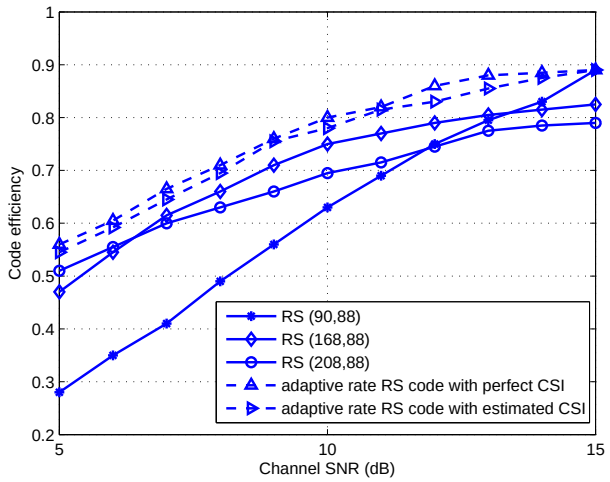


Figure 3.6: Comparisons of code efficiency of fixed-rate RS codes and adaptive rate RS codes as outer codes under different SNRs.

## 3.8 Conclusions

In this chapter, we investigate our system model considering erasures as well as errors. To deal with the fading channel, we implement erasure codes with the help of cyclic redundancy check codes to detect errors. We also propose a scheme to adopt Reed-Solomon codes as an outer codes to correct the errors to increase the overall code efficiency. Adaptive transmission schemes are applied to improve the system performance for all SNR regions.

# Chapter 4

# Network Coding in Wireless Line Network

In the previous chapter, to deal with the erroneous channels, we utilized a forward error correction code, RS code, as an outer code. This approach can be viewed as a straightforward way to apply the traditional error control codes in the point to point communication. Recently, a new type of coding scheme, network coding, which generalizes the traditional error control coding from the one-link scenario to the network scenario where the source and the destination are connected through a group of nodes which form a network. The information flow goes from the source through different links in the network to the destination. Due to the fact that the essence of network coding is in the same spirit of the traditional error control codes, it is also capable of correcting errors or erasures.

In this chapter, we analyze our wireless line network model and investigate coding schemes from a general network coding perspective. Network coding [19], [20] is a powerful tool for disseminating information in networks, yet it is susceptible to packet transmission errors caused by noise or intentional jamming. In a multi-link network, a

single error in one received packet would typically fail the entire transmission when the erroneous packet is combined with other received packets to deduce the transmitted message, which happens to be the situation when using erasure codes.

Network coding with proper design is capable of detecting and correcting the errors. It was first introduced for satellite communication networks in [21] and then fully developed in [22]. The reason why it can have the error-correction function in network scale is its relationship with the traditional point-to-point error-correction codes, i.e., algebraic coding. In fact, algebraic coding can be viewed as an instance of network coding. Readers are referred to [22] for details.

The chapter is organized as follows. First some fundamentals of network coding will be introduced. Then we investigate the error correction network coding in single source single path network. Then we extend the mathematical model and results to the multi-source single path network, which is indeed our line network model. Simulation results are followed.

# 4.1 Introduction

## 4.1.1 Network Coding

The simplest example for network coding has come to be known as the *butterfly network* [21], depicted in Figure 4.1. There is a single source $s$, which produces bits $A$ and $B$, and two destinations $t_1$ and $t_2$, both of which request both bits $A$ and $B$. As the same data must be transmitted to multiple destinations, this transmission type is known as multicasting. Each link in the network has capacity to transmit a single bit at a time. If no coding is allowed, then, as we can see from Figure 4.1a, there will be a "bottleneck" (congestion) at node $v$, as only one packet of either $A$ or $B$ can be transmitted through its outgoing link $l$. Packets would then be placed in a queue, to

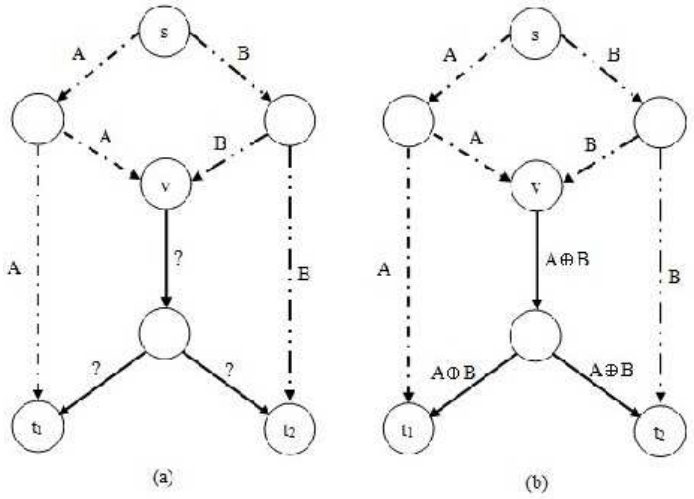


Figure 4.1: Butterfly network model.

recover both packets, two transmissions over link $l$ are required.

