ERROR CONTROLS FOR BROADCAST COMMUNICATION SYSTEMS: AN INTEGER PROGRAMMING APPROACH TO UEP CODING SCHEME AND A DETERMINISTIC APPROACH TO NETWORK CODING

by

Wook Jung
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Information Technology

Committee:

____________________	Dr. Shih-Chun Chang, Dissertation Director
____________________	Dr. Bijan Jabbari, Committee Member
____________________	Dr. Brian L. Mark , Committee Member
____________________	Dr. Bernd-Peter Paris, Committee Member
____________________	Dr. Robert Simon, Committee Member
____________________	Dr. Stephen Nash, Senior Associate Dean,
____________________	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering
Date: ____________________	Spring Semester 2015 George Mason University Fairfax, VA

Error Controls for Broadcast Communication Systems: An Integer Programming Approach to UEP Coding Scheme and A Deterministic Approach to Network Coding

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Wook Jung
Master of Science
George Mason University, 2003
Bachelor of Science
Ajou University, Korea, 2000

Director: Dr. Shih-Chun Chang, Associate Professor
Department of Electrical and Computer Engineering

Spring Semester 2015
George Mason University
Fairfax, VA

Dedication

I dedicate this dissertation to my parents, my wife, and my daughter.

Acknowledgments

First of all, I would like to express my heartfelt thanks to my advisor, Dr. Shih-Chun Chang who lead me to the field of coding theory and inspired me by showing his enthusiasms and ingenuity. This dissertation would not have been possible without his support, encouragement, guidance and persistent help.

I would like to thank Drs. Bijan Jabbari, Brian L. Mark, Bernd-Peter Paris and Robert Simon for their invaluable suggestions and serving as on my dissertation committee. I would like to give a special thanks to Dr. Pelin A. Kurtay and Director Lisa Nolder for supporting me whenever I had difficulties during the course of my Ph.D. study.

Last, but certainly not least, I thank my family for their endless and unconditional support and encouragement.

Table of Contents

List of Tables

List of Figures

Figure Page

Abstract

ERROR CONTROLS FOR BROADCAST COMMUNICATION SYSTEMS: AN INTEGER PROGRAMMING APPROACH TO UEP CODING SCHEME AND A DETERMINISTIC APPROACH TO NETWORK CODING

Wook Jung, PhD

George Mason University, 2015

Dissertation Director: Dr. Shih-Chun Chang

Traditional network protocols employ error control techniques for reliable information dissemination over noisy communication channels. In this dissertation, two main topics are investigated for efficient error controls over a broadcast channel. First, unequal error protection (UEP) coding schemes for multiuser communications are investigated, and we propose integer programming approaches to UEP coding and decoding. Second, reliable packet transmissions over a single-hop broadcast network are considered, and we propose a unified solution to use a deterministic network coding for a packet retransmission scheme and a packet-level forward error correction scheme.

For multiuser communications over a broadcast channel, integer programming approaches are introduced to the construction and the decoding of a binary linear UEP code. First, optimal UEP codes are constructed from integer programming for maximum efficiency, and lower bounds of UEP codes are derived to show the efficiency. Then, performance of the UEP coding scheme for multiuser communications are analyzed on a degraded broadcast channel. Finally, a decoding method of the binary UEP code is proposed by using iterative integer programming and majority logic. By presenting numerical results, examples, and

comparisons, we demonstrate that the UEP coding scheme effectively provides efficient forward error correction for multiuser broadcast communications.

For reliable packet transmissions over a single-hop broadcast network, we propose packet-level error control schemes by using a deterministic linear network coding. We first construct a deterministic network code based on Reed-Solomon (RS) code. Then, we provide an adaptive way to apply the deterministic network code for both retransmissions and forward error corrections by puncturing the RS code. Numerical analysis and simulations are performed to show the efficiency of the error control schemes.

Chapter 1: Introduction

Error controls for communication network over a noisy channels are classified into two categories:

1. *Forward error correction* (FEC) scheme that corrects channel errors by using error correcting codes; or,

2. *Automatic repeat request* (ARQ) scheme that recovers erroneous or lost information by retransmissions based on feedback.

In this dissertation, we investigate both FEC and ARQ schemes for reliable communications over a broadcast channel.

For the forward error correction, a communication system usually employs conventional error correcting codes that have equal error protection capability. However, when multiple users communicate over a noisy channel, each user is likely to have unequal errors on the received information. An *unequal error protection* (UEP) coding scheme provides more efficient error controls for the communication system than the conventional coding scheme.

When information has different significance, whether it is a bit or groups of bits, unequal error protection (UEP) codes separate the information and provide different levels of error protection. The notion of a UEP code was first introduced by Masnick and Wolf in [1]. Since then, construction methodologies of UEP codes have been actively studied in many coding theory papers. A majority of the linear UEP code constructions are based on combining shorter length linear codes. In this dissertation, unlike those approaches, we propose a construction methodology of optimal UEP codes for broadcast communications based on integer programming. We also propose a decoding methodology of the UEP codes based on integer programming and majority logic.

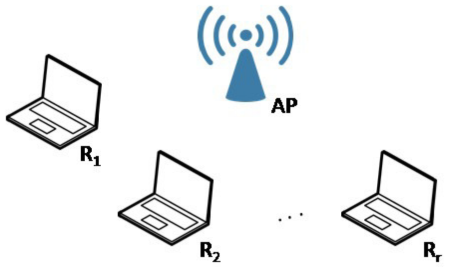


Figure 1.1: Single-hop wireless broadcast network.

Furthermore, consider packet transmissions over a broadcasting network that a single source has data packets to transmit to all receivers within its broadcasting range. One can consider the broadcasting source as an access point (AP) in wireless network and each receiver as a mobile station as depicted in Figure 1.1. Because of uncertainty of the broadcast channel, traditional network protocols employ error controls (either FEC scheme or ARQ scheme) for reliability of packet deliveries.

The packet retransmission scheme requires a feedback channel to receive packet error/loss information. The broadcasting source first collects acknowledgments of the transmitted packets (ACK/NACK) from its receivers through the feedback channel, then the source retransmits packets based on the acknowledgments. On the other hand, the forward error correction scheme does not require the feedback channel. Instead the forward error correction scheme adds redundancy into transmitting packets to ensure that an intended receiver can recover packets without requesting retransmissions when packet errors occur during transmission.

Since Ahlswede *et al.* introduced the concept of a *network coding* in [2], which allows to process packets at a network node, there have been some studies to apply the network coding concept into reliable packet transmissions. Our research investigates the methodologies of network coding for reliable packet transmissions and provides a unified solution based on linear network coding for both a retransmission scheme and a packet-level forward error correction scheme on a single-hop wireless broadcast network.

1.1 Contributions

The contributions of this dissertation can be categorized into two parts.

Unequal error protection coding and decoding by integer programming for broadcast channels

1. An integer programming approach is introduced to construct optimal UEP codes for multiuser communication over a broadcast channel.

   - An integer programming problem is formulated based on unequal error protection requirements of multiple users to construct an optimal binary UEP code.
   - An integer programming bound and an asymptotically achievable code rate are derived to show efficiency of the integer programming approach.
   - Performance analysis of the UEP coding scheme is presented for multiuser communications over a degraded broadcast channel.

2. A decoding algorithm is developed for the UEP codes based on iterative integer programming and majority logic.

Reliable packet transmission on single-hop broadcast networks using deterministic network coding

1. A methodology of deterministic linear network coding is investigated based on a punctured Reed-Solomon (RS) code.

2. A unified solution is provided for efficient reliable packet transmissions on a single-hop broadcast network:

   - A packet retransmission scheme (ARQ scheme) based on deterministic network coding.
   - A packet-level forward error correction scheme (FEC scheme) based on deterministic network coding.

1.2 Outline

The organization of this dissertation is as follows. First, Chapter 2 provides background and a literature review. Next, in Chapter 3, we propose an integer programming approach to construct binary UEP codes, and we provide an integer programming bound and numerical results that show the constructed UEP codes are optimal. Based on the bound, we illustrate asymptotic code rates with comparisons to throughput of the broadcast channel, and we also measure bit error performance of the UEP coding scheme on a degraded broadcast channel. Moreover, we present an iterative decoding method by using integer programming and majority logic. Then, in Chapter 4, we construct a deterministic linear network code from Reed-Solomon codes, and we present a unified solution based on the deterministic network coding for both a retransmission scheme and a packet-level forward error correction scheme on a single-hop wireless broadcast network. We also provide numerical analysis and simulation results to show the efficiency of the schemes. At the end of both Chapter 3 and Chapter 4, we discuss limitations and brief research directions. Finally, Chapter 5 presents a summary and future developments.

Chapter 2: Background and related works

2.1 Unequal error protection codes

The concept of *unequal error protection (UEP) codes* was first introduced by Masnick and Wolf in [1]. Consider a coded communication system depicted in Figure 2.1. Let $\mathcal{C}$ be a binary linear code that protects a message vector $\mathbf{m} = (m_1, m_2, \ldots, m_k)$ sent over a noisy channel,

$$\mathcal{C} = \left\{ \mathbf{c} \mid \mathbf{c} = \mathbf{mG}, \quad \mathbf{m} \in \{0,1\}^k \right\}$$

where $\mathbf{G}$ is a $k \times n$ generator matrix. Suppose that the code $\mathcal{C}$ protects each message m_i against t_i channel errors for $1 \leq i \leq k$. For a conventional t-error correcting code, $t_i = t$ for $1 \leq i \leq k$. However, if $t_i \neq t_j$ for $1 \leq i \neq j \leq k$, the code $\mathcal{C}$ has an unequal error protection capability, and it is called an unequal error protection code.

In [3], Dunning and Robbins introduced the notion of a *separation vector* which specifies the unequal error protection capability of a UEP code.

Separation vector For a binary linear (n, k) code $\mathcal{C}$, a separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ is defined as

$$s_i = \min \left\{ w_H(\mathbf{mG}) \mid m_i \neq 0 \right\} \quad \text{for} \quad 1 \leq i \leq k. \tag{2.1}$$

where $w_H(\cdot)$ denotes the Hamming weight.

It follows from the definition that a message bit m_i can be separated and protected against up to $\lfloor \frac{s_i - 1}{2} \rfloor$ channel errors for a given separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$. Suppose that each message m_i requires t_i unequal error protection for $1 \leq i \leq k$. Then, a linear

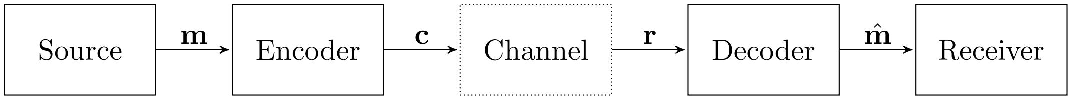


Figure 2.1: Overview of channel coding

code $\mathcal{C}$ can satisfy the requirements if its separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ satisfies

$$s_i \geq 2t_i + 1 \quad \text{for} \quad 1 \leq i \leq k.$$

Earlier works on UEP codes [1, 3–5] have mainly focused on establishing theoretical foundations. Particularly, we are interested in bounds on lengths of UEP codes derived in [1, 5] since the bounds shall be compared to the lengths of UEP codes that we construct in Section 3. Masnick and Wolf have derived an upper bound of a UEP code in [1], and we rewrite the upper bound for a binary UEP code in the following.

Upper bound When a message m_i is protected against t_i errors for $1 \leq i \leq k$, an upper bound of a binary UEP code can be given by

$$n \;\leq\; k + r_U \tag{2.2}$$

where r_U is the smallest number of check bits r such that

$$2^r \;>\; \sum_{i=0}^{2t_k-1} \binom{n-1}{i} - \sum_{i=2}^{k} \binom{n-1-\Xi_i}{2t_i-2} - \sum_{i=2}^{k} \binom{n-1-\Xi_i}{2t_i-1} \tag{2.3}$$

when Ξ_i is the number of message bits that are protected against i-bit or more errors (See [1, pp. 606–607] for the derivation).

On the other hand, van Gils has derived a lower bound on lengths of UEP codes for a non-increasing separation vector (i.e., $s_i \geq s_j$ for $1 \leq i \neq j \leq k$) in [5], and we rewrite the lower bound for a binary UEP code in the following.

Lower bound When a message m_i is protected against t_i errors and a non-decreasing separation vector is given as $s_i = 2t_i + 1$ for $1 \leq i \leq k$ (i.e., $t_i \geq t_j$ for $1 \leq i \neq j \leq k$), a lower bound of a binary UEP code can be given by

$$n \geq \sum_{i=1}^{k} \left\lceil \frac{s_i}{2^{i-1}} \right\rceil \tag{2.4}$$

(See [5, pp. 869–870] for the derivation).

In [1, pp. 603–604], Masnick and Wolf have also presented a binary UEP code construction method. However, it has been claimed by the authors that the optimal code construction becomes difficult for large dimensions (beyond $k = 5$). This observation motivates us to develop an integer programming approach to construct optimal UEP codes.

Since the pioneering work of Masnick and Wolf, constructions of UEP codes have been actively studied, and most construction methods of linear UEP codes are based on combining (e.g., direct sum, $|\mathbf{u}|\mathbf{u}+\mathbf{v}|$, concatenation, etc.) shorter length linear codes as presented in [4, 6–11]. Also there have been studies that investigate information theoretic features of UEP codes [12, 13] and studies that combines UEP coding and modulation for unequal error protection [14–19]. Furthermore, linear network coding has been applied for unequal error protection in [20–25].

2.2 Degraded broadcast channels

The notion of a *broadcast channel* was introduced by Cover in [26]. When k users are simultaneously communicated through a broadcast channel, the channel can be described by a single input (denoted by S) and multiple outputs (denoted by T_i for $1 \leq i \leq k$) as depicted in Figure 2.2.

In [27], Bergmans introduced the notion of *degraded broadcast channels* by decomposing a broadcast channel into multiple degraded component channels. Let CH_i denote component channels and let D_i be artificial channels that represent degradation for $1 \leq i \leq k$

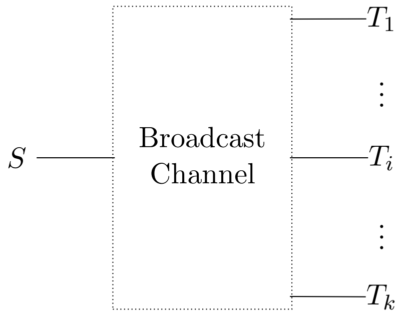


Figure 2.2: Broadcast channel with a channel input S and channel outputs T_i for $1 \leq i \leq k$.

where $CH_1 = D_1$. If a channel CH_j is represented by the cascade of CH_i and D_j, then the channel CH_j is a degraded version of the channel CH_i where $j = i + 1$ for $2 \leq j \leq k$ and the broadcast channel is called degraded (Figure 2.3 [27]).

By using the decomposed channel model (Figure 2.3), the error characteristics of the degraded broadcast channel can be described with k cascaded binary symmetric channels (BSCs) with transition probabilities $\alpha_i \in [0, \frac{1}{2}]$ as shown in Figure 2.4 [27]. Let p_i denote the bit error probability of the each component channel, then

$$\begin{aligned}
p_1 &= \alpha_1, \\
p_2 &= \alpha_1(1-\alpha_2) + (1-\alpha_1)\alpha_2 = p_1(1-\alpha_2) + (1-p_1)\alpha_2, \\
p_3 &= (1-\alpha_1)(1-\alpha_2)\alpha_3 + (1-\alpha_1)\alpha_2(1-\alpha_3) + \alpha_1(1-\alpha_2)(1-\alpha_3) + \alpha_1\alpha_2\alpha_3 \\
&= \{\alpha_1(1-\alpha_2) + (1-\alpha_1)\alpha_2\}(1-\alpha_3) + \{1 - \alpha_1(1-\alpha_2) - (1-\alpha_1)\alpha_2\}\alpha_3 \\
&= p_2(1-\alpha_3) + (1-p_2)\alpha_3, \\
\vdots \quad & \vdots
\end{aligned}$$

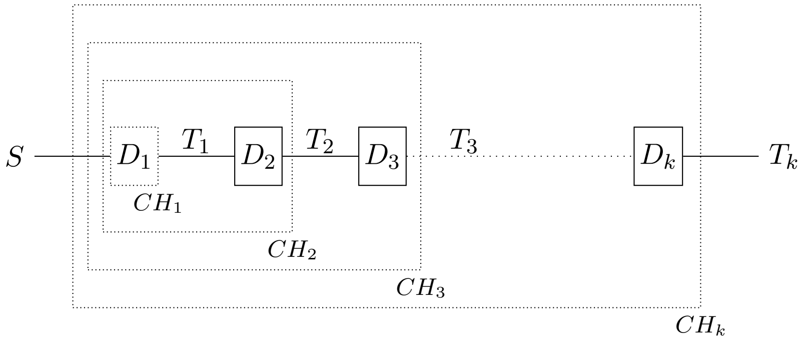


Figure 2.3: Degraded broadcast channel with k component channels.

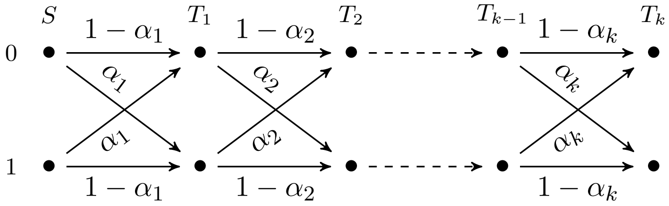


Figure 2.4: Cascaded BSCs for a degraded broadcast channel.

Similarly, the bit error probability of each component channel can be written by

$$p_i = p_{i-1}(1-\alpha_i) + (1-p_{i-1})\alpha_i \quad \text{for} \quad 1 \le i \le k \tag{2.5}$$

where $p_0 = 0$.

Let Γ_i represent an information rate for each component channel CH_i for $1 \le i \le k$, then the throughput rate of the degraded broadcast channel, R_T can be given by

$$R_T = \sum_{i=1}^{k} \Gamma_i = \Gamma_1 + \Gamma_2 + \cdots + \Gamma_k, \tag{2.6}$$

and the throughput rate is upper bounded by the capacity of channel CH_1.

Since the notion of degraded broadcast channel was introduced in [27], information theoretic features (e.g., coding theorem or capacity region) of the degraded broadcast channel have been studied in [28–34].

2.3 Linear (Integer) programming

Linear programming is an optimization method to find a minimum or maximum of a linear function (objective function) subject to certain linear equations or inequalities. If the variables of the objective function are restricted to be integers, then the linear programming is called an *integer programming*.

A typical linear programming problem is formulated as the following [35–37]:

Minimize or maximize

$$f_1x_1 + f_2x_2 + \cdots + f_nx_n$$

subject to

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \begin{pmatrix} \geq \\ = \\ \leq \end{pmatrix} b_1$$

$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \begin{pmatrix} \geq \\ = \\ \leq \end{pmatrix} b_2$$

$$\vdots \qquad \vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \begin{pmatrix} \geq \\ = \\ \leq \end{pmatrix} b_m$$

where

$$x_j \geq 0 \quad \text{for} \quad 1 \leq j \leq n$$

The function $f_1x_1 + f_2x_2 + \cdots + f_nx_n$ is an *objective function* to linearly optimize and the variables $x_1, x_2, \ldots x_n$ are called *decision variables.* If the decision variables are strictly integers, then the optimization refers to an integer programming. If some of the decision variables are restricted to be integers, then the optimization refers to a mixed integer programming. Based on the formulated problem, a linear programming solver provides a feasible solution for the decision variables x_j for $1 \leq j \leq n$ that satisfy all linear constraints, and the best feasible solution is called an optimal solution.

Linear (integer) programming has been applied to coding theory for obtaining bounds on lengths of linear codes ([38, Ch. 17, p. 537]), but mostly it has been used for alternative methods to decode linear codes [39–51].

As an example, we provide the concept of an integer programming approach for maximum likelihood decoding. Consider a communication system where a codeword $\mathbf{c}$ of a binary linear code $\mathcal{C}$ is transmitted over a noisy channel as depicted in Figure 2.1. At the receiving end of the coded system, a decoder tries to recover $\hat{\mathbf{m}}$ from the received $\mathbf{r} = (r_1, r_2, \ldots, r_n)$ such that $\hat{\mathbf{m}} = \mathbf{m}$. A decoding rule of a maximum likelihood decoder is to find a particular codeword $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ that maximizes $\Pr(\mathbf{x}|\mathbf{r})$. Assuming that all codewords of $\mathcal{C}$ are equally likely and the channel is memoryless [52, Ch. 1], it is equivalent to minimizing $\sum_{j=1}^{n} \gamma_j x_j$ where

$$\gamma_j = \log\left(\frac{\Pr(r_j|x_j = 0)}{\Pr(r_j|x_j = 1)}\right). \tag{2.7}$$

Then, the ML decoder can determine that the transmitted codeword $\mathbf{c}$ is the codeword $\mathbf{x}$ that has the minimum sum of $\sum_{j=1}^{n} \gamma_j x_j$. Let $\mathbf{H}$ be a parity check matrix of the linear code $\mathcal{C}$. Then an integer programming problem for the ML decoding can be formulated as

minimize

$$\sum_{j=1}^{n} \gamma_j x_j$$

subject to

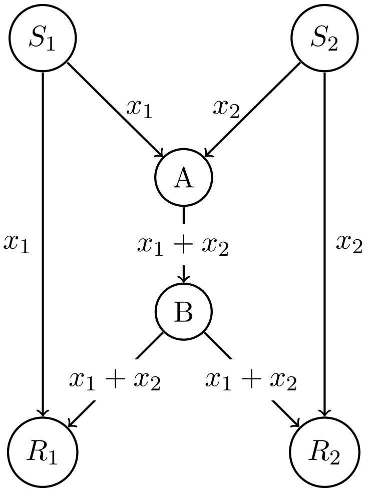


Figure 2.5: The butterfly network: network coding on multicast

$$\mathbf{H}\mathbf{x}^{\top} = \mathbf{0} \quad \text{over} \quad \mathbb{GF}(2)$$

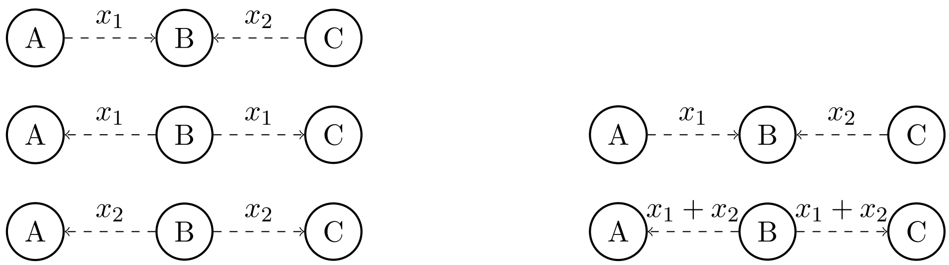
where

$$x_j \in \{0, 1\} \quad \text{for} \quad 1 \leq j \leq n.$$

2.4 Network coding

The concept of network coding was first introduced by Ahlswede *et al.* [2]. In their work, Ahlswede *et al.* have showed that coding at network nodes is necessary in multicast connection for optimal utilization. Figure 2.5 shows an example of network coding in a multicast network.

With network coding, unlike the traditional store and forward packet-switching, a network node not just forwards packets but also combines packets before transmitting. This new concept in communication networks has been proposed as a promising solution in a wide range of areas such as network throughput improvement, network resource utilization, security, and so on. A simple example of benefit of network coding in terms of wireless



(a) Without network coding

(b) With network coding

Figure 2.6: Benefit of network coding

resource utilization is shown in Figure 2.6. Compared to the "without network coding" scenario (Figure 2.6 (a)) which requires three time slots to exchange packets, network coding allows an exchange of packets in two time slots by processing packets in an intermediate node (Figure 2.6 (b)).

In the early stage of network coding, research has focused on theoretical foundations [53–60]. Li *et al* [53] and Koetter and Médard [54] have provided the concept of *linear network coding*, and Ho *et al* [59] have introduced the concept of *random network coding* in a multicast packet network. With the random network coding, a network node transmits linearly encoded packets with randomly chosen coding coefficients from a large enough finite field.

Along with the development of the theoretical frameworks on network coding, researches to practically implement the network coding concept have been made. Chou *et al.* [61] provided a practical and a distributed approach to apply random network coding into traditional packet networks by injecting the encoding information (i.e., coding coefficients) in the packet header so that receivers can decode the packet. Katti *et al.* [62] developed another practical approach to apply network coding using XOR to encode packets at wireless nodes.

2.5 Packet retransmissions using network coding

Network coding has been adopted to provide reliable communications over lossy packet networks. Coding schemes and forward error corrections using network coding have been discussed in [63–69]. Additionally, a couple of studies have shown packet retransmission schemes using network coding on a single-hop wireless broadcast network [70, 71]. In those retransmission schemes, network coding is deployed for retransmitting packets at a broadcasting source to reduce the number of retransmissions.

Traditional retransmission scheme (ARQ) retransmits unsuccessfully delivered packets based on feedbacks (ACK/NACK) from receivers. Nguyen *et al.* [70] uses XOR-based network coding to retransmit packets, i.e., a broadcasting source performs XOR operation on retransmission packets based on the feedbacks from its receivers. Xiao *et al.* [71] uses random network coding instead of XOR based network coding, i.e., a broadcasting source encodes retransmission packets with randomly chosen coding coefficients from a large enough finite field of $\mathbb{GF}(q = 2^m)$ (e.g., $\mathbb{GF}(2^8)$) and injects its encoding information in the packet header.

We consider the following communication scenario over a single-hop wireless broadcast network to compare the number of retransmissions among a traditional ARQ, XOR based retransmission scheme, and random network coding based retransmission scheme.

- A single source broadcasts packets to all receivers within its broadcast range, and every receiver wants to receive all packets from the source.
- Each receiver sends feedback to the source whether it has successfully received packets or not.
- The broadcasting source retransmits packets based on the feedbacks.

An example of the collected feedbacks at the broadcasting source listed in Table 2.1 after transmitting 4 packets, p_1, p_2, p_3, and p_4 to receivers, R_1, R_2, R_3, and R_4.

receiver	p_1	p_2	p_3	p_4
R_1	o	o	x	x
R_2	x	x	o	o
R_3	o	x	o	x
R_4	o	x	x	o

Table 2.1: Example of feedbacks: the symbol 'o' represents a successfully delivered packet and symbol 'x' represents a lost packet at each receiver.

Traditional ARQ scheme Since none of the 4 packets is received successfully at all 4 receivers simultaneously, the broadcasting source has to retransmit all 4 packets to receivers. Let q_1, q_2, q_3, and q_4 be the retransmission packets, then

$$q_i = p_i \quad \text{for} \quad 1 \leq i \leq 4.$$

XOR-based retransmission scheme If XOR operation is allowed for retransmission, the number of retransmissions can be reduced to 3 encoded packets. Let q_1, q_2, and q_3 be the retransmission packets, then

$$q_1 = p_1 \oplus p_3, \quad q_2 = p_2, \quad \text{and} \quad q_3 = p_4.$$

At the receiving end, every receiver can recover its lost packets using XOR operation (See Figure 2.7). For example, since the receiver R_2 has already successfully received p_3 from the previous transmission, the lost packets, p_1 and p_2 can be recovered from

$$p_1 = q_1 \oplus p_3 = (p_1 \oplus p_3) \oplus p_3,$$

$$p_2 = q_2.$$

Retransmission scheme based on random network coding Suppose that a broadcasting source can linearly encode packets with coding coefficients. Since the packet loss pattern in Table 2.1 shows that every receiver wants to receive two out of four

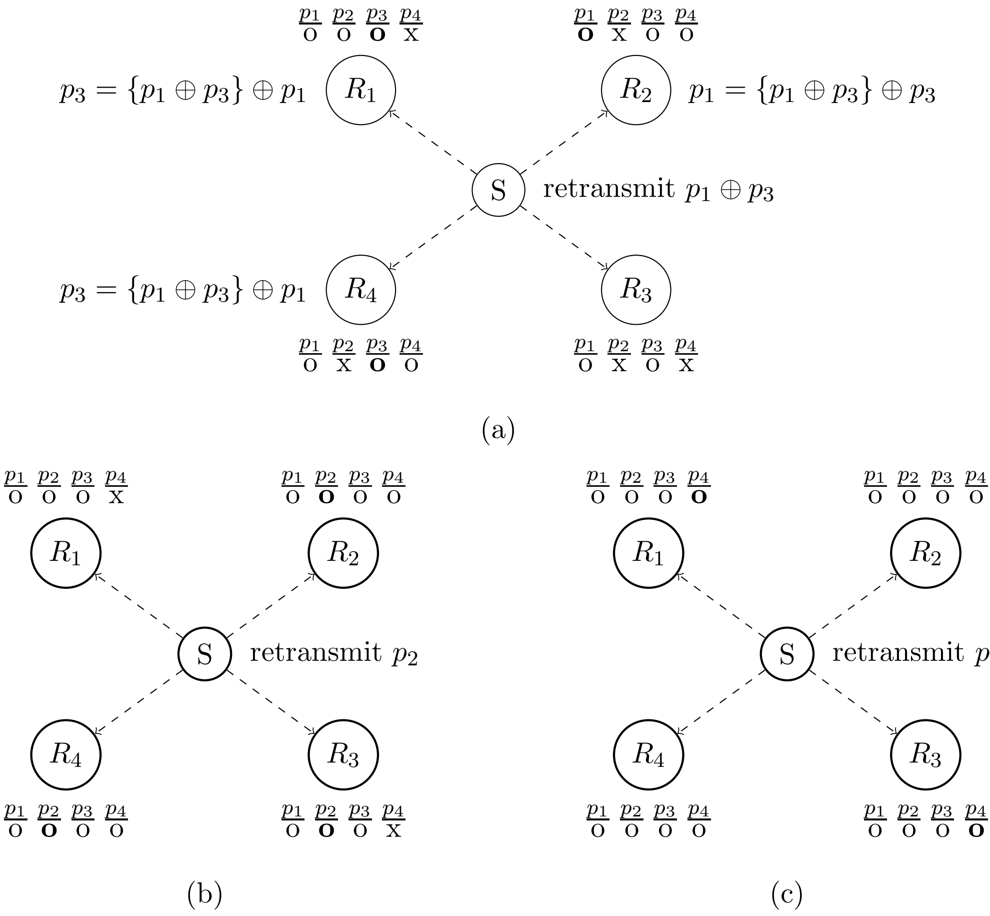


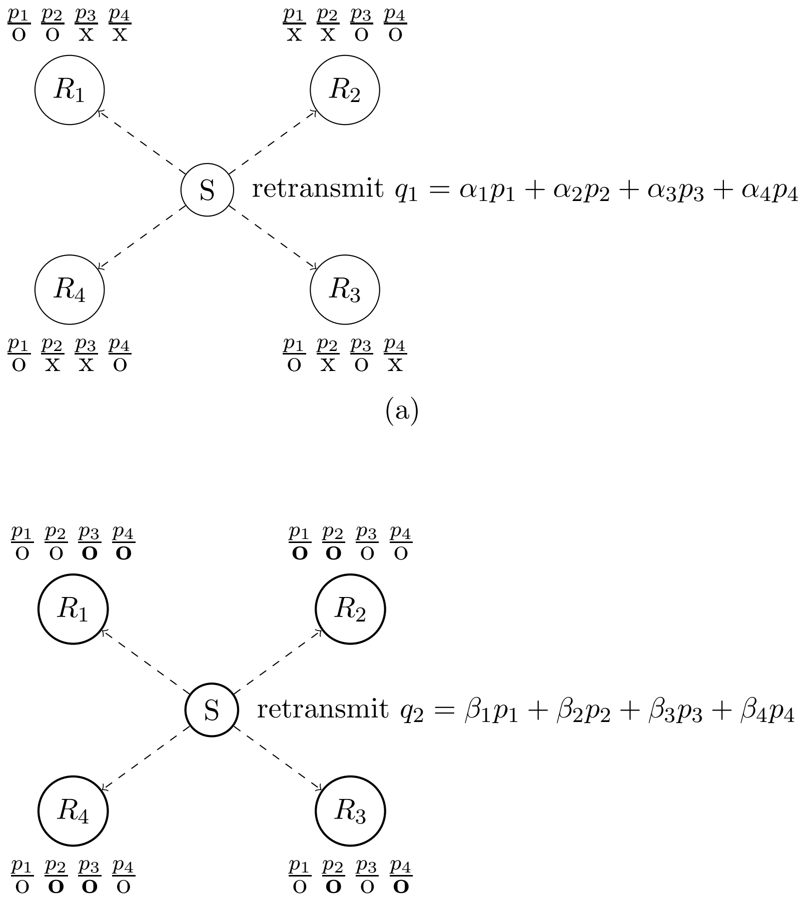
Figure 2.7: Retransmissions with XOR-based network coding

packets, two linearly combined retransmission packets, q_1 and q_2 will be enough for all receivers to recover their lost packets:

$$q_1 = \alpha_1 p_1 + \alpha_2 p_2 + \alpha_3 p_3 + \alpha_4 p_4$$

$$q_2 = \beta_1 p_1 + \beta_2 p_2 + \beta_3 p_3 + \beta_4 p_4$$

where $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ and $\beta_1, \beta_2, \beta_3, \beta_4$ are coding coefficients over a certain finite field. If the source linearly combines packets with proper coding coefficients and if all receivers know what coding coefficients are used to combine the retransmitted packets,



(a)

linear operation at R_1: $\begin{pmatrix} \alpha_3 & \alpha_4 \\ \beta_3 & \beta_4 \end{pmatrix} \begin{pmatrix} p_3 \\ p_4 \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} + \begin{pmatrix} \alpha_1 & \alpha_2 \\ \beta_1 & \beta_2 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}$

linear operation at R_2: $\begin{pmatrix} \alpha_1 & \alpha_2 \\ \beta_1 & \beta_2 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} + \begin{pmatrix} \alpha_3 & \alpha_4 \\ \beta_3 & \beta_4 \end{pmatrix} \begin{pmatrix} p_3 \\ p_4 \end{pmatrix}$

linear operation at R_3: $\begin{pmatrix} \alpha_2 & \alpha_4 \\ \beta_2 & \beta_4 \end{pmatrix} \begin{pmatrix} p_2 \\ p_4 \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} + \begin{pmatrix} \alpha_1 & \alpha_3 \\ \beta_1 & \beta_3 \end{pmatrix} \begin{pmatrix} p_1 \\ p_3 \end{pmatrix}$

linear operation at R_4: $\begin{pmatrix} \alpha_2 & \alpha_3 \\ \beta_2 & \beta_3 \end{pmatrix} \begin{pmatrix} p_2 \\ p_3 \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} + \begin{pmatrix} \alpha_1 & \alpha_4 \\ \beta_1 & \beta_4 \end{pmatrix} \begin{pmatrix} p_1 \\ p_4 \end{pmatrix}$

(b)

Figure 2.8: Retransmissions with random network coding

receivers can recover their lost packets by solving linear equations (See Figure 2.8). For example, since R_3 already successfully received p_1 and p_3 from the previous packet transmissions, the receiver R_3 can recover p_2 and p_4 by solving the following linear equations:

$$\begin{pmatrix} \alpha_2 & \alpha_4 \\ \beta_2 & \beta_4 \end{pmatrix} \begin{pmatrix} p_2 \\ p_4 \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} + \begin{pmatrix} \alpha_1 & \alpha_3 \\ \beta_1 & \beta_3 \end{pmatrix} \begin{pmatrix} p_1 \\ p_3 \end{pmatrix},$$

hence,

$$\begin{pmatrix} p_2 \\ p_4 \end{pmatrix} = \begin{pmatrix} \alpha_2 & \alpha_4 \\ \beta_2 & \beta_4 \end{pmatrix}^{-1} \left\{ \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} + \begin{pmatrix} \alpha_1 & \alpha_3 \\ \beta_1 & \beta_3 \end{pmatrix} \begin{pmatrix} p_1 \\ p_3 \end{pmatrix} \right\}.$$

Chapter 3: UEP coding schemes for broadcast channels

A multiuser communication system over a broadcast channel is depicted in Figure 3.1. Due to the inequality of errors that the broadcast channel introduces, each user requires a different level of error protection denoted by

$$\mathbf{t} = (t_1, t_2, \ldots, t_k) \quad \text{for} \quad 1 \leq i \leq k. \tag{3.1}$$

For a message vector $\mathbf{m} = (\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_k)$ whose element $\mathbf{m}_i$ represents a l-bit message of each user,

$$\mathbf{m}_i = \left(m_1^{(i)}, m_2^{(i)}, \ldots, m_l^{(i)}\right) \quad \text{for} \quad 1 \leq i \leq k,$$

a binary UEP code $\mathcal{C}$ with a generator matrix $\mathbf{G}$ can be represented as

$$\mathcal{C} = \Big\{\mathbf{mG} \mid \mathbf{m} = (\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_k)\,,\ \mathbf{m}_i \in \{0,1\}^l \quad \text{for} \quad 1 \leq i \leq k\Big\}\,.$$

As defined in [3], the unequal error protection capability of the code can be specified by its separation vector denoted by $\hat{\mathbf{s}} = (\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_k)$ where

$$\hat{s}_i = \min\left\{w_H\left(\mathbf{mG}\right) \mid \mathbf{m}_i \neq \mathbf{0}\right\} \quad \text{for} \quad 1 \leq i \leq k. \tag{3.2}$$

where $w_H(\cdot)$ denotes the Hamming weight. If the separation vector satisfies

$$\hat{s}_i \geq 2t_i + 1 \quad \text{for} \quad 1 \leq i \leq k, \tag{3.3}$$

then the UEP code can correct up to t_i errors and recover the l-bit message $\hat{\mathbf{m}}_i = \mathbf{m}_i$ for each user.

