

國立交通大學

網路工程研究所

碩 士 論 文

適用於可調式視訊編碼的多點傳播技術上的無比率非均
等抹除碼之編碼技術

Rateless Unequal Erasure Protection Coding Techniques Using on
H.264/SVC Video Multicasting

研 究 生： 沈子晉

指導教授： 邵家健 教授

共同指導教授：王忠炫 教授

中 華 民 國 九 十 九 年 七 月

適用於可調式視訊編碼的多點傳播技術上的無比率非均等抹除碼之編碼技術

Rateless Unequal Erasure Protection Coding Techniques Using on H.264/SVC Video
Multicasting

研究生：沈子晉

Student : Tzu-Ching, Shen

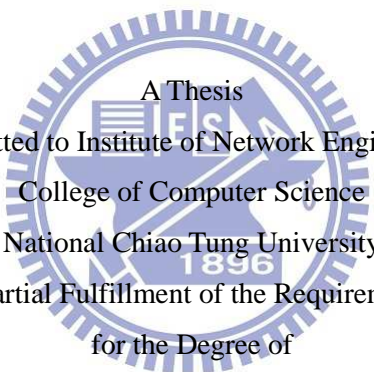
指導教授：邵家健

Advisor : John Kar-Kin Zao

共同指導教授：王忠炫

Co-Advisor : C-H, Wang

國立交通大學
網路工程研究所
碩士論文



A Thesis
Submitted to Institute of Network Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in
Computer Science

July 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年七月

適用於可調式視訊編碼的多點傳播技術上的無比率非均 等抹除碼之編碼技術

學生：沈子晉

指導教授：邵家健 博士

共同指導教授：王忠炫 博士

國立交通大學網路工程研究所 碩士班



我們的目的是在於建立一套用在可調式視訊編碼的多點傳播技術上的編碼機制，因為目前的研究大多沒有提供具體的方法，說明如何實作出一套極短長度無比例特性之非均等抹除碼的編碼系統，並且具有 (一)極佳的非均等保護能力以及良好的改錯能力 (二) 可調整不同層次保護程度的能力 (三) 可調整不同保護能力所佔有的訊息量比例 (四) 可以實際應用在可調式視訊編碼的多點傳輸上。

我們的編碼機制的組成包含了一個非均等前置的線性編碼器，一個交錯/多工器、和一個非均等無比例後置編碼器，並且清楚地描述如何最佳化以及各部分的參數要如何調整搭配來達成我們的目標。

Rateless Unequal Erasure Protection Coding Techniques Using on H.264/SVC Video Multicasting

Student: Tzu-Ching Shen

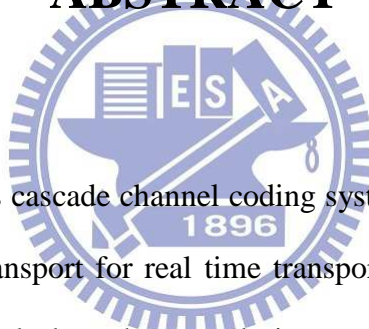
Advisor: Dr. John Kar-Kin Zao

Co-Advisor: Dr. C.-H. Wang

Institute of Network Engineering

National Chiao Tung University

ABSTRACT



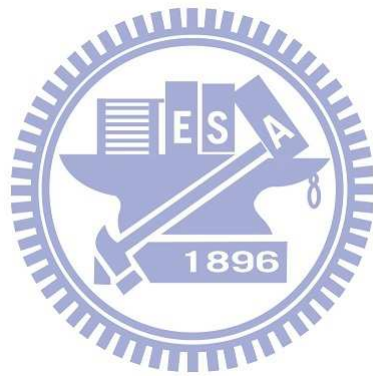
In this thesis, we propose a rateless cascade channel coding system with Unequal Erasure Protection capability using on H.264/SVC transport for real time transport because so far most of researches does not provide a specific method about how to design a short-length rateless unequal erasure protection code which is (1) with good UEP capability and graceful degradation (2) each UEP layer size is adjustable (3) protection capability for each UEP layer is adjustable (4) the coding system is practical to use on a real H.264/SVC transport.

The coding scheme we provide in this thesis is composed by an UEP Precoder, Interleaver/Multiplexer, and an UEP rateless Postcoder, and we also describe how to design, adjust parameters to optimize performance of them.

Content

摘要.....	i
ABSTRACT.....	ii
Chapter 1. Introduction.....	1
1.1 Research Objectives.....	1
1.2 Research Approach.....	1
1.3 Thesis Outlines.....	2
Chapter 2. Technical Background.....	3
2.1 Short-Length Rateless Erasure Correction Codes.....	3
2.1.1 Digital Fountain Codes.....	3
2.1.2 Short-Length LT Codes.....	4
Chapter 3. Design Strategy.....	7
3.1 Linear UEP Precoder.....	7
3.2 Rateless UEP Postcoder.....	8
3.3 Interleaver and Multiplexer.....	9
Chapter 4. Optimization of Short-Length LT Codes Degree Distribution using Evolutionary Strategies.....	12
4.1 LT Degree Distribution.....	12
4.2 Optimization of Short Length LT Degree Distribution.....	13
4.2.1 Algorithm Design.....	14
4.2.2 Optimization Attributes and Parameter Adjustment.....	14
4.2.3 Preliminary Results.....	15
4.2.4 Observation.....	20
Chapter 5. Design of Interleaver and Multiplexer.....	21
5.1 Multiplexer Design.....	21
5.1.1 Decoding Failure Probability of Linear UEP Block Precoder.....	21
5.1.2 Error Bounds for Viterbi Decoding.....	28
5.1.3 Realization.....	29
5.2 Design of Interleaver.....	31
5.2.1 Good Properties for Convolutional Decoding Process.....	31
5.2.2 Objective Function for Interleaver Design.....	34
5.2.3 Realization/Application.....	36
Chapter 6. Realization of Rateless Cascaded UEP Codec.....	44
6.1 Convolutional Precoder + Reed-Solomon Postcoder.....	44
6.2 Convolutional Codes + Equal Erasure Protected LT Codes.....	50
6.3 Convolutional Codes + Unequal Erasure Protected LT Codes.....	53
Chapter 7. Conclusions.....	55

7.1 Accomplishment.....55
7.2 Future Work.....55



List of Figures

Figure. 1 : System Flow Chart.....	1
Figure. 2 : Relations between source and output symbols of a Digital Fountain Code.....	3
Figure. 3: BP Decoding Process	4
Figure. 4 : Overall System Structure	7
Figure. 5 : LT decoding result.....	9
Figure. 6 : A simple example for Interleaver	10
Figure. 7 : Histogram of LT decoding performance with different rateless overhead	13
Figure. 8 : Average failure rate to different rateless overhead.....	13
Figure. 9 : Decoding performance for No.1 degree distribution	16
Figure. 10 : Decoding performance for No.2 degree distribution	17
Figure. 11 : Decoding performance for No.3 degree distribution.....	18
Figure. 12 : Decoding performance for No.4 degree distribution	19
Figure. 13 : Comparison between four degree distributions.....	20
Figure. 14 : Minimum Weight of Fundamental Path in Viterbi Decoding Process	32
Figure. 15 : Error Distances in Postcoder Output Sequence.....	32
Figure. 16 : Sliding Window for Error Ratio Estimation.....	33
Figure. 17 : Error Ratio before interleaving	33
Figure. 18 : Error Ratio before interleaving	34
Figure. 19 : Error Ratio boundary for different UEP layer.....	36
Figure. 20 : Error distribution for Interleaver size 100Kbit.....	38
Figure. 21 : Bit error rate for Interleaver size 100Kbit.....	39
Figure. 22 : Error distribution for Interleaver size 50Kbit.....	40
Figure. 23 : Bit error rate for Interleaver size 50Kbit.....	41
Figure. 24 : Error distribution for Interleaver size 10Kbit.....	42
Figure. 25 : Bit error rate for Interleaver size 10Kbit.....	43
Figure. 26 : System flow chart for UEP Conv. PreCoder cascaded with Reed-Solomon Coder	44
Figure. 27 : (10, 1)-Block interleaving to Reed-Solomon Coder	44
Figure. 28 : Allocation for primary channel and secondary channel	45
Figure. 29 : Comparison of Bit Error Rate among UEP Codes, football.....	46
Figure. 30 : Comparison of Bit Error Rate among UEP Codes, foreman.....	46
Figure. 31 : Latency-Distortion Tradeoffs of UEP Codes, football.....	47
Figure. 32 : Latency-Distortion Tradeoffs of UEP Codes, foreman	47
Figure. 33 : Comparison between Max. Latency UEP and RS Codes, football	48
Figure. 34 : Comparison between Max. Latency UEP and RS Codes, foreman	48

Figure. 35 : Comparison between Min. Latency UEP and RS Codes, football49

Figure. 36 : Comparison between Min. Latency UEP and RS Codes, foreman49

Figure. 37 : System flow chart for UEP Conv. Precoder cascaded with LT rateless coder50

Figure. 38 : Block interleaver used in the Conv. UEP Precoder cascaded with LT rateless coder system50

Figure. 39 : Latency-Distortion Tradeoffs of LT Codes, football51

Figure. 40 : Latency-Distortion Tradeoffs of LT Codes, foreman51

Figure. 41 : Comparison between Max. Latency LT code and RS Codes, football.....52

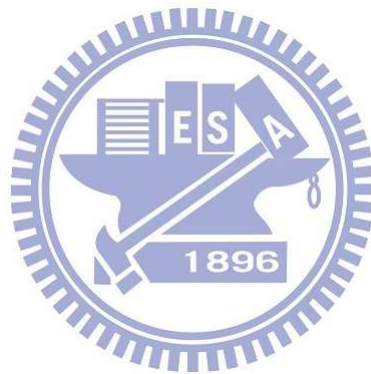
Figure. 42 : Comparison between Max. Latency LT code and RS Codes, foreman.....52

Figure. 43 : Decoding flow chart for UEP Precoder + UEP LT Postcoder.....53

Figure. 44 : UEP Precoder + UEP LT Postcoder with interleaver size 100Kb53

Figure. 45 : UEP Precoder + UEP LT Postcoder with interleaver size 50Kb54

Figure. 46 : UEP Precoder + UEP LT Postcoder with interleaver size 10Kb54



List of Tables

Table. 1 Comparison between different objective function	15
Table. 2 : Parameters in Viterbi decoding process	35
Table. 3 : Parameters used in block interleaver design.....	36
Table. 4 : Parameters used in S-random interleaver design.....	37



Chapter 1. Introduction

1.1 Research Objectives

In this thesis, we propose a rateless cascade channel coding system with Unequal Erasure Protection capability using on H.264/SVC transport for real time transport. Scalable Extension of the H.264-AVC standard (commonly known as H.264-SVC or simply SVC) [1,2,3] supports *spatial*, *temporal* and *fidelity scalability* by organizing a video bit stream into *scalable layers* that are related through adaptive inter-layer predictions and hierarchical references. This multi-layer-structure of SVC bit streams enables different types of viewing devices to extract and decode different parts of the bit streams. In video streaming applications, H.264-SVC is often used to provide two useful services:

- ❖ *Heterogeneous Multicast*, in which coarse grain scalable (CGS) layers of an SVC bit stream are funneled to viewing devices according to their display formats, processing power and network connectivity. This transport mechanism can incur dramatic saving of communication bandwidth when it is used to serve large clusters of viewing devices.
- ❖ *Graceful Degradation*, in which viewing devices are enabled to decode uncorrupted *medium grain scalable* (MGS) layers of an SVC bit stream they receive through unreliable communication. This fault tolerant mechanism can drastically improve video quality when it is deployed over wireless or peer-to-peer networks.