On the other hand, if coding is allowed, then node $v$ can simply transmit the XOR of bits $A$ and $B$, as illustrated in Figure 4.1b. This allows destination $t_1$ to recover $B = A \oplus (A \oplus B)$ and destination $t_2$ to recover $A = B \oplus (A \oplus B)$. Thus, by overcoming congestion, network coding is able to increase the throughput of a communication network. To achieve the capacity of a multicast network in general, it is strictly necessary to use network coding.

In contrast to the simple example of Figure 4.1, real networks can be hugely complex. If network coding is to be widely used, efficient coding (and decoding) operations must be developed so that the benefits of network coding can be obtained without increasing the implementation costs to a prohibitive level.

In this context, two contributions can be said to have raised network coding from a theoretical curiosity to a potential engineering application: *linear network coding* [22, 23] and *random linear network coding* [20, 24]. With linear network coding, all coding operations at nodes are constrained to be linear combinations of packets over a finite-field. The importance of linear operations is that they are arguably the simplest

possible to perform in practice, and the fundamental contribution of [22] is to show that no loss in multicast capacity is incurred if the field size is sufficiently large. What exactly to choose as the coefficients of these linear combinations, however, remains a problem. The contribution of [20] shows that a distributed design is sufficient in most cases. More precisely, if nodes choose coefficients uniformly at random and independently from each other, then no capacity is lost with high probability if the field size is sufficiently large. The actual coefficients used can be easily recorded in the packet headers. As decoding corresponds to performing the well-known algorithm of Gaussian elimination, random linear network coding becomes a practical approach that can potentially be implemented virtually over any network.

### 4.1.2 Error Control

How elegant and compelling, the principle of network coding is not without its drawbacks. Network coding achieves its benefits, essentially, by making every packet in the network statistically dependent on almost every other packet. However, this dependence creates a problem. What if some of the packets are corrupted? As Figure 4.2 illustrates, corrupted packets may contaminate other packets when coded at the internal nodes, leading to an error propagation problem. Indeed, even a single corrupted packet has the potential, when linearly combined with legitimate packets, to affect all packets gathered by a destination node. This is in contrast to routing (no coding), where an error in one packet affects only one source-destination path.

Thus, the phenomenon of error propagation in network coding can overwhelm the error correction capability of any classical error-correcting code used to protect the data end-to-end. “Classical” here means designed for the Hamming metric. In short, we might say the Hamming metric is not well-suited to the end-to-end channel induced by network coding. Thus, novel coding techniques are needed to solve this

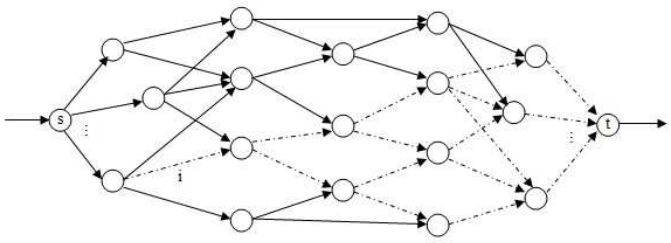


Figure 4.2: Error propagation.

new error correction problem.

At this point, it is necessary to point out that the physical-layer code is not enough to address the above mentioned packet corruption problem. One might argue that the corrupted packets would be detected and immediately rejected by the lower network layers leading to an erasure rather than an undetected error. However, it must be noticed that the physical-layer code may not always be perfect. Very reliable physical-layer codes require very long block lengths, and there are several situations where a very short block length is required for practical reasons [25]. Thus, we must be able to cope with eventual packet errors, without having to drop and request retransmission of all the received packets.

Another possible source of errors might be the presence of an *adversary*, i.e., a user who does not comply with the protocol. Since the adversary injects corrupt packets at the application layer, its effects cannot possibly be detected at the physical layer. Adversarial error represents an even more severe threat than random errors, since they can be specially designed to defeat the end-to-end error-correcting code.

Research on error-correcting codes for network coding started with [26–30], which investigated fundamental limits for adversarial error correction under a deterministic (i.e., non-random) network coding setting. Code constructions for random network coding were later proposed in [31, 32]; these constructions rely on using arbitrarily

large packet length and field size. In contrast, the framework proposed in [33, 34] is valid for any field or packet size.

In this chapter, we first review the results in [35] that for a given realization of error and erasure locations, successful decoding can be characterized in terms of the rank of certain matrices that correspond to useful and erroneous information received at the sink node. The probability of successful decoding for random coding and routing strategies on a simple one source multiple hop subgraph is given. Then we extend the results to the line model we are interested in. We derive the probability of successful decoding as well as the capacity region.