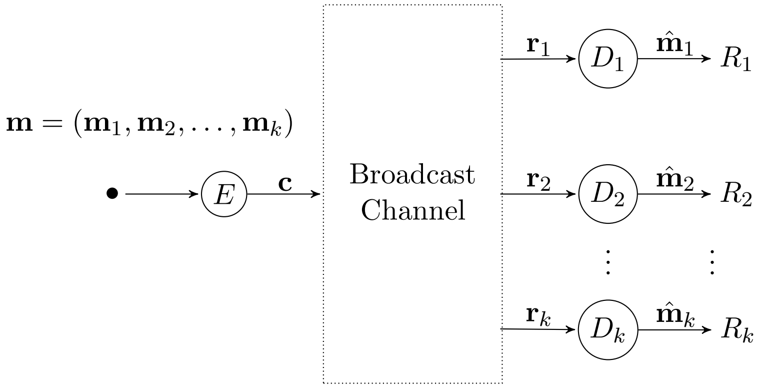


Figure 3.1: Multiuser communication system over a broadcast channel

Suppose that the unequal error protection requirement from each user is given as a separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ such that

$$s_i = 2t_i + 1 \quad \text{for} \quad 1 \leq i \leq k. \tag{3.4}$$

Then, by using integer programming, our objective is to construct a binary UEP code whose separation vector $\hat{\mathbf{s}}$ satisfies

$$\hat{s}_i \geq s_i \quad \text{for} \quad 1 \leq i \leq k, \tag{3.5}$$

i.e.,

$$\min \left\{ w_H\left(\mathbf{mG}\right) \mid \mathbf{m}_i \neq \mathbf{0} \right\} \geq s_i \quad \text{for} \quad 1 \leq i \leq k. \tag{3.6}$$

In addition, we develop a decoding algorithm by using integer programming such that each receiver R_i recovers the message $\mathbf{m}_i$ when the broadcast channel introduces less than or equal to t_i errors.

3.1 Integer programming approach to UEP coding scheme for single-bit messages

In this section, we limit our scope to the case of single-bit message for each user, i.e., $\mathbf{m}_i = m_i$ where $m_i \in \{0, 1\}$ for $1 \leq i \leq k$.

3.1.1 UEP code constructions using integer programming

First of all, a separation vector which specifies the unequal error protection requirement of each user is defined as the following;

Definition 3.1 (Non-decreasing separation vector)**.** *A separation vector* $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ *is defined for unequal error protection as in non-decreasing order, i.e.,*

$$s_i \leq s_j \quad for \quad 1 \leq i < j \leq k. \tag{3.7}$$

Let $\mathbf{A}_b$ be a $k \times (2^k - 1)$ matrix consisting of all nonzero binary k-tuples as columns in increasing order,

$$\mathbf{A}_b = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,2^k-1} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,2^k-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k,1} & a_{k,2} & \cdots & a_{k,2^k-1} \end{pmatrix} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_k \end{pmatrix} \tag{3.8}$$

$$= \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 1 & \cdots & 1 & 1 \\ 0 & 0 & 0 & \cdots & 1 & 0 & \cdots & 1 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & 1 & \cdots & 1 & 0 & \cdots & 1 & 1 \\ 1 & 0 & 1 & \cdots & 1 & 0 & \cdots & 0 & 1 \end{pmatrix} \tag{3.9}$$

where $\mathbf{a}_i = \left(a_{i,1}, a_{i,2}, \dots, a_{i,2^k-1}\right)$ is a row vector, and let $\mathbf{G}$ be a generator matrix of UEP code written as

$$\mathbf{G} = \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \vdots \\ \mathbf{g}_k \end{pmatrix}, \tag{3.10}$$

then, for $1 \leq i \leq k$, each row vector $\mathbf{g}_i$ can be represented as

$$\mathbf{g}_i = \Big(\underbrace{a_{i,1}, \dots, a_{i,1}}_{x_1 \text{ times}}, \underbrace{a_{i,2}, \dots, a_{i,2}}_{x_2 \text{ times}}, \cdots, \underbrace{a_{i,2^k-1}, \dots, a_{i,2^k-1}}_{x_{2^k-1} \text{ times}}\Big) \tag{3.11}$$

where $x_j \geq 0$ for $1 \leq j \leq 2^k - 1$, and

$$n = \sum_{j=1}^{2^k-1} x_j \tag{3.12}$$

represents the code length.

For a given separation vector $\mathbf{s} = (s_1, s_2, \dots, s_k)$, it follows from (3.2) and (3.7) that

$$w_H(\mathbf{g}_1) \geq s_1, \tag{3.13a}$$

$$w_H\Big(\mathbf{g}_i + \sum_{j=1}^{i-1} \omega_j \mathbf{g}_j\Big) \geq s_i \quad \text{for} \quad 2 \leq i \leq k \tag{3.13b}$$

where $\omega_j \in \{0,1\}$. Then, from (3.11),

$$w_H\left(\mathbf{g}_1\right) = \sum_{j=1}^{2^k-1} a_{1,j}x_j = \mathbf{a}_1\mathbf{x}^\top, \tag{3.14a}$$

$$w_H\Big(\mathbf{g}_i + \sum_{j=1}^{i-1}\omega_j\mathbf{g}_j\Big) = \Big(\mathbf{a}_i + \sum_{j=1}^{i-1}\omega_j\mathbf{a}_j\Big)\mathbf{x}^\top \quad \text{for} \quad 2 \le i \le k. \tag{3.14b}$$

where $\mathbf{x} = \big(x_1, x_2, \dots, x_{2^k-1}\big)$.

Therefore, from (3.13) and (3.14), we can have the following inequalities

$$\mathbf{A}_i\mathbf{x}^\top \ge \mathbf{b}_i^\top \quad \text{for} \quad 1 \le i \le k \tag{3.15}$$

where $\mathbf{A}_i$ is a $2^{i-1} \times (2^k - 1)$ matrix that contains all 2^{i-1} rows of

$$\mathbf{a}_1 \qquad\qquad \text{for} \quad i = 1 \tag{3.16a}$$

$$\mathbf{a}_i + \sum_{j=1}^{i-1}\omega_j\mathbf{a}_j \qquad\qquad \text{for} \quad 2 \le i \le k, \tag{3.16b}$$

and $\mathbf{b}_i = (s_i, s_i, \dots, s_i)$ is a row vector of length 2^{i-1}.

Based on (3.15), we formulate an integer programming problem for UEP code construction with the separation vector $\mathbf{s} = (s_1, s_2, \dots, s_k)$ whose elements are given as $s_i = 2t_i + 1$ for $1 \le i \le k$.

UEP code construction

Minimize

$$n = x_1 + x_2 + \cdots + x_{2^k-1}$$

subject to

$$\mathbf{A}\mathbf{x}^\top \geq \mathbf{b}^\top$$

where

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_k \end{pmatrix}, \quad \mathbf{b}^\top = \begin{pmatrix} \mathbf{b}_1^\top \\ \mathbf{b}_2^\top \\ \vdots \\ \mathbf{b}_k^\top \end{pmatrix}$$

A feasible solution or the best feasible solution (i.e., optimal solution) to the integer programming problem shall provide the vector $\mathbf{x} = \big(x_1, x_2, \ldots, x_{2^k-1}\big)$ whose sum is the objective value n. Let an index set $\mathcal{J}_s$ be

$$\mathcal{J}_s = \Big\{ j \mid x_j \neq 0, \quad 1 \leq j \leq 2^k - 1 \Big\},$$

$$= \{j_1, j_2, \ldots, j_\tau\} \quad \text{where} \quad 1 \leq j_1 < j_2 < \ldots < j_\tau \leq 2^k - 1, \tag{3.17}$$

and let $\mathbf{q}_j$ be j-th column of the matrix $\mathbf{A}_b$, i.e.,

$$\mathbf{A}_b = \big(\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_{2^k-1}\big). \tag{3.18}$$

Then, a generator matrix of UEP code can be constructed from the integer programming solution as

$$\mathbf{G} = \big(\underbrace{\mathbf{q}_{j_1}, \ldots, \mathbf{q}_{j_1},}_{x_{j_1}\text{ times}} \underbrace{\mathbf{q}_{j_2}, \ldots, \mathbf{q}_{j_2},}_{x_{j_2}\text{ times}} \ \ldots\ , \underbrace{\mathbf{q}_{j_\tau}, \ldots, \mathbf{q}_{j_\tau}}_{x_{j_\tau}\text{ times}}\big). \tag{3.19}$$

Example 3.1 $(k=3)$**.** *For* $\mathbf{s}=(3,5,7)$*, it follows from* (3.8)*,*

$$\mathbf{A}_a = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{pmatrix},$$

then the generator matrix can be represented as

$$\mathbf{G} = \left(\begin{array}{c|c|c|c} 0 \cdots 0 & 0 \cdots 0 & & 1 \cdots 1 \\ 0 \cdots 0 & 1 \cdots 1 & \cdots & 1 \cdots 1 \\ 1 \cdots 1 & 0 \cdots 0 & & 1 \cdots 1 \end{array}\right) = \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{pmatrix}.$$

$$\leftarrow x_1 \rightarrow \leftarrow x_2 \rightarrow \cdots \leftarrow x_7 \rightarrow$$

From (3.16)*,*

$$\mathbf{A}_1 = \begin{pmatrix} 0001111 \end{pmatrix} = \mathbf{a}_1,$$

$$\mathbf{A}_2 = \begin{pmatrix} 0110011 \\ 0111100 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_2 \\ \mathbf{a}_2 + \mathbf{a}_1 \end{pmatrix},$$

$$\mathbf{A}_3 = \begin{pmatrix} 1010101 \\ 1011010 \\ 1100110 \\ 1101001 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_3 \\ \mathbf{a}_3 + \mathbf{a}_1 \\ \mathbf{a}_3 + \mathbf{a}_2 \\ \mathbf{a}_3 + \mathbf{a}_1 + \mathbf{a}_2 \end{pmatrix}.$$

Therefore, the UEP code construction is to find $\mathbf{x}$ *that minimizes the sum* $x_1 + x_2 + \cdots + x_7$

satisfying

$$x_4 + x_5 + x_6 + x_7 \geq 3,$$

$$x_2 + x_3 + x_6 + x_7 \geq 5,$$

$$x_2 + x_3 + x_4 + x_5 \geq 5,$$

$$x_1 + x_3 + x_5 + x_7 \geq 7,$$

$$x_1 + x_3 + x_4 + x_6 \geq 7,$$

$$x_1 + x_2 + x_5 + x_6 \geq 7,$$

$$x_1 + x_2 + x_4 + x_7 \geq 7.$$

The optimal solution obtained from the integer programming is

$$n = \sum_{j=1}^{7} x_j = 11$$

where

$$\begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ \hline 3 & 2 & 2 & 1 & 1 & 1 & 1 \end{array}.$$

From (3.19)*, the corresponding generator matrix is*

$$\mathbf{G} = \begin{pmatrix} 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1 \\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1 \\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1 \end{pmatrix}.$$

Since $\hat{\mathbf{s}} = (4, 6, 7)$ *and*

$$\hat{s}_i \geq s_i \quad \textit{for} \quad 1 \leq i \leq k,$$

the UEP code generated from $\mathbf{G}$ *satisfies the unequal error protection requirement.*

3.1.2 Bounds, results, and comparisons

Integer programming bounds

We begin by introducing a lower bound on the length of UEP codes in the following theorem.

Theorem 3.1 (Integer programming bound)**.** *For a given non-decreasing separation vector* $\mathbf{s} = (s_1, s_2, \ldots, s_k)$*, an integer programming bound is given by*

$$n \geq \sum_{i=1}^{k} \left\lceil \frac{s_i}{2^{k-i}} \right\rceil . \tag{3.20}$$

Proof. For the integer programming approach to construct UEP code, we formulate $2^k - 1$ inequalities in matrix form as

$$\mathbf{A}\mathbf{x}^\top \geq \mathbf{b}^\top.$$

Since the rows of $\mathbf{A}$ consist of $2^k - 1$ linear combinations of $\mathbf{a}_i$ for $1 \leq i \leq k$, each column of $\mathbf{A}$ has exactly 2^{k-1} ones [52, p.97]. Therefore, the sum of $2^k - 1$ inequalities can be given as

$$2^{k-1} \left(x_1 + x_2 + \cdots + x_{2^k - 1} \right) \geq \sum_{i=1}^{k} 2^{i-1} s_i, \tag{3.21}$$

hence,

$$2^{k-1} \, n \; \geq \; \sum_{i=1}^{k} 2^{i-1} s_i. \tag{3.22}$$

To preserve the integer property of the length n, we add ceiling functions into (3.22); as a result,

$$n \ \geq \ \sum_{i=1}^{k} \left\lceil \frac{2^{i-1}}{2^{k-1}} \, s_i \right\rceil . \tag{3.23}$$

Therefore,

$$n \ \geq \ \sum_{i=1}^{k} \left\lceil \frac{s_i}{2^{k-i}} \right\rceil .$$

□

We observe that the integer programming bound in (3.20) is consistent with the lower bound derived in [5, Corollary 14]. As introduced in (2.4), van Gils has derived a lower bound of a UEP code for a non-increasing separation vector $\mathbf{s}' = (s'_1, s'_2, \ldots, s'_k)$ as

$$n_{\mathbf{s}'} \geq \sum_{i=1}^{k} \left\lceil \frac{s'_i}{2^{i-1}} \right\rceil . \tag{3.24}$$

Since the separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ is in non-decreasing order, we set $s'_i = s_{k-i+1}$. Then, we transform the bound of (3.24) into

$$\begin{aligned} n_{\mathbf{s}'} &\geq \sum_{i=1}^{k} \left\lceil \frac{s'_i}{2^{i-1}} \right\rceil \\ &= \sum_{i=1}^{k} \left\lceil \frac{s_{k-i+1}}{2^{i-1}} \right\rceil \\ &= \sum_{i=1}^{k} \left\lceil \frac{s_i}{2^{k-i}} \right\rceil . \end{aligned}$$

In the following, we also include an upper bound on the length of UEP codes derived by Masnick and Wolf in [1]. As we have reviewed in (2.2), the length of a UEP code with a given error protection requirement $\mathbf{t} = (t_1, t_2, \ldots, t_k)$ can be upper bounded by

$$n \;\leq\; k + r_U \tag{3.25}$$

where r_U is the smallest number of check bits r such that

$$2^r \;>\; \sum_{i=0}^{2t_k-1} \binom{n-1}{i} - \sum_{i=2}^{k} \binom{n-1-T_i}{2t_i-2} - \sum_{i=2}^{k} \binom{n-1-T_i}{2t_i-1}. \tag{3.26}$$

Integer programming results

Numerical results of integer programming by using IBM ILOG CPLEX v12.5.1 [1] [72] for a given $\mathbf{s} = (3, 5, \ldots, 2k+1)$ are presented in Table 3.1. From $k = 2$ to 8, lengths are exactly the same as the lower bound, therefore the corresponding UEP codes are optimal. These optimal results are listed in Table 3.2. For $9 \leq k \leq 15$, instead of finding optimal solutions, we limit our search for sub-optimal solutions due to time constraint. The results and the corresponding lower and upper bounds are plotted in Figure 3.2.

In Table 3.2, for each k that has the optimal solution, we show nonzero variables x_j that construct the generator matrix[2] of the UEP code. We note that there are many combinations of x_j for $1 \leq j \leq 2^k - 1$ that are summed to the optimal value of n (e.g., there are 3 solutions to $n = 7$ when $k = 2$, 7 solutions to $n = 11$ when $k = 3$, 2063 solutions to $n = 16$ when $k = 4$, and so on).

Comparisons

In Table 3.3, we compare the length of UEP codes to the length of time sharing repetition codes and the length of shortened BCH codes [73]. First of all, the length of time sharing repetition codes that provides the error protection requirement $\mathbf{t} = (t_1, t_2, \ldots, t_k)$ can be

[1] The IBM ILOG CPLEX v12.5.1 were run on ARGO, a research computing cluster provided by the Office of Research Computing at George Mason University, VA. (URL: http://orc.gmu.edu)

[2] The corresponding generator matrices are provide in Appendix A

.

Table 3.1: Numerical results from integer programming and the corresponding upper and lower bounds for $\mathbf{s} = (3, 5, \dots, 2k+1)$

k	Lower bounds	Integer programming	Upper bounds
	(3.20)	results, n	(3.25)
2	7	7	7
3	11	11	12
4	16	16	19
5	20	20	25
6	25	25	32
7	30	30	39
8	35	35	46
9	39	$\underline{40}$[a]	53
10	44	$\underline{45}$	60
11	49	$\underline{52}$	67
12	55	$\underline{58}$	74
13	59	$\underline{64}$	81
14	64	$\underline{70}$	88
15	69	$\underline{76}$	95

[a] Underline indicates sub-optimal length

obtained by

$$n_{TS} = \sum_{i=1}^{k} n_i \tag{3.27}$$

where n_i denotes the length of component code that provide t_i error correction for $1 \le i \le k$. Since the lengths of the component codes can by given by

$$n_i = 2t_i + 1 \quad \text{for} \quad 1 \le i \le k, \tag{3.28}$$

Table 3.2: Optimal code construction from integer programming for a separation vector $\mathbf{s} = (3, 5, \ldots, 2k+1)$

k	n	x_j for $j \in \mathcal{J}_s$
2	7	x_1 x_2 x_3 4 2 1
3	11	x_1 x_2 x_3 x_4 x_5 x_6 x_7 3 2 2 1 1 1 1
4	16	x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1
5	20	x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} 2 1 1 1 1 1 1 1 1 1 x_{11} x_{12} x_{13} x_{14} x_{15} x_{16} x_{17} x_{18} x_{19} 1 1 1 1 1 1 1 1 1
6	25	x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} 1 1 1 1 1 1 1 1 1 1 1 1 1 x_{14} x_{15} x_{16} x_{17} x_{18} x_{19} x_{20} x_{21} x_{32} x_{33} x_{34} x_{35} 1 1 1 1 1 1 1 1 1 1 1 1
7	30	x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 x_{16} x_{17} x_{18} x_{19} x_{20} x_{21} x_{22} x_{23} x_{32} x_{33} x_{34} x_{35} x_{64} x_{65} x_{104} 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
8	35	x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 x_{16} x_{17} x_{18} x_{19} x_{20} x_{21} x_{22} x_{23} x_{32} x_{33} x_{34} x_{35} x_{64} x_{65} x_{66} 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 x_{126} x_{127} x_{128} x_{129} x_{184} 1 1 1 1 1

the length of time sharing repetition codes with a given separation vector $\mathbf{s} = (3, 5, \ldots, 2k+1)$ is

$$n_{TS} = \sum_{i=1}^{k} (2t_i + 1) = k(k+2) \tag{3.29}$$

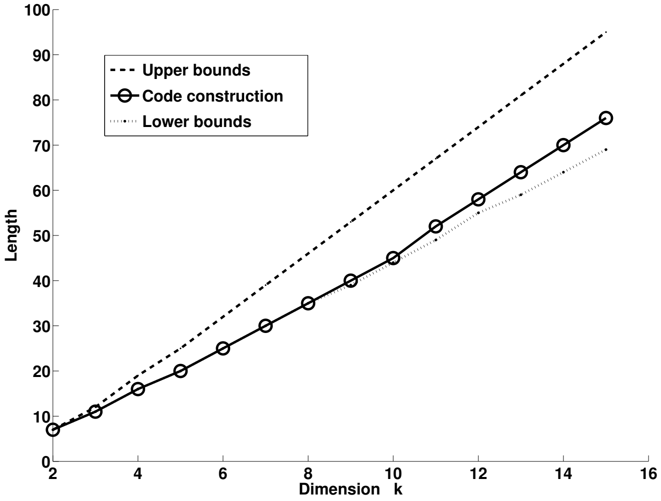


Figure 3.2: Integer programming results with bounds for $\mathbf{s} = (3, 5, \ldots, 2k+1)$

Secondly, since BCH codes are equal error protection codes, a minimum length of shortened BCH code that provides $\mathbf{t} = (t_1, t_2, \ldots, t_k)$ is equivalent to a minimum length of shortened BCH code that provides at least $t = \max_{\forall i} t_i = t_k$ error corrections. Therefore, for a given separation vector $\mathbf{s} = (3, 5, \ldots, 2k+1)$, a minimum length of shortened BCH code can be obtained by the following procedures.

1. Let (n_B, k_B, t_B) be parameters for t_B-error-correcting binary BCH codes, then we find a minimum length BCH code whose dimension k_B is at least k and error correction capability t_B is at least $t_k = \left\lfloor \frac{s_k}{2} \right\rfloor = k$.

2. Then, a minimum length of shortened BCH code with k error correction capability can be given as

$$n_{SB} = n_B - (k_B - k)\,. \tag{3.30}$$

Table 3.3: Codelength comparisons among UEP codes, Time sharing repetition codes, and shortened BCH codes for $\mathbf{s} = (3, 5, \ldots, 2k+1)$

k	UEP	Time Sharing	Shortened BCH
	n	n_{TS}	$(n_{SB}, k, \geq k)$
2	7	8	(10, 2, 2)
3	11	15	(13, 3, 3)
4	16	24	(24, 4, 5)
5	20	35	(25, 5, 5)
6	25	48	(31, 6, 7)
7	30	63	(46, 7, 7)
8	35	80	(53, 8, 10)
9	$\underline{40}$[a]	99	(54, 9, 10)
10	$\underline{45}$	120	(55, 10, 10)
11	$\underline{52}$	143	(58, 11, 11)
12	$\underline{58}$	168	(89, 12, 13)
13	$\underline{64}$	195	(90, 13, 13)
14	$\underline{70}$	224	(98, 14, 14)
15	$\underline{76}$	255	(106, 15, 15)

[a] Underline indicates sub-optimal length

3.1.3 Performance analysis

Asymptotic code rates

Based on the integer programming bounds, we compute asymptotically achievable code rates in the following theorem.

Theorem 3.2. *Let R_U be the rate of UEP code whose separation vector is given as $\mathbf{s} = (3, 5, \ldots, 2k+1)$. Then*

$$R_U \approx 0.2 \quad \textit{when} \quad k \gg 1. \tag{3.31}$$

Proof. Let n be the length of the optimal UEP code whose separation vector is $\mathbf{s} =$

$(3, 5, \ldots, 2k+1)$, then, from (3.20),

$$\begin{aligned} n &= \sum_{i=1}^{k} \left\lceil \frac{2i+1}{2^{k-i}} \right\rceil \\ &= \sum_{j=0}^{k-1} \left\lceil \frac{2(k-j)+1}{2^j} \right\rceil . \end{aligned} \tag{3.32}$$

Let

$$\eta_j = \left\lceil \frac{(2k-2j+1)}{2^j} \right\rceil ,$$

and

$$k = a_x 2^x + a_{x-1} 2^{x-1} + \cdots + a_0$$

where $x = \lfloor \log_2 k \rfloor$, then

$$\eta_j = \begin{cases} 2k+1 & , \ j=0 \\ k & , \ j=1 \\ a_x 2^{x-j+1} + \cdots + a_{j-1} 2^0 + \delta_j & , 2 \le j \le x+1 \\ 1 & , \ x+2 \le j \le k-1 \end{cases} \tag{3.33}$$

where $\delta_j \in \{0,1\}$ [74, p.47,51]. By letting $\Delta_j \triangleq \eta_j - \delta_j$ for $2 \le j \le x+1$, we can have

$$\begin{aligned} \sum_{j=2}^{x+1} \Delta_j &= a_x \left(2^{x-1} + \cdots + 1\right) + a_{x-1} \left(2^{x-2} + \cdots + 1\right) + \cdots + a_2 (2+1) + a_1 \\ &= a_x (2^x - 1) + a_{x-1} \left(2^{x-1} - 1\right) + \cdots + a_0 (1-1) \\ &= k - \sum_{w=0}^{x} a_w \end{aligned} \tag{3.34}$$

From (3.33) and (3.34),

$$\sum_{j=0}^{1} \eta_j = 3k + 1, \tag{3.35a}$$

$$\sum_{j=2}^{x+1} \eta_j = k - \sum_{w=0}^{x} a_w + \sum_{j=2}^{x+1} \delta_j, \tag{3.35b}$$

and

$$\sum_{j=x+2}^{k-1} \eta_j = k - \lfloor \log_2 k \rfloor - 2. \tag{3.35c}$$

Therefore,

$$n \;=\; \sum_{j=0}^{k-1} \eta_j \;\approx\; 5k \quad \text{when} \quad k \gg 1. \tag{3.36}$$

Consequently, the code rate R_U is converged to

$$R_U = \frac{k}{n} \approx \frac{k}{5k} = 0.2 \quad \text{when} \quad k \gg 1.$$

□

Simulation results based on $R = \frac{k}{n}$ of the UEP codes are illustrated in Figure 3.3 for $\mathbf{s} = (3, 5, \ldots, 2k + 1)$ and $2 \leq k \leq 2^{15}$, which shows asymptotic convergence to 0.2 as proven in Theorem 3.2.

Throughput of the degraded broadcast channels

Recall from Section 2.2, the proposed broadcast channel model depicted in Figure 3.1 can be considered as a degraded broadcast channel that has k component channels [27]. From k cascaded binary symmetric channels (BSCs) with parameters $\alpha_i \in [0, \frac{1}{2}]$ as shown in

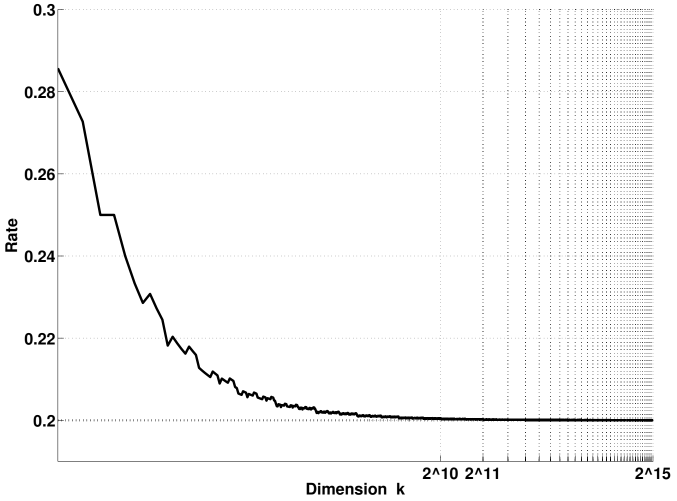


Figure 3.3: Rates of optimal UEP codes for $2 \leq k \leq 2^{15}$ when a separation vector is given as $\mathbf{s} = (3, 5, \ldots, 2k+1)$.

Figure 2.4, the bit error probability of the each component channel can be given by

$$p_i = p_{i-1}(1 - \alpha_i) + (1 - p_{i-1})\alpha_i \quad \text{for} \quad 1 \leq i \leq k \tag{3.37}$$

where $p_0 = 0$ [27]. For simplicity, let $\alpha_i = p$ for $1 \leq i \leq k$, then

$$\begin{aligned}
p_1 &= p, \\
p_2 &= (1 - p_1)p + p_1(1 - p) = p(1 + \Phi), \\
p_3 &= (1 - p_2)p + p_2(1 - p) = p(1 + \Phi + \Phi^2), \\
&\vdots \\
p_i &= (1 - p_{i-1})p + p_{i-1}(1 - p) = p(1 + \Phi + \Phi^2 + \cdots + \Phi^{i-1})
\end{aligned}$$

where $\Phi = (1 - 2p)$. Therefore, (3.37) can be written as in a closed form,

$$p_i = \frac{1 - \Phi^i}{2} \quad \text{for} \quad 1 \leq i \leq k. \tag{3.38}$$

Let n satisfy the bound (3.20) for a given separation vector $\mathbf{s} = (3, 5, \ldots, 2k + 1)$, then m_i can be successfully delivered to receiver i when the corresponding component channel introduces less than or equal to i errors. Therefore, the average rate of successful transmission of a bit to receiver R_i, denoting by θ_i, can be given as

$$\theta_i = \sum_{j=0}^{i} \binom{n}{j} (1 - p_i)^{n-j} p_i^j, \quad 1 \leq i \leq k, \tag{3.39}$$

consequently,

$$\Gamma_i = \frac{\theta_i}{n}, \quad 1 \leq i \leq k \tag{3.40}$$

where Γ_i denote the effective transmission rate of each component channel.

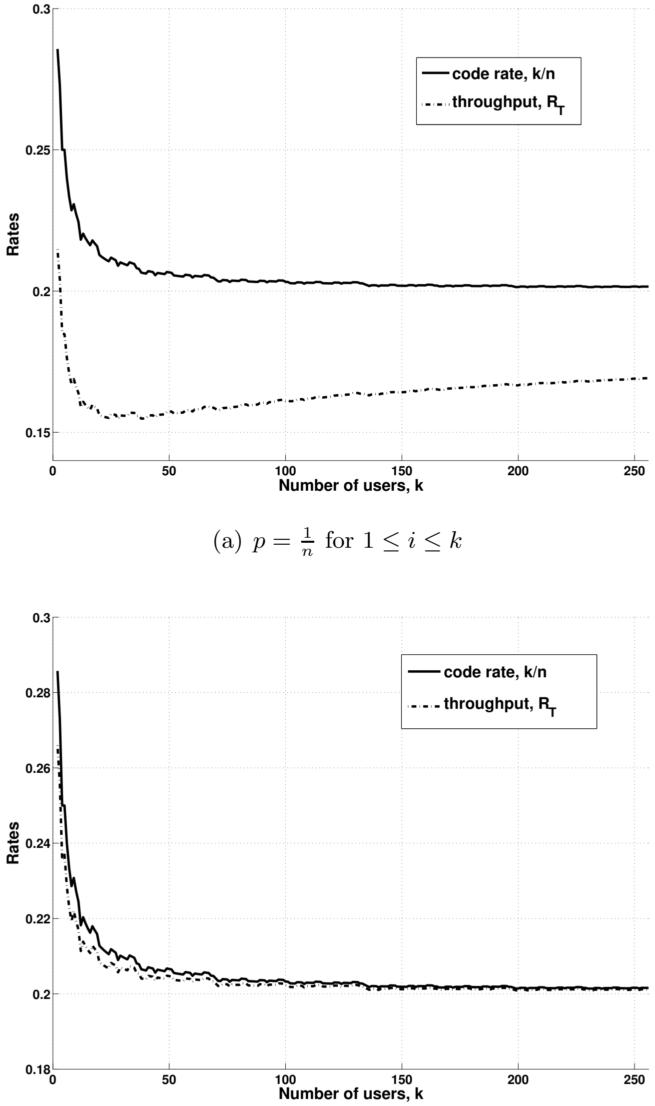
From (2.6), throughput of the degraded broadcast channel can be given by

$$R_T = \sum_{i=1}^{k} \Gamma_i \;\leq\; \frac{k}{n}. \tag{3.41}$$

Numerical results of the throughput R_T for $2 \leq k \leq 2^8$ and the corresponding code rate, $\frac{k}{n}$ for a given $\mathbf{s} = (3, 5, \ldots, 2k + 1)$, are illustrated in Figure 3.4.