The Development of a *short-length rateless unequal erasure protection (UEP) code* that is designed to maximize graceful degradation effects of SVC multicasting over wireless LANs/MANs. Unlike existing researches, we provide a UEP rateless coding scheme which (1) with good UEP capability and graceful degradation (2) each UEP layer size is adjustable (3) protection capability for each UEP layer is adjustable (4) the coding system is practical to use on a real H.264/SVC transport.

1.2 Research Approach

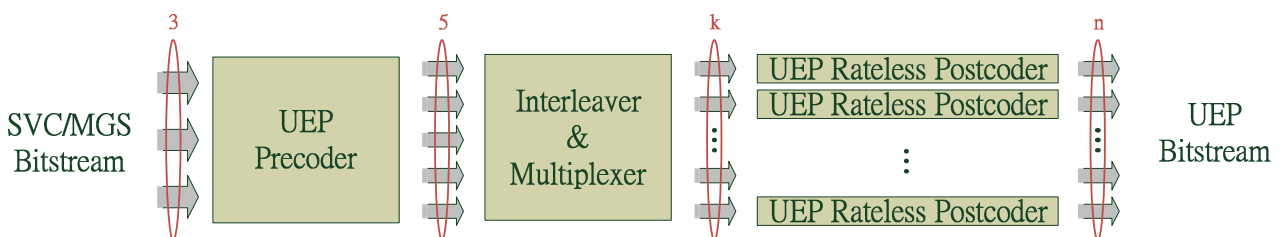


Figure. 1 : System Flow Chart

To achieve our goals, we apply a system which is composed by an UEP Precoder, Interleaver/Multiplexer, and an UEP rateless Postcoder. We chose to employ *unequal error/erasure protection* and *probabilistic rateless coding* techniques because of the following reasons those are related to needs of SVC multicasting,

- ❖ Channel and network codes with unequal erasure protection (UEP) capability are best suited for helping SVC bit streams to achieve rate-distortion optimized performance because the dependence relations existing among scalable layers cause the playback video quality to depend unevenly on successful reception of different layers — generally, the base layer and other low-level layers are more essential for good quality playback than the upper layers.
- ❖ Rateless erasure correction codes (also known as Digital Fountain Codes [4,5]) are best suited for implementing tradeoffs between network throughput and transport latency for robust and good quality video streaming. This is because Digital Fountain Codes can produce variable amount of randomly coded symbols over time and allow the receivers to decode more source symbols simply by capturing more of the coded symbols.

In this thesis, challenges would be separate into two different parts, one is finding suitable codes for each part of our coders, and the second is optimization and match up each block in the coding system. Following chapters will elaborate on details about designs for each block.

1.3 Thesis Outlines

Chapter 2 summarizes some fundamental background which is needed to comprehend the thesis, including Digital Fountain Codes, and short length LT codes. Chapter 3 describes the role of our channel coding system of the whole H.264/SVC system. Chapter 4 provides a novel idea about optimizing LT codes, in Chapter 5, we gives a detailed analysis about how the erasure probability of output codeword sequences effects the UEP ability of input information sequence, and how to design an interleaver coupled with Precoder and Postcoder. Chapter 6 shows our experiment results about channel coding system design according to principles from previous chapters. And finally, and in the final chapter, we conclude and envision.

Chapter 2. Technical Background

2.1 Short-Length Rateless Erasure Correction Codes

2.1.1 Digital Fountain Codes

Digital Fountain Codes are probabilistic erasure correction codes that allow an encoder to derived practically infinite amount of randomized output symbols from a fixed block of source symbols. They then permit the decoders to improving their successful decoding probability simply by capturing more output symbols. These codes are regarded as rateless codes because the ratios between the amount of source codes and output codes are not fixed rather they are determined by the decoding process. The encoding process of Digital Fountain Codes usually generates an output symbol by adding a random collection of source symbols over the finite field of the code. Both the number of source symbols (known as the degree distribution) and the selection of source symbols (expressed as the connectivity matrix) are determined by a pseudo-random process. On the other hand, the decoding process usually uses a belief propagation algorithm to recover the source symbols from the captured symbols. Success of decoding is not guaranteed; however, its probability can be enhanced simply by capturing more output symbols. Current, there are several implementations of Digital Fountain Codes including Luby Transform (LT) codes, Tornado codes and Raptor codes. [Figure. 2 : Relations between source and output symbols of a Digital Fountain Code] shows a systematic code that produces n output symbols from k source symbols by adding l redundant symbols to the source symbols.

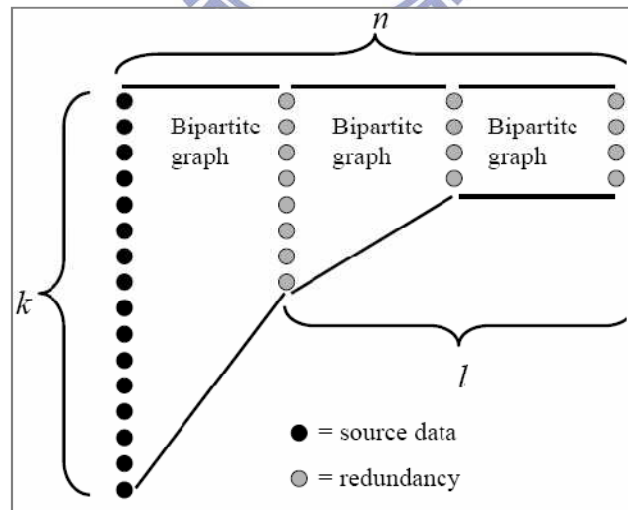


Figure. 2 : Relations between source and output symbols of a Digital Fountain Code

2.1.2 Short-Length LT Codes

A Luby Transform (LT) code is completely defined by the number of source symbols K and its degree distribution $P(d)$. The degree d of each output symbol is an arbitrary integer between 1 to K . Every block of source symbols has s symbols.

(A) Rateless LT Codes

The LT encoding process of an output symbol C_i can be divided into three steps:

1. Choose the degree d of an output symbol C_i according to the degree distribution.
2. Choose randomly and uniformly d distinct source symbols as the neighbors of output symbol C_i . Connections between output symbol C_i and its neighbors are also generated.
3. Perform addition on the values of all d source symbols as the value of output symbol C_i .

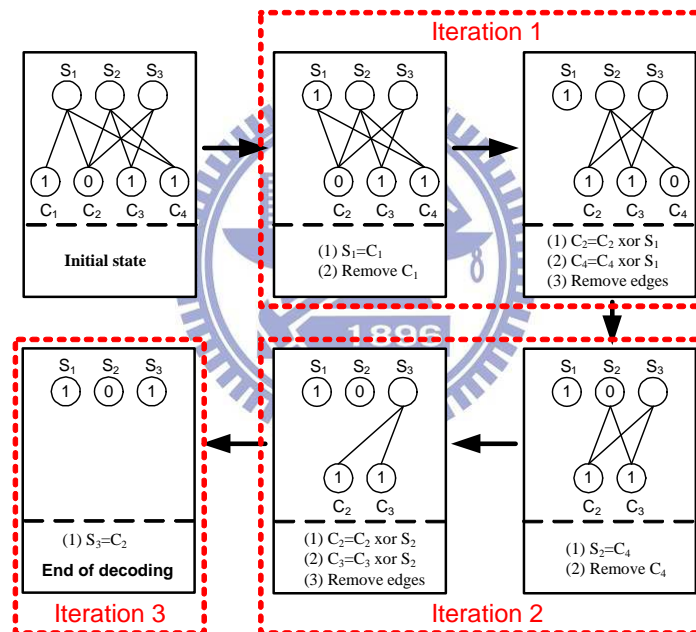


Figure. 3: BP Decoding Process

An encoder operates step 1 through step 3 iteratively to generate each codeword. A simple LT encoding example of the output symbol C_1 with $d_1=2$ is depicted as [Figure. 3: BP Decoding Process]. After two source symbols S_1 and S_3 are chosen uniformly and randomly as neighbors of C_1 , the value of C_1 is generated by performing addition on the values of S_1 and S_3 . The generation of the first output symbol C_1 is completed and C_1 is sent to receivers.

(B) Degree Distribution

The degree distribution is derived according to the released probability of an output symbol in BP decoding procedure. When K is fixed, the degree distributions of LT codes mainly influence not only the recovery probability of source symbols but also the complexities of encoding and decoding. Thus, a good degree distribution should have these properties:

1. High probability of recovering source symbols can be achieved with as few output symbols as possible.
2. Average connection number of source symbols should be as low as possible.

The first property is to minimize transmission bandwidth while the second property is to minimize complexities of encoding and decoding. Based on these two properties, a mathematical degree distribution called Soliton distribution Soliton (d) is derived. Notice that the probability of degree one in a Soliton distribution is $1/K$ that only one output symbol with degree one is generated in average. All output symbols with degree one can probably be erased during transmission. This follows that BP decoding fails to start. As a result, the robust Soliton distribution is developed for practical usage and it is described as follows:

$$S \equiv c \log_e \left(\frac{K}{\delta} \right) \sqrt{K} \quad (1)$$

$$P_{robust}(d) = \frac{P_{Soliton}(d) + \tau(d)}{Z} \quad (3)$$

$$\tau(d) = \begin{cases} \frac{S}{K} \frac{1}{d} & \text{for } d = 1, 2, \dots, \frac{K-S}{S} \\ \frac{S}{K} \log\left(\frac{S}{\delta}\right) & \text{for } d = \frac{K}{S} \\ 0 & \text{for } d > \frac{K}{S} \end{cases} \quad (2)$$

The extra parameters c and δ are an arbitrary parameter and a bound of decoding failure probability respectively. In comparison with Soliton distribution, the robust Soliton distribution has higher probability of degree one to ensure that BP decoding succeeds with high probability.

(C) Belief Propagation (BP) Decoding

After a receiver collects sufficient output symbols, belief propagation (BP) decoding which operates iteratively is able to start, every iteration can be divided into two steps:

1. Find output symbols with degree one. Assign their value to their neighboring source symbols and these source symbols are decoded. Remove output symbols with degree one.

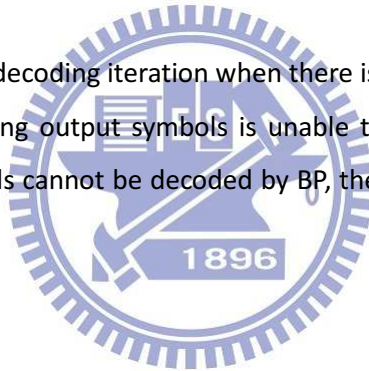
2. Perform addition on each remaining output symbols with their neighbors which are already decoded.
Remove these neighbors and connections from each codeword.