## 4.2 Single source network single path network error correction

In this section, we study single-source single-path network error correction. First we review several results on the necessary conditions for successful decoding subject to the relationship between the number of network errors and erasures and the minimum code distance. We then simplify our line network model to a single-path graph with single source and apply the existing results to derive the decoding probability.

We consider a single-source network model over an acyclic network $\mathcal{G}$ with source $\mathcal{S}$ and a set of sink nodes $\mathcal{T}$. We define that an erasure on a network link occurs when no packet is received over a link, and that an error occurs when a packet of arbitrary value is received.

Now we define the necessary notations to facilitate the analysis following [34]. The set of all possible values of packets transmitted and received in the network can be represented by the vector space, denoted by $V$, of length-$b$ vectors over the finite field $\mathbf{F}_q$. Denote the set of all subspaces of $V$ by $\mathcal{P}(V)$. Thus, an original source message

can be represented by a nonempty subset of $\mathcal{P}(V)$, where each source message $U \in \mathcal{C}$[1] is a subspace of constant dimension $K$, denoted by $\dim(U) = K$. According to the definition of coding [7], in order to transmit the source message $U \in \mathcal{C}$, the source transmits a set of packets whose corresponding vectors span $U$. The transmitted packets go through the channel from the source to the sink. In this channel, define the erasure operator and the error space [34].

**Definition 4.** *An operator channel associated with the vector space $V$ is a channel with input and output alphabet $\mathcal{P}(V)$. The channel input $U$ and the channel output $U'$ are related as*

$$U' = \mathcal{H}_k(U) \oplus E \tag{4.1}$$

*where $\mathcal{H}_k(U)$ is an erasure operator, $E \in \mathcal{P}(V)$ is an arbitrary error space, and $\oplus$ denotes the direct sum. If $\dim(U) > k$, the erasure operator $\mathcal{H}_k(U)$ acts to project $U$ onto a randomly chosen $k$-dimensional subspace of $U$; otherwise, $\mathcal{H}_k(U)$ leaves $U$ unchanged. If the erasure operator $\mathcal{H}_k(U)$ satisfies $\dim(U) - \dim(\mathcal{H}_k(U)) = \rho$, we say that $\mathcal{H}_k(U)$ corresponds to $\rho$ erasures and therefore $\rho = K - k$. The dimension of $E$ is called the error norm and define $t = \dim(E)$.*

Although the above definitions from [34] use $t$ and $\rho$ to refer the number of errors and erasures, they are essentially the number of dimension addition and deletions in the concept of subspaces. Thus, to avoid confusion, we refer to $t$ as the number of additions, and $\rho$ as the number of deletions. The distance between two spaces $U_1, U_2$ is defined as

$$d(U_1, U_2) = \dim(U_1 + U_2) - \dim(U_1 \cap U_2). \tag{4.2}$$

[1]Note that the source message here is a codeword after source coding, but it is not encoded with channel coding yet.

The minimum distance of $\mathcal{C}$ is denoted by [34]

$$\Delta = \min_{U_1, U_2 \in \mathcal{C}: U_1 \neq U_2} d(U_1, U_2) \tag{4.3}$$

To investigate the successful decoding probability, we will need the definition of a minimum distance decoder [34].

**Definition 5.** *A minimum distance decoder for a code $\mathcal{C}$ is one that takes the received subspace $U'$ and returns a nearest codeword $U \in \mathcal{C}$, i.e., a codeword $U \in \mathcal{C}$ satisfying, for all $\tilde{U} \in \mathcal{C}$, $d(U, U') \leq d(\tilde{U}, U')$ $\tilde{U} \neq U$ in $\mathcal{C}$ for which $d(\tilde{U}, U') \leq d(U, U')$.*

In [34], the following result is shown:

**Theorem 1.** *The transmitted subspace $U \in \mathcal{C}$ can be successfully produced by a minimum distance decoder from the received subspace $U'$ if*

$$2(t + \rho) < \Delta. \tag{4.4}$$

Theorem 1 provides an upper bound for the sum of the number of additions and deletions. However, it does not discuss the tightness of the bound. Therefore, to calculate the decoding probability, we still need to analyze the converse to the result in Theorem 1. In [36], the converse of Theorem 1 is shown and proven.

**Theorem 2.** *Let $\mathcal{C}$ have dimension $K$, minimum distance $\Delta$, and code rate $r > (K - \Delta/2 + 1)/K$. If $2(t + \rho) \geq \Delta$, then decoding is unsuccessful for some value of the transmitted subspace and the error packets.*

Let $\mathcal{P}$ denote the probability of successful decoding. According to Theorem 1 and 2, we can now compute that $\mathcal{P} = \Pr\left[(t + \rho) \leq \Delta/2 - 1\right]$. In the next subsection, we will utilize this result to calculate the decoding probability of a single source single path graph.