Bit-error performance of the degraded broadcast channels

For the degrade broadcast channel model with cascaded BSCs, a message bit m_i cannot be recovered at a receiver i if the channel introduces more than $\left\lfloor \frac{s_i}{2} \right\rfloor$ errors when the proposed UEP coding scheme is used for multiuser communications. Therefore we can obtain a



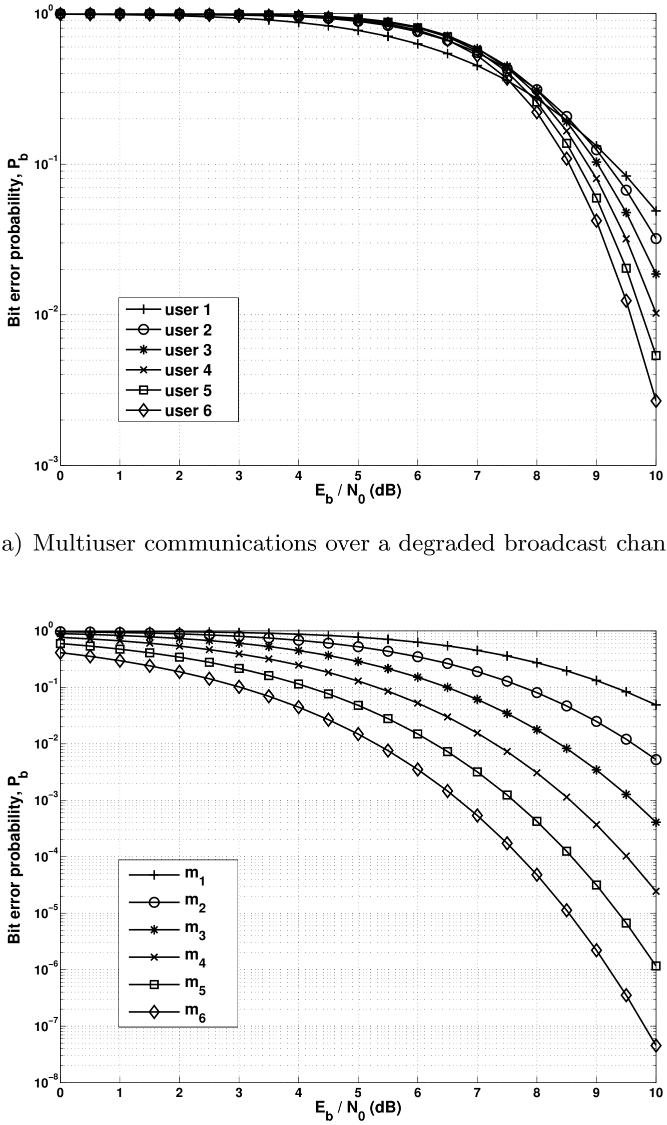
(a) $p = \frac{1}{n}$ for $1 \leq i \leq k$

(b) $p = \frac{1}{2n}$ for $1 \leq i \leq k$

Figure 3.4: Code rate and throughput of the degraded broadcast channel for $2 \leq k \leq 2^8$.

probability of bit error for a receiver i as

$$P_b(i) = \Pr(\hat{m}_i \neq m_i) = \sum_{j=i+1}^{n} \binom{n}{j} (1 - p_i)^{n-j} p_i^j, \quad 1 \leq i \leq k \tag{3.42}$$



(a) Multiuser communications over a degraded broadcast channel

(b) Multilevel error protection

Figure 3.5: Bit error performance with a binary UEP code where $n = 25$ and $k = 6$ for a given $\mathbf{s} = (3, 5, \ldots, 13)$.

where p_i is a bit error probability of the component channel as derived in (3.38). In Fig. 3.5(a), assuming an AWGN channel with BPSK signaling (i.e., $p = Q\left(\sqrt{\frac{2kE_b}{nN_0}}\right)$), we illustrate the bit error performances of the 6-user communication system over the degraded broadcast channel with a binary UEP code whose length is $n = 25$ for a given

$\mathbf{s} = (3, 5, \ldots, 13)$.

For comparison, we investigate the bit error performance when the optimal UEP codes are applied for a conventional multilevel error protection. Consider a communication scenario that a k-bit message vector $\mathbf{m} = (m_1, m_2, \ldots, m_k)$ is transmitted to a single receiver where the message bit m_i for $1 \leq i \leq k$ has levels of error protection requirement. Suppose that a separation vector $\mathbf{s} = (3, 5, \ldots, 2k+1)$ specifies the multilevel error protection, i.e., a message bit m_i is protected against i-bit channel errors. Then, a probability of bit error for each message bit can be written as

$$P_b(i) = \Pr(\hat{m}_i \neq m_i) = \sum_{j=i+1}^{n} \binom{n}{j} (1-p)^{n-j} p^j, \quad 1 \leq i \leq k. \tag{3.43}$$

where p denotes a transition probability of a single BSC (i.e., $p = Q\left(\sqrt{\frac{2kE_b}{nN_0}}\right)$ by assuming an AWGN channel with BPSK signaling). In Fig. 3.5(b), we illustrate the bit error performances of the 6-level error protection with binary UEP codes whose length is $n = 25$ for a given $\mathbf{s} = (3, 5, \ldots, 13)$.

3.1.4 Decoding of UEP codes using integer programming

Decoding by majority logic

Let a generator matrix of UEP code be represented with its column vectors $\mathbf{f}_j$ for $1 \leq j \leq n$,

$$\begin{aligned} \mathbf{G} &= \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k,1} & g_{k,2} & \cdots & g_{k,n} \end{pmatrix}, \\ &= (\mathbf{f}_1, \ \mathbf{f}_2, \ \ldots, \ \mathbf{f}_n), \end{aligned} \tag{3.44}$$

and let a received vector at receiver R_i for $1 \leq i \leq k$ be written as $\mathbf{r}_i = (r_1, r_2, \ldots, r_n)$. Then each received bit r_j can be represented by

$$r_j = \mathbf{m}\mathbf{f}_j + e_j, \tag{3.45}$$

$$= \sum_{i=1}^{k} m_i g_{i,j} + e_j \quad \text{for} \quad 1 \leq j \leq n$$

where $\mathbf{m} = (m_1, m_2, \ldots, m_k)$ is a message vector, $g_{i,j}$ is an element at the i-th row and the j-th column of the generator matrix $\mathbf{G}$, and e_j is a channel error.

Let $\mathcal{J}_h$ be a subset of $\mathcal{J}$ where $\mathcal{J} = \{1, 2, \ldots, n\}$ for a given h, and $\hat{r}_h$ be a modified received bit where

$$\hat{r}_h = \sum_{j \in \mathcal{J}_h} r_j \tag{3.46}$$

under modulo-2 additions. If the subset $\mathcal{J}_h$ satisfies

$$\sum_{j \in \mathcal{J}_h} \mathbf{f}_j = \mathbf{i}_i \tag{3.47}$$

where $\mathbf{i}_i$ denotes the i-th column of $k \times k$ identity matrix,

$$\hat{r}_h = \sum_{j \in \mathcal{J}_h} \left(\mathbf{m}\mathbf{f}_j + e_j \right),$$

$$= m_i + \sum_{j \in \mathcal{J}_h} e_j. \tag{3.48}$$

Consider the generator matrix $\mathbf{G}$ from Example 3.1 and the index subsets $\mathcal{J}_1 = \{8\}$, $\mathcal{J}_2 = \{5, 10\}$, and $\mathcal{J}_3 = \{7, 11\}$. Then, the corresponding modified received bits are written as

$$
\begin{aligned}
\hat{r}_1 &= \sum_{j \in \mathcal{J}_1} r_j = r_8 = m_1 + e_8, \\
\hat{r}_2 &= \sum_{j \in \mathcal{J}_2} r_j = r_5 + r_{10} = (\cancel{m_2} + e_5) + (m_1 + \cancel{m_2} + e_{10}) \\
&= m_1 + (e_5 + e_{10}), \\
\hat{r}_3 &= \sum_{j \in \mathcal{J}_3} r_j = r_7 + r_{11} \\
&= (\cancel{m_2} + \cancel{m_3} + e_7) + (m_1 + \cancel{m_2} + \cancel{m_3} + e_{11}) \\
&= m_1 + (e_7 + e_{11}).
\end{aligned}
$$

Suppose that the channel has introduced no error or single-bit error (i.e., $\sum_{j \in \forall j} e_j \leq 1$). Then, it is easy to see that the message bit m_1 can be determined at the receiver R_1 by majority logic since at most one of $\hat{r}_1$, $\hat{r}_2$, and $\hat{r}_3$ has an error component. Based on this observation, we develop the following theorem.

Theorem 3.3. *Suppose that, at receiver R_i for $1 \leq i \leq k$, the number of errors are less than or equal to $t_i = \left\lfloor \frac{s_i}{2} \right\rfloor$. Then the receiver R_i can recover the message m_i if there exist disjoint subsets $\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_{s_i}$ of $\mathcal{J}$ where each subset satisfies*

$$\sum_{j \in \mathcal{J}_h} \mathbf{f}_j = \mathbf{i}_i \quad \textit{for} \quad 1 \leq h \leq s_i.$$

Proof. From (3.48), each subset modifies the corresponding received bits as

$$\hat{r}_h = \sum_{j \in \mathcal{J}_h} r_j = m_i + \sum_{j \in \mathcal{J}_h} e_j \quad \text{for} \quad 1 \leq h \leq s_i. \tag{3.49}$$

If the subsets $\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_{s_i}$ are disjoint, i.e.,

$$\mathcal{J}_h \cap \mathcal{J}_{h'} = \emptyset \quad \text{for} \quad 1 \leq h \neq h' \leq s_i, \tag{3.50}$$

where $\bigcup_{h=1}^{s_i} \mathcal{J}_h \subseteq \mathcal{J}$, then less than or equal to $\left\lfloor \frac{s_i}{2} \right\rfloor$ of $\hat{r}_h$ has nonzero error components since $\sum_{\forall j} e_j \leq \left\lfloor \frac{s_i}{2} \right\rfloor$. Therefore, the message m_i can be determined at the receiver R_i by using majority logic on $(\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_{s_i})$, i.e.,

$$m_i = \begin{cases} 0 & \text{if} \quad \sum_{h=1}^{s_i} \hat{r}_h \leq \left\lfloor \frac{s_i}{2} \right\rfloor \\ 1 & \text{otherwise} \end{cases} \tag{3.51}$$

□

Based on Theorem 3.3, we formulate an integer programming problem such that it finds a subset of $\mathcal{J}$ that satisfies (3.47). Then, we iterate the integer programming until all disjoint subsets, $\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_{s_i}$ are found.

Iterative integer programming

Consider a subset $\mathcal{J}_h$ of an index set $\mathcal{J} = \{1, 2, \ldots, n\}$ that satisfies the following constraint over $\mathbb{GF}(2)$ for a given i,

$$\sum_{j \in \mathcal{J}_h} g_{i,j} = 1 \tag{3.52a}$$

$$\sum_{j \in \mathcal{J}_h} g_{i',j} = 0 \quad \text{for} \quad 1 \leq i' \neq i \leq k. \tag{3.52b}$$

Then, from (3.45) and (3.46),

$$\hat{r}_h = \sum_{j \in \mathcal{J}_h} r_j = \sum_{j \in \mathcal{J}_h} (\mathbf{m}\mathbf{f}_j + e_j)$$

$$= m_i \sum_{j \in \mathcal{J}_h} g_{i,j} + \sum_{1 \le i' \ne i \le k} m_{i'} \sum_{j \in \mathcal{J}_h} g_{i',j} + \sum_{j \in \mathcal{J}_h} e_j$$

$$= m_i + \sum_{j \in \mathcal{J}_h} e_j.$$

Therefore, the subset $\mathcal{J}_h$ which satisfies (3.47) can be obtained from the constraints (3.52) under modulo-2 addition.

Let us define a vector $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ that consists of binary variables y_j where

$$y_j \triangleq \begin{cases} 0 & \text{if} \quad j \notin \mathcal{J}_h \\ 1 & \text{if} \quad j \in \mathcal{J}_h \end{cases} \quad \text{for} \quad 1 \le j \le n. \tag{3.53}$$

Then, the constraints (3.52) to find the subset $\mathcal{J}_h$ can be rewritten under modulo-2 addition as

$$\sum_{j \in \mathcal{G}_i} y_j = 1, \tag{3.54a}$$

$$\sum_{j \in \mathcal{G}_{i'}} y_j = 0, \quad \text{for} \quad 1 \le i' \ne i \le k \tag{3.54b}$$

where $\mathcal{G}_i$ denotes an index set for the i-th row of $\mathbf{G}$ that has nonzero component, i.e.,

$$\mathcal{G}_i = \{j \mid g_{i,j} \ne 0,\ 1 \le j \le n\} \quad \text{for} \quad 1 \le i \le k. \tag{3.55}$$

Furthermore, since the objective of iterative integer programming is to find s_i numbers of disjoint subsets $\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_{s_i}$ of $\mathcal{J}$, and since

$$|J_1| + |J_2| + \cdots + |J_{s_i}| \ \leq \ |J| = n \tag{3.56}$$

where $|\cdot|$ denotes the cardinality of a set, minimizing the number of elements in a subset is desired to maximize the possibility that there exist s_i numbers of disjoint subsets.

Therefore, for receiver R_i, finding a subset $\mathcal{J}_h$ becomes to an integer programming problem that minimizes the sum of $y_1 + y_2 + \cdots + y_n$ satisfying the constraints in (3.54):

Finding an index subset at receiver R_i

Minimize

$$y_1 + y_2 + \cdots + y_n$$

subject to

$$\mathbf{G}\mathbf{y}^\top - 2\mathbf{z}^\top = \mathbf{i}_i$$

where

$$\mathbf{y} = (y_1, y_2, \ldots, y_n)\,,$$

$$y_1, y_2, \ldots, y_n \in \{0, 1\},$$

$$\mathbf{z} = (z_1, z_2, \ldots, z_k)\,,$$

$$z_1, z_2, \ldots, z_k \in \{0, 1, 2, 3, \ldots\}.$$

In the formulated integer programming problem, artificial variables z_u for $1 \leq u \leq k$ are introduced to convert the modulo-2 additions in (3.54) into the linear constraints (i.e., real additions), and they are bounded by

$$\sum_{j \in \mathcal{G}_u} y_j - 1 \ \leq \ 2z_u \ \leq \ \sum_{j \in \mathcal{G}_u} y_j \quad \text{for} \quad 1 \leq u \leq k. \tag{3.57}$$

The following table shows an example of the conversion with an artificial variable z that is bounded by

$$y_1 + y_2 + y_3 - 1 \;\leq\; 2z \;\leq\; y_1 + y_2 + y_3$$

where $z \in \{0, 1, 2, \dots\}$.

y_1	y_2	y_3	$\bigoplus_y$	$\sum_y$	z	$\sum_y$ - $2z$
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	1	0	1
0	1	1	0	2	1	0
1	0	0	1	1	0	1
1	0	1	0	2	1	0
1	1	0	0	2	1	0
1	1	1	1	3	1	1

Decoding algorithm

Example 3.2 $(k = 3)$**.** *From* **G** *in Example 3.1 and* (3.55),

$$\mathcal{G}_1 = \{8, 9, 10, 11\}\,,$$

$$\mathcal{G}_2 = \{4, 5, 6, 7, 10, 11\}\,,$$

$$\mathcal{G}_3 = \{1, 2, 3, 6, 7, 9, 11\}\,.$$

Therefore, the integer programming problem at R_1 *becomes to minimize the sum* $y_1 + y_2 + \cdots + y_{11}$ *satisfying*

$$y_8 + y_9 + y_{10} + y_{11} - 2z_1 = 1,$$

$$y_4 + y_5 + y_6 + y_7 + y_{10} + y_{11} - 2z_2 = 0,$$

$$y_1 + y_2 + y_3 + y_6 + y_7 + y_9 + y_{11} - 2z_3 = 0.$$

The optimal solution of the integer programming problem finds the smallest subset that satisfy (3.47)*;*

$$\begin{array}{ccccccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 & y_9 & y_{10} & y_{11} \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \rightarrow \mathcal{J}_1 = \{8\},$$

then R_1 *sets* $y_8 = 0$ *and repeats the integer programming to obtain disjoint subset* $\mathcal{J}_2$*;*

$$\begin{array}{ccccccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 & y_9 & y_{10} & y_{11} \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{array} \rightarrow \mathcal{J}_2 = \{5, 10\}.$$

Since R_1 *needs* $s_i = 3$ *disjoint subsets to decode* m_1*,* R_1 *sets* $y_5 = y_8 = y_{10} = 0$ *and repeats the integer programming to obtain the last disjoint subset* $\mathcal{J}_3$*;*

$$\begin{array}{ccccccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 & y_9 & y_{10} & y_{11} \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array} \rightarrow \mathcal{J}_3 = \{7, 11\}.$$

Using the disjoint subsets $\mathcal{J}_1$, $\mathcal{J}_2$, and $\mathcal{J}_3$, R_1 can determine m_1 by majority logic. We note here that the iterative integer programming to find the s_i numbers of disjoint subsets does not depend on the received information (i.e., it only depends on the structure of a generator matrix). Therefore, only the majority logic decoding is applied upon receiving information with the predetermined index subsets, which reduces the complexity of the decoding.

The detailed descriptions of the decoding algorithm is shown in Algorithm 1, and diagrams of the iterative integer programming and the majority logic decoding are illustrated in Fig. 3.6. Additionally, complete decoding procedures are listed in Table 3.4 for all receivers R_1, R_2, and R_3 with a message vector $\mathbf{m} = (1,1,1)$ and the corresponding codewords $\mathbf{c} = (1,1,1,1,1,0,0,1,0,0,1)$. Let $\mathbf{e}_1$, $\mathbf{e}_2$, and $\mathbf{e}_3$ denote error vectors at receiver R_1, R_2, and R_3, respectively. Then, the error vectors and received vectors are given as

$$\mathbf{e}_1 = (0,0,0,0,0,0,0,0,0,1,0) \;\rightarrow\; \mathbf{r}_1 = (1,1,1,1,1,0,0,1,0,1,1)\,,$$

$$\mathbf{e}_2 = (0,0,0,0,0,0,0,0,1,1,0) \;\rightarrow\; \mathbf{r}_2 = (1,1,1,1,1,0,0,1,1,1,1)\,,$$

$$\mathbf{e}_3 = (0,0,0,0,0,0,1,0,1,1,0) \;\rightarrow\; \mathbf{r}_3 = (1,1,1,1,1,0,1,1,1,1,1)\,.$$

Algorithm 1 UEP decoding algorithm at receiver R_i

```
procedure ITERATIVE INTEGER PROGRAMMING
    $h \leftarrow 0$
    repeat                                          ▷ Finding $s_i$ numbers of subsets
        $h \leftarrow h + 1$
        $\mathcal{J}_h \leftarrow$ integer programming solution
        $y_j \leftarrow 0$ for $j \in \bigcup_{1 \le w \le h} \mathcal{J}_w$
    until $h = s_i$
end procedure
Upon receiving $\mathbf{r}_i = (r_1, r_2, \ldots, r_n)$
procedure MAJORITY LOGIC DECODING
    $h \leftarrow 0$
    repeat                                          ▷ Modifying received bits
        $h \leftarrow h + 1$
        $\hat{r}_h \leftarrow \sum_{j \in \mathcal{J}_h} r_j$
    until $h = s_i$
    if $\sum_{h=1}^{s_i} \hat{r}_h \le \left\lfloor \frac{s_i}{2} \right\rfloor$ then          ▷ Majority logic
        $\hat{m}_i \leftarrow 0$
    else
        $\hat{m}_i \leftarrow 1$
    end if
end procedure
```

Table 3.4: UEP decoding example using integer programming and majority logic

$\mathbf{r}_1 = (1,1,1,1,1,0,0,1,0,1,1)$ at R_1

h	$\mathcal{J}_h$	$\hat{r}_h$	
1	$\{8\}$	$\hat{r}_1 = r_8 = 1$	
2	$\{5,10\}$	$\hat{r}_2 = r_5 + r_{10} = 0$	
3	$\{7,11\}$	$\hat{r}_3 = r_7 + r_{11} = 1$	$\to m_1 = 1$

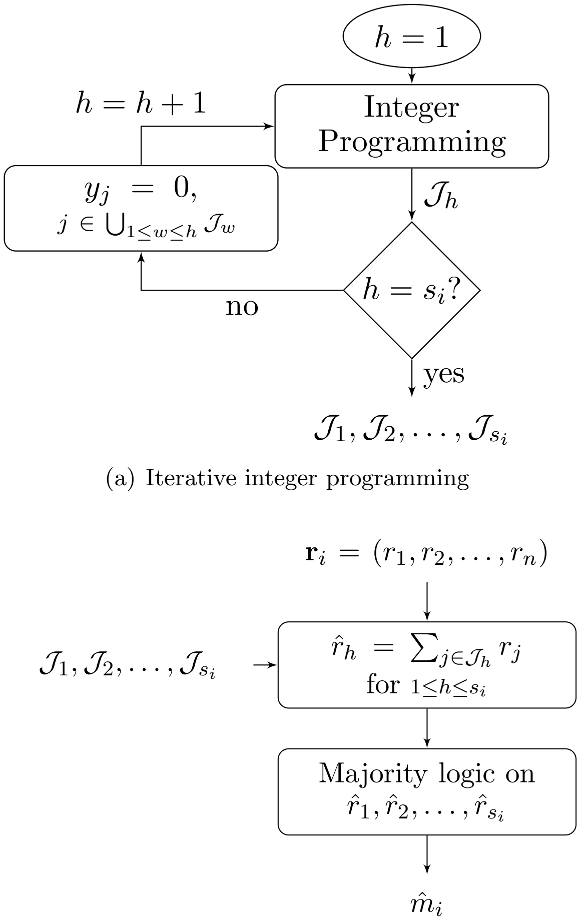
$\mathbf{r}_2 = (1,1,1,1,1,0,0,1,1,1,1)$ at R_2

h	$\mathcal{J}_h$	$\hat{r}_h$	
1	$\{4\}$	$\hat{r}_1 = r_4 = 1$	
2	$\{5\}$	$\hat{r}_2 = r_5 = 1$	
3	$\{8,10\}$	$\hat{r}_3 = r_8 + r_{10} = 0$	
4	$\{1,6\}$	$\hat{r}_4 = r_1 + r_6 = 1$	
5	$\{9,11\}$	$\hat{r}_5 = r_9 + r_{11} = 0$	$\to m_2 = 1$

$\mathbf{r}_3 = (1,1,1,1,1,0,1,1,1,1,1)$ at R_3

h	$\mathcal{J}_h$	$\hat{r}_h$	
1	$\{1\}$	$\hat{r}_1 = r_1 = 1$	
2	$\{2\}$	$\hat{r}_2 = r_2 = 1$	
3	$\{3\}$	$\hat{r}_3 = r_3 = 1$	
4	$\{8,9\}$	$\hat{r}_4 = r_8 + r_9 = 0$	
5	$\{4,6\}$	$\hat{r}_5 = r_4 + r_6 = 1$	
6	$\{10,11\}$	$\hat{r}_6 = r_{10} + r_{11} = 0$	
7	$\{5,7\}$	$\hat{r}_7 = r_5 + r_7 = 0$	$\to m_3 = 1$

The optimal UEP codes listed in Table 3.2 can be successfully decoded by using the integer programming decoding. However, not every optimal code constructed from the integer programming approach is decodable by the proposed decoding algorithm since it may not have the disjoint subsets. For example, there are $2,063$ solutions for $k = 4$ that have optimal value $n = 11$, but $1,719$ solutions (i.e., $1,719$ generator matrices for UEP codes) have the disjoint subsets. Therefore, we investigate the decodability of the proposed decoding algorithm in Section 3.3.2.


$h = 1$
Integer Programming
$h = h + 1$
$y_j = 0,$
$j \in \bigcup_{1 \le w \le h} \mathcal{J}_w$
$\mathcal{J}_h$
$h = s_i?$
no
yes
$\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_{s_i}$
$\mathbf{r}_i = (r_1, r_2, \ldots, r_n)$
$\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_{s_i}$
$\hat{r}_h = \sum_{j \in \mathcal{J}_h} r_j$
for $1 \le h \le s_i$
Majority logic on
$\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_{s_i}$
$\hat{m}_i$

(a) Iterative integer programming

(b) Majority logic decoding

Figure 3.6: Diagrams of UEP decoding method for single-bit message m_i at receiver R_i

3.2 Integer programming approach to UEP coding scheme for multi-bit messages

In this section, we extend the integer programming approach for the case of single-bit messages introduced in Section 3.1 to l-bit message case.

3.2.1 UEP code constructions using integer programming

Recall that a message vector $\mathbf{m} = (\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_k)$ consist of l-bit messages for k users denoted by

$$\mathbf{m}_i = \left(m_1^{(i)}, m_2^{(i)}, \ldots, m_l^{(i)}\right) \quad \text{for} \quad 1 \leq i \leq k \tag{3.58}$$

where $m_u^{(i)} \in \{0, 1\}$ for $1 \leq u \leq l$.

Furthermore, for a given non-decreasing separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ where $s_i = 2t_i + 1$ for $1 \leq i \leq k$, it follows from (3.3) that a UEP code can correct up to $t_i = \left\lfloor \frac{s_i}{2} \right\rfloor$ channel errors for the message $\mathbf{m}_i$ if

$$w_H(\mathbf{mG}) \ \geq \ s_i, \quad \mathbf{m}_i \neq 0 \quad \text{for} \quad 1 \leq i \leq k. \tag{3.59}$$

Let

$$m'_{(i-1)l+u} = m_u^{(i)} \quad \text{for} \quad 1 \leq i \leq k \quad \text{and} \quad 1 \leq u \leq l, \tag{3.60}$$

and

$$k' = kl. \tag{3.61}$$

Then, we can rewrite the message vector $\mathbf{m}$ as

$$\begin{aligned}\mathbf{m} &= (\mathbf{m}_1 \mid \mathbf{m}_2 \mid \ \ldots \ \mid \mathbf{m}_k) \\ &= \left(m_1^{(1)}, m_2^{(1)}, \ldots, m_l^{(1)} \mid m_1^{(2)}, m_2^{(2)}, \ldots, m_l^{(2)} \mid \ \ldots \ \mid m_1^{(k)}, m_2^{(k)}, \ldots, m_l^{(k)}\right) \\ &= \left(m'_1, m'_2, \ldots, m'_l \mid m'_{l+1}, m'_{l+2}, \ldots, m'_{2l} \mid \ \ldots \ \mid m'_{k'-l+1}, m'_{k'-l+2}, \ldots, m'_{k'}\right) \end{aligned} \tag{3.62}$$

Also, let us define the modified separation vector of length k' as $\bar{\mathbf{s}} = (\bar{s}_1, \bar{s}_2, \ldots, \bar{s}_{k'})$ where each element is written by

$$\bar{s}_{i'} = s_i \quad \text{for} \quad (i-1)l + 1 \leq i' \leq il \quad \text{and} \quad 1 \leq i \leq k, \tag{3.63}$$

i.e.,

$$\bar{\mathbf{s}} = \big(\underbrace{s_1, s_1, \ldots, s_1}_{l \text{ times}},\ \underbrace{s_2, s_2, \ldots, s_2}_{l \text{ times}},\ \ldots\ ,\ \underbrace{s_k, s_k, \ldots, s_k}_{l \text{ times}}\big). \tag{3.64}$$

Then, the constraints (3.59) is equivalent to

$$w_H\left(\mathbf{mG}\right) \ \geq\ \bar{s}_{i'}, \quad m'_{i'} = 1 \quad \text{for} \quad 1 \leq i' \leq k', \tag{3.65}$$

consequently, using the integer programming approach introduced in Section 3.1.1, we can construct a UEP code that satisfies (3.65).

Let $\mathbf{A}_b$ be the $k' \times (2^{k'} - 1)$ matrix as defined in (3.8), i.e.,

$$\mathbf{A}_b = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 1 & \cdots & 1 & 1 \\ 0 & 0 & 0 & \cdots & 1 & 0 & \cdots & 1 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & 1 & \cdots & 1 & 0 & \cdots & 1 & 1 \\ 1 & 0 & 1 & \cdots & 1 & 0 & \cdots & 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_{k'-1} \\ \mathbf{a}_{k'} \end{pmatrix} \tag{3.66}$$

then, from (3.15), we have the following inequalities for integer programming problem:

$$\mathbf{A}_{i'}\mathbf{x}^\top \geq \mathbf{b}_{i'}^\top \quad \text{for} \quad 1 \leq i' \leq k' \tag{3.67}$$

where

$$\mathbf{A}_{i'} = \begin{cases} \mathbf{a}_1 & \text{if} \quad i' = 1 \\ \mathbf{a}_{i'} + \sum_{u=1}^{i'-1} \omega_u \mathbf{a}_u, & \text{if} \quad 2 \leq i' \leq k' \end{cases} \tag{3.68a}$$

where $\omega_u \in \{0, 1\}$,

$$\mathbf{x} = \left(x_1, x_2, \ldots, x_{2^{k'}-1}\right), \tag{3.68b}$$

and

$$\mathbf{b}_{i'} = \big(\underbrace{s_{i'}, s_{i'}, \ldots, s_{i'}}_{2^{i'-1}}\big). \tag{3.68c}$$

Therefore, based on (3.67), we formulate an integer programming problem for UEP code construction with a given separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ whose elements are given as $s_i = 2t_i + 1$ for $1 \le i \le k$.

UEP code construction

Minimize

$$n = x_1 + x_2 + \cdots + x_{2^{k'}-1}$$

subject to

$$\mathbf{A}\mathbf{x}^\top \ge \mathbf{b}^\top$$

where

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_{k'} \end{pmatrix}, \quad \mathbf{b}^\top = \begin{pmatrix} \mathbf{b}_1^\top \\ \mathbf{b}_2^\top \\ \vdots \\ \mathbf{b}_{k'}^\top \end{pmatrix}$$

Example 3.3 ($k = 2$ & $l = 2$)**.** *For a given* $\mathbf{s} = (3, 5)$*, it follows from* (3.66)*,*

$$\mathbf{A}_b = \begin{pmatrix} 000000011111111 \\ 000111100001111 \\ 011001100110011 \\ 101010101010101 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{a}_4 \end{pmatrix},$$

and, from (3.68a)*,*

$$\mathbf{A}_1 = \begin{pmatrix} 000000011111111 \end{pmatrix} = (\mathbf{a}_1),$$

$$\mathbf{A}_2 = \begin{pmatrix} 000111100001111 \\ 000111111110000 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_2 \\ \mathbf{a}_2 + \mathbf{a}_1 \end{pmatrix},$$

$$\mathbf{A}_3 = \begin{pmatrix} 011001100110011 \\ 011001111001100 \\ 011110000111100 \\ 011110011000011 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_3 \\ \mathbf{a}_3 + \mathbf{a}_1 \\ \mathbf{a}_3 + \mathbf{a}_2 \\ \mathbf{a}_3 + \mathbf{a}_1 + \mathbf{a}_2 \end{pmatrix},$$

$$\mathbf{A}_4 = \begin{pmatrix} 101010101010101 \\ 101010110101010 \\ 101101001011010 \\ 101101010100101 \\ 110011001100110 \\ 110011010011001 \\ 110100101101001 \\ 110100110010110 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_4 \\ \mathbf{a}_4 + \mathbf{a}_1 \\ \mathbf{a}_4 + \mathbf{a}_2 \\ \mathbf{a}_4 + \mathbf{a}_1 + \mathbf{a}_2 \\ \mathbf{a}_4 + \mathbf{a}_3 \\ \mathbf{a}_4 + \mathbf{a}_1 + \mathbf{a}_3 \\ \mathbf{a}_4 + \mathbf{a}_2 + \mathbf{a}_3 \\ \mathbf{a}_4 + \mathbf{a}_1 + \mathbf{a}_2 + \mathbf{a}_3 \end{pmatrix}.$$

Therefore, the UEP code construction becomes to find $\mathbf{x}$ *that minimizes the sum* $x_1 + x_2 + \cdots + x_{15}$ *satisfying*

$$x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} \geq 3$$

$$x_4 + x_5 + x_6 + x_7 + x_{12} + x_{13} + x_{14} + x_{15} \geq 3$$

$$x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} \geq 3$$

$$x_2 + x_3 + x_6 + x_7 + x_{10} + x_{11} + x_{14} + x_{15} \geq 5$$

$$x_2 + x_3 + x_6 + x_7 + x_8 + x_9 + x_{12} + x_{13} \geq 5$$

$$x_2 + x_3 + x_4 + x_5 + x_{10} + x_{11} + x_{12} + x_{13} \geq 5$$

$$x_2 + x_3 + x_4 + x_5 + x_8 + x_9 + x_{14} + x_{15} \geq 5$$

$$x_1 + x_3 + x_5 + x_7 + x_9 + x_{11} + x_{13} + x_{15} \geq 5$$

$$x_1 + x_3 + x_5 + x_7 + x_8 + x_{10} + x_{12} + x_{14} \geq 5$$

$$x_1 + x_3 + x_4 + x_6 + x_9 + x_{11} + x_{12} + x_{14} \geq 5$$

$$x_1 + x_3 + x_4 + x_6 + x_8 + x_{10} + x_{13} + x_{15} \geq 5$$

$$x_1 + x_2 + x_5 + x_6 + x_9 + x_{10} + x_{13} + x_{14} \geq 5$$

$$x_1 + x_2 + x_5 + x_6 + x_8 + x_{11} + x_{12} + x_{15} \geq 5$$

$$x_1 + x_2 + x_4 + x_7 + x_9 + x_{10} + x_{12} + x_{15} \geq 5$$

$$x_1 + x_2 + x_4 + x_7 + x_8 + x_{11} + x_{13} + x_{14} \geq 5.$$

The optimal value obtained by solving the integer programming problem is

$$n = \sum_{j=1}^{15} x_j = 10,$$

where

$$\begin{array}{cccccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_7 & x_8 & x_9 & x_{10} & x_{11} \\ \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}.$$

Therefore, the corresponding generator matrix is

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,1\,1\,1\,0\,0\,0\,0 \\ 0\,1\,1\,0\,0\,1\,0\,0\,1\,1 \\ 1\,0\,1\,0\,1\,1\,0\,1\,0\,1 \end{pmatrix}.$$

3.2.2 Bounds, results, and comparisons

A lower bound on the length of the UEP code for the case of l-bit messages are given in the following corollary.