A receiver repeats steps 1 and 2 iteratively until source symbols are all decoded or there are no output symbols with degree one. A simple LT decoding example is illustrated in 0. Initially, four output symbols [C1, C2, C3, and C4] = [1, 0, 1, 1] are received to recover source symbols S1~S3 as follows:

1. C1 value is assigned to S1; C1 is removed and S1 is decoded. Addition is performed on C2 and C4 with S1 respectively. S1 is removed from the neighbors of C2 and C4.
2. C4 value is assigned to S2; C4 is removed and S2 is decoded. Addition is performed on C2 and C3 with S2 respectively. S2 is removed from the neighbors of C2 and C3.
3. C2 value is assigned to S3. C2 and C3 are removed. S3 is decoded and the BP decoding is completed.

(D) Drawbacks

BP decoding may stop at an arbitrary decoding iteration when there is no degree-one output symbol left. The information contained in the remaining output symbols is unable to be exploited by BP algorithm. In the cases that the received output symbols cannot be decoded by BP, the remaining output symbols will mean a waste of transmission bandwidth.



Chapter 3. Design Strategy

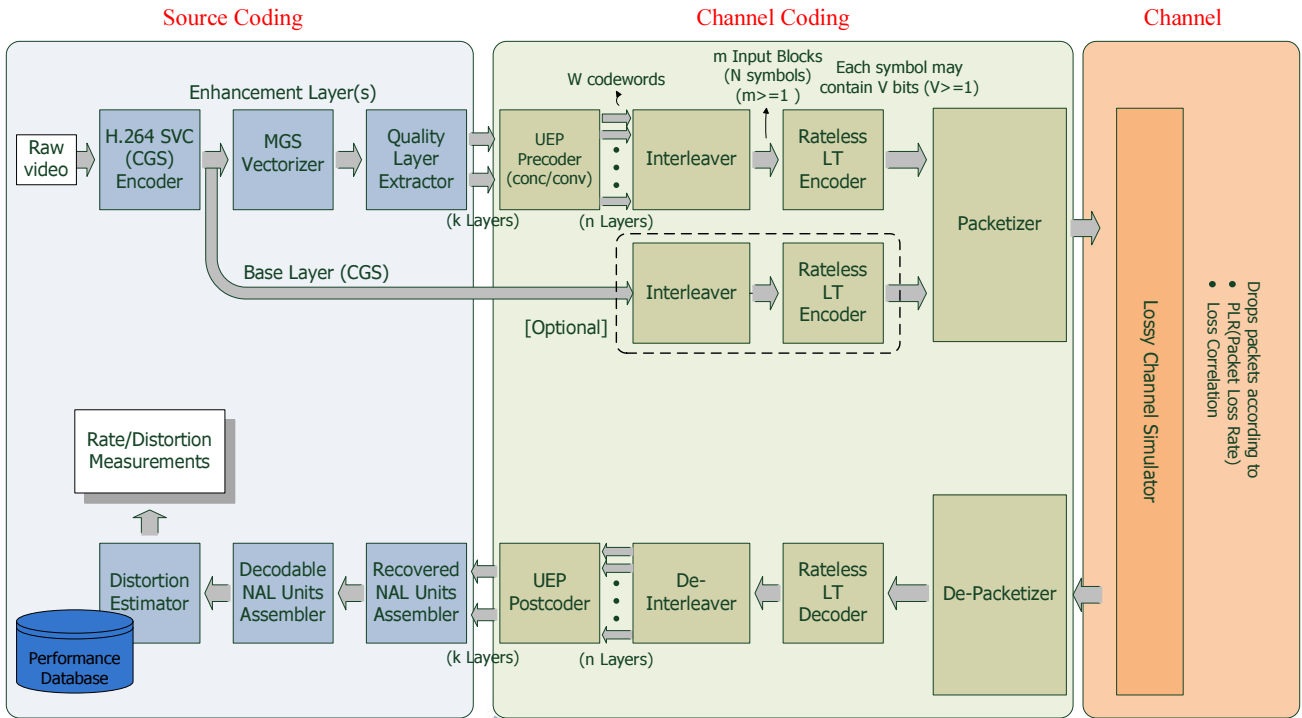
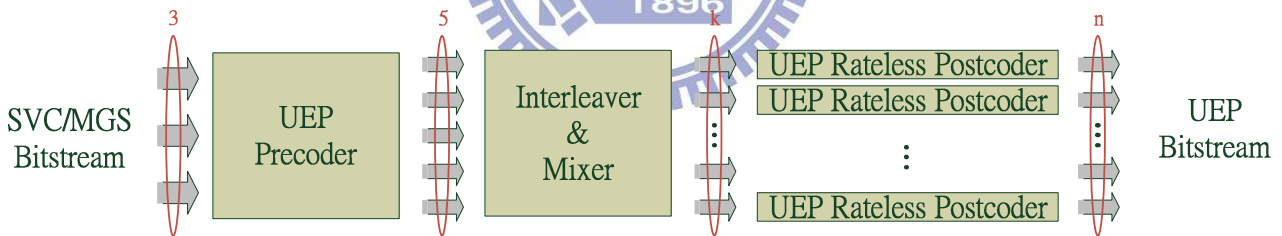


Figure 4 : Overall System Structure



3.1 Linear UEP Precoder

In our system, the main objective for Precoder is providing unequal erasure protection ability, and there are two different classes of codes can be candidates, block codes and convolutional codes. This section we will describe differences between block codes and convolutional codes, and summarize which of them will be the better choice for our system design.


We will only operate our convolutional codes on the $GF(2)$, its decoding complexity is lower than other $GF(p^m)$. Unlike convolutional codes, we observe that for linear block codes, information size for different protection layer is quite differently, it means if we chose linear block codes to be our Precoder design, we have to separate our SVC input streams into different protection layers with different size, and

higher protection ability with lower ratio of all information size. Contrary to the linear block codes, convolutional codes can provide different protection layers with the same size.

In addition, convolution code does not restrict its codeword length but only ratio of input to output sequence length, which means that it is more flexible to couple with codeword size of rateless code. On the other hand, block size of a linear block codes is fixed by given a (n,k) -concatenation linear block code. The constraint made the design of interleaver much harder.

Error correction ability to convolution codes is positive correlated to memory size. The more memories used in convolutional codes, usually the higher recover ability it has. At the mean time, the decoding complexity for convolutional codes grows exponentially. That is the reason why the average separation value of convolution code is lower than the linear block codes because the limitation of memory used, and the average code rate of the convolution code is higher than the linear block code.

For equally protection layer size to couple with source coding system design, we intend to use convolutional codes for our Precoder, and we choose convolutional codes for following experiments according to coding tables form [6], which provides good performance with UEP ability. [Table. 1 : Parameters for convolutional codes in our system design]



n	k	m	Canonical PGM	Forney Indices	Separation Vector
5	3	7	$\begin{bmatrix} 3 & 2 & 1 & 0 & 0 \\ 0 & 7 & 4 & 2 & 1 \\ 11 & 35 & 20 & 25 & 30 \end{bmatrix}$	1 2 4	4 6 10

Table. 1 : Parameters for convolutional codes in our system design

3.2 Rateless UEP Postcoder

The performance of Rateless Coder is determined by two factors,

1. Rx Symbol Inflation Ratio with respect to Rateless Encoder Input.
2. Design of degree distribution in Rateless Code.

If we define Δ_T to be Rx Symbol Inflation Ratio with respect to Rateless Encoder Input, which is rateless overhead, and $\xi(\Delta_T)$ be the failure probability of Rateless code as Rx Symbol Inflation Ratio with respect to Rateless Encoder Input. By now we know that $S_{l,i=[1...K]}$ can be decoded iff

$$\xi(\Delta_T) \cdot n \leq (\sigma_1 - 1)$$

Hence, probability of successful decoding of S_1 is

$$p_1(\Delta_T) = P \left[\xi(\Delta_T) = \frac{(\sigma_1 - 1)}{n} \right] = \int_0^{\frac{(\sigma_1 - 1)}{n}} p(\xi) d\xi$$

If $\xi(\Delta_T)$ has negligible variance, then

$$p_1(\Delta_T) \begin{cases} 1 & \text{iff } \frac{(\sigma_1 - 1)}{n} \leq \overline{\xi(\Delta_T)} \\ 0 & \text{otherwise} \end{cases}$$

The curve of the failure rate has usually shape similar to the one in [Figure. 5 : LT decoding result], x-axis represents the value of Δ_T and y-axis represents the failure rate.

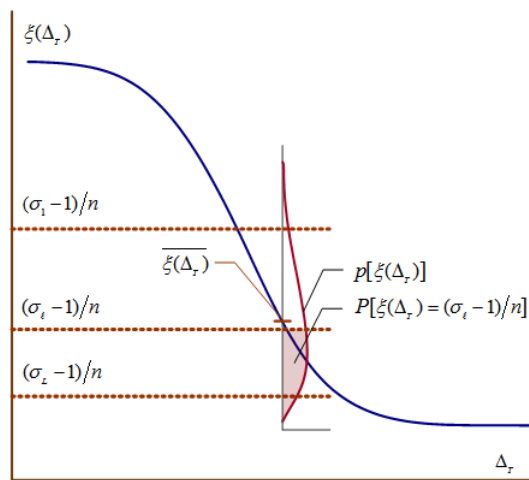


Figure. 5 : LT decoding result

In our experiments, we observed that standard deviation of decoding failure rate for waterfall region is quite large, which implies the worst case of decoding result may be very terrible, we call that bimodal feature for decoding results. In the [Figure. 5 : LT decoding result], three dotted horizontal lines are cordons for decoding ability with respect to each UEP layer in the Precoder. Once error probability above the line, errors may not be recovered in the specific UEP layer, hence our design criteria for rateless coder is trying to find some degree distributions with trifling bimodal feature or lower failure rate to minimize the area above cordons.

3.3 Interleaver and Multiplexer

In our system, the interleaver is a bridge between the Precoder and the Postcoder, the main objective of using interleaver is trying to maintain the UEP features after cascaded two coders, interleaver is a permutation table relocating erasure indices and avoiding burst erasure event. In the decoding process, the statistical properties of rate-less Postcoder is not ergodic because the erasure probability of each input vector

is highly correlated then causes burst error events which Precoder cannot afford. Fortunately, our ensemble process of Postcoder are uncorrelated, it implies that we can design an interleaver with properties which are suitable for our Precoder.

Also, to minimize our decoding time procrastination and coupled with parameters in the source coding process, the interleaver has to adjust its size to one GOP, the minimal decodable unit of Scalable Video Coding, which means

$$InterleaverSize = \frac{1}{R} \cdot \max\{GOPSize\}$$

R is the code rate of Precoder.

Example

The first approach of our interleaver performs permutation to relocate the index of input symbol sequence and avoiding burst erasure event, we apply a classic block interleaver: the input data is written along columns and read along rows. The design of our block interleaver is different from traditional one since our Postcoder has UEP ability, and we have to keep this feature after interleaving.

An intuitive design is separating interleaver into several regions corresponded to our UEP layer counts of Postcoder, each region performs indices relocation individually. In other words, the specific region contains all indices from the same UEP layer of Postcoder.