### 4.2.1 Single path graph

In Section 2.2, we have modeled the line network by a single path graph with multiple sources. In this subsection, to ease the analysis, we first analyze the single source single path graph. Let $\mathbf{F}_q^{m\times n}$ denote the set of all $m \times n$ matrices over finite field $\mathbf{F}_q$. Let $\mathcal{C}$ be a subspace code with codeword dimension $K$, minimum distance $\Delta$, and code rate greater than $(K - \Delta/2)/K$. Let matrix $W \in \mathbf{F}_q^{K\times b}$ represent the transmitted codeword.

We now present some results in [34] which applies the theorems in the previous section to study the error and erasure performance of coding and routing strategies on the single source networks with randomly located errors and erasures.

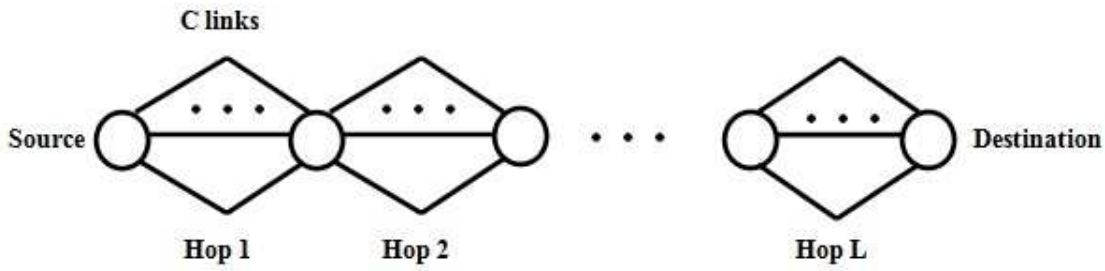


Figure 4.3: The network model of single path subgraph

Consider a simple single path subgraph with $L$ multiple hops shown in Figure 4.3. Let $C$ be the number of parallel links on each hop of $\mathcal{G}_L$. Let $M = C \cdot L$ denote the total number of links in $\mathcal{G}$. Let $G \in \mathbf{F}_q^{C\times K}$ denote the source coding matrix. An error on a link is modeled as addition of an arbitrary error packet to the packet being transmitted at that link. Let $Z \in \mathbf{F}_q^{M\times b}$ denote the error matrix whose $i$th row corresponds to the error packet that is injected on the $i$the link of $\mathcal{G}$, $A \in \mathbf{F}_q^{C\times C}$ denote the transfer matrix from all links in the network to the packets received at $\mathcal{T}$ and $B \in \mathbf{F}_q^{C\times M}$ denote the transfer matrix from error packets to the packets received

at $\mathcal{T}$. The received codeword can be obtained

$$Y = AGW + BZ \tag{4.5}$$

*Remarks:* At this point, it is necessary to make a connection of the network coding model described above with the traditional error correction coding model applied in the previous chapters. In the point to point error control coding, $K$ packets are encoded into $N$ packets, which will be transmitted in $N$ time units. If we think of $N$ time units as $N$ separate links, it is exactly the case in the network coding where the number of links is represented by $C$. Another way to implement the $C$ parallel links is to apply frequency division, either by simple usage of different frequency bands or by Orthogonal Frequency Division Multiplexing Access (OFDMA). Again, in this chapter we always follow the algebraic interpretation of error control coding and network coding. We also would like to stress again that the errors and erasures in this chapter are actually in the sense of additions and deletions of vector space.

There are three possible events for the transmission of the $j$th hop: an erasure with probability $p$; an error with probability $s$; no errors or erasures with probability $(1-p-s)$.

The random linear coding (RLC) is adopted here. Each node creates random linear combinations of all received packets and sends them to the outgoing links. In case of a link erasure, the node replaces the erased packet by creating a random linear combination of the successfully received packets.

The following definitions are necessary [34]. Let $I$ denote the $C \times C$ identity matrix. Define $A_j \in \mathbf{F}_q^{C\times C}$ as a random matrix with entries from $\mathbf{F}_q$. If an erasure occurs on link $i$ of $j$-th hop, define $E_j \in \mathbf{F}_q^{C\times C}$ as $I$ with the $i$th row equal to the zero vector. If no error occurs, define $D_j \in \mathbf{F}_q^{C\times C}$ as $D_j = I$. If an error occurs on the $i$th link, define $D_j \in \mathbf{F}_q^{C\times C}$ as $I$ with $i$th row equal to the zero vector. Define $D_j^* \in \mathbf{F}_q^{C\times C}$

as $D_j^* = I - D_j$.