Corollary 3.1 (Integer programming bound for l-bit messages)**.** *For a given non-decreasing separation vector* $\mathbf{s} = (s_1, s_2, \ldots, s_k)$*, an integer programming bound is given by*

$$n \geq \sum_{i=1}^{k} \sum_{j=il-l+1}^{il} \left\lceil \frac{s_i}{2^{kl-j}} \right\rceil . \tag{3.69}$$

Proof. For the given separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$, we formulate $2^{kl} - 1$ inequalities

Table 3.5: Optimal results of integer programming: l-bit message

l	k	IP bound	IP result	nonzero x_j for $1 \le j \le 2^{kl} - 1$
2	2	10	10	$\begin{array}{cccccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_7 & x_8 & x_9 & x_{10} & x_{11} \\ \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$
2	3	16	16	$\begin{array}{cccccccccccccccc} x_1 & x_2 & x_4 & x_5 & x_6 & x_8 & x_9 & x_{10} & x_{11} & x_{16} & x_{17} & x_{19} & x_{32} & x_{35} & x_{47} & x_{61} \\ \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$
3	2	13	13	$\begin{array}{ccccccccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_{16} & x_{32} & x_{44} & x_{54} & x_{61} \\ \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$
4	2	15	15	$\begin{array}{ccccccccccccccc} x_1 & x_2 & x_4 & x_7 & x_8 & x_{11} & x_{13} & x_{14} & x_{16} & x_{32} & x_{64} & x_{127} & x_{128} & x_{179} & x_{213} \\ \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$

for constraints of the integer programming problem as

$$\mathbf{A}\mathbf{x}^\top \ \geq \ \mathbf{b}^\top.$$

Similar to the proof of Theorem 3.20, each column of $\mathbf{A}$ has 2^{kl-1} ones, and the sum of the inequalities can be written as

$$\begin{aligned} & 2^{kl-1}\left(x_1 + x_2 + \cdots + x_{2^{kl}-1}\right) \\ & \qquad \geq (2^l - 1)s_1 + 2^l(2^l - 1)s_2 + \cdots + 2^{(k-1)l}(2^l - 1)s_k. \end{aligned} \tag{3.70}$$

Consequently,

$$n\, 2^{kl-1} \geq \sum_{i=1}^{l} 2^{i-1} s_1 + \sum_{i=l+1}^{2l} 2^{i-1} s_2 + \cdots + \sum_{i=(k-1)l+1}^{kl} 2^{i-1} s_k. \tag{3.71}$$

Table 3.6: Code length comparisons for $\mathbf{s} = (3, 5, \ldots, 2k+1)$

l	k	UEP code	Shortened BCH[a]	Time sharing	
		n	n_{SB}	n_{TS}	component codes[b]
2	2	10	(12, 4, 2)	15	(5, 2, 1) (10, 2, 2)
2	3	16	(21, 6, 3)	27	(5, 2, 1) (10, 2, 2) (12, 2, 3)
3	2	13	(14, 6, 2)	17	(6, 3, 1) (11, 3, 2)
4	2	15	(18, 8, 2)	19	(7, 4, 1) (12, 4, 2)

[a] Shortened BCH codes with parameters: $(n_{SB},\ kl,\ t \geq k)$
[b] Shortened BCH codes with parameters: $(n_i,\ l,\ t \geq= i)$

By applying ceiling functions,

$$n \geq \sum_{i=1}^{l} \left\lceil \frac{2^{i-1}s_1}{2^{kl-1}} \right\rceil + \sum_{i=l+1}^{2l} \left\lceil \frac{2^{i-1}s_2}{2^{kl-1}} \right\rceil + \cdots + \sum_{i=(k-1)l+1}^{kl} \left\lceil \frac{2^{i-1}s_k}{2^{kl-1}} \right\rceil . \tag{3.72}$$

Therefore,

$$n \geq \sum_{i=1}^{k} \sum_{j=il-l+1}^{il} \left\lceil \frac{s_i}{2^{kl-j}} \right\rceil .$$

□

In Table 3.5, we show results of integer programming for the l-bit messages with $\mathbf{s} = (3, 5, \ldots, 2k+1)$. For the listed l and k, the optimal UEP codes are constructed by integer programming such that their lengths satisfy the integer programming bounds derived in (3.69). Generator matrices for the optimal UEP codes are constructed in Appendix A.

In addition, we compare the length of UEP codes to the length of t-error-correcting shortened BCH codes and the length of time sharing shortened BCH codes in Table 3.6. Similar to (3.30), for a separation vector given as $\mathbf{s} = (3, 5, \ldots, 2k+1)$, a minimum length of shortened BCH code that provides at least $t = \lfloor \frac{s_k}{2} \rfloor = k$ error corrections can be obtained by

$$n_{SB} = n_B - (k_B - kl) \tag{3.73}$$

where n_B is a minimum length of binary BCH code whose dimension, k_B is at least kl and whose error correction capability is at least $t = k$. Also, a length of time sharing shortened BCH codes can be written as the sum of the length of components codes,

$$n_{TS} = \sum_{i=1}^{k} n_i \tag{3.74}$$

where n_i is a minimum length of shortened BCH code whose dimension is l and whose error correction capability is at least $t = \left\lfloor \frac{s_i}{2} \right\rfloor = i$.

3.2.3 Asymptotic code rates and throughput of broadcast channels

In the following theorem, an asymptotically achievable code rate of the optimal UEP code is derived for the case of l-bit messages.

Theorem 3.4. *Let $R(l)$ be a rate of the optimal UEP code for the l-bit messages when the non-decreasing separation vector is given as $\mathbf{s} = (3, 5, \ldots, 2k+1)$. Then,*

$$R(l) \approx \frac{l}{l+4} \quad \textit{when} \quad k \gg 1. \tag{3.75}$$

Proof. From (3.69), a length of the optimal UEP code for a given $\mathbf{s} = (3, 5, \ldots, 2k+1)$ can be written as

$$n = \sum_{i=1}^{k} \sum_{j=il-l+1}^{il} \left\lceil \frac{s_i}{2^{kl-j}} \right\rceil .$$

Since $s_i = 2i + 1$ for $1 \leq i \leq k$,

$$n = \sum_{i=1}^{k} \sum_{j=il-l+1}^{il} \left\lceil \frac{2i+1}{2^{kl-j}} \right\rceil . \tag{3.76}$$

Then, from (3.63),

$$n = \sum_{u=1}^{kl} \left\lceil \frac{2\left\lceil \frac{u}{l} \right\rceil + 1}{2^{kl-u}} \right\rceil , \tag{3.77a}$$

$$= \sum_{v=0}^{kl-1} \left\lceil \frac{2k+1-2\left\lfloor \frac{v}{l} \right\rfloor}{2^{v}} \right\rceil . \tag{3.77b}$$

Let

$$\eta_v = \left\lceil \frac{2k+1-2\left\lfloor \frac{v}{l} \right\rfloor}{2^{v}} \right\rceil ,$$

and

$$k = a_x 2^x + a_{x-1} 2^{x-1} + \cdots + a_0$$

where $x = \lfloor \log_2 k \rfloor$, then

$$\eta_v = \begin{cases} 2k+1 & , \quad v = 0 \\ k+1 & , \quad v = 1 \\ a_x 2^{x-v+1} + \cdots + a_{v-1} 2^0 + \delta_v, & 2 \leq v \leq x+1 \\ 1 & , \quad x+2 \leq v \leq kl-1 \end{cases} \tag{3.78}$$

where $\delta_v \in \{0, 1\}$.

Let $\Delta_v = \eta_v - \delta_v$ for $2 \leq v \leq x + 1$, then

$$\Delta_v = a_x 2^{x-v+1} + a_{x-1} 2^{x-v} + \cdots + a_{v-1} 2^0 \tag{3.79}$$

and

$$\begin{aligned} \sum_{v=2}^{x+1} \Delta_v &= a_x \left(2^{x-1} + \cdots + 1\right) + a_{x-1} \left(2^{x-2} + \cdots + 1\right) + \cdots + a_2 (2+1) + a_1 \\ &= a_x (2^x - 1) + a_{x-1} \left(2^{x-1} - 1\right) + \cdots + a_0 (1 - 1) \\ &= k - \sum_{w=0}^{x} a_w. \end{aligned} \tag{3.80}$$

From (3.78) and (3.80),

$$\sum_{v=0}^{1} \eta_v = 3k + 2, \tag{3.81a}$$

$$\sum_{v=2}^{x+1} \eta_v = k - \sum_{w=0}^{x} a_w + \sum_{v=2}^{x+1} \delta_v, \tag{3.81b}$$

and

$$\sum_{v=x+2}^{kl-1} \eta_v = kl - \lfloor \log_2 k \rfloor - 2. \tag{3.81c}$$

Therefore,

$$n \;=\; \sum_{v=0}^{kl-1} \eta_v \;\approx\; (l+4)k \quad \text{when} \quad k \gg 1, \tag{3.82}$$

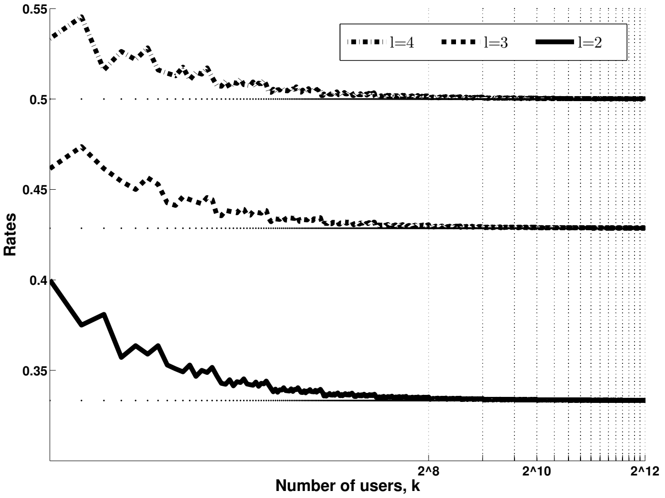


Figure 3.7: Rates of optimal UEP codes for $2 \leq k \leq 2^{12}$ when $l = 2, 3, 4$.

consequently, the code rate is converged to

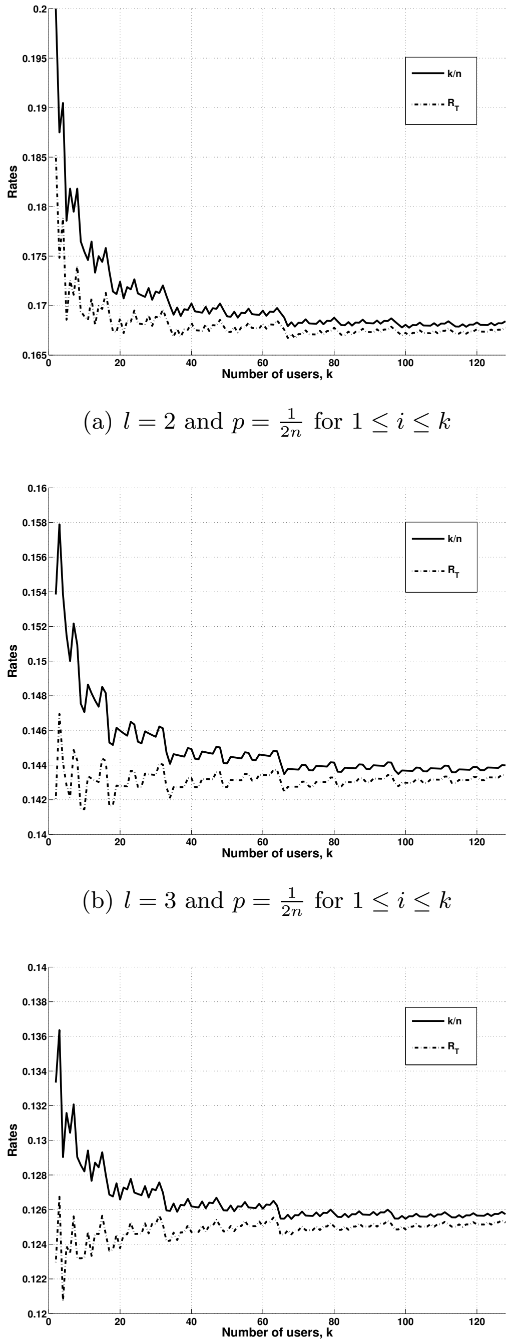
$$R(l) = \frac{kl}{n} \approx \frac{l}{(l+4)} \quad \text{when} \quad k \gg 1.$$

□

Simulation results for the rates of the UEP codes obtained by computing $R(l) = \frac{kl}{n}$ are illustrated in Fig. 3.7 for $2 \leq k \leq 2^{12}$, which shows asymptotic convergence $\frac{l}{l+4}$ for $l = 2, 3$, and 4 as proven in Theorem 3.4.

As explained in Section 2.2, the broadcast channel model can be viewed as k degraded component channels. Consequently, the model can be described by k cascaded BSCs with transition probabilities, α_i for $1 \leq i \leq k$ as illustrated in Figure 2.4.

Let n satisfy the bound in (3.69) for a given separation vector $\mathbf{s} = (3, 5, \ldots, 2k+1)$. It follows from (3.37) and (3.39), since the l-bit message $\mathbf{m}_i$ is successfully delivered to the receiver R_i when the corresponding component channel introduces less than or equal to i errors, the effective transmission rate of the l-bit message $\mathbf{m}_i$ to the receiver R_i through the



(a) $l = 2$ and $p = \frac{1}{2n}$ for $1 \leq i \leq k$

(b) $l = 3$ and $p = \frac{1}{2n}$ for $1 \leq i \leq k$

(c) $l = 4$ and $p = \frac{1}{2n}$ for $1 \leq i \leq k$

Figure 3.8: Throughput of the degraded broadcast channel for $2 \leq k \leq 2^7$

component channel can be given by

$$\Gamma_i = \frac{1}{n}\sum_{j=0}^{i}\binom{n}{j}(1-p_i)^{n-j}p_i^j, \quad 1 \le i \le k \tag{3.83}$$

where p_i is a bit error probability of the component channel as derived in (3.38). Consequently, the throughput of the degraded broadcast channel can be given by

$$R_T = \sum_{i=1}^{k}\Gamma_i \ \le \ \frac{k}{n}. \tag{3.84}$$

We illustrate the throughput of the degraded broadcast channel for the case of $l = 2$, 3, and 4 in Figure 3.8 when $\alpha_i = p = \frac{1}{2n}$ for $1 \le i \le k$.

3.2.4 Decoding of UEP codes using integer programming

In this section, we extend the decoding algorithm introduced in Section 3.1.4 to decode l-bit messages, $\mathbf{m}_i$ at each receiver R_i for $1 \le i \le k$. Since a message vector is given as $\mathbf{m} = (\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_k)$ where $\mathbf{m}_i = \left(m_1^{(i)}, m_2^{(i)}, \ldots, m_l^{(i)}\right)$ for $1 \le i \le k$, each element of the received vector $\mathbf{r}_i$ can be written as

$$r_j = \sum_{i=1}^{k}\sum_{u=1}^{l} m_u^{(i)} g_{l(i-1)+u,j} + e_j \quad \text{for} \quad 1 \le j \le n \tag{3.85}$$

where $g_{i,j}$ represents the i-th row and the j-th column element of $\mathbf{G}$ and e_j is a channel error. In the following, we explain the decoding method with an example of $k = 2$ and $l = 2$ to recover $\mathbf{m}_1 = \left(m_1^{(1)}, m_2^{(1)}\right)$ at receiver R_1 .

Example 3.4 ($k = 2$ & $l = 2$)**.** *From Example 3.3, the corresponding generator matrix is*

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,1\,1\,1\,0\,0\,0\,0 \\ 0\,1\,1\,0\,0\,1\,0\,0\,1\,1 \\ 1\,0\,1\,0\,1\,1\,0\,1\,0\,1 \end{pmatrix}.$$

Iterative integer programming *Receiver R_1 first finds subsets $\mathcal{J}_1^1, \mathcal{J}_2^1, \mathcal{J}_3^1$ using iterative integer programming to decode $m_1^{(1)}$. From the generator matrix, we have*

$$\mathcal{G}_1 = \{7, 8, 9, 10\},$$

$$\mathcal{G}_2 = \{4, 5, 6\},$$

$$\mathcal{G}_3 = \{2, 3, 6, 9, 10\},$$

$$\mathcal{G}_4 = \{1, 3, 5, 6, 8, 10\}.$$

Therefore, the integer programming problem that finds a subset $\mathcal{J}_h^1$ for $1 \leq h \leq s_i$ becomes to minimize the sum $y_1 + y_2 + \cdots + y_{10}$ satisfying

$$\begin{aligned} y_7 + y_8 + y_9 + y_{10} - 2z_1 &= 1, \\ y_4 + y_5 + y_6 - 2z_2 &= 0, \\ y_2 + y_3 + y_6 + y_9 + y_{10} - 2z_3 &= 0, \\ y_1 + y_3 + y_5 + y_6 + y_8 + y_{10} - 2z_4 &= 0. \end{aligned}$$

The subsets obtained from integer programming are $\mathcal{J}_1^1 = \{7\}$*,* $\mathcal{J}_2^1 = \{2, 9\}$*, and* $\mathcal{J}_3^1 = \{1, 8\}$*. Since the subsets for decoding* $m_1^{(1)}$ *has been found,* R_1 *sets* $g_{1,j} = 0$ *for* $j \in \mathcal{G}_1$*, and* R_1 *finds subsets* $\mathcal{J}_1^2, \mathcal{J}_2^2, \mathcal{J}_3^2$ *to decode* $m_2^{(1)}$*. Since we have*

$$\mathcal{G}_1 = \{\} = \emptyset,$$

$$\mathcal{G}_2 = \{4, 5, 6\},$$

$$\mathcal{G}_3 = \{2, 3, 6, 9, 10\},$$

$$\mathcal{G}_4 = \{1, 3, 5, 6, 8, 10\},$$

the integer programming problem that finds a subset $\mathcal{J}_h^2$ *for* $1 \leq h \leq s_i$ *becomes to minimize the sum* $y_1 + y_2 + \cdots + y_{10}$ *satisfying*

$$\begin{aligned} y_4 + y_5 + y_6 - 2z_2 = &\quad 1, \\ y_2 + y_3 + y_6 + y_9 + y_{10} - 2z_3 = &\quad 0, \\ y_1 + y_3 + y_5 + y_6 + y_8 + y_{10} - 2z_4 = &\quad 0. \end{aligned}$$

The subsets obtained from integer programming are $\mathcal{J}_1^2 = \{4\}$*,* $\mathcal{J}_2^2 = \{6, 10\}$*, and* $\mathcal{J}_3^2 = \{1, 5\}$*.*

Majority logic decoding *Upon receiving* $\mathbf{r}_i = (r_1, r_2, \ldots, r_n)$*,* R_1 *recovers* $\mathbf{m}_1 = \left(m_1^{(1)}, m_2^{(1)}\right)$ *using majority logic based on the subsets found from iterative integer programming. Since the subsets for* $m_1^{(1)}$ *are* $\mathcal{J}_1^1 = \{7\}$*,* $\mathcal{J}_2^1 = \{2, 9\}$*, and* $\mathcal{J}_3^1 = \{1, 8\}$*,*

$$\hat{r}_1 = r_7 = m_1^{(1)} + e_7,$$

$$\hat{r}_2 = r_2 + r_9 = \left(\cancel{m_1^{(2)}} + e_2\right) + \left(m_1^{(1)} + \cancel{m_1^{(2)}} + e_9\right)$$

$$= m_1^{(1)} + (e_2 + e_9),$$

$$\hat{r}_3 = r_1 + r_8 = \left(\cancel{m_2^{(2)}} + e_1\right) + \left(m_1^{(1)} + \cancel{m_2^{(2)}} + e_8\right)$$

$$= m_1^{(1)} + (e_1 + e_8).$$

Consequently, $m_1^{(1)}$ *can be determined by majority logic if* $\sum_{\forall j} e_j \leq t_i$*.*

To cancel out $m_1^{(1)}$ *in the received vector,* R_1 *updates* $\mathbf{r}_i = (r_1, r_2, \ldots, r_n)$ *by*

$$r_j = r_j + m_1^{(1)} \quad \textit{for} \quad j \in \mathcal{G}_1,$$

i.e.,

$$r_7 = r_7 + m_1^{(1)} = \cancel{m_1^{(1)}} + \cancel{m_1^{(1)}} + e_7,$$

$$r_8 = r_8 + m_1^{(1)} = \cancel{m_1^{(1)}} + \cancel{m_1^{(1)}} + m_2^{(2)} + e_8,$$

$$r_9 = r_9 + m_1^{(1)} = \cancel{m_1^{(1)}} + \cancel{m_1^{(1)}} + m_1^{(2)} + e_9,$$

$$r_{10} = r_{10} + m_1^{(1)} = \cancel{m_1^{(1)}} + \cancel{m_1^{(1)}} + m_1^{(2)} + m_2^{(2)} + e_{10}.$$

Then, R_1 *use the subsets* $\mathcal{J}_1^2 = \{4\}$*,* $\mathcal{J}_2^2 = \{6, 10\}$*, and* $\mathcal{J}_3^2 = \{1, 5\}$ *to obtain the following* *modified received bits:*

$$\begin{aligned}
\hat{r}_1 &= r_4 = m_2^{(1)} + e_4, \\
\hat{r}_2 &= r_6 + r_{10} \\
&= \left(m_2^{(1)} + \cancel{m_1^{(2)}} + \cancel{m_2^{(2)}} + e_6\right) + \left(\cancel{m_1^{(2)}} + \cancel{m_2^{(2)}} + e_{10}\right) \\
&= m_2^{(1)} + (e_6 + e_{10}), \\
\hat{r}_3 &= r_1 + r_5 = \left(\cancel{m_2^{(2)}} + e_1\right) + \left(m_2^{(1)} + \cancel{m_2^{(2)}} + e_5\right) \\
&= m_2^{(1)} + (e_1 + e_5).
\end{aligned}$$

Consequently, R_1 recovers $m_2^{(1)}$ by majority logic if $\sum_{\forall j} e_j \leq t_i$. Complete decoding procedures are listed in Table 3.7 for receivers R_1 and R_2 with a message vector $\mathbf{m} = (1,1,1,1)$ and the corresponding codewords $\mathbf{c} = (1,1,0,1,0,1,1,0,0,1)$.

In general, R_i first finds subsets $\mathcal{J}_1^u, \mathcal{J}_2^u, \ldots, \mathcal{J}_{s_i}^u$ using iterative integer programming for $1 \leq u \leq l$. Then, upon receiving the received bit $\mathbf{r}_i$, the receiver R_i applies majority logic on the modified received bits,

$$\hat{r}_h = \sum_{j \in \mathcal{J}_h^u} r_j = m_u^{(i)} + \sum_{j \in \mathcal{J}_h^u} e_j \quad \text{for} \quad 1 \leq h \leq s_i \tag{3.86}$$

to determine $m_u^{(i)}$, and R_i repeats procedures for $1 \leq u \leq l$. The detailed descriptions of the decoding algorithm is shown in Algorithm 2, and diagrams of the iterative integer programming and the majority logic decoding are illustrated in Fig. 3.9.

Algorithm 2 Decoding algorithm at R_i for l-bit message

```
procedure ITERATIVE INTEGER PROGRAMMING
    $u \leftarrow 0$
    repeat                                                    ▷ Finding subsets for $m_u^{(i)}$
        $u \leftarrow u + 1$
        $v \leftarrow l(i-1) + u$
        Formulate an integer programming problem
            $\min \quad y_1 + y_2 + \cdots + y_n$
            $\text{s.t.} \quad \mathbf{G}\mathbf{y}^\top - 2\mathbf{z}^\top = \mathbf{i}_v$
        $h \leftarrow 0$
        repeat                                                ▷ Finding $s_i$ numbers of subsets
            $h \leftarrow h + 1$
            $\mathcal{J}_h^u \leftarrow$ integer programming solution
            $y_j \leftarrow 0$ for $j \in \bigcup_{1 \leq w \leq h} \mathcal{J}_w^u$
        until $h = s_i$
        $g_{v,j} \leftarrow 0$ for $j \in \mathcal{G}_v$
    until $u = l$
end procedure
Upon receiving $\mathbf{r}_i = (r_1, r_2, \ldots, r_n)$
procedure MAJORIY LOGIC DECODING
    $u \leftarrow 0$
    repeat                                                    ▷ Determine $m_u^{(i)}$
        $u \leftarrow u + 1$
        $h \leftarrow 0$
        repeat                                                ▷ Modifying received bits
            $h \leftarrow h + 1$
            $\hat{r}_h = \sum_{j \in \mathcal{J}_h^u} r_j$
        until $h = s_i$
        if $\sum_{h=1}^{s_i} \hat{r}_h \leq \left\lfloor \frac{s_i}{2} \right\rfloor$ then                      ▷ Majority logic
            $\hat{m}_u^{(i)} \leftarrow 0$
        else
            $\hat{m}_u^{(i)} \leftarrow 1$
        end if
        $r_j \leftarrow r_j + \hat{m}_u^{(i)}$ for $j \in \mathcal{G}_{l(i-1)+u}$
    until $u = l$
end procedure
```

Table 3.7: Decoding example for $l = 2$ and $k = 2$

$\mathbf{r}_1 = (1, 1, 0, 1, 0, 1, 1, 0, 1, 1)$ at R_1

u	h	$\mathcal{J}_h$	$\hat{r}_h$
1	1	$\{7\}$	$\hat{r}_1 = r_7 = 1$
	2	$\{2, 9\}$	$\hat{r}_2 = r_2 + r_9 = 0$
	3	$\{1, 8\}$	$\hat{r}_3 = r_1 + r_8 = 1 \qquad \to m_1^{(1)} = 1$

$\mathbf{r}_1 \leftarrow \mathbf{r}_1 + (0, 0, 0, 0, 0, 0, 1, 1, 1, 1)$

$\mathbf{r}_1 = (1, 1, 0, 1, 0, 1, 0, 1, 0, 0)$

u	h	$\mathcal{J}_h$	$\hat{r}_h$
2	1	$\{4\}$	$\hat{r}_1 = r_4 = 1$
	2	$\{6, 10\}$	$\hat{r}_2 = r_6 + r_{10} = 1$
	3	$\{1, 5\}$	$\hat{r}_3 = r_1 + r_5 = 1 \qquad \to m_2^{(1)} = 1$

$\mathbf{r}_2 = (1, 1, 0, 1, 0, 1, 1, 1, 1, 1)$ at R_2

u	h	$\mathcal{J}_h$	$\hat{r}_h$
1	1	$\{2\}$	$\hat{r}_1 = r_2 = 1$
	2	$\{7, 9\}$	$\hat{r}_2 = r_7 + r_9 = 0$
	3	$\{1, 3\}$	$\hat{r}_3 = r_1 + r_3 = 1$
	4	$\{8, 10\}$	$\hat{r}_4 = r_8 + r_{10} = 0$
	5	$\{5, 6\}$	$\hat{r}_5 = r_5 + r_6 = 1 \qquad \to m_1^{(2)} = 1$

$\mathbf{r}_2 \leftarrow \mathbf{r}_2 + (0, 1, 1, 0, 0, 1, 0, 0, 1, 1)$

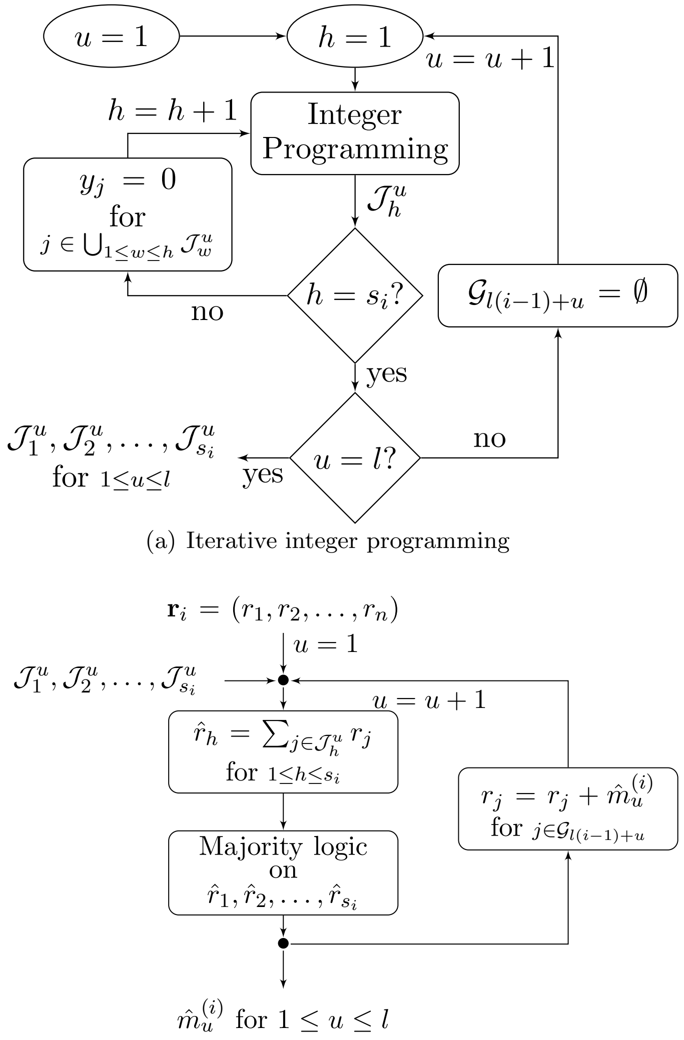
$\mathbf{r}_2 = (1, 0, 1, 1, 0, 0, 1, 1, 0, 0)$

2	1	$\{1\}$	$\hat{r}_1 = r_1 = 1$
	2	$\{3\}$	$\hat{r}_2 = r_3 = 1$
	3	$\{9, 10\}$	$\hat{r}_3 = r_9 + r_{10} = 0$
	4	$\{7, 8\}$	$\hat{r}_4 = r_7 + r_8 = 0$
	5	$\{4, 6\}$	$\hat{r}_5 = r_4 + r_6 = 1 \qquad \to m_2^{(2)} = 1$

3.3 Discussions

3.3.1 On the complexity of UEP code construction

In Section 3.1.1 (also, in Section 3.2.1), we introduce an integer programming approach to construct optimal UEP codes for k users in broadcast communications. Essentially,


$u = 1$
$h = 1$
$u = u + 1$
$h = h + 1$
Integer Programming
$y_j = 0$ for $j \in \bigcup_{1 \le w \le h} \mathcal{J}_w^u$
$\mathcal{J}_h^u$
$h = s_i?$
no
$\mathcal{G}_{l(i-1)+u} = \emptyset$
yes
$\mathcal{J}_1^u, \mathcal{J}_2^u, \ldots, \mathcal{J}_{s_i}^u$ for $1 \le u \le l$
yes
$u = l?$
no
$\mathbf{r}_i = (r_1, r_2, \ldots, r_n)$
$u = 1$
$\mathcal{J}_1^u, \mathcal{J}_2^u, \ldots, \mathcal{J}_{s_i}^u$
$u = u + 1$
$\hat{r}_h = \sum_{j \in \mathcal{J}_h^u} r_j$ for $1 \le h \le s_i$
$r_j = r_j + \hat{m}_u^{(i)}$ for $j \in \mathcal{G}_{l(i-1)+u}$
Majority logic on $\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_{s_i}$
$\hat{m}_u^{(i)}$ for $1 \le u \le l$

(a) Iterative integer programming

(b) Majority logic decoding

Figure 3.9: Diagrams of UEP decoding method for l-bit message $\mathbf{m}_i = (m_1^{(i)}, m_2^{(i)}, \ldots, m_l^{(i)})$ at receiver R_i

when a separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ is given, an integer programming problem is formulated based on a basis matrix, $\mathbf{A}_b$ where $\mathbf{A}_b$ is $k \times (2^k - 1)$ matrix consisting of all nonzero binary k-tuples as columns in increasing order. Then the UEP code construction becomes to minimize the sum $x_1 + x_2 + \cdots + x_{2^k-1}$ subject to satisfying certain linear constraints.

However, we observe that the number of variables for the objective function, $x_1 + x_2 +$

$\cdots + x_{2^k-1}$ exponentially increases as the number of users, k is increased. Furthermore, since integer programming is NP-hard problem, it may not provide an optimal solution in reasonable time for a large number of variables as results indicate in Section 3.1.2. These observations make the integer programming approach less practical when k is a large number; hence, we wish to direct further researches on the complexity of the integer programming approach. In the following, we present an idea to reduce the complexity of the approach.

UEP code construction based on shortened BCH codes In the proposed integer programming approach, we use a $k \times (2^k - 1)$ basis matrix, but we can reduce the complexity of integer programming significantly by using a generator matrix of a binary BCH code as the basis matrix. Suppose that a non-decreasing separation vector is given as $\mathbf{s} = (s_1, s_2, \ldots, s_k)$, then the UEP code construction based on shortened BCH codes can be described by the following procedures.

1. Choose a binary BCH code whose dimension is at least k and whose error correction capability is at least $t = \left\lfloor \frac{s_k}{2} \right\rfloor = k$, and let the parameters of the BCH code be (n_B, k_B, t).

2. Obtain a shortened the BCH code with parameters (n_{SB}, k, t) where $n_{SB} = n_B - (k_B - k)$.

   - example for $k = 5$: $(31, 11, 5)$ BCH code $\rightarrow$ $(25, 5, 5)$ shortened BCH code.