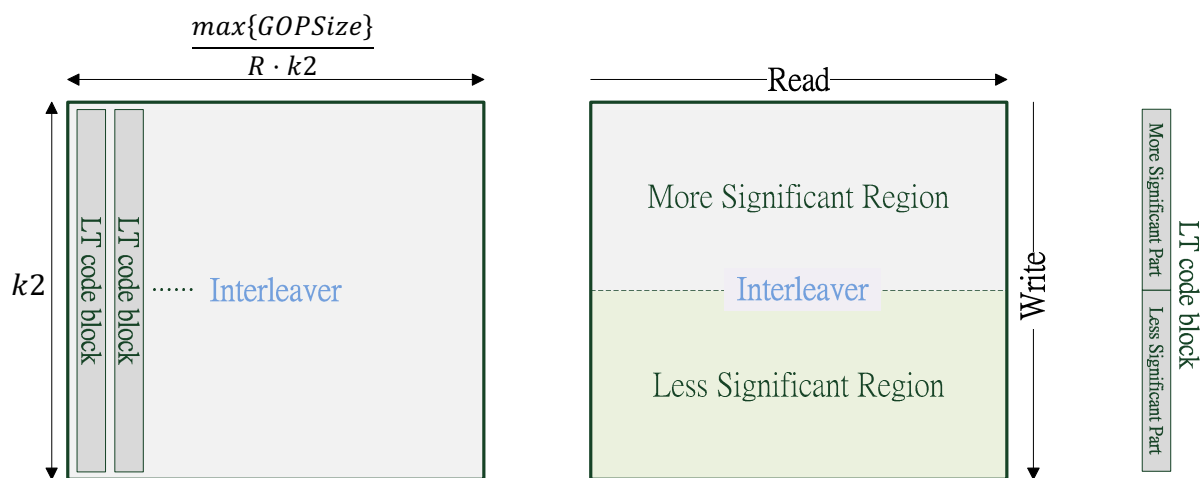


Figure. 6 : A simple example for Interleaver

[Figure. 6 : A simple example for Interleaver] gives an example that rate-less Postcoder has two different protection ability parts, the more significant part has lower average erasure probability compare with the less significant part. To keep this feature after interleaving, we also separate the interleaver into two different regions, called more significant region and the less significant region with respect to the Postcoder.



Chapter 4. Optimization of Short-Length LT Codes Degree Distribution using Evolutionary Strategies

4.1 *LT Degree Distribution*

There are two major reasons we have to do optimization for the LT degree distribution, the first reason is from our observation of experiment results: The burst error event, which exist in short-length LT decoding results because its block length is not infinity. It is a special feature correlated to decoding-case distribution, although the average failure probability is very low, it does not guarantee each of failure probability is very low, the erasure probability is either close to zero or close to the one. When some case with large enough erasure probability, its behavior is similar to burst error. In this iteration, LT decoder equivalently introduces a burst error channel behavior that is harmful for our Postcoder design.

As we know, during the waterfall region, different decoding iteration with the same epsilon value has bimodal recover probability. In other words, standard deviation of erasure probability of the waterfall region are quite large, it forced us to find some better degree distribution to eliminate this phenomenon. The second reason is for a better performance, minimizing the failure rate. The two objectives are equivalent to fine a kind of degree distribution: the lower failure rate (failure probability) at the same epsilon value compared with the old one, with no bimodal decoding results. To optimize the failure rate we apply an evolutionary strategy called CMA-ES (Covariance Matrix Adaptation).

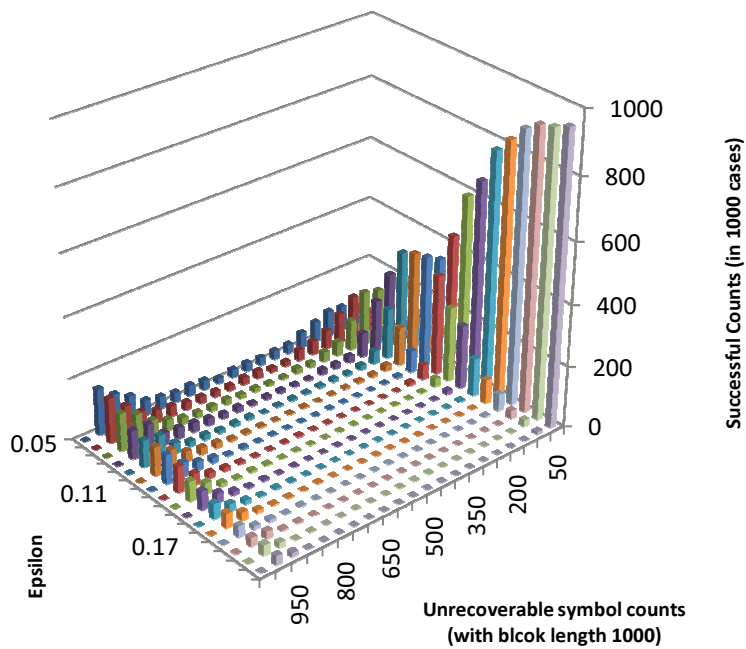


Figure. 7 : Histogram of LT decoding performance with different rateless overhead

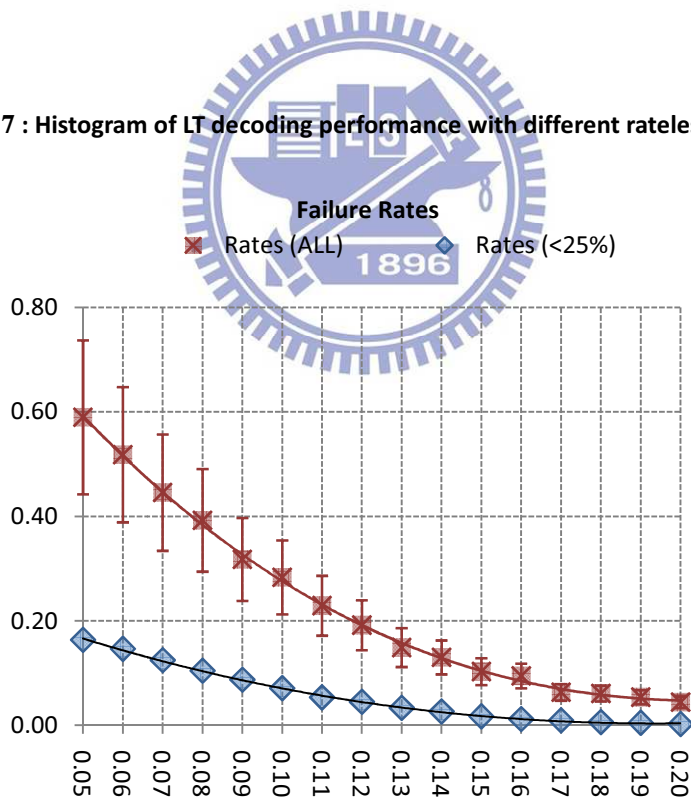


Figure. 8 : Average failure rate to different rateless overhead

4.2 Optimization of Short Length LT Degree Distribution

An important step in the development of Rateless UEP Channel Codes is to design short data length Rateless Erasure Corrections Codes that exhibit good tradeoff between received symbol counts and decoding failure

rates. Since the beginning of 2009, the research team has been trying to design these codes by searching for the optimal degree distribution of short data length LT codes using an evolutionary strategy based on covariance matrix adaptation (CMA-ES).

4.2.1 Algorithm Design

In our initial experiments with the goal to find the optimal degree distribution, the initial LT parameters we set is about two things as following,

- ❖ Tag number for different kind of connection counts
- ❖ Tag value for connected input symbol counts

We set the number of tags equal to 10, actually in the short length LT codes design, this number will be set around 10 to 30 because more tag counts does not give the better performance, another reason for our experiments are the variable counts: For CMA-ES, fewer parameters are easier to be controlled.

The tag values (e.g. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10) represent the number of input symbols connected to one output symbol and were assigned to numbers (mainly primes) between 1 and 200 with dense distribution at the beginning. As the values of tags become higher the tag distribution is becoming more and more spars (according to empirical results this pattern seems to be the most suitable). In all our experiments were LT blocks consisting of 1000 symbols.

4.2.2 Optimization Attributes and Parameter Adjustment

We performed experiments with various optimization criteria and parameter values. Four experiments with the promising results are summarized in the table below.

No	Optimization (minimization) attribute	€																		
1	Failure rate (number of undecoded symbols within block) of LT blocks, which after decoding had at least half of the symbols correct (500 out of 1000).	0.05																		
2	Failure rate (number of undecoded symbols within block) of all the LT blocks.	0.15																		
3	Cumulative failure rate after taking into consideration following weights :	0.05																		
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Failure Rate</th> <th><200</th> <th><200</th> <th><200</th> <th><200</th> <th>>200</th> <th>>200</th> <th>>200</th> <th>>200</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Failure Rate	<200	<200	<200	<200	>200	>200	>200	>200										0.1
Failure Rate	<200	<200	<200	<200	>200	>200	>200	>200												
		0.15																		

	ε	0.05	0.1	0.15	0.2	0.05	0.1	0.15	0.2		0.2
	Weight	1	2	3	4	6	7	8	10		
4	Cumulative failure rate after taking into consideration following weights:										0.05
	Failure Rate	<200	<150	<100	<50	>200	>150	>100	>50		0.1
	ε	0.05	0.1	0.15	0.2	0.05	0.1	0.15	0.2		0.15
	Weight	1	1	1	1	4	4	4	4		0.2

Table. 2 Comparison between different objective function

The first objective function is trying to minimize the failure rate averaged by trails which has more than half of the symbols correct because LT decoding process has the bimodal feature, and the second objective function is trying to minimize the overall failure probability from an intuitive thinking. The last two objective functions are applying the concept of weighting.

4.2.3 Preliminary Results

[Figure. 9 : Decoding performance for No.1 degree distribution] to [Figure. 12 : Decoding performance for No.4 degree distribution] show histograms of four cases mentioned in the previous section.



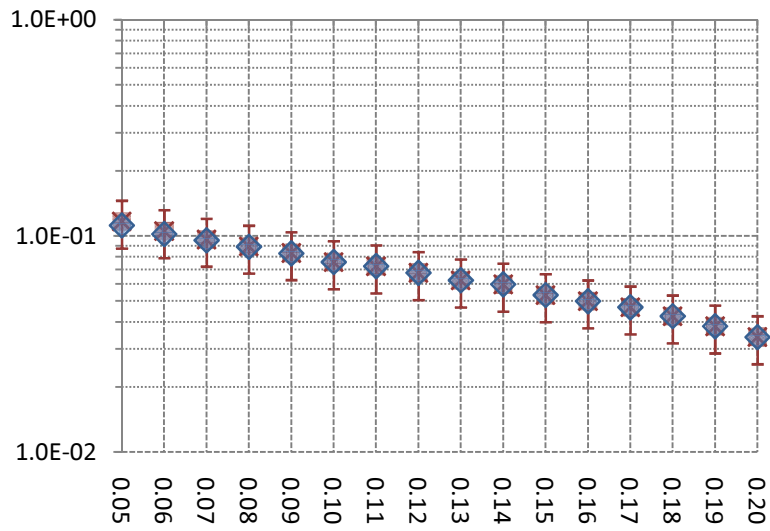
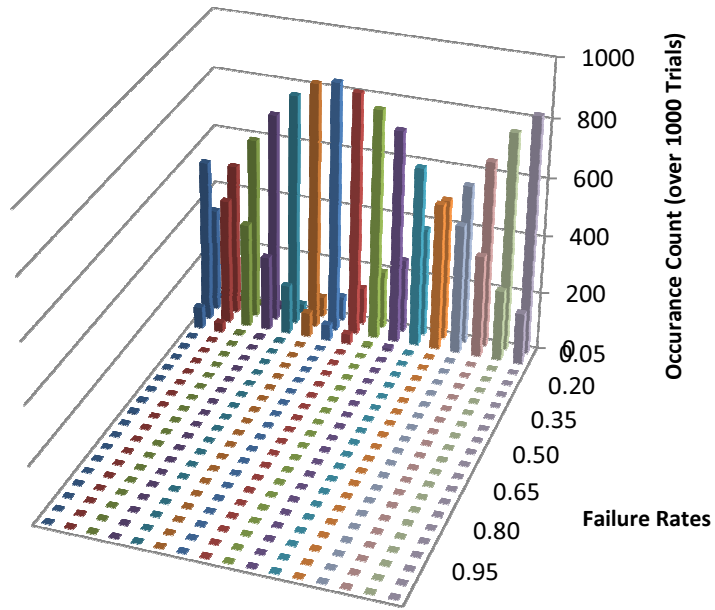