Define

$$F_j = \begin{cases} D_j & \text{if an error occurs at the } j\text{th hop,} \\ E_j & \text{if an erasure occurs at the } j\text{th hop,} \\ I & \text{if no errors or erasures occur at the } j\text{th hop.} \end{cases} \tag{4.6}$$

Thus, for RLC we can compute $A$ and $B$ as

$$\begin{aligned} A &= F_L A_L F_{L-1} A_{L-1} \cdots F_2 A_2 F_1 A_1 \\ B &= \begin{bmatrix} F_L A_L \cdots F_2 A_2 D_1^* & F_L A_L \cdots F_3 A_3 D_2^* & \cdots & F_L A_L D_{L-1}^* & D_L^* \end{bmatrix} \end{aligned} \tag{4.7}$$

Let $\mathcal{P}$ denote the probability of successful decoding. Let $\mathsf{A}$ and $\mathsf{D}$ be the random variables denoting the number of dimension additions/deletions to/from $rowspace(S)$[2] in $\mathcal{G}_M$ respectively. Thus we have $\mathcal{P} = \Pr(\mathsf{A} + \mathsf{D} \leq \Delta/2 - 1)$.

Let $Y^j$ denote the subspace spanned by received packets at the $j$th node of $\mathcal{G}_L$. Let $a_j$ and $d_j$ be the number of dimension additions/deletions to/from $rowspace(S)$ present in $Y^j$. The $j$th node of $\mathcal{G}_L$ is in state $i$ if $a_j + d_j = i$ after RLC is performed at the $j$th node. Let $P_{i,k}^j$ denote the probability that the $j$th node of $\mathcal{G}_L$ will be in state $k$ after the transmission from $(j-1)$-th to the $j$th hop, given that the $(j-1)$-th node of $\mathcal{G}_L$ is in state $i$.

The following lemma on the decoding probability of single path subgraph using RLC is given in [35].

**Lemma 1.** *When RLC is performed at every node of $\mathcal{G}_L$, for every node $j = 1, \ldots, L$, we have:*

[2]In linear algebra, the rowspace of a matrix is the set of all possible linear combinations of its row vectors.

*if* $0 \leq i < C - K$

$$P^j_{i,i} = 1 - s, P^j_{i,i+1} = s, P^j_{i,k} = 0 \text{ for } k \neq i, i+1 \tag{4.8}$$

*if* $i = C - K + 2m$, $m = 0, \ldots, K - 1$

$$P^j_{i,i} = 1 - p - s, P^j_{i,i+1} = p, P^j_{i,i+2} = s, P^j_{i,k} = 0 \text{ for } k \neq i, i+1, i+2 \tag{4.9}$$

*if* $i = C - K + 2m + 1, m = 0, \ldots, K - 1$

$$P^j_{i,i} = 1 - s, P^j_{i,i+1} = s, P^j_{i,k} = 0 \text{ for } k \neq i, i+1 \tag{4.10}$$

*if* $i = C + K$

$$P^j_{i,i-1} = p, P^j_{i,i} = 1 - p, P^j_{i,k} = 0 \text{ for } k \neq i-1, i \tag{4.11}$$

Therefore, the system is modeled as a Markov chain which has a probability transition matrix with entries $P^j_{i,k}$ for $i, k = 0 \ldots C + R$. After $L$ transitions of the Markov chain, $\mathcal{P}$ is obtained.

## 4.3 Multiple source multicast network error correction

In last section, we reviewed the results on single source multicast network error correction with an example of single path subgraph using RLC. Now, we are going to study multiple source multicast network error correction. We will use the wireless line network as the system model and investigate the decoding probability with RLC.

Capacity regions for multisource network error correction will be discussed in the second subsection. Please note that the wireless line network can be abstracted as a multiple source unicast network. However, we can apply the results in this section to the wireless line network by assuming the sink node number to be 1.

### 4.3.1 Decoding probability of multiple source multicast network error correction

In this section, by using the multiple source multicast network model to fully describe the line network model, we are able to analyze the decoding probability of such network as well as the capacity region. The wireless line network can be viewed as a single path subgraph with every node along the path being a source node except for the sink node.

Consider a multicast network error correction problem on a graph $\mathcal{G}$ with $L$ source nodes $\mathcal{S} = \{s_1, s_2, \ldots, s_L\}$ and a sink node $\mathcal{T}$, which can be seen as $R_1$, $R_2$, $\ldots$, $R_L$ and the substation in previous chapters. Each link has unit capacity. For any non-empty subset $\mathcal{S}' \subseteq \mathcal{S}$, let $I(\mathcal{S}')$ be the indices of the source nodes that belong to $\mathcal{S}'$. Let $m_{\mathcal{S}'}$ be the minimum cut capacity between any sink and $\mathcal{S}'$. For each $i, i = 1, \ldots, L$, let $\mathcal{C}_i$ be the code used by source $i$. Let $\mathcal{C}_{\mathcal{S}'}$ be the Cartesian product of the individual codes of the sources in $\mathcal{S}'$.