3. Set the generator matrix of the shortened BCH code as the basis matrix $\mathbf{A}_b$.

- example for $k = 5$:

$$\mathbf{A}_b = \begin{pmatrix} 1010101101100100011010000 \\ 0101010110110010001101000 \\ 0010101011011001000110100 \\ 0001010101101100100011010 \\ 0000101010110110010001101 \end{pmatrix}$$

4. Formulate integer programming as described in section 3.1.1.

5. Find the optimal solution to the formulated problem.

   - example for $k = 5$: optimal value $n = 21$ where

$$\begin{array}{ccccccccccc} x_4 & x_7 & x_{10} & x_{11} & x_{13} & x_{15} & x_{16} & x_{17} & x_{19} & x_{23} & x_{25} \\ \hline 2 & 1 & 1 & 1 & 2 & 2 & 1 & 3 & 1 & 2 & 5 \end{array},$$

hence,

$$\mathbf{G} = \begin{pmatrix} 001110000000010000000 \\ 110010011000010000000 \\ 001101100100001100000 \\ 110111100011100000000 \\ 001010011000001111111 \end{pmatrix}$$

As it is shown in the example of $k = 5$, the complexity of integer programming is reduced since the number of variables are reduced from 31 (see Section 3.1.1) to 25. However, since the dimension of basis matrix $\mathbf{A}_b$ is decreased to reduce the complexity, the optimal solution from integer programming no longer guarantees to satisfy the bound derived in Section 3.1.2. Therefore, there is a trade-off between the complexity of UEP code construction and the

Table 3.8: Comparisons of UEP code construction for $\mathbf{s} = (3, 5, \ldots, 2k+1)$

k	Code construction based on $k \times (2^k - 1)$ matrix		Code construction based on shortened BCH codes	
	no. of variables	code length	no. of variables	code length
2	3	7	10	7
3	7	11	13	11
4	15	16	24	16
5	31	20	25	21
6	63	25	31	25
7	127	30	46	31
8	255	35	53	38

efficiency of the codes as we compared in Table 3.8. From the observation, reducing the complexity of the integer programming approach would be an interesting subject to research.

3.3.2 On the decodability of the UEP decoding algorithm

From Section 3.1.1, we find an integer programming solution denoted by $\mathbf{x} = \big(x_1, x_2, \ldots, x_{2^k-1}\big)$ whose nonzero components construct a generator matrix of a UEP code. Let $\mathcal{I}$ be an index set for the nonzero components of $\mathbf{x}$, i.e.,

$$\mathcal{I} = \Big\{ j \mid x_j \neq 0,\ 1 \leq j \leq 2^k - 1 \Big\}, \tag{3.87}$$

and let $\mathcal{I}_h$ be a subset of $\mathcal{I}$ for $1 \leq h \leq s_i$. Then, for a given i, the sets $\mathcal{I}_h$ for $1 \leq h \leq s_i$ satisfy (3.47) if

$$\sum_{j \in \mathcal{I}_h} \mathbf{q}_j = \mathbf{i}_i \quad \text{for} \quad 1 \leq h \leq s_i \tag{3.88}$$

under modulo-2 additions where $\mathbf{q}_j$ is the j-th column of $\mathbf{A}_b$ from (3.8), i.e.,

$$\begin{aligned}\mathbf{A}_b &= \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,2^k-1} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,2^k-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k,1} & a_{k,2} & \cdots & a_{k,2^k-1} \end{pmatrix} \\ &= \big(\mathbf{q}_1,\ \mathbf{q}_2,\ \ldots,\ \mathbf{q}_{2^k-1},\big) \end{aligned} \tag{3.89}$$

and $\mathbf{i}_i$ is i-th column of $k \times k$ identity matrix.

Let $\mathbf{x}_h = \big(x_1^{(h)}, x_2^{(h)}, \ldots, x_{2^k-1}^{(h)}\big)$ be defined as

$$x_j^{(h)} \triangleq \begin{cases} 1 & \text{if} \quad j \in \mathcal{I}_h \\ 0 & \text{otherwise} \end{cases} \quad \text{for} \quad 1 \leq j \leq 2^k - 1. \tag{3.90}$$

If

$$\sum_{h=1}^{s_i} x_j^{(h)} \leq x_j \quad \forall j, \tag{3.91}$$

then the s_i numbers of disjoint subsets in Theorem 3.3 exist. Therefore, if the integer programming solution $\mathbf{x}$ satisfies (3.88) and (3.91) for $1 \leq i \leq k$, the corresponding UEP code is majority-logic-decodable by the proposed decoding algorithm. For example, the optimal UEP codes shown in Table 3.2 have the s_i numbers of the disjoint subsets, hence they are majority-logic-decodable.

Modified integer programming approach of UEP code construction Let $\mathcal{A}_i$ be an index set for i-th row of $\mathbf{A}_b$ that has non-zero component, i.e.,

$$\mathcal{A}_i = \left\{ j \mid a_{i,j} = 1,\ 1 \leq j \leq 2^k - 1 \right\}.$$

Then, consider the variable x_j for a given i that satisfies the following constraints:

$$x_j \geq 1, \qquad j \neq \xi,\ j \in \mathcal{A}_i \tag{3.92a}$$

$$x_{j'} - x_j \geq 0, \qquad j' = j - \xi \tag{3.92b}$$

where $\xi = 2^{k-i}$. Recall that if $x_j \geq 1$, then $\mathbf{q}_j$ appears x_j times in columns of $\mathbf{G}$. Consequently, the constraints (3.92) guarantee that if $\mathbf{q}_j$ appears x_j times in columns of $\mathbf{G}$, then $\mathbf{q}_{j'}$ appears at least x_j times in columns of $\mathbf{G}$. Furthermore, since

$$\mathbf{q}_j + \mathbf{q}_{j'} = \mathbf{i}_i$$

where $\mathbf{i}_i$ is i-th column of $k \times k$ identity matrix, the constraints (3.92) also guarantee that there exist x_j numbers of subsets that satisfy (3.47) if $x_j \geq 1$ for $j \neq \xi$ and $j \in \mathcal{A}_i$.

Let a binary indicator variable w_{ij} be defined as

$$w_{ij} \triangleq \begin{cases} 1 & \text{if} \quad x_j \geq 1 \text{ and } x_{j'} - x_j \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.93}$$

for $j \in \mathcal{A}_i$, $j \neq \xi$, and $1 \leq i \leq k$. Then, there exist s_i numbers of disjoint subsets that satisfy (3.47) if

$$\sum_{\substack{j \neq \xi \\ j \in \mathcal{A}_i}} w_{ij} x_j + x_\xi \geq s_i \quad \text{for} \quad 1 \leq i \leq k. \tag{3.94}$$

Therefore, (3.94) can be additional constraints for integer programming so that the corresponding UEP codes are majority-logic-decodable.

The logical constraints in (3.93) can be rewritten as integer programming constraints. First, let binary indicator variables u_{ij} and v_{ij} be

$$u_{ij} = 1 \quad \text{if} \quad x_j \geq 1, \tag{3.95a}$$

$$v_{ij} = 1 \quad \text{if} \quad x_{j'} - x_j \geq 0. \tag{3.95b}$$

Then, since $w_{ij} = u_{ij}v_{ij}$, the indicator variable w_{ij} can be expressed with the following inequalities:

$$u_{ij} + v_{ij} - 1 \;\leq\; 2w_{ij} \;\leq\; u_{ij} + v_{ij}. \tag{3.96}$$

Similarly, the quadratic terms (i.e., $w_{ij}x_j$) in (3.94) can be rewritten as linear constraints by introducing artificial variables z_{ij} such that $z_{ij} = w_{ij}x_j$. Suppose the variables x_j are bounded by $0 \leq x_j \leq M$ for $1 \leq j \leq 2^k - 1$ where M is a sufficiently large constant. Then, z_{ij} can be expressed with the following pairs of inequalities:

$$0 \;\leq z_{ij} \;\leq\; Mw_{ij}, \tag{3.97a}$$

$$x_j - M(1 - w_{ij}) \;\leq z_{ij} \;\leq\; x_j. \tag{3.97b}$$

Therefore, from the integer programming problem formulated in Section 3.1.1 and the additional constraints (3.94), we can construct UEP codes that are majority-logic-decodable

Modified UEP code construction

Minimize

$$n = x_1 + x_2 + \cdots + x_{2^k-1}$$

subject to

$$\mathbf{A}\mathbf{x}^\top \geq \mathbf{b}^\top,$$

$$\sum_{j \neq 2^{k-i},\, j \in \mathcal{A}_i} z_{ij} + x_{2^{k-i}} \geq s_i, \quad \text{for} \quad 1 \leq i \leq k,$$

where

$$\begin{aligned} 0 \ &\leq z_{ij} \ \leq \ M w_{ij}, \\ x_j - M(1 - w_{ij}) \ &\leq z_{ij} \ \leq \ x_j. \end{aligned}$$

3.3.3 On the non-binary UEP code construction

The integer programming approach to construct binary UEP codes can be extended to construct non-binary UEP codes. Let the l-bit message $\mathbf{m}_i = (m_1^{(i)}, m_2^{(i)}, \ldots, m_l^{(i)})$ for $1 \leq i \leq k$ be represented with symbols in $\mathbb{GF}(q)$ where $q = 2^l$, then, using the integer programming approach introduced in Section 3.1.1, we construct non-binary UEP codes over $\mathbb{GF}(q)$ with a given separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ where $s_i = 2t_i + 1$ for $1 \leq i \leq k$, which satisfies

$$w_H(\mathbf{mG}) \geq s_i, \quad m_i \neq 0 \quad \text{for} \quad 1 \leq i \leq k. \tag{3.98}$$

where $\mathbf{m} = (m_1, m_2, \ldots, m_k)$ and $m_1, m_2, \ldots, m_k \in \mathbb{GF}(q)$.

Let the basis matrix $\mathbf{A}_b$ be a $k \times (q^k - 1)$ matrix consisting of all nonzero k-tuples over

$\mathbb{GF}(q)$ as columns in increasing order; for $\mathbb{GF}(q) = \{0, 1, \alpha, \alpha^2, \ldots, \alpha^{q-2}\}$,

$$\mathbf{A}_b = \begin{pmatrix} 00\ 0 \cdots \alpha^{q-2} \\ 00\ 0 \cdots \alpha^{q-2} \\ \vdots \vdots\ \vdots\ \ddots\ \ \vdots \\ 1\alpha\alpha^2 \cdots \alpha^{q-2} \end{pmatrix} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_k \end{pmatrix}, \tag{3.99}$$

and, for any vector $\mathbf{v} = (v_1, v_2, \ldots, v_w)$ over $\mathbb{GF}(q)$, let us define $[\mathbf{v}] = (\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_w)$ as

$$\hat{v}_j = \begin{cases} 0 & \text{if} \quad v_j = 0 \\ 1 & \text{if} \quad v_j \neq 0 \end{cases} \quad \text{for} \quad 1 \leq j \leq w. \tag{3.100}$$

Then, similar to the formulation in Section 3.1.1, we can have the following inequalities for integer programming,

$$\mathbf{A}_i \mathbf{x}^\top \geq \mathbf{b}_i^\top \quad \text{for} \quad 1 \leq i \leq k \tag{3.101}$$

where $\mathbf{A}_i$ is a $q^{i-1} \times (q^k - 1)$ matrix whose rows are

$$[\mathbf{a}_1] \quad \text{if} \quad i = 1, \tag{3.102a}$$

$$\left[\mathbf{a}_i + \sum_{u=1}^{i-1} \omega_u \mathbf{a}_u\right] \quad \text{if} \quad 2 \leq i \leq k \tag{3.102b}$$

where $w_u \in \mathbb{GF}(q)$ for all u,

$$\mathbf{x} = \left(x_1, x_2, \ldots, x_{q^k - 1}\right), \tag{3.103}$$

and

$$\mathbf{b}_i = (\underbrace{s_i, s_i, \ldots, s_i}_{q^{i-1}}) \quad \text{for} \quad 1 \leq i \leq k. \tag{3.104}$$

Based on (3.101), we can formulate integer programming problem for non-binary UEP code construction when a non-decreasing separation vector $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ is given:

UEP code construction over $\mathbb{GF}(q)$

Minimize

$$n = x_1 + x_2 + \cdots + x_{q^k-1}$$

subject to

$$\mathbf{A}\mathbf{x}^\top \geq \mathbf{b}^\top$$

where

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_k \end{pmatrix}, \quad \mathbf{b}^\top = \begin{pmatrix} \mathbf{b}_1^\top \\ \mathbf{b}_2^\top \\ \vdots \\ \mathbf{b}_k^\top \end{pmatrix}$$

Example 3.5 ($k = 2$ & $l = 2$)**.** *Let* $GF(2^2) = \{0, 1, \alpha, \alpha^2 = \alpha + 1\}$ *and* $\mathbf{s} = (3, 5)$*, then, from* (3.99)*,*

$$\mathbf{A}_b = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & \alpha & \alpha & \alpha & \alpha & \alpha^2 & \alpha^2 & \alpha^2 & \alpha^2 \\ 1 & \alpha & \alpha^2 & 0 & 1 & \alpha & \alpha^2 & 0 & 1 & \alpha & \alpha^2 & 0 & 1 & \alpha & \alpha^2 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \end{pmatrix},$$

and, from (3.102a),

$$\mathbf{A}_1 = \begin{pmatrix} 000111111111111 \end{pmatrix} = [\mathbf{a}_1]\,,$$

$$\mathbf{A}_2 = \begin{pmatrix} 111011101110111 \\ 111101111011110 \\ 111110111101011 \\ 111111010111101 \end{pmatrix} = \begin{pmatrix} [\mathbf{a}_2] \\ [\mathbf{a}_2 + \mathbf{a}_1] \\ [\mathbf{a}_2 + \alpha\mathbf{a}_1] \\ \left[\mathbf{a}_2 + \alpha^2\mathbf{a}_1\right] \end{pmatrix}.$$

Therefore, the UEP code construction is to find $\mathbf{x}$ *that minimizes the sum* $x_1+x_2+\cdots+x_{15}$ *satisfying*

$$x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} \geq 3$$

$$x_1 + x_2 + x_3 + x_5 + x_6 + x_7 + x_9 + x_{10} + x_{11} + x_{13} + x_{14} + x_{15} \geq 5$$

$$x_1 + x_2 + x_3 + x_4 + x_6 + x_7 + x_8 + x_9 + x_{11} + x_{12} + x_{13} + x_{14} \geq 5$$

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_7 + x_8 + x_9 + x_{10} + x_{12} + x_{14} + x_{15} \geq 5$$

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_8 + x_{10} + x_{11} + x_{12} + x_{13} + x_{15} \geq 5.$$

The optimal solution obtained from integer programming is

$$n = \sum_{j=1}^{15} x_j = 6$$

Table 3.9: Length comparisons between optimal UEP codes and punctured RS codes over $\mathbb{GF}(q)$ where $q = 2^k$

k	q	UEP codes	punctured RS codes
4	16	12	12
5	32	15	15
6	64	18	18
7	128	21	21
8	256	24	24

where

$$\begin{array}{ccccc} x_2 & x_6 & x_7 & x_{10} & x_{12} \\ \hline 2 & 1 & 1 & 1 & 1 \end{array}.$$

Consequently, the corresponding generator matrix is

$$\mathbf{G} = \begin{pmatrix} 0 & 0 & 1 & 1 & \alpha & \alpha^2 \\ \alpha & \alpha & \alpha & \alpha^2 & \alpha & 0 \end{pmatrix}.$$

Similar to Theorem 3.1, an integer programming bound on a length of the non-binary UEP code can be given by

$$n \geq \sum_{i=1}^{k} \left\lceil \frac{s_i}{q^{k-i}} \right\rceil . \tag{3.105}$$

In Table 3.9, lengths of the optimal binary UEP codes over $\mathbb{GF}(q)$ that satisfy the bound (3.105)[3] are provided for a given separation vector $\mathbf{s} = (3, 5, \ldots, 2k+1)$ with comparison to lengths of the t-error-correcting punctured RS codes where $t = \left\lfloor \frac{s_k}{2} \right\rfloor = k$. We observe that there is no difference between the lengths of UEP codes over $\mathbb{GF}(q)$ and the lengths the punctured RS codes.

[3] Derivation of (3.105) given in Appendix B

Chapter 4: Deterministic network coding for reliable packet transmissions on single-hop broadcast network

Network coding concept has been adopted to many applications in order to give benefits over traditional store-and-forward networks by allowing a network node to combine packets before transmitting. Consider a single-hop, single-source, and, multiple-receiver broadcast network depicted in Figure 1.1, traditional network protocol provides reliable packet transmissions over the broadcast network by either a forward error correction (FEC) or a retransmission (ARQ). In this chapter we provide a unified solution for reliable packet transmissions on the broadcast network by applying a deterministic approach of linear network coding into both forward error correction scheme and retransmission scheme.

4.1 Deterministic network coding

Practical approach of applying linear network coding to a packet retransmission scheme is to choose coding coefficients uniformly at random from a large enough finite field and inject coding coefficients in the packet header so that receivers can decode [71]. To eliminate the overhead caused by injecting encoding information in the packet header, we apply a deterministic approach to choose coding coefficients to encode packets; in other words, the coding coefficients are known to both a sender and receivers.

4.1.1 Reed-Solomon codes

Reed-Solomon (RS) code is an widely applied error correcting code in many applications. Conventional t-error-correcting $(n,\, k,\, d)$ RS code is described by the parameters the length of the code n, the dimension k, and the minimum distance d where $d = n - k + 1$ and $n - k = 2t$.

Code construction

We define a generator matrix of $(n,\ k,\ n-k+1)$ Reed-Solomon code by following the original formulation of Reed and Solomon in [75]. Let $\mathbf{G}_{RS}$ be the $k \times n$ generator matrix for the $(n,\ k,\ n-k+1)$ RS code, then

$$\mathbf{G}_{RS} \triangleq \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{k-1} & (\alpha^2)^{k-1} & \cdots & (\alpha^{n-1})^{k-1} \end{pmatrix} \tag{4.1}$$

where α is a primitive element in $\mathbb{GF}(2^m)$ whose order is $n = 2^m - 1$ and $k < n$. Let $\mathbf{p} = (p_0, p_1, \ldots, p_{k-1})$ consist of k symbols in $\mathbb{GF}(2^m)$ with its polynomial representation,

$$P(x) = \sum_{i=0}^{k-1} p_i x^i = p_0 + p_1 x + p_2 x^2 + \cdots + p_{k-1}x^{k-1},$$

then a codeword, $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ can be written as[1]

$$\begin{aligned} \mathbf{c} =& \mathbf{p}\ \mathbf{G}_{RS} \\ =& (P(1)\ P(\alpha)\ P(\alpha^2)\ \ldots\ P(\alpha^{n-1})). \end{aligned}$$

Let $C(x) = c_0 + c_1x + c_2x^2 + \cdots + c_{n-1}x^{n-1}$ be a polynomial representation of codeword

[1] The original formulation of code in [75] encoded with all elements of $\mathbb{GF}(2^m)$, i.e., $\mathbf{c} = (P(0)\ P(\alpha)\ P(\alpha^2)\ \ldots\ P(\alpha^{n-1})\ P(1))$. We note that the $(n,\ k,\ n-k+1)$ RS code is encoded with only non-zero elements for this dissertation, i.e., $\mathbf{c} = (P(1)\ P(\alpha)\ P(\alpha^2)\ \ldots\ P(\alpha^{n-1}))$.

c, then

$$
\begin{aligned}
C(\alpha^h) &= \sum_{j=0}^{n-1} c_j (\alpha^h)^j \\
&= \sum_{j=0}^{n-1} \left(\sum_{i=0}^{k-1} p_i (\alpha^j)^i \right) (\alpha^h)^j \\
&= \sum_{i=0}^{k-1} p_i \sum_{j=0}^{n-1} \alpha^{(i+h)j} \\
&= \sum_{i=0}^{k-1} p_i \frac{\alpha^{(i+h)n} + 1}{\alpha^{(i+h)} + 1}.
\end{aligned}
$$

Since

$$\frac{\alpha^{(i+h)n} + 1}{\alpha^{(i+h)} + 1} = 0 \quad \text{for} \quad i + h \neq n \quad \text{and} \quad 0 \leq i \leq k-1, \tag{4.2}$$

$C(x)$ have $\alpha,\ \alpha^2,\ \ldots\ \alpha^{n-k}$ as its roots; therefore, the code generated by $\mathbf{G}_{RS}$ is Reed-Solomon code with the minimum distance of $n - k + 1$ and its corresponding parity check matrix can be defined as

$$\mathbf{H}_{RS} \triangleq \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^2)^{n-1} \\ 1 & \alpha^3 & (\alpha^3)^2 & \cdots & (\alpha^3)^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{n-k} & (\alpha^{n-k})^2 & \cdots & (\alpha^{n-k})^{n-1} \end{pmatrix}. \tag{4.3}$$

Punctured RS codes

Reed-Solomon code is a *maximum distance separable* (MDS) code which satisfies the Singleton bound with equality. One property of MDS codes is that a punctured MDS code is also maximum distance separable. Generally, a punctured MDS code can be obtained by deleting columns of its generator matrix. Let $\mathbf{G}_{RS}^{(u)}$ denote a generator matrix of the punctured RS code by deleting the last u columns of $k \times n$ generator matrix defined in (4.1),

$$\mathbf{G}_{RS}^{(u)} \triangleq \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-u-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^{n-u-1})^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{k-1} & (\alpha^2)^{k-1} & \cdots & (\alpha^{n-u-1})^{k-1} \end{pmatrix}. \tag{4.4}$$

Since deleting the last u columns of $\mathbf{G}_{RS}$ can reduce minimum distance by at most u, the minimum distance of punctured code is greater than or equal to $n-k+1-u$. Furthermore, since the Singleton bound implies that the minimum distance of $(n-u,\ k,\ d-u)$ punctured RS code is less than or equal to $n-u-k+1$, the code generated by $\mathbf{G}_{RS}^{(u)}$ has the minimum distance of $n-u-k+1$.

4.1.2 Deterministic linear network codes

Using the generator matrix of the Reed-Solomon code in (4.1), we can construct a deterministic linear network code for a single-hop broadcast network (Figure 1.1). Suppose a single source in the broadcast network has k packets to broadcast. Each packet is represented as s symbols in a finite field, $\mathbb{GF}(2^m)$. A linear network coding allows the broadcasting source to transmit l linearly encoded packets instead of transmitting k uncoded packets. Let $\mathbf{p}_i = (p_{0,i}, p_{1,i}, \ldots, p_{(s-1),i})^\top$ be a column vector of i-th uncoded packet and $\mathbf{c}_j = (c_{0,j}, c_{1,j}, \ldots, c_{(s-1),j})^\top$ be a column vector of j-th encoded packet, then an encoded packet with a deterministic linear network code can be described by

$$\mathbf{c}_j = \sum_{i=0}^{k-1} \mathbf{p}_i \alpha^{ij} \quad \text{for} \quad 0 \le j \le l-1 \tag{4.5}$$

where α is a primitive element in $\mathbb{GF}(2^m)$ whose order is $n = 2^m - 1$ and $k, l < n$. We represent our deterministic linear network coding model as the following:

$$\mathbf{C}_{s\times l} = \mathbf{P}_{s\times k}\mathbf{G}_{k\times l} \tag{4.6}$$

where $\mathbf{C}_{s\times l}$ is a $s \times l$ matrix whose columns are linearly encoded packets, $\mathbf{P}_{s\times k}$ is a $s \times k$ matrix whose columns are uncoded packets, and $\mathbf{G}_{k\times l}$ is a $k \times l$ generator matrix of deterministic linear network code defined as

$$\mathbf{G}_{k\times l} \triangleq \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{l-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^{l-1})^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{k-1} & (\alpha^2)^{k-1} & \cdots & (\alpha^{l-1})^{k-1} \end{pmatrix}. \tag{4.7}$$

We note that the generator matrix of the deterministic linear network code, $\mathbf{G}_{k\times l}$, is obtained from puncturing last $n - l$ columns of generator matrix of $(n,\ k,\ n - k + 1)$ RS code (i.e., $\mathbf{G}_{k\times l} = \mathbf{G}_{RS}^{(n-l)}$). As a result, a code generated by $\mathbf{G}_{k\times l}$ has a minimum distance of $l - k + 1$.

4.2 Deterministic network coding for reliable packet transmission

4.2.1 Packet retransmissions using deterministic network coding

In a single-hop broadcast network, traditional packet retransmission scheme (ARQ) retransmits unsuccessfully delivered packets based on packet loss information collected from receivers within broadcasting range(See Figure 4.1). It has been shown that allowing a broadcasting source to combine retransmission packet reduces the number of retransmission packets over the broadcast network [70,71]. Specifically, the number of retransmissions depends on the maximum number of packet losses among all receivers when linear network coding is applied for packet retransmissions. For example, suppose that there are r number of receivers in a single-hop broadcast network, and let R_j denote j-th receiver where $1 \leq j \leq r$. If a receiver R_j has been experienced e_j packet losses, then the number of retransmission packets for a broadcasting source to retransmit is $\max_{\forall j} \{e_j\}$ for $1 \leq j \leq r$.

Encoding

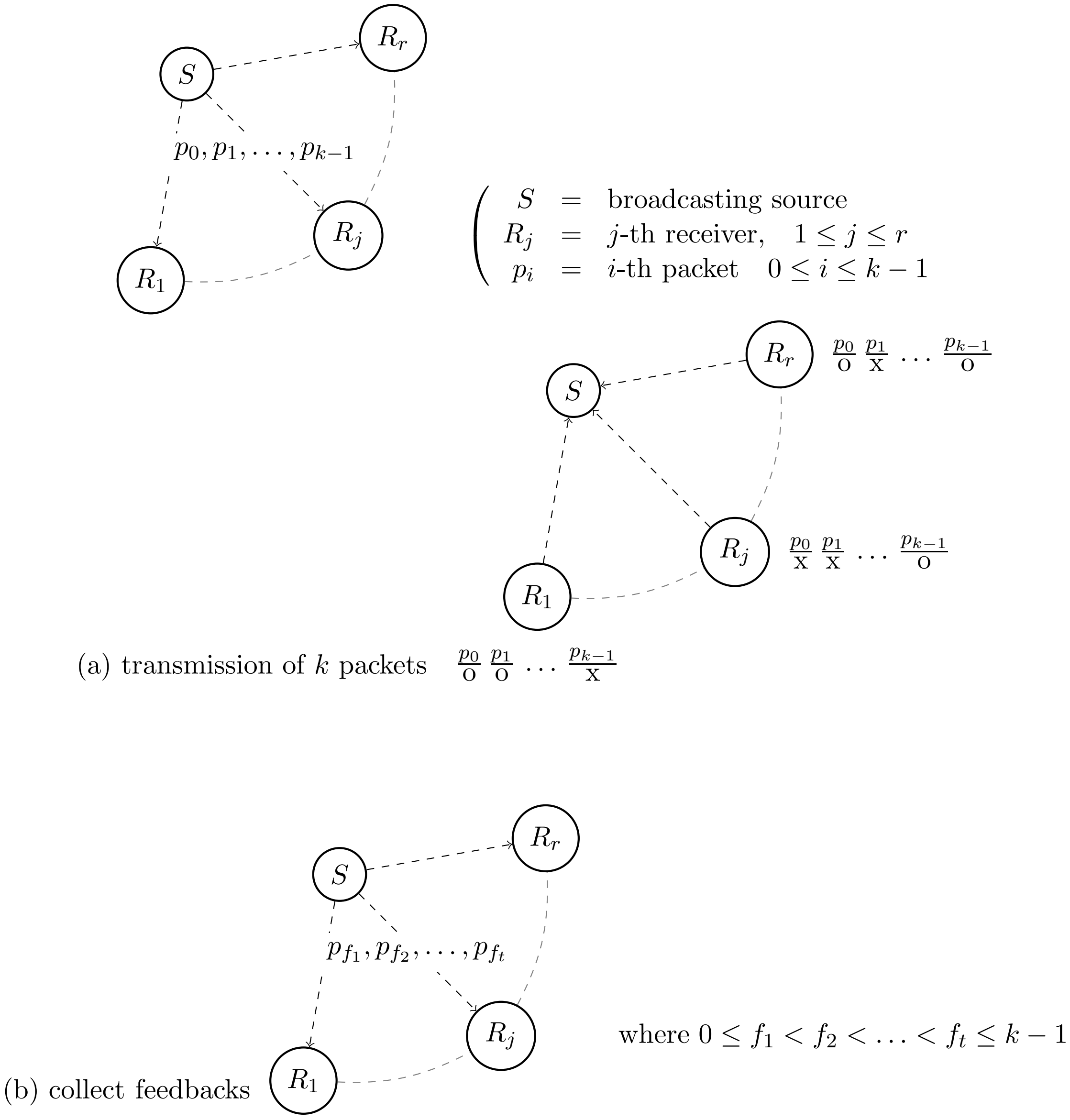
With a deterministic linear network coding, every retransmission packet is linearly encoded by pre-determined coding coefficients. Using the deterministic linear network code defined in Section 4.1.2, an encoded j-th retransmission packet can be written as

$$\mathbf{c}_j = \sum_{i=0}^{k-1} \mathbf{p}_i \alpha^{ij} \quad \text{for} \quad 0 \leq j \leq t-1. \tag{4.8}$$

where α is a primitive element in $\mathbb{GF}(2^m)$ whose order is $n = 2^m - 1$ and t is the number of packets needed to retransmit (See Figure 4.2). The packet retransmission scheme with the deterministic network coding can be represented in matrix form as

$$\mathbf{C}_{s\times t} = \mathbf{P}_{s\times k}\mathbf{G}_{k\times t} \tag{4.9}$$

where $\mathbf{C}_{s\times t}$ is $s \times t$ matrix whose columns are linearly combined packets for retransmission, $\mathbf{P}_{s\times k}$ is $s \times k$ matrix whose columns are uncoded packets, and $\mathbf{G}_{k\times t}$ is $k \times t$ generator matrix of deterministic linear network code for retransmission scheme. Note that the generator matrix, $\mathbf{G}_{k\times t}$ is obtained by puncturing the last $n - t$ columns of the generator matrix of


S
R_r
R_j
R_1
$p_0, p_1, \ldots, p_{k-1}$
S = broadcasting source
R_j = j-th receiver, $1 \le j \le r$
p_i = i-th packet $0 \le i \le k-1$
S
R_r
$\frac{p_0}{\text{O}}\ \frac{p_1}{\text{X}}\ \ldots\ \frac{p_{k-1}}{\text{O}}$
R_j
$\frac{p_0}{\text{X}}\ \frac{p_1}{\text{X}}\ \ldots\ \frac{p_{k-1}}{\text{O}}$
R_1
$\frac{p_0}{\text{O}}\ \frac{p_1}{\text{O}}\ \ldots\ \frac{p_{k-1}}{\text{X}}$
S
R_r
$p_{f_1}, p_{f_2}, \ldots, p_{f_t}$
R_j
R_1
where $0 \le f_1 < f_2 < \ldots < f_t \le k-1$

(a) transmission of k packets

(b) collect feedbacks

(c) retransmission of t packets ($t \le k$)

Figure 4.1: ARQ scheme

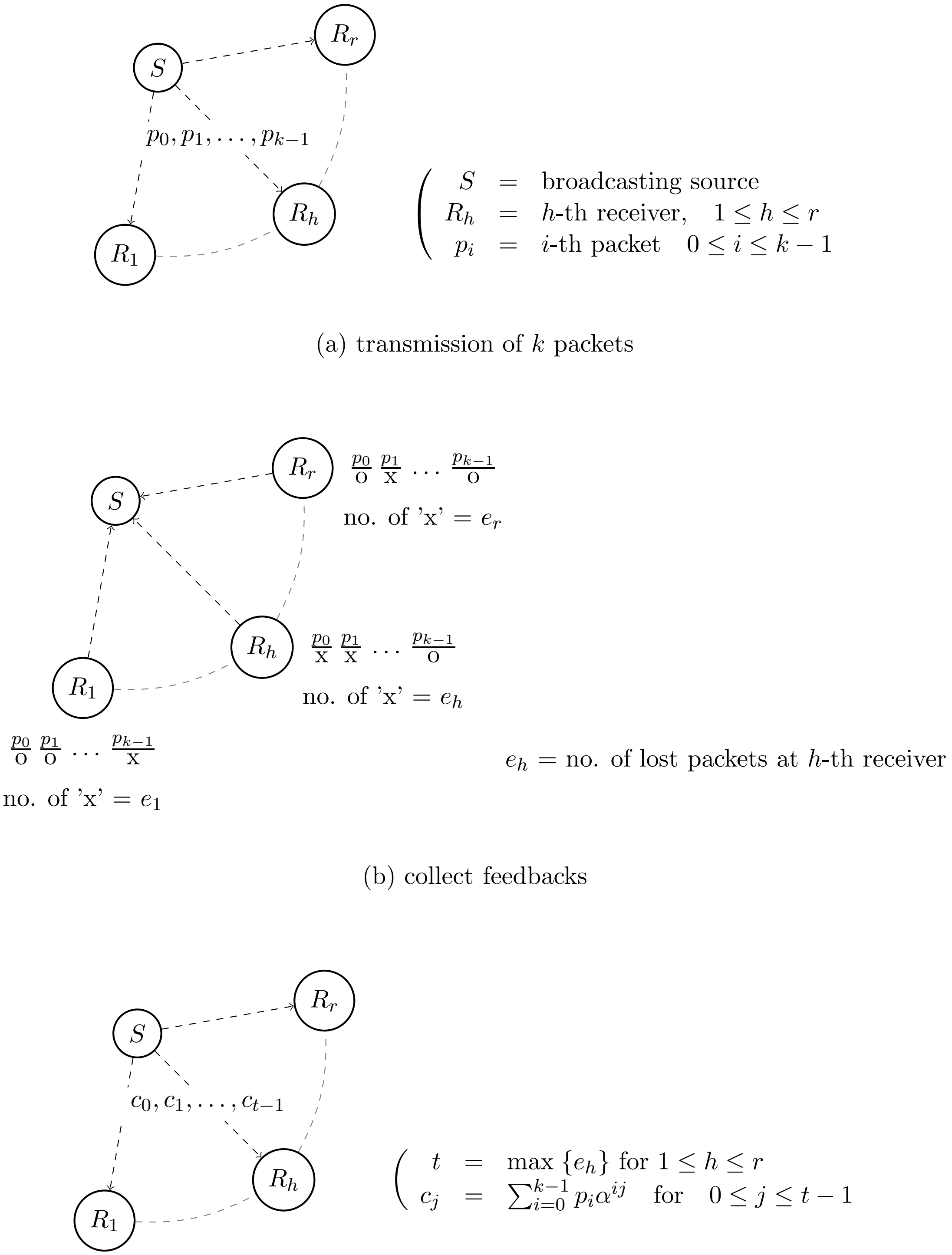
$(n,\, k,\, n-k+1)$ RS code defined in (4.1),

$$\mathbf{G}_{k\times t} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{t-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^{t-1})^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{k-1} & (\alpha^2)^{k-1} & \cdots & (\alpha^{t-1})^{k-1} \end{pmatrix}. \tag{4.10}$$

Decoding

Upon receiving encoded retransmission packets, since each receiver has knowledge of coding coefficients, retransmitted packets, and previously successfully received packets, lost packets can be recovered at each receiver by solving linear equations. Because every receiver has experienced a different number of lost packets and a different set of lost packets, each receiver has different demands on retransmission packets. Thus, the coding coefficients of linear transform have to be carefully chosen to ensure that packet retransmissions cover all demands of receivers.