Figure. 9 : Decoding performance for No.1 degree distribution

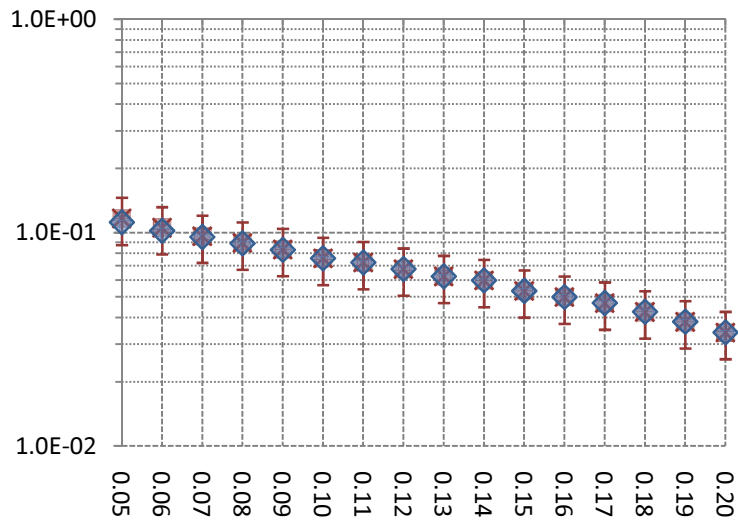
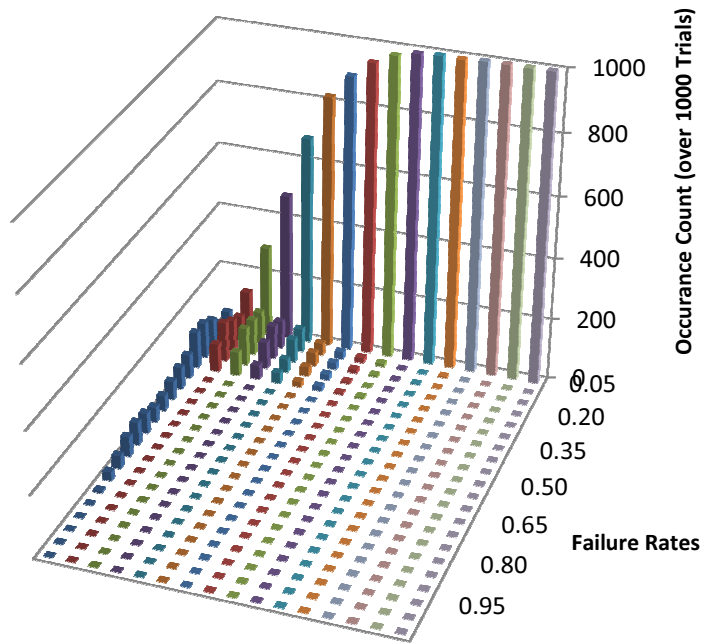


Figure. 10 : Decoding performance for No.2 degree distribution

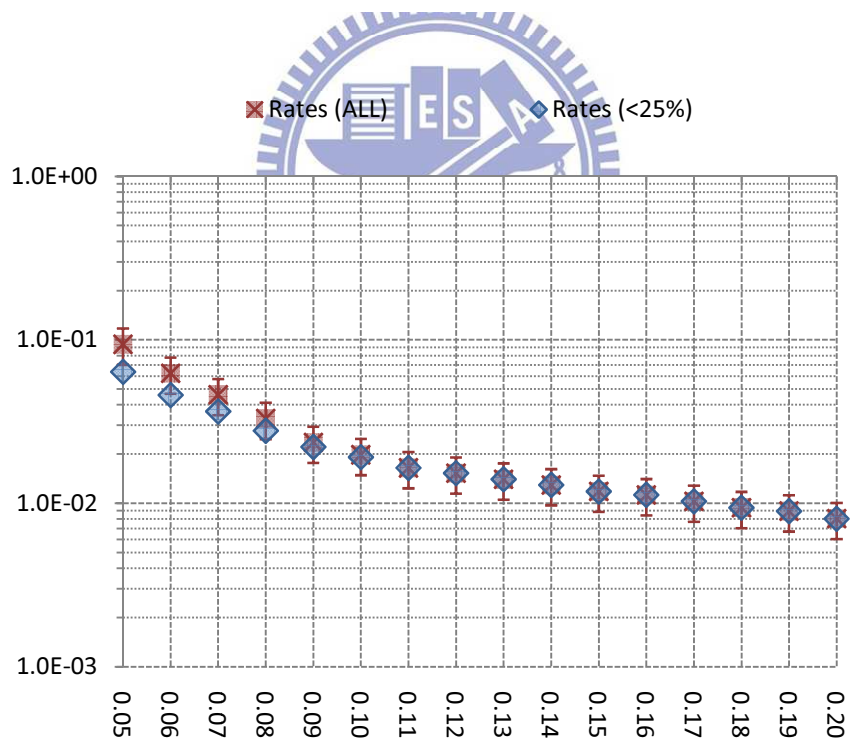
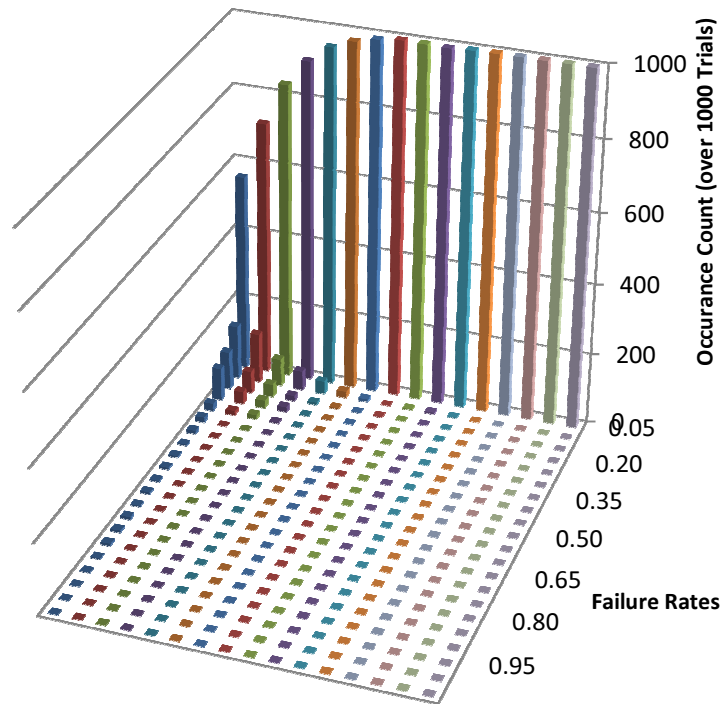
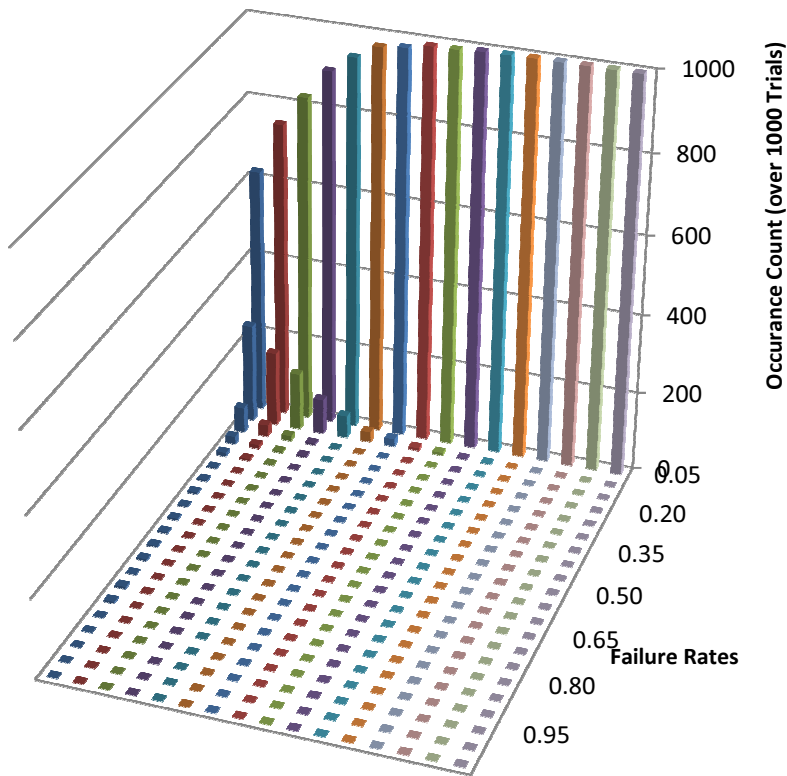


Figure. 11 : Decoding performance for No.3 degree distribution



✕ Rates (ALL)

◆ Rates (<25%)

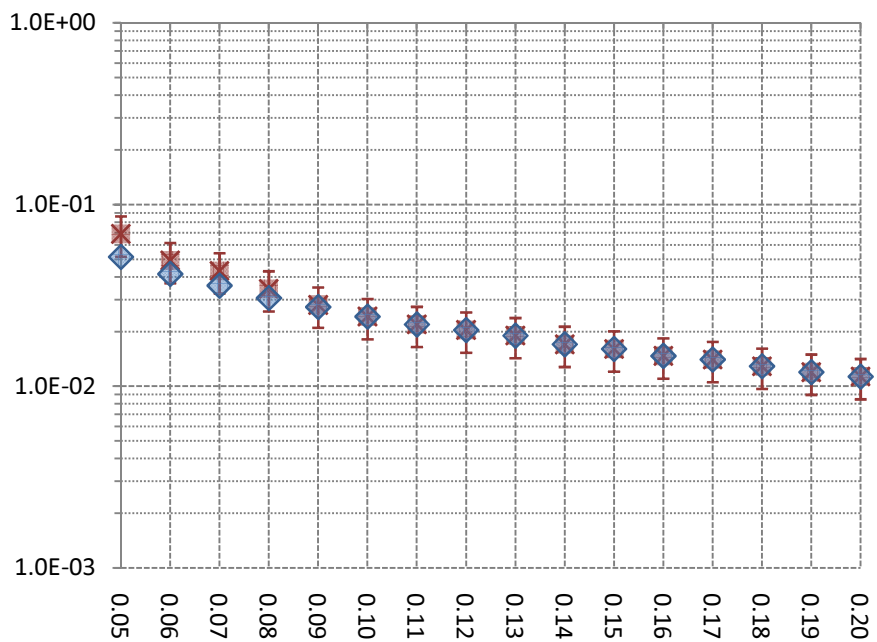


Figure. 12 : Decoding performance for No.4 degree distribution

The degree distributions produced by the four experiments as well as that of the Soliton distribution are shown in [Figure. 13 : Comparison between four degree distributions].