The packets received at $\mathcal{T}$ can be written as

$$Y = (A_1 G_1 W_1 + A_2 G_2 W_2 + \cdots + A_L G_L W_L) + (B_1 Z_1 + B_2 Z_2 + \cdots + B_L Z_L) \quad (4.12)$$

By applying the same technique adopted in Sec. 4.2.1, we can model the system as a Markov chain to calculate the error probability.

### 4.3.2 Capacity regions of multiple source multicast network error correction

In [7] it is shown that the minimum subspace distance decoder can successfully recover the transmitted subspace from the received subspace if

$$2(\rho + t) < D_s^{\min}, \tag{4.13}$$

where $D_s^{\min}$ is the minimum subspace distance of the code. Note that $D_s$ treats insertions and deletions of subspaces symmetrically.

The following theorem characterizes the capacity of the network error correction in a multiple-source multicast scenario.

**Theorem 3.** *Consider a multiple-source multicast network error correction problem on network $\mathcal{G}$ with known topology. For any arbitrary errors on up to $z$ links, the capacity region is given by:*

$$\sum_{i \in I(\mathcal{S}')} r_i \leq m_{\mathcal{S}'} - 2z, \forall \mathcal{S}' \subseteq \mathcal{S}. \tag{4.14}$$

*Proof:* Let $l_{i,j}, j = 1, \ldots, n_i$, be the outgoing links of each source $s_i, i = 1, \ldots, n$. Let $\mathcal{S}' \subseteq \mathcal{S}$. We add a virtual source node $w_{\mathcal{S}'}$ to $\mathcal{G}$ and obtain $\mathcal{G}_{\mathcal{S}'}$. In the new graph $w_{\mathcal{S}'}$ are linked to each source $s_i$ in $\mathcal{S}'$ by $n_i$ links $l'_{i,j}, j = 1, \ldots, n_i$. Note that the minimum cut capacity between $w_{\mathcal{S}'}$ and any sink is at least $m_{\mathcal{S}'}$. Any network code that multicasts at rate $r_i$ from each source $s_i, i \in I(\mathcal{S}')$ over $\mathcal{G}$ corresponds to a network code that multicasts at rate $\sum_{i \in I(\mathcal{S}')} r_i$ from $w_{\mathcal{S}'}$ to the sink node; the symbol on each link $l'_{i,j}$ is the same as that on link $l_{i,j}$, and the coding operations at all other nodes are identical for $\mathcal{G}$ and $\mathcal{G}_{\mathcal{S}'}$. Then, by applying the network Singleton bound in [27], Theorem 3 follows.

### 4.3.3 Simulation results

Figure 4.4 shows the decoding probability of a line network with 2 sources and 1 sink with random linear coding. For $L = 2$, each hop has $C = 5$ parallel links. For simplicity, we assume all the links in the system have identical error probability $s$ and erasure probability $p$. The successful decoding probability are draw versus the erasure probability $p$. Two systems with $s = 0.05$ and $s = 0.1$ are compared. Figure 4.5 shows the capacity of network with the mincut for each source $m_1 = m_2 = 5$ under arbitrary errors on up to $z = 1$ links.

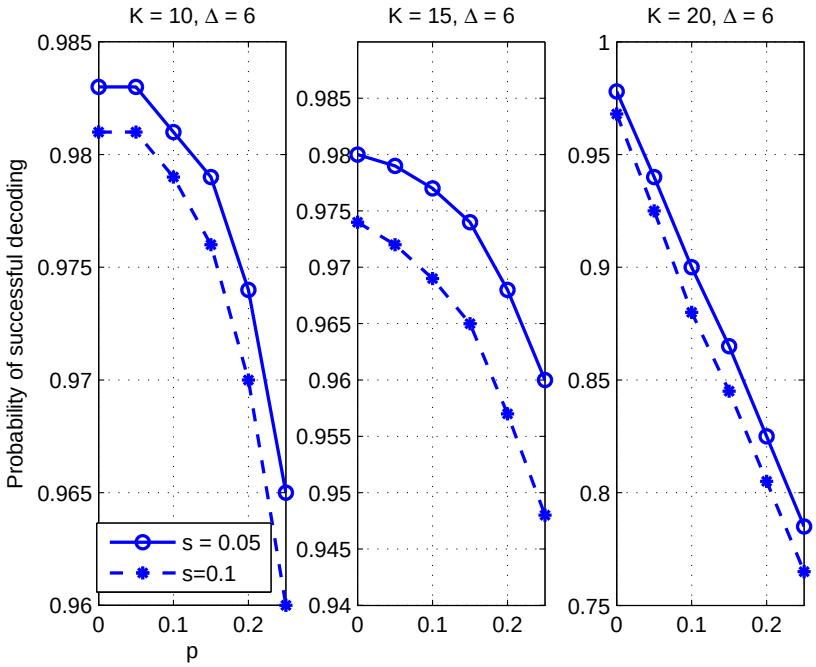


Figure 4.4: Simulation results for decoding probability of 2 sources 1 sink line network with random linear coding. $L = 2, C = 5$.