Suppose that a receiver lost t packets and received t retransmission packets from a broadcasting source. Let $f_1, f_2, \ldots, f_t$ denote the indexes for t lost packets and $s_1, s_2, \ldots, s_{k-t}$ denote the indexes for the $k-t$ successfully received packets where $0 \le f_1 < f_2 < \ldots < f_t \le k-1$ and $0 \le s_1 < s_2 < \ldots < s_{k-t} \le k-1$, then the t retransmitted packets can be


S
R_r
R_h
R_1
$p_0, p_1, \ldots, p_{k-1}$
$\frac{p_0}{\text{o}}\ \frac{p_1}{\text{x}} \ldots \frac{p_{k-1}}{\text{o}}$
no. of 'x' $= e_r$
$\frac{p_0}{\text{x}}\ \frac{p_1}{\text{x}} \ldots \frac{p_{k-1}}{\text{o}}$
no. of 'x' $= e_h$
$\frac{p_0}{\text{o}}\ \frac{p_1}{\text{o}} \ldots \frac{p_{k-1}}{\text{x}}$
no. of 'x' $= e_1$
$c_0, c_1, \ldots, c_{t-1}$

$$\left(\begin{array}{rcl} S & = & \text{broadcasting source} \\ R_h & = & h\text{-th receiver}, \quad 1 \le h \le r \\ p_i & = & i\text{-th packet} \quad 0 \le i \le k-1 \end{array}\right.$$

(a) transmission of k packets

e_h = no. of lost packets at h-th receiver

(b) collect feedbacks

$$\left(\begin{array}{rcl} t & = & \max\{e_h\} \text{ for } 1 \le h \le r \\ c_j & = & \sum_{i=0}^{k-1} p_i \alpha^{ij} \quad \text{for} \quad 0 \le j \le t-1 \end{array}\right.$$

(c) packet retransmission using deterministic network code

Figure 4.2: Packet retransmission scheme with deterministic network coding

written as

$$
\begin{aligned}
(\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{t-1}) &= (\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{k-1})\,\mathbf{G}_{k\times t} \\
&= (\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{k-1}) \begin{pmatrix} g_{0,0} & g_{0,1} & \cdots & g_{0,t-1} \\ g_{1,0} & g_{1,1} & \cdots & g_{1,t-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,t-1} \end{pmatrix}, \text{ where } \quad g_{i,j} = \alpha^{ij} \\
&= \left(\mathbf{p}_{s_1}, \mathbf{p}_{s_2}, \ldots, \mathbf{p}_{s_{k-t}}\right) \begin{pmatrix} g_{s_1,0} & g_{s_1,1} & \cdots & g_{s_1,t-1} \\ g_{s_2,0} & g_{s_2,1} & \cdots & g_{s_2,t-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{s_{k-t},0} & g_{s_{k-t},1} & \cdots & g_{s_{k-t},t-1} \end{pmatrix} \\
&\quad + (\mathbf{p}_{f_1}, \mathbf{p}_{f_2}, \ldots, \mathbf{p}_{f_t}) \begin{pmatrix} g_{f_1,0} & g_{f_1,1} & \cdots & g_{f_1,t-1} \\ g_{f_2,0} & g_{f_2,1} & \cdots & g_{f_2,t-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{f_t,0} & g_{f_t,1} & \cdots & g_{f_t,t-1} \end{pmatrix}.
\end{aligned}
\tag{4.11}
$$

Let the two separated coefficient matrices in (4.11) be $\mathbf{W}_s$ and $\mathbf{W}_f$ respectively, then

$$
(\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{t-1}) = \left(\mathbf{p}_{s_1}, \mathbf{p}_{s_2}, \ldots, \mathbf{p}_{s_{k-t}}\right)\mathbf{W}_s + (\mathbf{p}_{f_1}, \mathbf{p}_{f_2}, \ldots, \mathbf{p}_{f_t})\,\mathbf{W}_f. \tag{4.12}
$$

Because the receiver knows all coding coefficients (i.e., $\mathbf{W}_s$ and $\mathbf{W}_f$), t retransmitted packets $\{\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{t-1}\}$, and $k-t$ successfully delivered packets $\{\mathbf{p}_{s_1}, \mathbf{p}_{s_2}, \ldots, \mathbf{p}_{s_{k-t}}\}$, the t lost packets $\{\mathbf{p}_{f_1}, \mathbf{p}_{f_2}, \ldots, \mathbf{p}_{f_t}\}$ can be recovered by solving linear equations if the square matrix, $\mathbf{W}_f$ is invertible.

Let $\mathbf{V}_t(a_0, a_1, \ldots, a_{t-1})$ denote a $t\times t$ Vandermonde matrix with elements of $a_0, a_1, \ldots, a_{t-1}$, then

$$\mathbf{W}_f{}^\top = \mathbf{V}_t(\alpha^{f_1}, \alpha^{f_2}, \ldots, \alpha^{f_t})$$

$$= \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha^{f_1} & \alpha^{f_2} & \cdots & \alpha^{f_t} \\ \alpha^{f_1}{}^2 & \alpha^{f_2}{}^2 & \cdots & \alpha^{f_t}{}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{f_1}{}^{t-1} & \alpha^{f_2}{}^{t-1} & \cdots & \alpha^{f_t}{}^{t-1} \end{pmatrix}. \tag{4.13}$$

Since $\det\left(\mathbf{A}^\top\right) = \det\left(\mathbf{A}\right)$ where A is a square matrix,

$$\det\left(\mathbf{W}_f\right) = \det\Big(\mathbf{W}_f{}^\top\Big) \tag{4.14}$$

$$= \det\Big(\mathbf{V}_t(\alpha^{f_1}, \alpha^{f_2}, \ldots, \alpha^{f_t})\Big)$$

$$= \prod_{f_1 \le i < j \le f_t} (\alpha^j - a^i). \tag{4.15}$$

Since the elements, $\alpha^{f_1}, \alpha^{f_2}, \ldots, \alpha^{f_t}$ are non-zero and distinct over $\mathbb{GF}(2^m)$, the determinant of $\mathbf{W}_f$ is non-zero. Therefore, the matrix $\mathbf{W}_f$ is non-singular as required.

Now consider a case that a receiver requested e retransmission packets where e is less than t and the receiver successfully received e consecutively encoded retransmission packets

$\{\mathbf{c}_h, \mathbf{c}_{h+1}, \ldots, \mathbf{c}_{h+e-1}\}$ where $0 \leq h \leq t - e$. For simplicity, we assume $h = 0$, then

$$(\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{e-1}) = \left(\mathbf{p}_{s_1}, \mathbf{p}_{s_2}, \ldots, \mathbf{p}_{s_{k-e}}\right) \begin{pmatrix} g_{s_1,0} & g_{s_1,1} & \cdots & g_{s_1,e-1} \\ g_{s_2,0} & g_{s_2,1} & \cdots & g_{s_2,e-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{s_{k-e},0} & g_{s_{k-e},1} & \cdots & g_{s_{e-t},t-1} \end{pmatrix}$$

$$+ (\mathbf{p}_{f_1}, \mathbf{p}_{f_2}, \ldots, \mathbf{p}_{f_e}) \begin{pmatrix} g_{f_1,0} & g_{f_1,1} & \cdots & g_{f_1,e-1} \\ g_{f_2,0} & g_{f_2,1} & \cdots & g_{f_2,e-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{f_e,0} & g_{f_e,1} & \cdots & g_{f_e,e-1} \end{pmatrix}. \tag{4.16}$$

Similar to (4.12), e retransmitted packets can be written as

$$(\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{e-1}) = \left(\mathbf{p}_{s_1}, \mathbf{p}_{s_2}, \ldots, \mathbf{p}_{s_{k-e}}\right) \mathbf{W}^*{}_s + (\mathbf{p}_{f_1}, \mathbf{p}_{f_2}, \ldots, \mathbf{p}_{f_e}) \mathbf{W}^*{}_f \tag{4.17}$$

where $\mathbf{W}^*{}_s$ and $\mathbf{W}^*{}_f$ are corresponding coefficient matrices. Because the determinant of $\mathbf{W}^*{}_f$ is equal to the determinant of $\mathbf{V}_e(\alpha^{f_1}, \alpha^{f_2}, \ldots, \alpha^{f_e})$, as long as e consecutively encoded packet are successfully delivered at the receiver, $\mathbf{W}^*{}_f$ is invertible. Therefore, the encoding based on Vandermonde matrix does guarantee to decode packets if a receiver (who requested e packet retransmissions) successfully receives e consecutively encoded retransmission packets where e is less than or equal to t.

Example

Let $k = 4$, $s = 4$. Suppose

$$\mathbf{P}_{4\times 4} = \begin{pmatrix} 1 & \alpha & 1 & 1 \\ 1 & 1 & \alpha & \alpha^2 \\ \alpha^4 & \alpha & \alpha & \alpha^3 \\ \alpha^5 & 1 & \alpha^6 & \alpha^4 \end{pmatrix}$$

where α is a primitive element in $\mathbb{GF}(2^3)$ and is a root of the primitive polynomial x^3+x+1.

Assume that a broadcasting source collects feedbacks from receivers and there are at most 2 unsuccessfully delivered packets among all receivers within broadcasting range. Then the source needs to retransmit 2 linearly combined packets using (4.9):

$$\mathbf{C}_{4\times 2} = \mathbf{P}_{4\times 4}\mathbf{G}_{4\times 2}$$

$$= \begin{pmatrix} 1 & \alpha & 1 & 1 \\ 1 & 1 & \alpha & \alpha^2 \\ \alpha^4 & \alpha & \alpha & \alpha^3 \\ \alpha^5 & 1 & \alpha^6 & \alpha^4 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & \alpha \\ 1 & \alpha^2 \\ 1 & \alpha^3 \end{pmatrix} = \begin{pmatrix} \alpha^3 & \alpha \\ \alpha^4 & \alpha^5 \\ \alpha^6 & \alpha^2 \\ \alpha^6 & \alpha^4 \end{pmatrix}$$

At the receiving end, each receiver wants to recover its lost packets from the linearly combined packets. For example, a receiver R has $\mathbf{p}_0 = (1, 1, \alpha^4, \alpha^5)^\top$ and $\mathbf{p}_2 = (1, \alpha, \alpha, \alpha^6)^\top$

and wants to recover $\mathbf{p}_1$ and $\mathbf{p}_3$ from retransmitted packets. The linearly encoded retransmission packets can be represented by

$$
\begin{aligned}
(\mathbf{c}_0, \mathbf{c}_1) &= (\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)\, \mathbf{G}_{4\times 2} \\
&= (\mathbf{p}_0, \mathbf{p}_2) \begin{pmatrix} g_{0,0} & g_{0,1} \\ g_{2,0} & g_{2,1} \end{pmatrix} + (\mathbf{p}_1, \mathbf{p}_3) \begin{pmatrix} g_{1,0} & g_{1,1} \\ g_{3,0} & g_{3,1} \end{pmatrix} \\
&= \begin{pmatrix} 1 & 1 \\ 1 & \alpha \\ \alpha^4 & \alpha \\ \alpha^5 & \alpha^6 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & \alpha^2 \end{pmatrix} + (\mathbf{p_1}, \mathbf{p_3}) \begin{pmatrix} 1 & \alpha \\ 1 & \alpha^3 \end{pmatrix}.
\end{aligned}
$$

Since the receiver R knows $\mathbf{p}_0$, $\mathbf{p}_2$, and $\mathbf{G}_{4\times 2}$,

$$
\begin{aligned}
(\mathbf{p}_1, \mathbf{p}_3) &= \left\{ (\mathbf{c}_0, \mathbf{c}_1) + (\mathbf{p}_0, \mathbf{p}_2) \begin{pmatrix} 1 & 1 \\ 1 & \alpha^2 \end{pmatrix} \right\} \begin{pmatrix} 1 & \alpha \\ 1 & \alpha^3 \end{pmatrix}^{-1} \\
&= \left\{ \begin{pmatrix} \alpha^3 & \alpha \\ \alpha^4 & \alpha^5 \\ \alpha^6 & \alpha^2 \\ \alpha^6 & \alpha^4 \end{pmatrix} + \begin{pmatrix} 1 & 1 \\ 1 & \alpha \\ \alpha^4 & \alpha \\ \alpha^5 & \alpha^6 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & \alpha^2 \end{pmatrix} \right\} \begin{pmatrix} 1 & \alpha \\ 1 & \alpha^3 \end{pmatrix}^{-1} \\
&= \begin{pmatrix} \alpha & 1 \\ 1 & \alpha^2 \\ \alpha & \alpha^3 \\ 1 & \alpha^4 \end{pmatrix}
\end{aligned}
$$

4.2.2 Forward error corrections using deterministic network coding

Because a deterministic linear network code constructed by puncturing the generator matrix of RS code (4.7), the deterministic linear network coding for packet transmission can easily provide forward error corrections.

Encoding

In a single-hop broadcast network, a source broadcasts $k + 2t$ linearly encoded packets to provide t packet-error-correction. Using the deterministic linear network code defined in Section 4.1.2, the packet encoding for forward error correction can be described by

$$\mathbf{c}_j = \sum_{i=0}^{k-1} \mathbf{p}_i \alpha^{ij} \quad \text{for} \quad 0 \leq j \leq n' - 1, \tag{4.18}$$

where α is a primitive element in $\mathbb{GF}(2^m)$ and $n' = k + 2t$ (See Figure 4.3). A deterministic linear network coding model for forward error correction can also be expressed in matrix form,

$$\mathbf{C}_{s\times n'} = \mathbf{P}_{s\times k}\mathbf{G}_{k\times n'} \tag{4.19}$$

where $\mathbf{P}_{s\times k}$ is a $s \times k$ matrix whose columns are uncoded packets, $\mathbf{C}_{s\times n'}$ is a $s \times n'$ matrix whose columns are encoded packets, and $\mathbf{G}_{k\times n'}$ is a $k \times n'$ generator matrix for deterministic linear network code. Similar to the packet retransmission scheme, the generator matrix, $\mathbf{G}_{k\times n'}$ for forward error correction scheme is obtained from puncturing the last $n - (k + 2t)$

columns of the generator matrix in (4.1),

$$\mathbf{G}_{k\times n'} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n'-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^{n'-1})^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{k-1} & (\alpha^2)^{k-1} & \cdots & (\alpha^{n'-1})^{k-1} \end{pmatrix}. \tag{4.20}$$

Theorem 4.1. *The deterministic linear network code generated by $\mathbf{G}_{k\times n'}$ guarantees to correct up to t packet errors where $n' = k + 2t$.*

Proof. Each encoded packet consists of s symbols over $\mathbb{GF}(2^m)$ and any corrupted symbol in a packet would result in a packet error. Thus, t-packet-error affects t columns of $\mathbf{C}_{s\times n'}$ and it corrupts at most t symbols in each row of $\mathbf{C}_{s\times n'}$ (See Figure 4.3). Since rows of $\mathbf{C}_{s\times n'}$ are encoded with $(k + 2t,\ k,\ 2t + 1)$ punctured RS code, they can be corrected up to t symbols of errors. Therefore the deterministic linear network code generated by $\mathbf{G}_{k\times n'}$ guarantees to correct up to t packet errors where $n' = k + 2t$. □

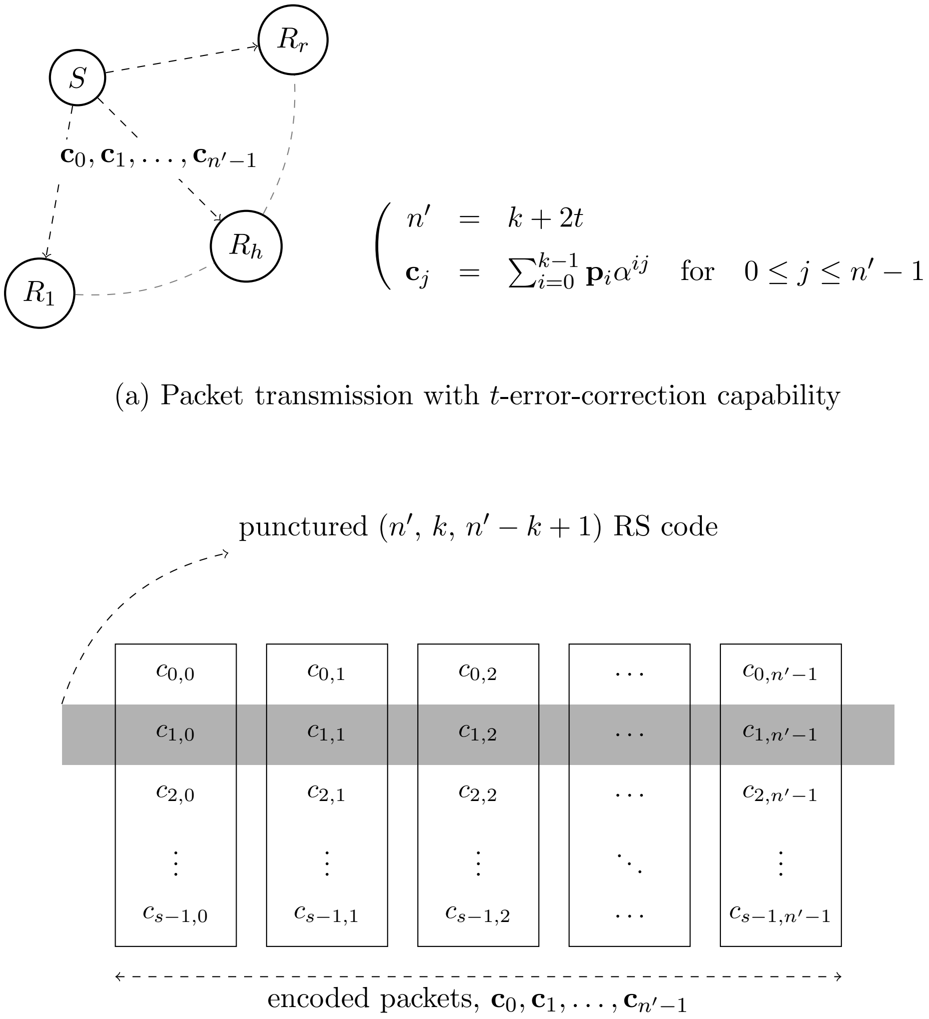
Decoding

Let $\mathbf{R}_{s\times n'}$ denote a $s \times n'$ matrix whose columns are received packets, $\mathbf{C}_{s\times n'}$ denote a $s \times n'$ matrix whose columns are transmitted packets and $\mathbf{E}_{s\times n'}$ denote a $s \times n'$ matrix of errors introduced by broadcast channel. Then,

$$\mathbf{R}_{s\times n'} = \mathbf{C}_{s\times n'} + \mathbf{E}_{s\times n'}$$

and

$$\mathbf{r}_h^* = \mathbf{c}_h^* + \mathbf{e}_h^* \quad \text{for} \quad 0 \le h \le s - 1$$


S
R_r
R_h
R_1
$\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{n'-1}$
$\left(\begin{array}{lcl} n' & = & k+2t \\ \mathbf{c}_j & = & \sum_{i=0}^{k-1} \mathbf{p}_i \alpha^{ij} \quad \text{for} \quad 0 \le j \le n'-1 \end{array} \right.$

(a) Packet transmission with t-error-correction capability

punctured $(n',\, k,\, n'-k+1)$ RS code
encoded packets, $\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{n'-1}$

$c_{0,0}$	$c_{0,1}$	$c_{0,2}$	$\cdots$	$c_{0,n'-1}$
$c_{1,0}$	$c_{1,1}$	$c_{1,2}$	$\cdots$	$c_{1,n'-1}$
$c_{2,0}$	$c_{2,1}$	$c_{2,2}$	$\cdots$	$c_{2,n'-1}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$c_{s-1,0}$	$c_{s-1,1}$	$c_{s-1,2}$	$\cdots$	$c_{s-1,n'-1}$

(b) A block of encoded packets

Figure 4.3: Forward error correction scheme with deterministic network coding

where $\mathbf{r}_h^* = (r_{h,0}, r_{h,1}, \ldots, r_{h,(n'-1)})$ and $\mathbf{e}_h^* = (e_{h,0}, e_{h,1}, \ldots, e_{h,(n'-1)})$ denote a row vector of the received words and error vectors respectively.

Upon receiving $\mathbf{R}_{s\times n'}$, row-by-row decoding is performed to correct packet errors. We can use standard error and erasure decoder for a $(n,\ k,\ n-k+1)$ RS code to decode punctured $(n',\ k,\ n'-k+1)$ RS code by treating the deleted columns as erasure positions. Let the received polynomial be $R_h(x) = r_{h,0} + r_{h,1}x + \ldots + r_{h,(n'-1)}x^{n'-1}$, then the standard error and erasure decoding procedures for the punctured RS code consist of the following: [52]

1. Compute erasure location polynomial, $\Gamma_h(x)$ for $0 \leq h \leq s-1$,

$$\Gamma_h(x) = \Gamma(x) = \prod_{i=n'}^{n-1} \left(1 - \alpha^i x\right). \tag{4.21}$$

2. Compute syndrome polynomial, $S_h(x)$,

$$S_h(x) = \sum_{j=1}^{n-k} s_{h,j} x^{(j-1)} \tag{4.22}$$

   where $s_{h,j} = r_h(\alpha^j) = \sum_{i=0}^{n'-1} r_{h,i}(\alpha^j)^i$ and $0 \leq h \leq s-1$.

3. Compute modified syndrome polynomial, $\Theta_h(x)$,

$$\Theta_h(x) \equiv \Gamma_h(x) S_h(x) \mod x^{n-k}. \tag{4.23}$$

4. Key equation:

$$\Lambda_h(x)\Theta_h(x) \equiv \Omega_h(x) \mod x^{n-k} \tag{4.24}$$

   where $\Lambda_h(x)$ is error location polynomial and $\Omega_h(x)$ is error value polynomial.

5. Apply Euclid's algorithm to solve the key equation and determine $\Lambda_h(x)$ and $\Omega_h(x)$.

6. Apply Forney's algorithm to find error location and value

$$e_{hj} = \begin{cases} \frac{\Omega_h(\alpha^{-j})}{\Lambda_h{}'(\alpha^{-j})\Gamma_h(\alpha^{-j})} & : \text{ if } \quad \Lambda_h(\alpha^{-j}) = 0 \\ 0 & : \text{ otherwise} \end{cases} \quad \text{for} \quad 0 \leq j \leq n' - 1 \tag{4.25}$$

7. Find the transmitted codeword, $\mathbf{c}_h^*$.

Suppose that the broadcast channel introduced less than t packet errors. Through row-by-row decoding of the punctured RS code, each receiver can recover the encoded packets, $\mathbf{C}_{s\times n'}$. Now consider a $s \times k$ sub-matrix of $\mathbf{C}_{s\times n'}$ by deleting the last $n' - k$ columns and define as $\mathbf{C'}_{s\times k}$, then

$$\mathbf{C'}_{s\times k} = \mathbf{P}_{s\times k}\mathbf{V}_k(\alpha^0, \alpha^1, \ldots, \alpha^{k-1})$$

where $\mathbf{V}_k(\alpha^0, \alpha^1, \ldots, \alpha^{k-1})$ is the $k \times k$ Vandermonde matrix. Since the elements of $\mathbf{V}_k(\alpha^0, \alpha^1, \ldots, \alpha^{k-1})$ are non-zero and distinct over $\mathbb{GF}(2^m)$, $\mathbf{V}_k(\alpha^0, \alpha^1, \ldots, \alpha^{k-1})$ has non-zero determinant. Therefore each receiver can recover the original packet information by solving linear equations: $\mathbf{P}_{s\times k} = \mathbf{C'}_{s\times k}\mathbf{V}_k(\alpha^0, \alpha^1, \ldots, \alpha^{k-1})^{-1}$.

Example

Let $k = 3$ and $s = 4$. Suppose

$$\mathbf{P}_{4\times 3} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 \\ \alpha^4 & \alpha & \alpha^3 \\ \alpha^5 & \alpha^6 & \alpha^4 \end{pmatrix}$$

where α is a primitive element in $\mathbb{GF}(2^3)$ and is a root of the primitive polynomial x^3+x+1.

From (4.1), a $(7,3,5)$ RS code can be generated by

$$\mathbf{G}_{RS} = \mathbf{G}_{3\times 7} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \alpha & \alpha^3 & \alpha^5 \end{pmatrix}.$$

Now assume that the broadcast channel needs single error correction capability. A $(5,3,3)$ punctured single-error-correcting RS code can be generated by puncturing the last two columns of $\mathbf{G}_{3\times 7}$,

$$\mathbf{G}_{3\times 5} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \alpha \end{pmatrix}.$$

Then, a broadcasting source obtains encoded packets:

$$\begin{aligned} \mathbf{C}_{4\times 5} &= \mathbf{P}_{4\times 3}\mathbf{G}_{3\times 5} \\ &= \begin{pmatrix} 1 & \alpha^5 & \alpha^3 & \alpha^5 & \alpha^6 \\ \alpha^5 & \alpha^3 & \alpha^5 & \alpha^6 & \alpha^6 \\ \alpha^5 & \alpha^6 & \alpha^2 & \alpha^2 & \alpha^5 \\ \alpha^2 & \alpha^3 & \alpha^5 & 0 & \alpha^3 \end{pmatrix}. \end{aligned}$$

For a receiver in a broadcasting range, suppose that received encoded packets are

$$\mathbf{R}_{4\times 5} = \begin{pmatrix} 1 & \alpha^5 & \alpha^3 & 1 & \alpha^6 \\ \alpha^5 & \alpha^3 & \alpha^5 & \alpha & \alpha^6 \\ \alpha^5 & \alpha^6 & \alpha^2 & \alpha & \alpha^5 \\ \alpha^2 & \alpha^3 & \alpha^5 & 1 & \alpha^3 \end{pmatrix},$$

and each row of received words are

$$r_0(x) = 1 + \alpha^5 x + \alpha^3 x^2 + x^3 + \alpha^6 x^4,$$

$$r_1(x) = \alpha^5 + \alpha^3 x + \alpha^5 x^2 + \alpha x^3 + \alpha^6 x^4,$$

$$r_2(x) = \alpha^5 + \alpha^6 x + \alpha^2 x^2 + \alpha x^3 + \alpha^5 x^4,$$

$$r_3(x) = \alpha^2 + \alpha^3 x + \alpha^5 x^2 + x^3 + \alpha^3 x^4.$$

From (4.21), the erasure location polynomial is

$$\begin{aligned}\Gamma_h(x) &= (1 + \alpha^5 x)(1 + \alpha^6 x), \\ &= 1 + \alpha x + \alpha^4 x^2 \quad \text{for} \quad h = 0, 1, 2, 3.\end{aligned}$$

Then a receiver can compute the syndrome polynomials using (4.22),

$$S_0(x) = \alpha^3 + \alpha^6 x + \alpha^6 x^2 + \alpha^4 x^3,$$

$$S_1(x) = \alpha^6 + \alpha^2 x + x^2 + \alpha^2 x^3,$$

$$S_2(x) = \alpha + \alpha^2 x + x^2 + \alpha^3 x^3,$$

$$S_3(x) = 1 + \alpha^2 x + x^2,$$

and the modified syndrome polynomials using (4.23),

$$\Theta_0(x) = \alpha^3 + \alpha^3 x + \alpha^6 x^2 + \alpha^2 x^3,$$

$$\Theta_1(x) = \alpha^6 + \alpha^6 x + x^2 + \alpha^3 x^3,$$

$$\Theta_2(x) = \alpha + \alpha^6 x^2 + \alpha^2 x^3,$$

$$\Theta_3(x) = 1 + \alpha^4 x + \alpha^2 x^2 + \alpha^5 x^3.$$

Using the Euclid's algorithm, the error location polynomials and the error value polynomial can be obtained as

$$\Lambda_0(x) = \alpha^2(1 + \alpha^3 x),$$

$$\Lambda_1(x) = \alpha(1 + \alpha^3 x),$$

$$\Lambda_2(x) = \alpha^2(1 + \alpha^3 x),$$

$$\Lambda_3(x) = \alpha^6(1 + \alpha^3 x),$$

and

$$\Omega_0(x) = \alpha^5 + \alpha^6 x,$$

$$\Omega_1(x) = 1 + \alpha x + x^2,$$

$$\Omega_2(x) = \alpha^3 + \alpha^6 x + \alpha x^2,$$

$$\Omega_3(x) = \alpha^6 + \alpha^5 x + \alpha^5 x^2.$$

Then using (4.25) one can locate error position and its value, hence the error vectors are

$$\mathbf{E}_{4\times 5} = \begin{pmatrix} 0 & 0 & 0 & \alpha^4 & 0 \\ 0 & 0 & 0 & \alpha^5 & 0 \\ 0 & 0 & 0 & \alpha^4 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Finally, a receiver recovers the encoded packets,

$$\mathbf{C}_{4\times 5} = \mathbf{R}_{4\times 5} + \mathbf{E}_{4\times 5}$$

$$= \begin{pmatrix} 1 & \alpha^5 & \alpha^3 & \alpha^5 & \alpha^6 \\ \alpha^5 & \alpha^3 & \alpha^5 & \alpha^6 & \alpha^6 \\ \alpha^5 & \alpha^6 & \alpha^2 & \alpha^2 & \alpha^5 \\ \alpha^2 & \alpha^3 & \alpha^5 & 0 & \alpha^3 \end{pmatrix}.$$

Since a receiver recovers the encoded packets, one can now obtain the original packets by multiplying inverse of the Vandermonde matrix;

$$\mathbf{P}_{4\times 3} = \mathbf{C}_{4\times 3} \cdot \mathbf{V^{-1}}_{3\times 3}$$

$$= \begin{pmatrix} 1 & \alpha^5 & \alpha^3 \\ \alpha^5 & \alpha^3 & \alpha^5 \\ \alpha^5 & \alpha^6 & \alpha^2 \\ \alpha^2 & \alpha^3 & \alpha^5 \end{pmatrix} \begin{pmatrix} \alpha & \alpha^2 & \alpha^5 \\ \alpha^2 & \alpha^6 & 1 \\ \alpha^5 & 1 & \alpha^4 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 \\ \alpha^4 & \alpha & \alpha^3 \\ \alpha^5 & \alpha^6 & \alpha^4 \end{pmatrix}.$$

4.3 Numerical analysis and simulations

4.3.1 Analysis

Packet Losses

We assume that packet loss characteristics over a broadcast channel are independent and identical with a packet loss probability of p at each receiver, hence a transmitted packet is successfully received at a receiver with probability of $1-p$. Let $\{X_i\}$ be a set of random variables which takes a value of 1 if i-th packet is not received successfully, or a value of 0 if i-th packet is received successfully for $1 \le i \le k$,

$$\Pr(X_i = 1) = p \quad \text{and} \quad \Pr(X_i = 0) = 1 - p. \tag{4.26}$$

Let $\{Y_j\}$ be a set of random variables denotes the number of lost packets for j-th receiver R_j where $1 \le j \le r$, then Y_j is the sum of a set of $\{X_i\}$ which follows a binomial distribution with parameters of k and p,

$$Y_j = \sum_{i=1}^{k} X_i \tag{4.27}$$

and

$$\Pr(Y_j = y) = \binom{k}{y} p^y (1-p)^{k-y}. \tag{4.28}$$

We define a random variable Z to denote that the maximum number of unsuccessfully delivered packets among all receivers,

$$Z = \max \{Y_1, Y_2, \ldots, Y_r\}, \tag{4.29}$$

then we have

$$\Pr(Z \leq z) = \Pr(\max\{Y_1, Y_2, \ldots, Y_r\} \leq z) \tag{4.30}$$

$$= \prod_{i=1}^{r} \Pr(Y_i \leq z) \tag{4.31}$$

$$= \{\Pr(Y_i \leq z)\}^r. \tag{4.32}$$

Therefore,

$$\Pr(Z = z) = \Pr(Z \leq z) - \Pr(Z \leq z-1) \tag{4.33}$$

$$= \prod_{i=1}^{r} \Pr(Y_i \leq z) - \prod_{i=1}^{r} \Pr(Y_i \leq z-1) \tag{4.34}$$

$$= \{\Pr(Y_i \leq z)\}^r - \{\Pr(Y_i \leq z-1)\}^r \tag{4.35}$$

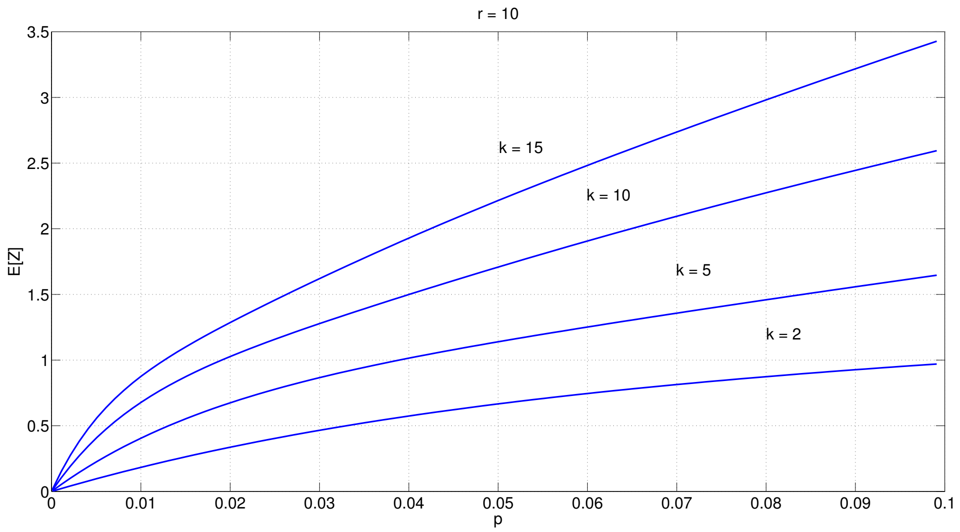


Figure 4.4: E[Z] versus packet error probability, p.

and

$$E[Z] = \sum_{i=0}^{k} i \Pr(Z = i) \tag{4.36}$$

$$= \Pr(Z = 1) + 2\Pr(Z = 2) + \cdots + k\Pr(Z = k). \tag{4.37}$$

From (4.33) and (4.36), on average, there are at most $E[Z]$ numbers of unsuccessfully delivered packets out of k transmitted packets among all receivers where

$$E[Z] = k - \sum_{i=0}^{k-1} \Pr(Y \le i)^r. \tag{4.38}$$

We numerically experiment with the packet losses. The results are depicted in Figure 4.4, Figure 4.5, Figure 4.6 and Figure 4.7. To evaluate the packet loss analysis, we simulate actual packet losses among receivers in a single-hop broadcast network and compare the simulation results with the analysis. The comparisons are shown in Figure 4.8.