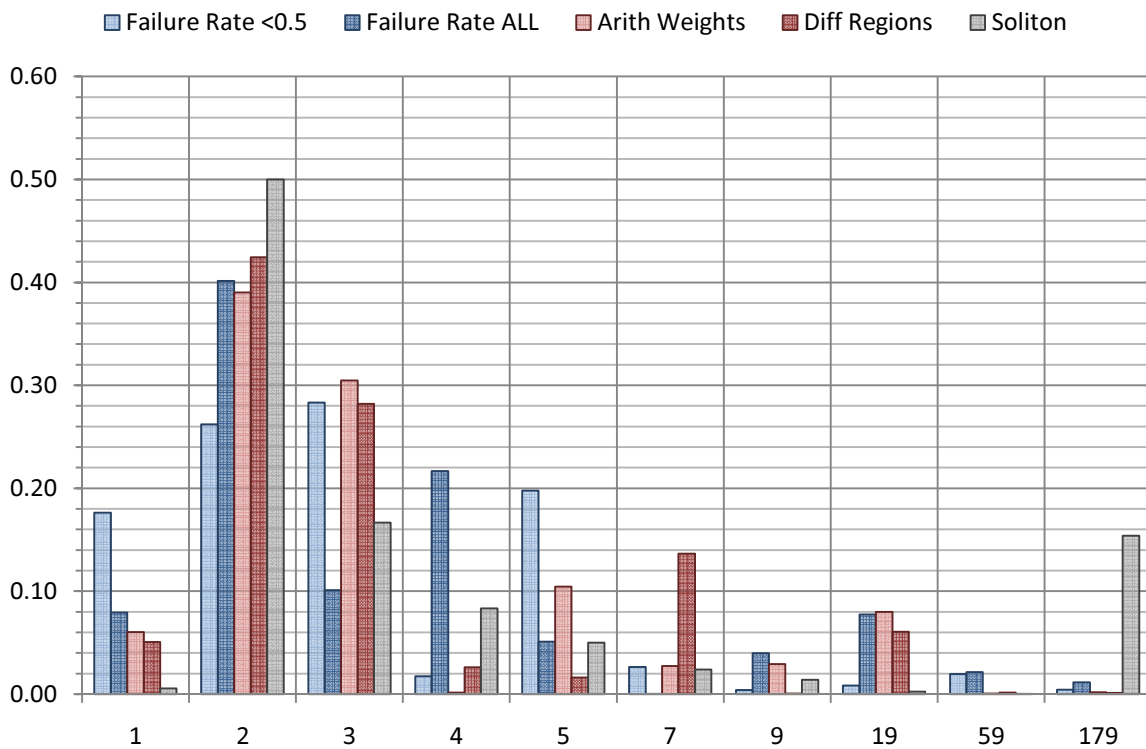


Figure. 13 : Comparison between four degree distributions

4.2.4 Observation

From these results, we can see that the four obtained degree distributions have probability of the second tag higher than probability of the first tag. The second important observation is that almost all the tags with significant probability lie in the interval [1, 19].

We plan to focus on experiments with the degree distributions consisting of more than 10 tags. We also plan to calibrate the weights assignment and elaborate the children selection in the evolutionary part of our algorithm. Since we observed that distributions with the probability of the second tag higher than the probability of the first tag are generating good results, we might want to prefer the children with this feature to be selected as parents for the next generation.

Chapter 5. Design of Interleaver and Multiplexer

5.1 Multiplexer Design

5.1.1 Decoding Failure Probability of Linear UEP Block Precoder

In this chapter we would like to describe how different erasure events caused by channel effect each input symbol's decoding result. We first consider when and which condition causes input symbols recover failures, without taking probabilities of different erasure events into account. The second part of the chapter will give a precise upper bound of erasure probability for each input symbol, and we deduce a special case suitable for our system at the end of this chapter.

(A) Deterministic Analysis

Consider an (n, k) linear block codes consisting of symbols over $GF(p^q)$. Let $\mathbf{s}^{(m)}$ be the input codeword and $\mathbf{c}^{(m)}$ be the output codeword that are generated from $\mathbf{s}^{(m)}$ with the use of G :

$$\mathbf{c}^{(m)} = \mathbf{s}^{(m)} \cdot G.$$

For $0 < m \leq 2^k - 1$, m is an integer.

Also, let $G^{\bar{J}}$ be the degenerated Generator Matrix derived from G after puncturing j columns, and \bar{J} is the collection of indices of punctured columns of G , the cardinality of \bar{J} is $|\bar{J}| = j$.

Lemma 1 [Decoding Failure, Special Case]

The k_{th} Input symbol s_k cannot be recovered if k_{th} row of $G^{\bar{J}}$ is zero (that is, all its entries are zero).

Proof: Let $\mathbf{s}_a = (s_1, s_2, \dots, s_k)$ and $\mathbf{s}_b = (s_1, s_2, \dots, s'_k)$ be two input codewords that differ only at their k_{th} symbol (i. e. $s_k \neq s'_k$). Also, assume $\mathbf{c}_a^{\bar{J}}$ and $\mathbf{c}_b^{\bar{J}}$ be the two corresponded received output codeword generated by

$$G^{\bar{J}} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1(n-j)} \\ g_{21} & g_{22} & \cdots & g_{2(n-j)} \\ \vdots & & & \vdots \\ g_{k1} & g_{k2} & \cdots & g_{k(n-j)} \end{bmatrix}$$

$G^{\bar{J}}$ is the degenerated Generator Matrix of G , which means

$$\begin{aligned}\bar{\mathbf{c}}_a^{\bar{\mathcal{J}}} &= \mathbf{s}_a \cdot G^{\bar{\mathcal{J}}} \\ \bar{\mathbf{c}}_b^{\bar{\mathcal{J}}} &= \mathbf{s}_b \cdot G^{\bar{\mathcal{J}}}\end{aligned}$$

and their corresponded transmitted codewords are

$$\begin{aligned}\mathbf{c}_a &= \mathbf{s}_a \cdot G \\ \mathbf{c}_b &= \mathbf{s}_b \cdot G\end{aligned}$$

\mathbf{c}_a and \mathbf{c}_b are differ at those positions which related to k_{th} symbol of \mathbf{s}_a and \mathbf{s}_b because other positions of \mathbf{s}_a and \mathbf{s}_b are the same.

If $\bar{\mathbf{c}}_a^{\bar{\mathcal{J}}} = \bar{\mathbf{c}}_b^{\bar{\mathcal{J}}}$, it implies that after puncturing, $\bar{\mathbf{c}}_a^{\bar{\mathcal{J}}}$ and $\bar{\mathbf{c}}_b^{\bar{\mathcal{J}}}$ have been removed all of positions those related to s_k , which means those punctured columns $\bar{\mathcal{J}}$ covers all positions which value is equal to 1 of row k .

Therefore, as $\bar{\mathcal{J}}$ covers all positions of row k equals to 1, we are not able to recognize whether our receiving codeword is encoded from \mathbf{s}_a or \mathbf{s}_b since $\bar{\mathbf{c}}_a^{\bar{\mathcal{J}}} = \bar{\mathbf{c}}_b^{\bar{\mathcal{J}}}$. ■

Proposition 1 [Decoding Failure, General Case]

The k_{th} input symbol s_k cannot be recovered if there exists a family of rows of $G^{\bar{\mathcal{J}}}$ except the k_{th} one, which is linearly dependent with k_{th} row of $G^{\bar{\mathcal{J}}}$.

Proof: Let $\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_d$ are d rows of $G^{\bar{\mathcal{J}}}$ which are linear dependent, by definition that there exist a vector $\mathbf{s} = (s_1, s_2, \dots, s_d)$, $s_1 \mathbf{r}_1 + s_2 \mathbf{r}_2 + \dots + s_d \mathbf{r}_d = 0$, we assume \mathbf{r}_1 is the k_{th} row of $G^{\bar{\mathcal{J}}}$ without loss of generality. Then,

$$(s_1 + \bar{s}_1) \cdot \mathbf{r}_1 + (s_2 + \bar{s}_2) \cdot \mathbf{r}_2 + \dots + (s_d + \bar{s}_d) \cdot \mathbf{r}_d = 0$$

where \bar{s}_i is the additive inverse of s_i , for $0 < i < d$,

$$(s_1 \cdot \mathbf{r}_1 + s_2 \cdot \mathbf{r}_2 + \dots + s_d \cdot \mathbf{r}_d) + (\bar{s}_1 \cdot \mathbf{r}_1 + \bar{s}_2 \cdot \mathbf{r}_2 + \dots + \bar{s}_d \cdot \mathbf{r}_d) = 0$$

Therefore,

$$(\bar{s}_1 \cdot \mathbf{r}_1 + \bar{s}_2 \cdot \mathbf{r}_2 + \dots + \bar{s}_d \cdot \mathbf{r}_d) = 0$$

There exists an input vector $(\bar{s}_1, \bar{s}_2, \dots, \bar{s}_d)$ which is different from $\bar{\mathbf{s}}$ will generate the same output codeword by $G^{\bar{\mathcal{J}}}$. In this case we are not able to recognize the original input value of these positions, including k_{th} input symbol s_k . ■

(B) Probabilistic Analysis

Lemma 2

Let p_k be the probability that k_{th} input symbol s_k cannot be recovered, also let e_i be the erasure probability of i_{th} column of G , then

$$p_k \geq \prod_{\forall i, i_{th} \text{ entry of row } k \text{ of } G \text{ is } 1} e_i$$

Proof: Let $p_{(\bar{J},k)}$ be the probability that k_{th} row of $G^{\bar{J}}$ is zero, it is clearly that

$$p_{(\bar{J},k)} \leq \prod_{\forall i, i_{th} \text{ entry of row } k \text{ of } G \text{ is } 1} e_i, \quad |\bar{J}| \geq i$$

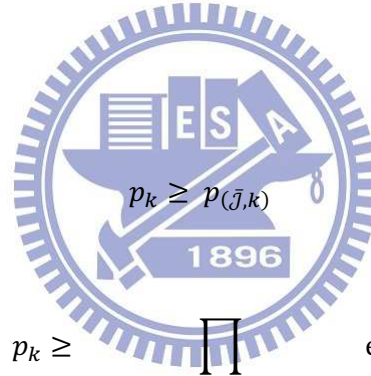
The equality hold when $|\bar{J}| = i$ with each related entry to G is 1.

According to **Lemma 1**, it is only a special case of decoding failure so that the real unrecoverable probability

$$p_k = p_{(\bar{J},k)} + C$$

$C \geq 0$.

Therefore,



which implies

$$p_k \geq \prod_{\forall i, i_{th} \text{ entry of row } k \text{ of } G \text{ is } 1} e_i$$



Definition Given an (n, k) linear block code \mathcal{C} , consider two codewords \mathbf{c} and \mathbf{c}' belong to \mathcal{C} , *Hamming distance* $d_H(\mathbf{c}, \mathbf{c}')$ is the number of distinct components in \mathbf{c} and \mathbf{c}' , and

$$d_H(\mathbf{c}, \mathbf{c}') = W_H(\mathbf{c} - \mathbf{c}')$$

$W_H(\mathbf{c} - \mathbf{c}')$ is called *Hamming weight* of vector $\mathbf{c} - \mathbf{c}'$.