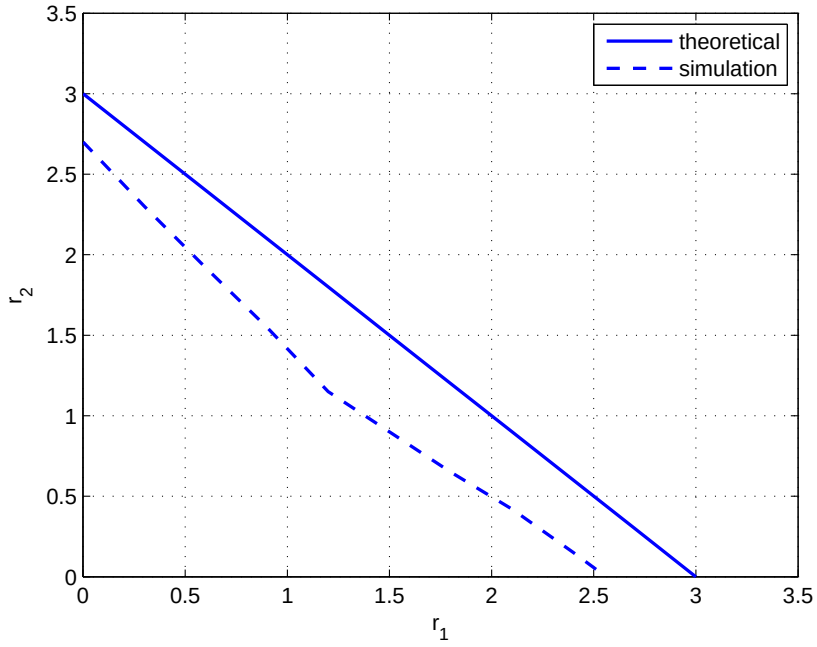


Figure 4.5: Capability probability of 2 sources 1 sink line network with random linear coding. $m_1 = m_2 = 5, z = 1$.

# Chapter 5

# Conclusions and Future work

## 5.1 Conclusions

In this thesis we have investigated a specific type of wireless sensor networks which could be used in Smart Grid for data collection. The system can be generalized as a multi-source single-path line network model. We have studied mainly two approaches to deal with the channel erasures and errors: erasure codes and network coding with error correction.

In Chapter 2, we first reviewed the basics of erasure codes and presented the simulation results of these codes applied to the line network model. These coding schemes differ from each other with respect to encoding and decoding complexity, decoding overhead, delay and other performances. We then proposed two general packet processing strategies: complete encoding and decoding and decode-at-destination. Different strategies suit different systems in terms of node capacity and delay and memory space constraints.

Chapter 2 only considers codes which are able to correct erasures. To deal with the fading channels, we proposed to use CRC and RS code to detect and correct the

errors. In particular, for every packet encoded with an erasure code, we append CRC bits to the end to detect the errors. In addition, we use an RS code as an outer code to correct the errors. Adaptive transmission is applied, which outperforms the systems with fixed-rate RS codes.

As addressed in Chapter 4, we investigated using network coding with error correction functions in our line networks model. Network coding with error correction can be viewed as a generalization of traditional point to point error correction codes. We first reviewed some existing results on single source single path network. Then we extended the results to multiple source single path network which is the abstraction of the line network model suitable for smart grid sensor network.

## 5.2 Future Work

There are several limitations of this thesis which can be topics of future work.

1. In Chapter 2, we apply the erasure codes and investigate its performance in erasure channels. For future work, the performance of different erasure codes in fading channels without the use of CRC or error correcting codes can be evaluated.

2. In Chapter 3, we assume Rayleigh fading channel model. However, Nakagami-$m$ fading can be used to describe a more general channel model, especially for our line network model where there usually exists a line-of-sight path between two adjacent poles.

3. In Chapter 4, we investigate the network coding in our line network model. Future work can be focused on applying the network coding in more sophisticated network models.

# Bibliography

[1] J. Chen, S. Kher, and A. Somani, "Energy efficient model for data gathering in structured multiclustered wireless sensor network," in *Performance, Computing, and Communications Conference*, 2006.

[2] R. A. Leon, V. Vittal, and G. Manimaran, "Application of sensor network of secure electric engergy infrastructure," *IEEE Transactions on Power Delivery*, vol. 22, Apr. 2007.

[3] D. J. C. MacKay, "Fountain codes," in *IEE Proceedings of Communications*, vol. 152, pp. 1062–1068, Dec. 2005.

[4] K. Hung, W. Lee, V. Li, K. Lui, P. Pong, K. Wong, G. Yang, and J. Zhong, "On Wireless Sensors Communication for Overhead Transmission Line Monitoring in Power Delivery Systems," in *First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Oct. 2010.