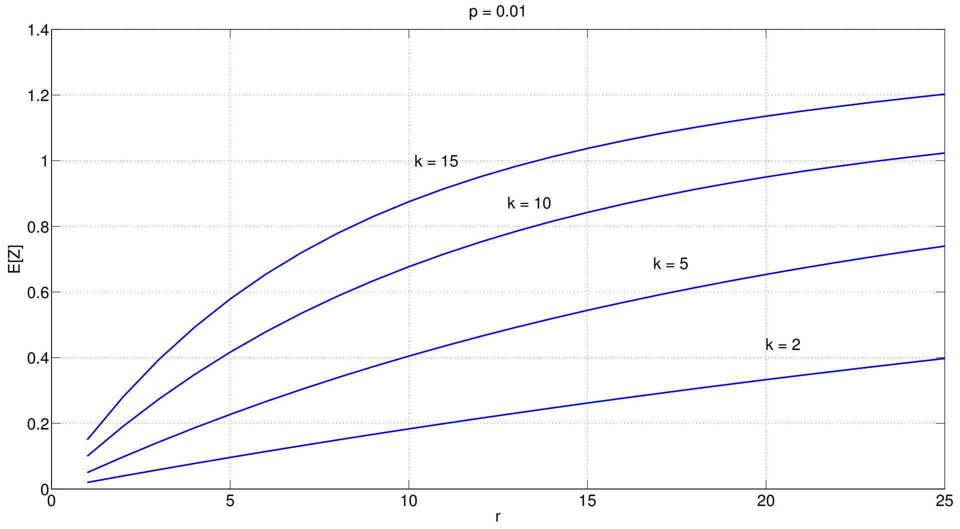


Figure 4.5: E[Z] versus number of receivers, r.

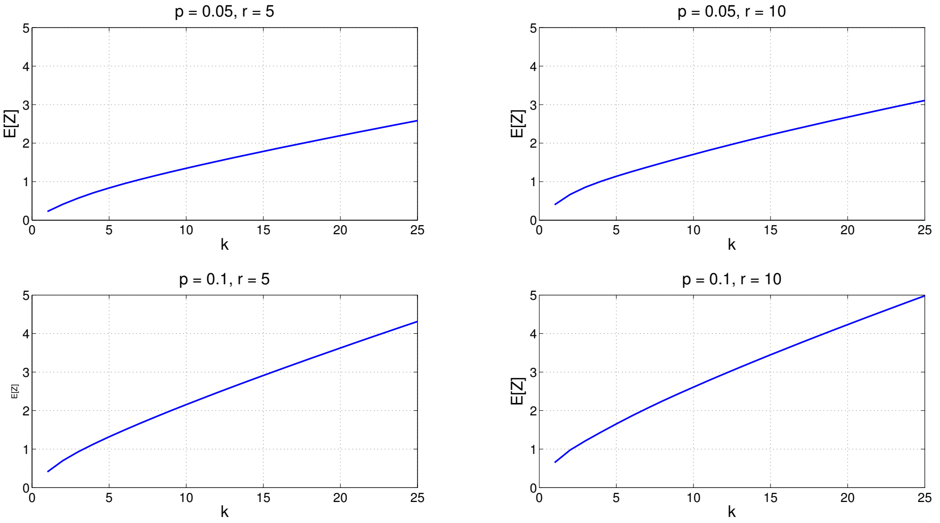


Figure 4.6: E[Z] versus number of transmitted packets, k.

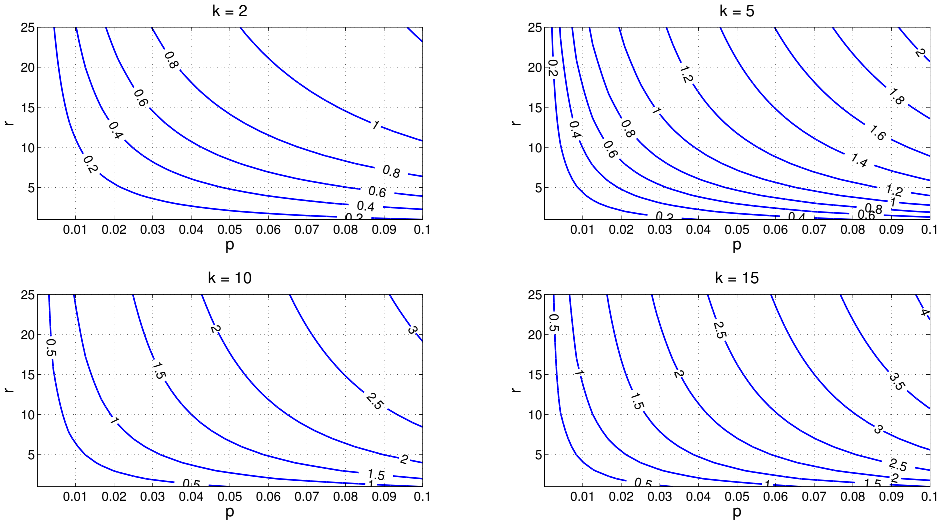


Figure 4.7: Contours of E[Z].

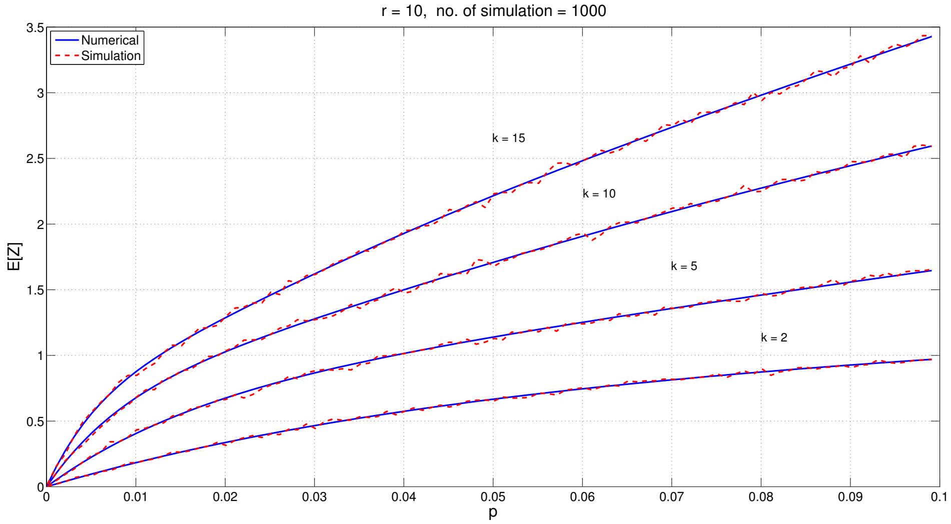


Figure 4.8: Comparison results: theory versus simulation.

Traditional ARQ scheme

For comparison, we review the average number of transmissions required for traditional ARQ scheme. Let N_{ARQ} denote the average number of transmissions required for ARQ scheme, r be the number of receivers, p be the probability of packet loss at each receiver, and $P_{success}$ be the probability that a retransmission packet is successfully delivered at all receivers. Then,

$$N_{ARQ} = P_{success} + (1 + N_{ARQ}) \times (1 - P_{success}). \tag{4.39}$$

By solving the above recursive equation, we have

$$N_{ARQ} = \frac{1}{P_{success}}. \tag{4.40}$$

Under the assumption that the packet loss is independent and identical among all receivers,

$$P_{success} = (1 - p)^r. \tag{4.41}$$

Therefore,

$$N_{ARQ} = \frac{1}{(1 - p)^r}. \tag{4.42}$$

Retransmissions Scheme

For an analysis of retransmission scheme, we assume that each receiver sends ACK/NAKs to its source based on current success or failure of packet receptions regardless of success/failure of previous retransmission attempts (i.e., memoryless receiver). Let N_{RET} denote the average number of transmissions required to successfully deliver a packet with retransmission scheme, r be the number of receivers, and p be the packet loss rate.

Proposition 4.1. *If t packets are unsuccessfully delivered out of k packet transmissions, the average number of transmissions required to deliver a packet with network coded packet* *retransmission is*

$$N_{RET} = 1 + \frac{t}{k(1-p)^{tr}} \tag{4.43}$$

where $0 < t \leq k$.

Proof. Let T_{RET} denote the total number of retransmissions for t unsuccessfully delivered packets, $P_{success}$ be the probability that the retransmission packets are successfully delivered to all receiver. Then,

$$T_{RET} = t \times P_{success} + (t + T_{RET}) \times (1 - P_{success}). \tag{4.44}$$

By solving the above recursive equation, we have

$$T_{RET} = \frac{t}{P_{success}}. \tag{4.45}$$

Under the assumption that the packet loss is independent and identical among all receivers,

$$P_{success} = (1-p)^{tr}. \tag{4.46}$$

From (4.45) and (4.46),

$$T_{RET} = \frac{t}{(1-p)^{tr}}. \tag{4.47}$$

Therefore,

$$N_{RET} = \frac{k + T_{RET}}{k} = 1 + \frac{t}{k(1-p)^{tr}}.$$

□

Forward Error Correction scheme

Let N_{FEC} denote the average number of transmissions required to successfully deliver a packet when forward error correction scheme is used with deterministic linear network coding.

Proposition 4.2. *Assume that there are t unsuccessfully delivered packets out of k broadcasting packets. The average number of transmissions required to deliver a packet for forward error correction scheme with deterministic linear network coding is*

$$N_{FEC} = 1 + 2\frac{t}{k} \tag{4.48}$$

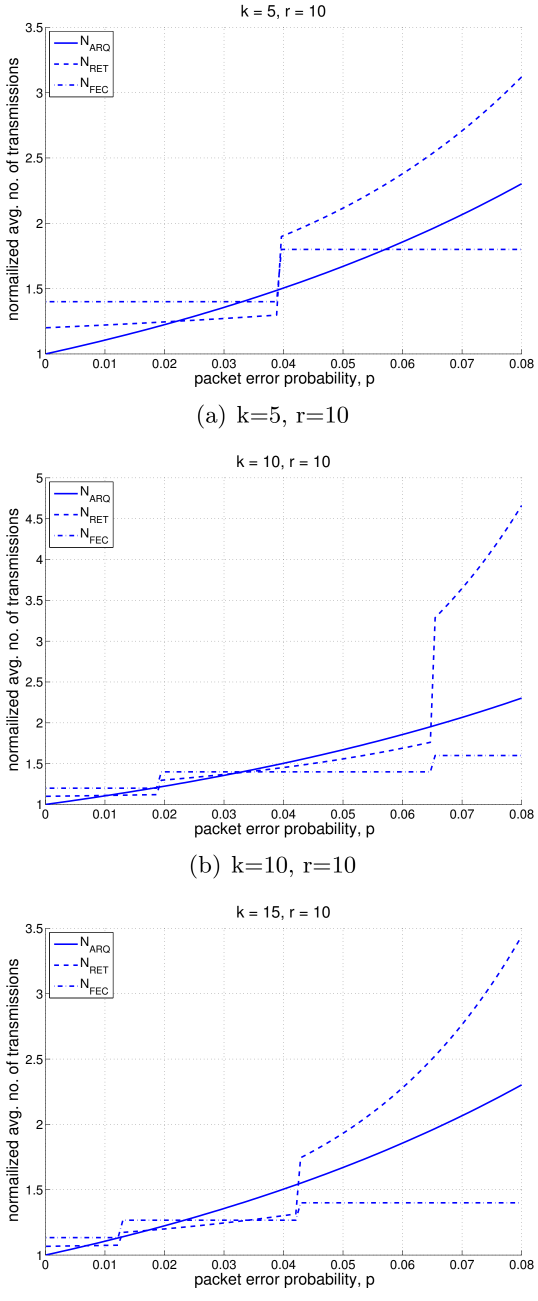
where $0 < t \leq k$.

Proof. For the forward error correction scheme using deterministic network coding, a broadcast source transmit n' encoded packets using $(n',\ k,\ n' - k + 1)$ punctured RS code to correct t packet errors, where $n' = k + 2t$. Since n' is the number of transmission required to transmit k packets while providing t-error correction capability, the average number of transmissions for successfully delivering a packet can be given by $N_{FEC} = \frac{n'}{k} = 1 + 2\frac{t}{k}$. □

4.3.2 Numerical results

We experiment numerically to evaluate the theoretical analysis of the expected number of transmissions to successfully deliver a packet in a single-hop wireless broadcast network. Both forward error correction scheme and retransmission scheme apply deterministic network coding for packet transmissions.

Figures 4.9 and 4.10 show the expected number of transmissions for delivering a packet with varying packet error probability, p, varying the number of packets for a broadcasting source to transmit, k, and varying the number of receivers within broadcasting range, r. Given k, r,and p, the maximum number of unsuccessfully delivered packets over the single-hop broadcast network among r receivers are calculated as $t = E[Z]$ (4.38).

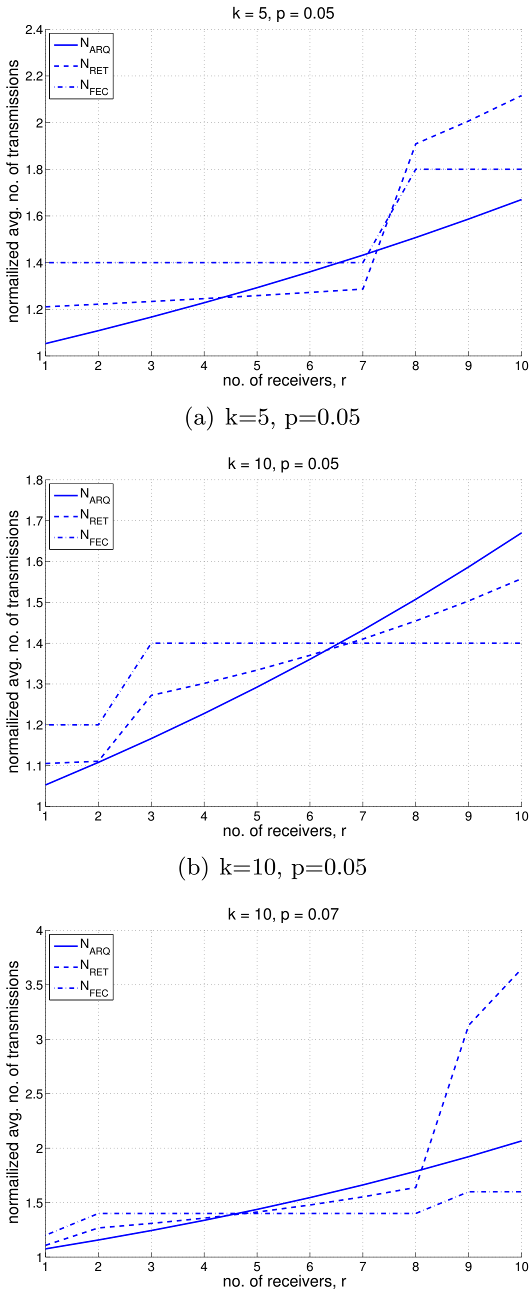


(a) k=5, r=10

(b) k=10, r=10

(c) k=15, r=10

Figure 4.9: Expected no. of transmission vs. packet error probability



(a) k=5, p=0.05

(b) k=10, p=0.05

(c) k=10, p=0.07

Figure 4.10: Expected no. of transmission vs. no. of receivers

We observe that the expected number of transmissions for retransmission scheme increases rapidly as p, k, or r increases. On the other hand, the expected number of transmissions for forward error correction scheme remains small even if the packet error probability, the number of packets to transmit, or the number of receivers are large.

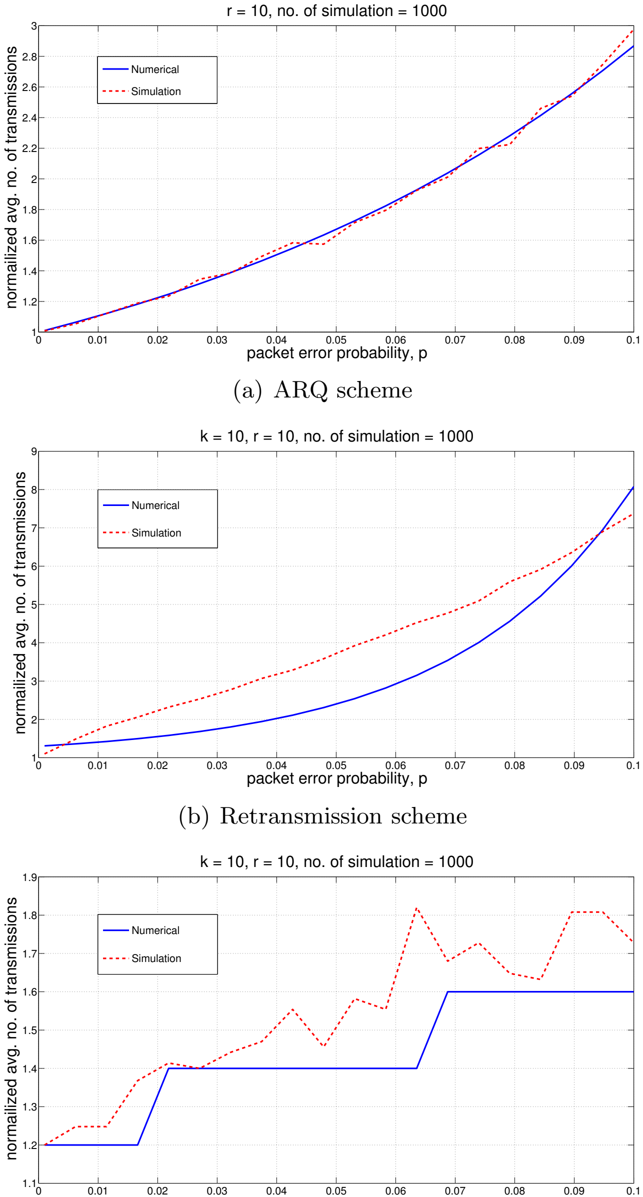
We also simulate the expected number of transmissions in a single-source, single-hop broadcast network to compare with the results of numerical analysis. Figure 4.11 shows the comparisons between analysis and simulation results for traditional ARQ scheme, retransmission scheme, and forward error correction scheme.

4.4 Discussions: unequal error protection using network coding

Theoretical analysis presented in this chapter assumes that error characteristics at all receivers in the broadcast network are identical. In the following, we provide an example of unequal error protection on a clustered broadcast network by using the proposed deterministic network coding in a hybrid way.

Unequal error protection with hybrid-ARQ scheme using network coding

Consider a clustered broadcast network in Figure 4.12, a single source has k packets to broadcast to receiver R_1 to R_{x+y}. Assume that the broadcast channel degrades signals physically, packet losses at receivers in the inner cluster is smaller than packet losses at receivers in the outer cluster (i.e., $p_1 < p_2$). To deal with the unequal error characteristics at receivers, we use the deterministic network coding for hybrid-ARQ (i.e., first apply forward error correction scheme with the proposed network code to recover lost packets at the receivers in the inner cluster, then apply network coded retransmissions in the outer cluster). The network coded packet transmissions for unequal error protection on the clustered broadcast network are described with the following steps:



(a) ARQ scheme

(b) Retransmission scheme

(c) Forward error correction scheme

Figure 4.11: Comparison result: numerical analysis vs. simulation

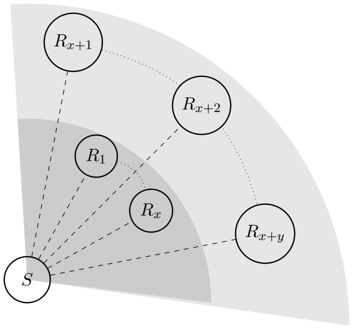


Figure 4.12: Clustered broadcast network

1. Send network coded packets with forward error correction capability

   The idea of unequal protection scheme is to recover packet losses at the receivers in the inner cluster with forward error correction and reduce the retransmission requests from the receivers in the outer cluster. The forward error correction scheme with deterministic network coding assumes that there is no help of error detection from lower layer. However, since we provide scheme that forward error correction and retransmission work together in a hybrid way, it seems that involvement of packet error detection from the lower layer is useful to the unequal error protection scheme. If CRC detects which packet comes with error, then the error location in the received packets are known to the receiver. In Theorem 4.1, we prove that network coded $k+2t$ packets correct up to t packet errors, but, with given error locations on packets, only $k+t$ encoded packets are needed to correct t packet erasures [52].

   With the clustered broadcast network, a source transmits $k+t_1$ encoded packets to recover t_1 packet erasures where t_1 denotes the number of packet losses at the inner cluster's receivers. From (4.18), a packet encoding for forward error correction using the proposed deterministic network code can be described by

$$\mathbf{c}_j = \sum_{i=0}^{k-1} \mathbf{p}_i \alpha^{ij} \quad \text{for} \quad 0 \le j \le k+t_1-1, \tag{4.49}$$

where α is a primitive element in $\mathbb{GF}(2^m)$.

2. Collect feedbacks from receivers

   Suppose packet losses at the receivers in the inner cluster is at most t_1 and at the receivers in the outer cluster is at most t_2. Since $k + t_1$ encoded packets can recover all packet erasures at the receivers in the inner cluster, packet retransmission depends only on the feedbacks from receivers in the outer cluster.

3. Send network coded packet for retransmission

   Based on the collected feedbacks from receivers, the broadcasting source knows that receivers in the inner cluster successfully received k packets. The source also knows that there are at most t_2 packet losses among receivers in the outer cluster. However, since the source initially broadcast $k + t_1$ packets, the receivers in the outer cluster successfully received at least $k - (t_2 - t_1)$ packets. Therefore the broadcasting source needs to retransmit $k - (k - (t_2 - t_1)) = t_2 - t_1$ packets for receivers in the outer cluster in order to deliver k packets.

   Let $f_1, f_2, \ldots, f_{t_2}$ be indexes for the t_2 unsuccessfully delivered packets to receivers in the outer cluster where $f_1 < f_2 < \ldots < f_{t_2}$, then a packet encoding for retransmission with the proposed deterministic network code using (4.8) can be described by

$$\mathbf{c}_j = \sum_{i=0}^{k-1} \mathbf{p}_i \alpha^{ij} \quad \text{for} \quad f_1 \leq j \leq f_{t_2 - t_1}. \tag{4.50}$$

As we have shown in the above example, both forward error correction scheme and retransmission scheme can adaptively work together to provide unequal error protection. Since we have limited our focus on a special case of the clustered broadcast network, an application of the hybrid ARQ scheme in general broadcast networks would be an interesting subject to research.

Chapter 5: Summary and future works

Error controls are important techniques for reliable communications over noisy channels. When information is corrupted by channel errors, the error control techniques are employed either to retransmit the corrupted information based on feedback (ARQ), or to correct the channel errors using error correcting codes (FEC). In this dissertation, we have investigated methodologies of both a retransmission scheme and a forward error correction scheme.

Unequal error protection (UEP) codes have been considered as an efficient error correction method when information has levels of significance. For multiuser communications over a broadcast channel, we have proposed a UEP coding scheme for a multiuser broadcast channel. First, we have introduced an integer programming approach to construct a binary UEP code. For a given unequal error protection requirement from each user, we have formulated an integer programming problem to be optimized for constructing an optimal UEP code. Numerical results and integer programming bounds have shown the efficiency of the code constructions. Also, we have investigated asymptotically achievable code rates for the multiuser communications and we have analyzed performance of the UEP coding schemes for the multiuser communications on degraded broadcast channels. Then, we have presented decoding methods that use integer programming and majority logic. When users receive information that is encoded by the UEP code, each user uses iterative integer programming to find sufficient numbers of index subsets so that each receiver can decode the information by majority logic. We have found that the UEP coding and decoding based on integer programming effectively provide unequal error protection to each user in broadcast communications.

For reliable packet transmissions over a single-hop broadcast network, we have applied a linear network coding into both a retransmission scheme and a forward error correction scheme. We have first constructed a deterministic network code from a Reed-Solomon error correcting code whose generator matrix is in the form of Vandermonde matrix. Then, we have provided an adaptive way to apply the deterministic network code for both a packet retransmission scheme and a packet-level forward error correction scheme by puncturing the generator matrix. For forward error correction scheme, we have constructed a generator matrix by puncturing the last $n - (k + 2t)$ columns of the generator matrix of a $(n,\ k,\ n-k+1)$ RS code to correct t packet errors. On the other hand, retransmission packets are generated by puncturing the last $n-t$ columns of the generator matrix of a $(n,\ k,\ n-k+1)$ RS code for t packet retransmissions. Therefore, we have found a unified solution to use the deterministic linear network code for reliable packet transmissions on a single-hop wireless broadcast network. We have provided the numerical analysis and simulation results to prove the efficiency of applying the deterministic network code for packet transmissions, and we have also shown that the unified solution can be applied in a hybrid fashion to provide efficient error controls.

5.1 Future works

In Section 3.3.2, we have discussed decodability of the proposed iterative decoding method that uses multiple integer programming and majority logic, and we have shown that not every optimal UEP code constructed by the proposed integer programming approach is decodable by the decoding method. We have also provided an idea of the modified integer programming approach to construct UEP codes that are majority-logic-decodable (i.e., received information can be decomposed into sufficient number of subsets that are decodable by majority logic). However, the modified integer programming approach in Section 3.3.2 does not guarantee to construct optimal UEP codes because of the additional constraints.

Based on what we have found, we would like to extend our research to find a UEP code construction method such that the codes are majority-logic-decodable and has optimal length. Currently, we focus on the following research directions:

1. Solution filtering: As it has been noted in Section 3.1.2, there are many optimal

solutions from integer programming for a UEP code construction that satisfies a given unequal error protection requirement. Suppose there are certain linear constraints that validate the decodability of each optimal solution. Then, every optimal solutions can be filtered by the additional constraints to obtain majority-logic-decodable UEP codes. Therefore, research objective would be finding the constraints for filtering solutions.

2. Modifying integer programming problem: The solution filtering approach ensures that the filtered solution constructs an optimal UEP code. However, since finding every or large numbers of optimal solutions from integer programming has to precede the filtering, the amount of computation time will be increased. Therefore, modifying an integer programming problem by adding linear constraints is preferred especially when k is large. Since, as pointed out, the additional constraints may change the optimal value of the solution, research objective would be finding the additional constraints for integer programming that validate decodability and optimality of the UEP code constructions.

Appendix A: Generator matrices of optimal UEP codes

Single-bit message Let $\mathbf{G}$ be a generator matrix of the optimal UEP codes constructed from Table 3.2, and $\hat{\mathbf{s}} = (\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_k)$ be the corresponding separation vector obtained from the generator matrix.

1. $k = 2$

   Integer programming solution

x_1	x_2	x_3
4	2	1

   Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,1\,1\,1 \\ 1\,1\,1\,1\,0\,0\,1 \end{pmatrix}$$

   Separation vector

$$\hat{\mathbf{s}} = (3, 5)$$

   Disjoint subsets

$$\mathcal{J}(1) = \big\{\{5\}, \{6\}, \{4, 7\}\big\}$$

$$\mathcal{J}(2) = \big\{\{1\}, \{2\}, \{3\}, \{4\}, \{6, 7\}\big\}$$

where $\mathcal{J}(i) \triangleq \{\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_{s_i}\}$ for $1 \leq i \leq k$.

2. $k = 3$

Integer programming solution

x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	2	2	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,1\,1\,1\,1\,0\,0\,1\,1 \\ 1\,1\,1\,0\,0\,1\,1\,0\,1\,0\,1 \end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = (4, 6, 7)$$

Disjoint subsets

$$\mathcal{J}(1) = \big\{\{8\}, \{5, 10\}, \{7, 11\}\big\}$$

$$\mathcal{J}(2) = \big\{\{4\}, \{5\}, \{8, 10\}, \{1, 6\}, \{9, 11\}\big\}$$

$$\mathcal{J}(3) = \big\{\{1\}, \{2\}, \{3\}, \{8, 9\}, \{4, 6\}, \{10, 11\}, \{5, 7\}\big\}$$

3. $k = 4$

Integer programming solution

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1 \\ 1\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1 \end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = \big(8, 8, 8, 9\big)$$

Disjoint subsets

$$\mathcal{J}(1) = \big\{\{9\}, \{3, 11\}, \{4, 12\}\big\}$$

$$\mathcal{J}(2) = \big\{\{5\}, \{11, 15\}, \{10, 14\}, \{4, 8\}, \{1, 6\}\big\}$$

$$\mathcal{J}(3) = \big\{\{3\}, \{13, 15\}, \{10, 12\}, \{6, 8\}, \{14, 16\}, \{2, 4\}, \{5, 7\}\big\}$$

$$\mathcal{J}(4) = \big\{\{1\}, \{2\}, \{13, 14\}, \{11, 12\}, \{7, 8\}, \{15, 16\}, \{3, 4\}, \{5, 6\}, \{9, 10\}\big\}$$

4. $k = 5$

Integer programming solution

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	x_{19}
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1 \\ 1\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1 \end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = \big(4, 8, 8, 10, 11\big)$$

Disjoint subsets

$$\mathcal{J}(1) = \big\{\{17\}, \{4, 20\}, \{1, 18\}\big\}$$

$$\mathcal{J}(2) = \big\{\{9\}, \{7, 15\}, \{6, 14\}, \{4, 12\}, \{1, 10\}\big\}$$

$$\mathcal{J}(3) = \big\{\{5\}, \{11, 15\}, \{10, 14\}, \{4, 8\}, \{1, 6\}, \{3, 7\}, \{9, 13\}\big\}$$

$$\mathcal{J}(4) = \big\{\{3\}, \{17, 19\}, \{13, 15\}, \{10, 12\}, \{6, 8\}, \{14, 16\}, \{2, 4\}, \{5, 7\}, \{9, 11\}\big\}$$

$$\mathcal{J}(5) = \big\{\{1\}, \{2\}, \{17, 18\}, \{7, 8\}, \{13, 14\}, \{11, 12\}, \{15, 16\}, \{3, 4\},$$
$$\{5, 6\}, \{9, 10\}, \{19, 20\}\big\}$$

5. $k = 6$

Integer programming solution

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	x_{19}	x_{20}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

x_{21}	x_{32}	x_{33}	x_{34}	x_{35}
1	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,0\,0\,0\,0 \\ 0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,0\,0\,1\,1 \\ 1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1 \end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = (4, 6, 8, 10, 12, 13)$$

Disjoint subsets

$$\begin{aligned}
\mathcal{J}(1) &= \big\{\{22\},\{1,23\},\{2,24\}\big\} \\
\mathcal{J}(2) &= \big\{\{16\},\{5,21\},\{1,17\},\{3,19\},\{2,18\}\big\} \\
\mathcal{J}(3) &= \big\{\{8\},\{4,12\},\{3,11\},\{7,15\},\{1,9\}, \\
&\qquad \{2,10\},\{5,13\}\big\} \\
\mathcal{J}(4) &= \big\{\{4\},\{3,7\},\{1,5\},\{2,6\},\{8,12\}, \\
&\qquad \{10,14\},\{9,13\},\{11,15\},\{16,20\}\big\} \\
\mathcal{J}(5) &= \big\{\{2\},\{16,18\},\{1,3\},\{5,7\},\{4,6\}, \\
&\qquad \{8,10\},\{12,14\},\{9,11\},\{13,15\},\{17,19\},\{22,24\}\big\} \\
\mathcal{J}(6) &= \big\{\{1\},\{6,7\},\{2,3\},\{4,5\},\{8,9\}, \\
&\qquad \{10,11\},\{12,13\},\{14,15\},\{16,17\},\{18,19\}, \\
&\qquad \{20,21\},\{22,23\},\{24,25\}\big\}
\end{aligned}$$

6. $k = 7$

Integer programming solution

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	x_{19}	x_{20}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

x_{21}	x_{22}	x_{23}	x_{32}	x_{33}	x_{34}	x_{35}	x_{64}	x_{65}	x_{104}
1	1	1	1	1	1	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1 \\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&0&1 \\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0 \\
0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1 \\
0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0 \\
0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0 \\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0
\end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = (3, 5, 8, 9, 12, 14, 15)$$

Disjoint subsets

$$\mathcal{J}(1) = \big\{\{28\}, \{1, 29\}, \{8, 24, 30\}\big\}$$

$$\mathcal{J}(2) = \big\{\{24\}, \{1, 25\}, \{2, 26\}, \{3, 27\}, \{8, 28, 30\}\big\}$$

$$\begin{aligned}\mathcal{J}(3) = \big\{&\{16\}, \{3, 19\}, \{7, 23\}, \{2, 18\}, \{1, 17\},\\ &\{4, 20\}, \{5, 21\}\big\}\end{aligned}$$

$$\begin{aligned}\mathcal{J}(4) = \big\{&\{8\}, \{1, 9\}, \{5, 13\}, \{4, 12\}, \{2, 10\},\\ &\{3, 11\}, \{6, 14\}, \{7, 15\}, \{24, 28, 30\}\big\}\end{aligned}$$

$$\begin{aligned}\mathcal{J}(5) = \big\{&\{4\}, \{17, 21\}, \{11, 15\}, \{8, 12\}, \{18, 22\},\\ &\{1, 5\}, \{3, 7\}, \{2, 6\}, \{10, 14\}, \{9, 13\}, \{16, 20\}\big\}\end{aligned}$$

$$\begin{aligned}\mathcal{J}(6) = \big\{&\{2\}, \{4, 6\}, \{12, 14\}, \{1, 3\}, \{5, 7\},\\ &\{8, 10\}, \{9, 11\}, \{16, 18\}, \{17, 19\}, \{20, 22\},\\ &\{13, 15\}, \{21, 23\}, \{24, 26\}\big\}\end{aligned}$$

$$\begin{aligned}\mathcal{J}(7) = \big\{&\{1\}, \{14, 15\}, \{22, 23\}, \{8, 9\}, \{12, 13\},\\ &\{6, 7\}, \{18, 19\}, \{20, 21\}, \{4, 5\}, \{16, 17\},\\ &\{26, 27\}, \{2, 3\}, \{10, 11\}, \{24, 25\}, \{28, 29\}\big\}\end{aligned}$$

7. $k = 8$

Integer programming solution

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	x_{19}	x_{20}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

x_{21}	x_{22}	x_{23}	x_{32}	x_{33}	x_{34}	x_{35}	x_{64}	x_{65}	x_{66}	x_{126}	x_{127}	x_{128}	x_{129}	x_{184}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix}
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0\,0 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,1\,1\,0\,0\,1 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1 \\
0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1 \\
0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0 \\
0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,1\,0\,0\,0 \\
1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,0\,1\,0\,1\,0
\end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = (3, 5, 7, 11, 11, 13, 15, 17)$$

Disjoint subsets

$$\mathcal{J}(1) = \big\{\{33\}, \{1,34\}, \{8,16,24,35\}\big\}$$

$$\mathcal{J}(2) = \big\{\{28\}, \{1,29\}, \{2,30\}, \{6,31,33,35\}, \{12,16,27,32\}\big\}$$

$$\mathcal{J}(3) = \big\{\{24\}, \{2,26\}, \{1,25\}, \{3,27\}$$

$$\{14,16,28,31\}\{10,20,29,32\}, \{7,13,18,33,35\}, \big\}$$

$$\mathcal{J}(4) = \big\{\{16\}, \{7,23\}, \{1,17\}, \{6,22\}, \{2,18\},$$

$$\{3,19\}, \{4,20\}, \{5,21\}, \{12,24,30,31\}\big\}$$

$$\mathcal{J}(5) = \big\{\{8\}, \{7,15\}, \{1,9\}, \{6,14\}, \{2,10\}, \{4,12\}, \{3,11\},$$

$$\{5,13\}, \{20,24,30,31\}, \{21,26,28,32\}, \{16,18,19,25,33,35\}\big\}$$

$$\mathcal{J}(6) = \big\{\{4\}, \{9,13\}, \{11,15\}, \{2,6\}, \{10,14\},$$

$$\{3,7\}, \{1,5\}, \{8,12\}, \{16,20\}, \{18,22\},$$

$$\{17,21\}, \{19,23\}, \{30,31,33,35\}\big\}$$

$$\mathcal{J}(7) = \big\{\{2\}, \{1,3\}, \{25,27\}, \{9,11\}, \{24,26\},$$

$$\{5,7\}, \{4,6\}, \{8,10\}, \{12,14\}, \{13,15\},$$

$$\{16,18\}, \{17,19\}, \{20,22\}, \{21,23\}, \{28,30\}\big\}$$

$$\mathcal{J}(8) = \big\{\{1\}, \{6,7\}, \{8,9\}, \{33,34\}, \{14,15\}, \{10,11\},$$

$$\{4,5\}, \{12,13\}, \{28,29\}, \{24,25\}, \{20,21\}, \{18,19\},$$

$$\{2,3\}, \{16,17\}, \{22,23\}, \{26,27\}, \{31,32\}\big\}$$

Multi-bit message Let $\mathbf{G}$ be a generator matrix of the optimal UEP codes constructed from Table 3.5, and $\hat{\mathbf{s}} = (\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_k,)$ be the corresponding separation vector with respect to the $\mathbf{G}$.