Definition Consider an (n, k) linear block code \mathcal{C} , *minimum weight* of code \mathcal{C} is

$$W_{min}(\mathcal{C}) = \min_{\mathbf{c} \in \mathcal{C}} W_H(\mathbf{c})$$

And *minimum distance* of code \mathcal{C} is,

$$D_{min}(\mathcal{C}) = \min_{\mathbf{x}, \mathbf{x}' \in \mathcal{C}} d_H(\mathbf{c}, \mathbf{c}')$$

$$\begin{aligned}
&= \min_{c, c' \in \mathcal{C}} W_H(c - c') \\
&= \min_{y \in \mathcal{C}, y \neq 0} W_H(y) \\
&= W_{min}(\mathcal{C})
\end{aligned}$$

Definition Consider an (n, k) linear block codes \mathcal{C} , its Weight Enumerator is defined by

$$A(Z) = \sum_{i=0}^n A_i \cdot Z^i, \quad 0 \leq i \leq n$$

Where A_i is the number of codewords of weight i .

Definition Consider an encoding η of an (n, k) linear block codes \mathcal{C} , the *separation vector* of η with respect to a weight function w , is denoted by

$$S_w(\eta)_i \triangleq W_{min}[\{\eta(\mathbf{m}) - \eta(\mathbf{m}'): m_i \neq m'_i\}], \quad 1 \leq i \leq k$$

Where \mathbf{m} and \mathbf{m}' range over F^k

It implies

$$S_w(\eta)_i = W_{min}[\{\eta(\mathbf{m}): m_i \neq 0\}]$$

Definition Consider an (n, k) linear block codes, its *k-weight enumerator* is defined by

$$A^{(k)}(Z) = \sum_{i=0}^n A_i^{(k)} \cdot Z^i$$

Where $A_i^{(k)}$ is the number of codeword with weight i , being encoded from all input symbols except the k_{th} position is zero, for $0 \leq i \leq n$, that is

$$A_i^{(k)} = \{ |W_H(\eta(\mathbf{m})) = i | : m_k \neq 0 \}$$

Definition Consider an (n, k) linear block codes \mathcal{C} , \mathcal{C}^k is a subset of \mathcal{C} that

$$\mathcal{C}^k \triangleq \{\eta(\mathbf{m}): m_k \neq 0\}$$

And we also define that

$$\mathcal{C}^{\bar{k}} = \mathcal{C} \setminus \mathcal{C}^k$$

Proposition 2 [Symbol failure rate as transmitting over an EEP channel]

Let $P_E^{(k)}$ be the failure probability of k_{th} position of input vector over an EEP Binary Erasure Channel, then

$$P_E^{(k)} \leq \left(\frac{1}{2}\right)^{n+1} \cdot A^{(k)} \left(\sqrt{\frac{p}{1-p}} \right)$$

Proof: By our definition, $P_E^{(k)}$ is the probability that an *erasure event* occurs after decoding (decoding failure) by transmitting codeword of Precoder \mathbf{c}_i , for all \mathbf{c}_i corresponded to input vector \mathbf{s}_i those k_{th} position is not equal to zero, which can be written as

$$P_E^{(k)} = \sum_{i=0}^{2^{(k-1)}-1} \Pr\{Error|\mathbf{c}_i\} \cdot \Pr\{\mathbf{c}_i\}$$

It is clear that

$$\begin{aligned} P_E^{(k)} &= \sum_{i=0}^{2^{(k-1)}-1} \Pr\{\hat{\mathbf{r}} \neq \mathbf{c}_i|\mathbf{c}_i\} \cdot \Pr\{\mathbf{c}_i\} \\ &= \sum_{i=0}^{2^{(k-1)}-1} \left(\sum_{j \neq i} \Pr\{\hat{\mathbf{r}} = \mathbf{c}_j|\mathbf{c}_i\} \right) \cdot \Pr\{\mathbf{c}_i\} \end{aligned}$$

\mathbf{r} is the received symbol corresponded to \mathbf{c}_i , $\hat{\mathbf{r}}$ is the decoding result of \mathbf{r} , and

$$\begin{aligned} &\Pr\{\hat{\mathbf{r}} = \mathbf{c}_j|\mathbf{c}_i\} \\ &= \Pr\left\{ \bigcap_{l \neq j} \{d_H(\mathbf{c}_i, \mathbf{c}_j) \leq d_H(\mathbf{c}_i, \mathbf{c}_l)\} \mid \mathbf{c}_i \right\} \\ &\leq \Pr\{d_H(\mathbf{c}_i, \mathbf{c}_j) \leq d_H(\mathbf{c}_i, \mathbf{c}_l) \mid \mathbf{c}_i\} \\ &= \Pr\{l \geq W_H(\mathbf{c}_i - \mathbf{c}_j)\}, \end{aligned}$$

l is the number of erasure positions of vector $\mathbf{c}_i - \mathbf{c}_j$ with value '1', and

$$\begin{aligned} &\Pr\{l \geq W_H(\mathbf{c}_i - \mathbf{c}_j)\} \\ &= \frac{1}{2} \cdot \sum_{a=W_H(\mathbf{c}_i - \mathbf{c}_j)}^n \binom{n - W_H(\mathbf{c}_i - \mathbf{c}_j)}{a - W_H(\mathbf{c}_i - \mathbf{c}_j)} \cdot p^a \cdot (1-p)^{n-a} \\ &= \frac{1}{2} \cdot p^{W_H(\mathbf{c}_i - \mathbf{c}_j)} \cdot \sum_{a=W_H(\mathbf{c}_i - \mathbf{c}_j)}^n \binom{n - W_H(\mathbf{c}_i - \mathbf{c}_j)}{a - W_H(\mathbf{c}_i - \mathbf{c}_j)} \cdot p^{a - W_H(\mathbf{c}_i - \mathbf{c}_j)} \cdot (1-p)^{n-a} \\ &\leq \frac{1}{2} \cdot p^{W_H(\mathbf{c}_i - \mathbf{c}_j)} \cdot \sum_{a=W_H(\mathbf{c}_i - \mathbf{c}_j)}^n \binom{n - W_H(\mathbf{c}_i - \mathbf{c}_j)}{a - W_H(\mathbf{c}_i - \mathbf{c}_j)} \cdot p^{\frac{(n - W_H(\mathbf{c}_i - \mathbf{c}_j))}{2}} \cdot (1-p)^{\frac{(n - W_H(\mathbf{c}_i - \mathbf{c}_j))}{2}} \end{aligned}$$

$$\begin{aligned}
&= \frac{1}{2} \cdot p^{\frac{(n+W_H(\mathbf{c}_i-\mathbf{c}_j))}{2}} \cdot (1-p)^{\frac{(n-W_H(\mathbf{c}_i-\mathbf{c}_j))}{2}} \cdot \sum_{a=W_H(\mathbf{c}_i-\mathbf{c}_j)}^n \binom{n-W_H(\mathbf{c}_i-\mathbf{c}_j)}{a-W_H(\mathbf{c}_i-\mathbf{c}_j)} \\
&\leq p^{\frac{(n+W_H(\mathbf{c}_i-\mathbf{c}_j))}{2}} \cdot (1-p)^{\frac{(n-W_H(\mathbf{c}_i-\mathbf{c}_j))}{2}} \\
&= (p(1-p))^{\frac{n}{2}} \cdot \left(\frac{p}{1-p}\right)^{\frac{W_H(\mathbf{c}_i-\mathbf{c}_j)}{2}}
\end{aligned}$$

p is the erasure probability of Binary Erasure Channel, thus

$$\begin{aligned}
&\Pr\{\hat{\mathbf{r}} = \mathbf{c}_j | \mathbf{c}_i\} \\
&\leq \Pr\{l \geq W_H(\mathbf{c}_i - \mathbf{c}_j)\} \\
&= (p(1-p))^{\frac{n}{2}} \cdot \left(\frac{p}{1-p}\right)^{\frac{W_H(\mathbf{c}_i-\mathbf{c}_j)}{2}} \\
&= (p(1-p))^{\frac{n}{2}} \cdot \left(\sqrt{\frac{p}{1-p}}\right)^{d_H(\mathbf{c}_i, \mathbf{c}_j)}
\end{aligned}$$

Thus,

$$\begin{aligned}
P_E^{(k)} &\leq \sum_{i=0}^{2^{(k-1)}-1} \left(\sum_{j \neq i} (p(1-p))^{\frac{n}{2}} \cdot \left(\sqrt{\frac{p}{1-p}}\right)^{d_H(\mathbf{c}_i, \mathbf{c}_j)} \right) \cdot \Pr\{\mathbf{c}_i\} \\
&= (p(1-p))^{\frac{n}{2}} \cdot \sum_{i=0}^{2^{(k-1)}-1} \left(\sum_{d=d_H(\mathbf{c}_i, \mathbf{c}_j) \neq 0} A_d^{(k)} \left(\sqrt{\frac{p}{1-p}}\right)^d \right) \cdot \Pr\{\mathbf{c}_i\} \\
&= (p(1-p))^{\frac{n}{2}} \cdot \sum_{i=0}^{2^{(k-1)}-1} \left(A^{(k)} \left(\sqrt{\frac{p}{1-p}}\right) \right) \cdot \Pr\{\mathbf{c}_i\} \\
&= (p(1-p))^{\frac{n}{2}} \cdot A^{(k)} \left(\sqrt{\frac{p}{1-p}}\right) \cdot \left(\sum_{i=0}^{2^{(k-1)}-1} \Pr\{\mathbf{c}_i\} \right) \\
&= \frac{(p(1-p))^{\frac{n}{2}}}{2} \cdot A^{(k)} \left(\sqrt{\frac{p}{1-p}}\right) \\
&\leq \left(\frac{1}{2}\right)^{n+1} \cdot A^{(k)} \left(\sqrt{\frac{p}{1-p}}\right)
\end{aligned}$$

We assume every kind of input vector has the same probability being chosen to transmit over channel. ■

Proposition 3 [Symbol failure rate as transmitting over an UEP channel]

Let $P_E^{(k)}$ be the failure probability of k_{th} position of input vector over an UEP Binary Erasure Channel, then

$$P_E^{(k)} \leq \frac{1}{2} \cdot \sum_{y \in \mathcal{C}^k} \left(\prod_{y_s=1} e_s \right)$$

Where e_i is the erasure probability of output symbol at i_{th} position.