[5] P. Pakzad and A. S. Fragouli, C., "Coding schemes for line networks," in *IEEE ISIT*, pp. 1852–1857, Sept. 2005.

[6] P. Elias, "Coding for two noisy channels," in *Third London Symposium in Information Theory*, Sept. 1955.

[7] S. Lin and D. J. Costello, *Error control coding: fundamentals and applications*. New Jersey: Prentice Hall, 1st ed., 1983.

[8] M. Luby, "LT codes," in *Proc. 43rd Ann. IEEE Symp. on Foundations of Computer Science*, Nov. 2002.

[9] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, pp. 2551–2567, Feb. 2006.

[10] C. Koller, M. Haenggi, and D. J. J. Costello, "On the optimal block length for joint channel and network coding," in *IEEE Information Theory Workshop (ITW)*, Oct. 2011.

[11] J. G. Proakis, *Digital Communications*. New York: McGraw-Hill, Inc, 4th ed., 2001.

[12] P. Koopman, "32-bit cyclic redundancy codes for Internet applications," in *International Conf. on Dependable Systems and Networks*, pp. 459–468, 2002.

[13] M. Luby, A. Shokrollahi, M. Watson, and T. Stockhammer, "Raptor forward error correction scheme for object delivery," in *IETF*, Oct. 2007.

[14] G. T. . V6.1.0, "Technical Specification Group Services and System Aspects; Multimedia Broadcast/Multicast Service; Protocols and Codecs," *3GPP Specification*, June 2005.

[15] T. Stockhammer, A. Shokrollahi, M. Watson, M. Luby, T. Gasiba, B. Furht, and S. Ahson, "Application layer forward error correction for mobile multimedia broadcasting," in *Handbook of Mobile Broadcasting: DVB-H, DMB, ISDB-T and Media FLO*.

[16] E. R. Berlekamp, "The technology of error correcting codes," in *IEEE ISIT*, pp. 653–663, Aug. 2002.

[17] D. McEliece and L. Swanson, "On the decoder error probability of Reed-Solomon codes," *IEEE Trans. Inf. Theory*, vol. 32, pp. 701–703, 1986.

[18] K. E. Baddour and N. C. Beaulieu, "Autoregressive models for fading channel simulation," in *Proc. of IEEE Globecom*, Aug. 2002.

[19] P. Chou, Y. Wu, and K. Jain, "Practical network coding," in *2003 Allerton Conf. on Commun., Control and Computing*, Oct. 2003.

[20] T. Ho, M. Medard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, pp. 4413–4430, Oct. 2006.

[21] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, pp. 1204–1216, July 2000.

[22] S.-Y. R. Li, W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, pp. 371–381, 2003.

[23] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, pp. 782–795, Oct. 2003.

[24] T. Ho, R. Koetter, R. Medard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *IEEE ISIT*, p. 442, June 2003.

[25] S. Katti, D. Katabi, H. Balakrishnan, and M. Medard, "Symbol-level network coding for wireless mesh networks," in *ACM SIGCOMM*, Aug. 2008.

[26] N. Cai and R. W. Yeung, "Network coding and error correction," in *IEEE ISIT*, pp. 119–122, Oct. 2002.

[27] R. W. Yeung and N. Cai, "Network error correction, part I: Basic concepts and upper bounds," *Commun. Inform. Syst.*, vol. 6, pp. 19–36, Oct. 2006.

[28] N. Cai and R. W. Yeung, "Network error correction, part II: Lower bounds," *Commun. Inform. Syst.*, vol. 6, pp. 37–54, Oct. 2006.

[29] Z. Zhang, "Network error correction coding in packetized networks," in *IEEE ISIT*, pp. 433–437, Oct. 2006.

[30] Z. Zhang, "Linear network error correction codes in packet networks," *IEEE Trans. Inf. Theory*, vol. 54, no. 1, pp. 209–218, 2008.

[31] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard, "Resilient network coding in the presence of Byzantine adversaries," in *IEEE Int. Conf. on Computer Commun.*, pp. 616–624, May 2007.

[32] Z. Zhang, "Resilient network coding in the presence of Byzantine adversaries," *IEEE Trans. Inf. Theory*, vol. 54, no. 6, pp. 2586–2603, 2008.

[33] R. Koetter and F. R. Kschischang, "Coding for errors and erasures in random network coding," in *IEEE ISIT*, pp. 24–29, 2007.

[34] R. Koetter and F. R. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3579–3591, 2008.

[35] R. Koetter and F. R. Kschischang, "Network coding for error correction," *Ph.D. Dissertation.*

[36] D. Silva and F. R. Kschischang, "On metrics for error correction in network coding," *IEEE Trans. Inf. Theory*, vol. 55, pp. 5479–5490, 2009.