1. $k = 2$ and $l = 2$

   Integer programming solution

x_1	x_2	x_3	x_4	x_5	x_7	x_8	x_9	x_{10}	x_{11}
1	1	1	1	1	1	1	1	1	1

   Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,1\,1\,1\,0\,0\,0\,0 \\ 0\,1\,1\,0\,0\,1\,0\,0\,1\,1 \\ 1\,0\,1\,0\,1\,1\,0\,1\,0\,1 \end{pmatrix}$$

   Separation vector

$$\hat{\mathbf{s}} = (3, 5)$$

2. $k = 3$ and $l = 2$

   Integer programming solution

x_1	x_2	x_4	x_5	x_6	x_8	x_9	x_{10}	x_{11}	x_{16}	x_{17}	x_{19}	x_{32}	x_{35}	x_{47}	x_{61}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0000000000001111 \\ 0000000001110001 \\ 0000011110000011 \\ 0011100000000011 \\ 0100100110010110 \\ 1001001010110111 \end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = (4, 5, 7)$$

3. $k = 2$ and $l = 3$

Integer programming solution

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_{16}	x_{32}	x_{44}	x_{54}	x_{61}
1	1	1	1	1	1	1	1	1	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0000000001111 \\ 0000000010011 \\ 0000000100101 \\ 0001111000111 \\ 0110011000010 \\ 1010101000001 \end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = (3, 5)$$

4. $k = 2$ and $l = 4$

Integer programming solution

x_1	x_2	x_4	x_7	x_8	x_{11}	x_{13}	x_{14}	x_{16}	x_{32}	x_{64}	x_{127}	x_{128}	x_{179}	x_{213}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Generator matrix

$$\mathbf{G} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,1\,0\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,1\,0\,1\,1 \\ 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,0\,1\,0\,0\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1\,0\,0\,0\,1\,0\,1\,0 \\ 1\,0\,0\,1\,0\,1\,1\,0\,0\,0\,0\,1\,0\,1\,1 \end{pmatrix}$$

Separation vector

$$\hat{\mathbf{s}} = (3, 5)$$

Appendix B: Derivation of (3.105)

From the integer programming formulation, we have $\frac{q^k-1}{q-1}$ inequalities in the form of

$$\mathbf{A}\mathbf{x}^\top \geq \mathbf{b}^\top.$$

Since each column of $\mathbf{A}$ has q^{k-1} ones, the sum of $\frac{q^k-1}{q-1}$ inequalities can be given by

$$q^{k-1}\left(x_1 + \cdots + x_{q^k-1}\right) \ \geq \ s_1 + qs_2 + \cdots + q^{(k-1)}s_k,$$

hence,

$$q^{k-1}\, n \ \geq \ \sum_{i=1}^{k} q^{i-1} s_i. \tag{B.1}$$

By applying ceiling functions for integer property,

$$n \ \geq \ \sum_{i=1}^{k} \left\lceil \frac{q^{i-1}}{q^{k-1}}\, s_i \right\rceil. \tag{B.2}$$

Therefore,

$$n \ \geq \ \sum_{i=1}^{k} \left\lceil \frac{s_i}{q^{k-i}} \right\rceil.$$

Bibliography

Bibliography

[1] B. Masnick and J. Wolf, "On linear unequal error protection codes," *IEEE Transactions on Information Theory*, vol. 13, no. 4, pp. 600–607, Oct. 1967.

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[3] L. A. Dunning and W. E. Robbins, "Optimal encodings of linear block codes for unequal error protection," *Informatioin and Control*, vol. 37, pp. 150–177, May 1978.

[4] I. Boyarinov and G. Katsman, "Linear unequal error protection codes," *IEEE Transactions on Information Theory*, vol. 27, no. 2, pp. 168–175, Mar. 1981.

[5] W. J. van Gils, "Two topics on linear unequal error protection codes: Bounds on their length and cyclic code classes," *IEEE Transactions on Information Theory*, vol. 29, no. 6, pp. 866–876, Nov. 1983.

[6] ——, "Linear unequal error protection codes from shorter codes (corresp.)," *IEEE Transactions on Information Theory*, vol. 30, no. 3, pp. 544–546, May 1984.

[7] M.-C. Lin and S. Lin, "Codes with multi-level error-correcting capabilities," *Discrete Mathematics*, vol. 83, no. 2-3, pp. 301–314, 1990.

[8] R. H. Morelos-Zaragoza and S. Lin, "On a class of optimal nonbinary linear unequal-error-protection codes for two sets of messages," *IEEE Transactions on Information Theory*, vol. 40, no. 1, pp. 196–200, Jan. 1994.

[9] U. Dettmar, Y. Gao, and U. Sorger, "Modified generalized concatenated codes and their application to the construction and decoding of LUEP codes," *IEEE Transactions on Information Theory*, vol. 41, no. 5, pp. 1499–1503, Sep. 1995.

[10] R. H. Morelos-Zaragoza and H. Imai, "Binary multilevel convolutional codes with unequal error protection capabilities," *IEEE Transactions on Communications*, vol. 46, no. 7, pp. 850–853, Jul. 1998.

[11] V. Kumar and O. Milenkovic, "On unequal error protection LDPC codes based on plotkin-type constructions," *IEEE Transactions on Communications*, vol. 54, no. 6, pp. 994–1005, Jun. 2006.

[12] S. Borade, B. Nakiboğlu, and L. Zheng, "Unequal error protection: An information-theoretic perspective," *IEEE Transactions on Information Theory*, vol. 55, no. 12, pp. 5511–5539, Dec. 2009.

[13] B. Nakiboğlu, S. K. Gorantla, L. Zheng, and T. P. Coleman, “Bit-wise unequal error protection for variable-length block codes with feedback,” *IEEE Transactions on Information Theory*, vol. 59, no. 3, pp. 1475–1504, Mar. 2013.

[14] A. R. Calderbank and N. Seshadri, “Multilevel codes for unequal error protection,” *IEEE Transactions on Information Theory*, vol. 39, no. 4, pp. 1234–1248, Jul. 1993.

[15] L.-F. Wei, “Coded modulation with unequal error protection,” *IEEE Transactions on Communications*, vol. 41, no. 10, pp. 1439–1449, Oct. 1993.

[16] R. H. Morelos-Zaragoza and S. Lin, “QPSK block-modulation codes for unequal error protection,” *IEEE Transactions on Information Theory*, vol. 41, no. 2, pp. 576–581, Mar. 1995.

[17] R. H. Morelos-Zaragoza, T. Kasami, S. Lin, and H. Imai, “On block-coded modulation using unequal error protection codes over Rayleigh-fading channels,” *IEEE Transactions on Communications*, vol. 46, no. 1, pp. 1–4, Jan. 1998.

[18] R. H. Morelos-Zaragoza, M. P. C. Fossorier, S. Lin, and H. Imai, “Multilevel coded modulation for unequal error protection and multistage decoding – part I: Symmetric constellations,” *IEEE Transactions on Communications*, vol. 48, no. 2, pp. 204–213, Feb. 2000.

[19] M. Isaka, M. P. C. Fossorier, R. H. Morelos-Zaragoza, S. Lin, and H. Imai, “Multilevel coded modulation for unequal error protection and multistage decoding – part II: Asymmetric constellations,” *IEEE Transactions on Communications*, vol. 48, no. 5, pp. 774–786, May 2000.

[20] A. Limmanee and W. Henkel, “UEP network coding for scalable data,” in *5th International Symposium on Turbo Codes and Related Topics*, Sep. 2008, pp. 333–337.

[21] N. Thomos, J. Chakareski, and P. Frossard, “Prioritized distributed video delivery with randomized network coding,” *IEEE Transactions on Multimedia*, vol. 13, no. 4, pp. 776–787, Aug. 2011.

[22] M. Iezzi, M. Di Renzo, and F. Graziosi, “Network code design from unequal error protection coding: Channel-aware receiver design and diversity analysis,” in *IEEE International Conference on Communications (ICC)*, Jun. 2011, pp. 1–6.

[23] C. Yao, J. K. Zao, C.-H. Wang, S.-Y. R. Li, N. A. Claude, and K.-K. Yen, “On separation vectors of static linear network codes with UEP capability,” in *International Symposium on Network Coding (NetCod)*, Jul. 2011, pp. 1–6.

[24] D. Vukobratović and V. Stanković, “Unequal error protection random linear coding strategies for erasure channels,” *IEEE Transactions on Communications*, vol. 60, no. 5, pp. 1243–1252, May 2012.

[25] J. Yue, Z. Lin, and B. Vucetic, “Distributed fountain codes with adaptive unequal error protection in wireless relay networks,” vol. 13, no. 8, pp. 4220–4231, Aug. 2014.

[26] T. M. Cover, “Broadcast channels,” *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 2–14, Jan. 1972.

[27] P. Bergmans, “Random coding theorem for broadcast channels with degraded components,” *IEEE Transactions on Information Theory*, vol. 19, no. 2, pp. 197–207, Mar. 1973.

[28] R. G. Gallager, “Capacity and coding for degraded broadcast channels,” *Problemy Peredachi Informatsii*, vol. 10, no. 3, pp. 3–14, Jul.-Sep. 1974.

[29] R. Ahlswede and J. Körner, “Source coding with side information and a converse for degraded broadcast channels,” *IEEE Transactions on Information Theory*, vol. 21, no. 6, pp. 629–637, Nov. 1975.

[30] J. Körner and K. Marton, “General broadcast channels with degraded message sets,” *IEEE Transactions on Information Theory*, vol. 23, no. 1, pp. 60–64, Jan. 1977.

[31] A. El Gamal, “The feedback capacity of degraded broadcast channels (corresp.),” *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 379–381, May 1978.

[32] T. Kasami, S. Lin, V. K. Wei, and S. Yamamura, “Coding for the binary symmetric broadcast channel with two receivers,” *IEEE Transactions on Information Theory*, vol. 31, no. 5, pp. 616–625, Sep. 1985.

[33] T. M. Cover, “Comments on broadcast channels,” *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2524–2530, Oct. 1998.

[34] C. Nair and A. El Gamal, “The capacity region of a class of three-receiver broadcast channels with degraded message sets,” *IEEE Transactions on Information Theory*, vol. 55, no. 10, pp. 4479–4493, Oct. 2009.

[35] A. Sultan, *Linear programming: an introduction with applications.* San Diego, CA: Academic press, Inc., 1993.

[36] R. K. Martin, *Large scale linear and integer optimization: a unified approach.* Norwell, MA: Kluwer academic publishers, 1999.

[37] J. C. Smith and Z. C. Taşkin, “A tutorial guide to mixed-integer programming models and solution techniques,” in *Optimization in Medicine and Biology*, G. J. Lim and E. K. Lee, Eds. Auerbach Publications, 2008.

[38] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes.* New York, NY: North-Holland publishing company, 1977.

[39] J. K. Omura, “Iterative decoding of linear codes by a modulo-2 linear program,” *Discrete Mathematics*, vol. 3, no. 1–3, pp. 193–208, 1972.

[40] M. Breitbach, M. Bossert, R. Lucas, and C. Kempter, “Soft-decision decoding of linear block codes as optimization problem,” *European Transactions on Telecommunications*, vol. 9, no. 3, pp. 289–293, May – Jun. 1998.

[41] T. Kasami, “On integer programming problems related to soft-decision iterative decoding algorithms,” in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC)*, ser. Lecture Notes in Computer Science, vol. 1719. Springer, 1999, pp. 43–54.

[42] J. Feldman, M. J. Wainwright, and D. R. Karger, “Using linear programming to decode binary linear codes,” *IEEE Transactions on Information Theory*, vol. 51, no. 3, pp. 954–972, Mar. 2005.

[43] S. C. Draper, J. S. Yedidia, and Y. Wang, “ML decoding via mixed-integer adaptive linear programming,” in *IEEE International Symposium on Information Theory (ISIT)*, 2007, pp. 1656–1660.

[44] K. Yang, X. Wang, and J. Feldman, “A new linear programming approach to decoding linear block codes,” *IEEE Transactions on Information Theory*, vol. 54, no. 3, pp. 1061–1072, Mar. 2008.

[45] M. F. Flanagan, V. Skachek, E. Byrne, and M. Greferath, “Linear-programming decoding of nonbinary linear codes,” *IEEE Transactions on Information Theory*, vol. 55, no. 9, pp. 4134–4154, Sep. 2009.

[46] A. Tanatmis, S. Ruzika, H. W. Hamacher, M. Punekar, F. Kienle, and N. Wehn, “A separation algorithm for improved LP-decoding of linear block codes,” *IEEE Transactions on Information Theory*, vol. 56, no. 7, pp. 3277–3289, Jul. 2010.

[47] M. Punekar, F. Kienle, N. Wehn, A. Tanatmis, S. Ruzika, and H. W. Hamacher, “Calculating the minimum distance of linear block codes via integer programming,” in *6th International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, Sep. 2010, pp. 329–333.

[48] M. Helmling, S. Ruzika, and A. Tanatmis, “Mathematical programming decoding of binary linear codes: Theory and algorithms,” *IEEE Transactions on Information Theory*, vol. 58, no. 7, pp. 4753–4769, Jul. 2012.

[49] S. Scholl, F. Kienle, M. Helmling, and S. Ruzika, “Integer programming as a tool for analysis of channel codes,” in *Proceedings of 2013 9th International ITG Conference on Systems, Communication and Coding (SCC)*, Jan. 2013, pp. 1–6.

[50] M. Esmaeili, A. Alampour, and T. A. Gulliver, “Decoding binary linear block codes using local search,” *IEEE Transactions on Communications*, vol. 61, no. 6, pp. 2138–2145, Jun. 2013.

[51] M. Punekar, P. Vontobel, and M. Flanagan, “Low-complexity LP decoding of nonbinary linear codes,” *IEEE Transactions on Information Theory*, vol. 61, no. 8, pp. 3073–3085, Aug. 2013.

[52] S. Lin and D. J. Costello, Jr., *Error control coding*, 2nd ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2004.

[53] S.-Y. R. Li, R. W. Yeung, and N. Cai, “Linear network coding,” *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[54] R. Koetter and M. Médard, “An algebraic approach to network coding,” *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[55] T. Ho, M. Médard, J. Shi, M. Effros, and D. R. Karger, "On randomized network coding," in *Proceedings of 41st Annual Allerton Conference on Communication, Control, and Computing*, Oct. 2003.

[56] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Transactions on Information Theory*, vol. 51, no. 6, pp. 1973–1982, Jun. 2005.

[57] N. J. A. Harvey, D. R. Karger, and K. Murota, "Deterministic network coding by matrix completion," in *Proceedings of 16th annual ACM-SIAM symposium on Discrete algorithms (SODA)*, 2005, pp. 489–498.

[58] A. Ramamoorthy, J. Shi, and R. D. Wesel, "On the capacity of network coding for random networks," *IEEE Transactions on Information Theory*, vol. 51, no. 8, pp. 2878–2885, Aug. 2005.

[59] T. Ho, M. Médard, R. Köetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[60] C. Fragouli and E. Soljanin, "Information flow decomposition for network coding," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 829–848, Mar. 2006.

[61] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proceedings of 41st Allerton Conference on Communication, Control and Computing*, Oct. 2003.

[62] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," *IEEE/ACM Transactions on Networking*, vol. 16, no. 3, pp. 497–510, Jun. 2008.

[63] R. W. Yeung and N. Cai, "Network error correction, part I: Basic concepts and upper bounds," *Communications in Information and Systems*, vol. 6, no. 1, pp. 19–36, 2006.

[64] ——, "Network error correction, part II: Lower bounds," *Communications in Information and Systems*, vol. 6, no. 1, pp. 37–54, 2006.

[65] D. S. Lun, M. Médard, R. Koetter, and M. Effros, "On coding for reliable communication over packet networks," *Physical Communication*, vol. 1, no. 1, pp. 3–20, Mar. 2008.

[66] Z. Zhang, "Linear network error correction codes in packet networks," *IEEE Transactions on Information Theory*, vol. 54, no. 1, pp. 209–218, Jan. 2008.

[67] R. Köetter and F. R. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Transactions on Information Theory*, vol. 54, no. 8, pp. 3579–3591, Aug. 2008.

[68] D. Silva, F. R. Kschischang, and R. Köetter, "A rank-metric approach to error control in random network coding," *IEEE Transactions on Information Theory*, vol. 54, no. 9, pp. 3951–3967, Sep. 2008.

[69] D. Silva and F. R. Kschischang, "On metrics for error correction in network coding," *IEEE Transactions on Information Theory*, vol. 55, no. 12, pp. 5479–5490, Dec. 2009.

[70] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 2, pp. 914–925, Feb. 2009.

[71] X. Xiao, L.-M. Yang, W.-P. Wang, and S. Zhang, "A wireless broadcasting retransmission approach based on network coding," in *Proceedings of 4th IEEE International Conference on Circuits and Systems for Communications, ICCSC 2008*, May 2008, pp. 782–786.

[72] *User's manual for CPLEX V12.5.1*, IBM ILOG, 2013. [Online]. Available: http://pic.dhe.ibm.com/infocenter/cosinfoc/v12r5/index.jsp

[73] H. J. Helgert and R. D. Stinaff, "Shortened BCH codes," *IEEE Transactions on Information Theory*, vol. 19, no. 6, pp. 818–820, Nov. 1973.

[74] D. E. Knuth, *The art of computer programming*, 3rd ed. Reading, MA: Addison-Wesley, 1997, vol. 1.

[75] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, Jun. 1960.

[76] M.-C. Lin and S. Lin, "Cyclic unequal error protection codes constructed from cyclic codes of composite length," *IEEE Transactions on Information Theory*, vol. 34, no. 4, pp. 867–871, Jul. 1988.

[77] M.-C. Lin, C.-C. Lin, and S. Lin, "Computer search for binary cyclic UEP codes of odd length up to 65," *IEEE Transactions on Information Theory*, vol. 36, no. 4, pp. 924–935, Jul. 1990.

[78] T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "On multilevel block modulation codes," *IEEE Transactions on Information Theory*, vol. 37, no. 4, pp. 965–975, Jul. 1991.

[79] T. Takata, S. Ujita, T. Kasami, and S. Lin, "Multistage decoding of multilevel block M-PSK modulation codes and its performance analysis," *IEEE Transactions on Information Theory*, vol. 39, no. 4, pp. 1204–1218, Jul. 1993.

[80] R. H. Morelos-Zaragoza and S. Lin, "On primitive BCH codes with unequal error protection capabilities," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 788–790, May 1995.

[81] J. Limbanyen and K. Yamaguchi, "Unequal error protection codes based on $|u|u+v|\cdot|$ construction," in *IEEE International Symposium on Information Theory (ISIT)*, Cambridge, MA, Aug.16 – 21, 1998, p. 198.

[82] E. Fujiwara, T. Ritthongpitak, and M. Kitakami, "Optimal two-level unequal error control codes for computer systems," *IEEE Transactions on Computers*, vol. 47, no. 12, pp. 1313–1325, Dec. 1998.

[83] U. Wachsmann, R. F. H. Fischer, and J. B. Huber, "Multilevel codes: Theoretical concepts and practical design rules," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1361–1391, Jul. 1999.

[84] N. Rahnavard, B. Vellambi, and F. Fekri, "Rateless codes with unequal error protection property," *IEEE Transactions on Information Theory*, vol. 53, no. 4, pp. 1521–1532, Apr. 2007.

[85] S. Borade, L. Zheng, and M. Trott, "Multilevel broadcast networks," in *IEEE International Symposium on Information Theory (ISIT)*, Jun. 2007, pp. 1151–1155.

[86] S. Borade, B. Nakiboğlu, and L. Zheng, "Some fundamental limits of unequal error protection," in *IEEE International Symposium on Information Theory (ISIT)*, Jul. 2008, pp. 2222–2226.

[87] E. Kuriata, "Creation of unequal error protection codes for two groups of symbols," *International Journal of Applied Mathematics and Computer Science*, vol. 18, no. 2, pp. 251–257, Jun. 2008.

[88] D. Sejdinović, D. Vukobratović, A. Doufexi, V. Senk, and R. J. Piechocki, "Expanding window fountain codes for unequal error protection," *IEEE Transactions on Communications*, vol. 57, no. 9, pp. 2510–2516, Sep. 2009.

[89] C.-H. Wang, M.-C. Chiu, and C.-C. Chao, "On unequal error protection of convolutional codes from an algebraic perspective," *IEEE Transactions on Information Theory*, vol. 56, no. 1, pp. 296–315, Jan. 2010.

[90] R. Morelos-Zaragoza and N. D'souza, "Two-level channel coding for cooperative wireless networks based on WiMAX LDPC codes," in *IEEE 22nd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2011, pp. 2349–2353.

[91] A. B. Fontaine and W. W. Peterson, "On coding for the binary symmetric channel," *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, vol. 77, no. 5, pp. 638–647, Nov. 1958.

[92] P. Bergmans and T. M. Cover, "Cooperative broadcasting," *IEEE Transactions on Information Theory*, vol. 20, no. 3, pp. 317–324, May 1974.

[93] P. Bergmans, "A simple converse for broadcast channels with additive white gaussian noise (corresp.)," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 279–280, Mar. 1974.

[94] T. M. Cover, "An achievable rate region for the broadcast channel," *IEEE Transactions on Information Theory*, vol. 21, no. 4, pp. 399–404, Jul. 1975.

[95] T. Kasami and S. Lin, "Coding for a multiple-access channel," *IEEE Transactions on Information Theory*, vol. 22, no. 2, pp. 129–137, Mar. 1976.

[96] K. Marton, "A coding theorem for the discrete memoryless broadcast channel," *IEEE Transactions on Information Theory*, vol. 25, no. 3, pp. 306–311, May 1979.

[97] A. El Gamal, "The capacity of a class of broadcast channels," *IEEE Transactions on Information Theory*, vol. 25, no. 2, pp. 166–169, Mar. 1979.

[98] A. El Gamal and T. M. Cover, "Multiple user information theory," *Proceedings of the IEEE*, vol. 68, no. 12, pp. 1466–1483, Dec. 1980.

[99] C. Nair, "Capacity regions of two new classes of two-receiver broadcast channels," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4207–4214, Sep. 2010.

[100] C.-C. Wang, "On the capacity of 1-to-K broadcast packet erasure channels with channel output feedback," *IEEE Transactions on Information Theory*, vol. 58, no. 2, pp. 931–956, Feb. 2012.

[101] E. J. McCluskey, "Error-correcting codes: a linear programming approach," *The Bell System Technical Journal*, vol. 38, no. 6, pp. 1485–1512, Nov. 1959.

[102] H. T. Moorthy, S. Lin, and T. Kasami, "Soft-decision decoding of binary linear block codes based on an iterative search algorithm," *IEEE Transactions on Information Theory*, vol. 43, no. 3, pp. 1030–1040, May 1997.

[103] M. P. C. Fossorier and S. Lin, "A unified method for evaluating the error-correction radius of reliability-based soft-decision algorithms for linear block codes," *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 691–700, Mar. 1998.

[104] A. Eryilmaz, A. Ozdaglar, and M. Médard, "On delay performance gains from network coding," in *Proceedings of 40th Annual Conference on Information Sciences and Systems*, Mar. 2006, pp. 864–870.

[105] P. Chou and Y. Wu, "Network coding for the internet and wireless networks," *IEEE Signal Processing Magazine*, vol. 24, no. 5, pp. 77–85, Sep. 2007.

[106] M. Ghaderi, D. Towsley, and J. Kurose, "Reliability gain of network coding in lossy wireless networks," in *Proceedings of IEEE 27th Conference on Computer Communications (INFOCOM)*, Apr. 2008, pp. 2171–2179.

[107] C. Fragouli, J. Widmer, and J.-Y. Le Boudec, "Efficient broadcasting using network coding," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 450–463, Apr. 2008.

[108] X. Xiao, L. Yang, W. Wang, and S. Zhang, "A broadcasting retransmission approach based on random linear network coding," in *Proceedings of the 9th International Conference for Young Computer Scientists, ICYCS 2008*, Nov. 2008, pp. 457–461.

[109] Y. Wu, V. Stankovic, Z. Xiong, and S.-Y. Kung, "On practical design for joint distributed source and network coding," *IEEE Transactions on Information Theory*, vol. 55, no. 4, pp. 1709–1720, Apr. 2009.

[110] I. Stojanovic, Z. Wu, M. Sharif, and D. Starobinski, "Data dissemination in wireless broadcast channels: Network coding versus cooperation," *IEEE Transactions on Wireless Communications*, vol. 8, no. 4, pp. 1726–1732, Apr. 2009.

[111] A. S. Avestimehr, S. N. Diggavi, and D. N. C. Tse, “Wireless network information flow: A deterministic approach,” *IEEE Transactions on Information Theory*, vol. 57, no. 4, pp. 1872–1905, Apr. 2011.

[112] D. Vukobratović, C. Khirallah, V. Stanković, and J. Thompson, “Random network coding for multimedia delivery services in LTE/LTE-Advanced,” *IEEE Transactions on Multimedia*, vol. 16, no. 1, pp. 277–282, Jan. 2014.

[113] Y. Lin, B. Liang, and B. Li, “Priority random linear codes in distributed storage systems,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 11, pp. 1653–1667, Nov. 2009.

[114] A. K. Ramasubramonian and J. W. Woods, “Multiple description coding and practical network coding for video multicast,” *IEEE Signal Processing Letters*, vol. 17, no. 3, pp. 265–268, Mar. 2010.

[115] D. Slepian, “A note on two binary signaling alphabets,” *IRE Transactions on Information Theory*, vol. 2, no. 2, pp. 84–86, Jun. 1956.

[116] M. Plotkin, “Binary codes with specified minimum distance,” *IRE Transactions on Information Theory*, vol. 6, no. 4, pp. 445–450, Sep. 1960.

[117] N. J. A. Sloane and D. S. Whitehead, “New family of single-error correcting codes,” *IEEE Transactions on Information Theory*, vol. 16, no. 6, pp. 717–719, Nov. 1970.

[118] N. J. A. Sloane, S. M. Reddy, and C.-L. Chen, “New binary codes,” *IEEE Transactions on Information Theory*, vol. 18, no. 4, pp. 503–510, Jul. 1972.

[119] R. Roth and G. Seroussi, “On generator matrices of MDS codes (corresp.),” *IEEE Transactions on Information Theory*, vol. 31, no. 6, pp. 826–830, Nov. 1985.

[120] Y. Xu and T. Zhang, “Variable shortened-and-punctured Reed-Solomon codes for packet loss protection,” *IEEE Transactions on Broadcasting*, vol. 48, no. 3, pp. 237–245, Sep. 2002.

[121] J. Lacan and J. Fimes, “Systematic MDS erasure codes based on Vandermonde matrices,” *IEEE Communications Letters*, vol. 8, no. 9, pp. 570–572, Sep. 2004.

[122] D. J. J. Versfeld, J. N. Ridley, H. C. Ferreira, and A. S. J. Helberg, “On systematic generator matices for Reed-Solomon codes,” *IEEE Transactions on Information Theory*, vol. 56, no. 6, pp. 2549–2550, Jun. 2010.

[123] W. J. van Gils, “On linear unequal error protection codes,” Master’s thesis, Eindhoven University of Technology, Jul. 1982.

[124] R. H. Morelos-Zaragoza, “Multi-level error correcting codes,” Ph.D. dissertation, University of Hawaii, May 1992.

[125] T. Ho, “Networking from a network coding perspective,” Ph.D. dissertation, Massachusetts Institute of Technology, May 2004.

[126] Y. Wu, "Network coding for multicasting," Ph.D. dissertation, Princeton University, Jan. 2006.

[127] D. S. Lun, "Efficient operation of coded packet networks," Ph.D. dissertation, Massachusetts Institute of Technology, Jun. 2006.

[128] N. Axvig, "Applications of linear programming to coding theory," Ph.D. dissertation, University of Nebraska, Aug. 2010.

[129] W. W. Peterson and E. J. Weldon, Jr., *Error-correcting codes*, 2nd ed. Cambridge, MA: MIT press, 1972.

[130] M. Y. Rhee, *Error-correcting coding theory.* McGraw-Hill, Inc., 1965.

[131] S. B. Wicker, *Error control systems for digital communication and storage.* Upper Saddle River, NJ: Prentice Hall, 1995.

[132] M. Bossert, *Channel coding for telecommunications.* New York, NY: John Wiley & Sons, Inc., 1999.

[133] R. E. Blahut, *Algebraic codes for data transmission.* New York, NY: Cambridge university press, 2003.

[134] W. C. Huffman and V. Pless, *Fundamentals of error-correcting codes.* New York, NY: Cambridge university press, 2003.

[135] R. H. Morelos-Zaragoza, *The art of error correcting coding*, 2nd ed. Hoboken, NJ: John Wiley & Sons, Inc., 2006.

[136] R. M. Roth, *Introduction to coding theory.* New York, NY: Cambridge university press, 2006.

[137] T. M. Cover and J. A. Thomas, *Elements of information theory*, 2nd ed. Hoboken, NJ: John Wiley & Sons, Inc., 2006.

[138] T. Ho and D. S. Lun, *Network coding: an introduction.* New York, NY: Cambridge univerisy press, 2008.

[139] R. W. Yeung, *Information theory and network coding.* New York, NY: Springer, 2008.

[140] M. C. Ferris, O. L. Magasarian, and S. J. Wright, *Linear programming with MATLAB.* Philadelphia, PA: Society for Industrual and Applied Mathematics (SIAM) and Mathematical Programming Society (MPS), 2007.

[141] J. M. Wozencraft and I. M. Jacobs, *Principles of communication engineering.* Prospect Heights, IL: Waveland press, Inc., 1965.

[142] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics*, 2nd ed. Reading, MA: Addison-Wesley, 1994.

[143] D. P. Bertsekas and J. N. Tsitsiklis, *Introduction to probability.* Belmont, MA: Athena Scientific, 2002.

Curriculum Vitae

Wook Jung received his Bachelor of Science in Information and Computer Engineering in 2000 from Ajou University, Republic of Korea. He then pursued graduate studies at George Mason University, Fairfax, VA, where he received his Master of Science in Computer Engineering in 2003. His main research interests lie in efficient error control coding for improving network performance. More generally, his interests lie in both theoretical and practical aspects of communication networks.