Proof: It is similar to **Proposition 2**, let $P_E^{(k)}$ is the probability that an *erasure event* occurs after decoding (decoding failure) by transmitting codeword of Precoder \mathbf{c}_i , for all \mathbf{c}_i corresponded to input vector \mathbf{s}_i those k_{th} position are not equal to zero, which can be written as

$$P_E^{(k)} = \sum_{i=0}^{(2^{k-1})-1} \left(\sum_{j \neq i} \Pr\{\hat{\mathbf{r}} = \mathbf{c}_j | \mathbf{c}_i\} \right) \cdot \Pr\{\mathbf{c}_i\}$$

$\hat{\mathbf{r}}$ is decoded result as receiving output codeword vector \mathbf{r} .

$$\begin{aligned} & \Pr\{\hat{\mathbf{r}} = \mathbf{c}_j | \mathbf{c}_i\} \\ &= \Pr\{\Pr\{\mathbf{r} | \mathbf{c}_i\} < \Pr\{\mathbf{r} | \mathbf{c}_j\}\} \\ &\leq \Pr\{\Pr\{r_1 | (\mathbf{c}_i)_1\} \cdot \Pr\{r_2 | (\mathbf{c}_i)_2\} \cdot \dots \cdot \Pr\{r_n | (\mathbf{c}_i)_n\} < \Pr\{r_1 | (\mathbf{c}_j)_1\} \cdot \Pr\{r_2 | (\mathbf{c}_j)_2\} \cdot \dots \cdot \Pr\{r_n | (\mathbf{c}_j)_n\}\} \\ &= \Pr\left\{ \prod_{(\mathbf{c}_j)_s \neq (\mathbf{c}_i)_s} \Pr\{r_s | \mathbf{c}_{js}\} - \prod_{(\mathbf{c}_j)_s = (\mathbf{c}_i)_s} \Pr\{r_s | \mathbf{c}_{is}\} > 0 \right\} \\ &= \frac{1}{2} \cdot \prod_{(\mathbf{c}_j)_s \neq (\mathbf{c}_i)_s} \Pr\{r_s \text{ is erased}\} \end{aligned}$$

The difference between **Proposition 2** and **Proposition 3** is each position of output codeword has the same erasure probability in the **Proposition 2**, on the other side, the erasure probability of each output codeword could have different erasure probability in the **Proposition 3** (i.e. e_i be the erasure probability of i_{th} position of output codeword), and

$$\begin{aligned} & \frac{1}{2} \cdot \prod_{(\mathbf{c}_j)_s \neq (\mathbf{c}_i)_s} \Pr\{r_s \text{ is erased}\} \\ &= \frac{1}{2} \cdot \prod_{(\mathbf{c}_i)_s - (\mathbf{c}_j)_s = 1} e_s \end{aligned}$$

Therefore,

$$\begin{aligned}
P_E^{(k)} &\leq \sum_{i=0}^{(2^{k-1})-1} \left(\sum_{j \neq i} \left(\frac{1}{2} \cdot \prod_{(c_j)_s \neq (c_i)_s} \Pr\{r_s \text{ is erased}\} \right) \right) \cdot \Pr\{\mathbf{c}_i\} \\
&\leq \sum_{i=0}^{(2^{k-1})-1} \left(\sum_{j \neq i} \left(\frac{1}{2} \cdot \prod_{(c_i)_s - (c_j)_s = 1} e_s \right) \right) \cdot \Pr\{\mathbf{c}_i\} \\
&= \frac{1}{2} \cdot \sum_{i=0}^{(2^{k-1})-1} \left(\sum_{W_H(\mathbf{y}) \neq 0, \mathbf{y} \in \mathcal{C}^k} \prod_{y_s = 1} e_s \right) \cdot \Pr\{\mathbf{c}_i\} \\
&= \frac{1}{2} \cdot \sum_{\mathbf{y} \in \mathcal{C}^k} \left(\prod_{y_s = 1} e_s \right)
\end{aligned}$$

\mathcal{C}^k is the collection of codewords that encoded by input vectors those k_{th} position are not equal to zero. ■

5.1.2 Error Bounds for Viterbi Decoding

The correct path of trellis diagram in Viterbi decoding process is eliminated for the first time in favor of an incorrect path at time unit t , we say that is the *First event error* represents as $P_f(E, t)$.

Let's consider the first event error probability over a symmetric binary erasure channel. If we define the set $E_t = \{E_{1,(t,l_1)}, E_{2,(t,l_2)}, \dots, E_{i,(t,l_i)}\}$ is a collection of i error events off the correct path at time unit $t - l_i$ occurs at time unit t . According to the union bound,

$$P_f(E, t) = \Pr \left\{ \bigcup_{E_{i,(t,l_i)} \in E_t} E_t \right\} \leq \sum_{E_{i,(t,l_i)} \in E_t} \Pr\{E_t\}$$

Where $\Pr\{E_{i,(t,l_i)}\}$ is the probability that decoder prefers error event i to the correct path at time unit t .

$$\Pr\{E_{i,(t,l_i)}\} = \Pr\{\hat{\mathbf{c}}_{(t,l_i)} = \mathbf{c}_{i,(t,l_i)} | \mathbf{c}_{(t,l_i)}\}$$

$\mathbf{c}_{i,(t,l_i)}$ is the incorrect codeword vector with length l_i which is off the correct path at time unit $t - l_i$, and $\hat{\mathbf{c}}_{(t,l_i)}$ is the decoded result corresponded to the receiving output sequence $\mathbf{r}_{(t,l_i)}$, then

$$\begin{aligned}
\Pr\{E_{i,(t,l_i)}\} &= \Pr\{\Pr\{\mathbf{r}_{(t,l_i)} | \mathbf{c}_{(t,l_i)}\} < \Pr\{\mathbf{r}_{(t,l_i)} | \mathbf{c}_{i,(t,l_i)}\}\} \\
&\leq \frac{1}{2} \cdot \prod_{c_{is} \neq c_s} \Pr\{r_s \text{ is erased}\}
\end{aligned}$$

Therefore,

$$\begin{aligned}
P_f(E, t) &\leq \sum_{E_{i,(t,l_i)} \in E_t} \left(\frac{1}{2} \cdot \prod_{c_{is} \neq c_s} \Pr\{r_s \text{ is erased}\} \right) \\
&= \frac{1}{2} \cdot \sum_{E_{i,(t,l_i)} \in E_t} \left(\prod_{c_{is}-c_s \neq 0} e_s \right) \\
&= \frac{1}{2} \cdot \sum_{W_H(c_i) \neq 0} \left(\prod_{c_{is}, l_i=1} e_s \right)
\end{aligned}$$

Because this bound is independent of t,

$$\begin{aligned}
P_f(E) &\leq \frac{1}{2} \cdot \sum_{c_i \in \mathcal{C}^k} \left(\prod_{c_{is}=1} e_s \right) \\
&= \frac{1}{2} \cdot \sum_{y \in \mathcal{C}^k} \left(\prod_{y_s=1} e_s \right)
\end{aligned}$$

5.1.3 Realization

The objective of using multiplexer in our coding scheme is trying to let the UEP layers of system be adjustable. By the requirements from the MGS encoder in the source coding part, MGS coder would ask for some expected protection layer counts and bit error rate with respect to each layer. Unfortunately, by given finite choices of Precoder and Postcoder, the possible combinations of them may not able to provide an optimal coding scheme for the source coding part, therefore we need a channel coding scheme which its UEP layers are adjustable, including bit error rate and layer counts. The next paragraph will describe how to adjust bit error rate and create more UEP layers by multiplexer.

If Precoder can provide enough layers which is equal to overall required UEP layer counts, that means we just need to adjust the protection ability of each layer to match MGS coder's requirement. According to the formula proved in Section 5.1.2, we know that

$$P_E^{(k)} \leq \frac{1}{2} \cdot \sum_{y \in \mathcal{C}^k} \left(\prod_{y_s=1} e_s \right)$$

If we consider an LT codes with two protection layers with respect to erasure probability e_{high} and e_{low} , which $e_{\text{high}} > e_{\text{low}}$, $e_s \in [e_{\text{high}}, e_{\text{low}}]$. And the bit error rate for layer k of UEP Precoder is \mathcal{L}_k . Then, in order to match up the requirement for SVC coder, multiplexer would proceed as following ways,

Step.1	Spread out the inequality for all k
Step.2	Find the dominated term for each layer. [If we would like to adjust down \mathcal{L}_k , the dominate term might be some common terms of e_i , which i_{th} position lots of $\mathbf{y} \in \mathcal{C}^k$ is '1'] [If we would like to adjust up \mathcal{L}_k , the dominate term might be the term with lower exp]
Step.3	assign e_{high} or e_{low} to the dominate term with respect to the \mathcal{L}_k be higher or lower
Step.4	Find the dominate term for each layer again. [Because after some e_s are determined after step.3, that would change e_s to be a constant]
Step.5	Repeat from Step.2 until all e_s are assigned a specific value.

After multiplexing, we need to use our formula to check whether for all k , $P_E^{(k)}$ is match our requirement.

Example

Consider a (4,3) linear block codes with generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

And the Rateless Postcoder has two UEP layer, with respect to the bit error rate 10^{-1} and 10^{-3} , if these two layers both has the same proportion of information, then the goal of Case.1 is trying to make the UEP behavior more obviously, and the Case.2 is trying to minimize the average bit error rate of three layers.

	e_1	e_2	e_3	e_4
Case.1	10^{-1}	10^{-1}	10^{-3}	10^{-3}
Case.2	10^{-3}	10^{-3}	10^{-1}	10^{-1}

C1	(1 0 0)	(1 0 0 0)	$\mathbf{e_1 + e_1 e_3 e_4 + e_1 e_2 e_4 + e_1 e_2 e_3}$
	(1 0 1)	(1 0 1 1)	Case.1 : $10^{-1} + 10^{-7} + 10^{-5} + 10^{-5}$
	(1 1 0)	(1 1 0 1)	Case.2 : $10^{-3} + 10^{-5} + 10^{-7} + 10^{-7}$

	(1 1 1)	(1 1 1 0)	
C2	(0 1 0)	(0 1 0 1)	$e_2e_4 + e_2e_3 + e_1e_2e_4 + e_1e_2e_3$
	(0 1 1)	(0 1 1 0)	Case. 1 : $10^{-4} + 10^{-4} + 10^{-5} + 10^{-5}$
	(1 1 0)	(1 1 0 1)	Case. 2 : $10^{-4} + 10^{-4} + 10^{-7} + 10^{-7}$
	(1 1 1)	(1 1 1 0)	
C3	(0 0 1)	(0 0 1 1)	$e_3e_4 + e_2e_3 + e_1e_3e_4 + e_1e_2e_3$
	(0 1 1)	(0 1 1 0)	Case. 1 : $10^{-6} + 10^{-4} + 10^{-7} + 10^{-4}$
	(1 0 1)	(1 0 1 1)	Case. 2 : $10^{-2} + 10^{-4} + 10^{-5} + 10^{-7}$
	(1 1 0)	(1 1 1 0)	

5.2 Design of Interleaver

5.2.1 Good Properties for Convolutional Decoding Process

This section will describe how the erasure channel behavior effect the performance of Convolutional decoder, that is Viterbi decoding process, once we knew which kind of erasure events will damage the performance of Viterbi decoding, we can design a suitable interleaver avoiding these bad erasure events which Viterbi decoding cannot afford.

In Viterbi decoding process, we say a fundamental path is a path start from all-zero path and end the all-zero state without intermediate return. The error event in the Viterbi decoding is that the correct path is eliminated in favor of the incorrect path. It is equivalent to the event that transmitting an all-zero sequence and the survivor path is a fundamental path which is not the all-zero one, the error event will occur when all coded bit on this fundamental path are erased.

Consider an (n, k) Convolutional codes, if we define the fundamental path with the minimum Hamming weight of coded bit as event E_{\min} , also the length of E_{\min} is $l(\text{branch}) = n \cdot l(\text{bit})$ and the Hamming weight of coded bit of E_{\min} is E_w .

Example

Given an $(5, 3)$ Convolutional codes and E_{\min} in [Figure. 14 : Minimum Weight of Fundamental Path in Viterbi Decoding Process], in other words there is no fundamental path which Hamming weight of coded bit is larger than E_{\min} ,

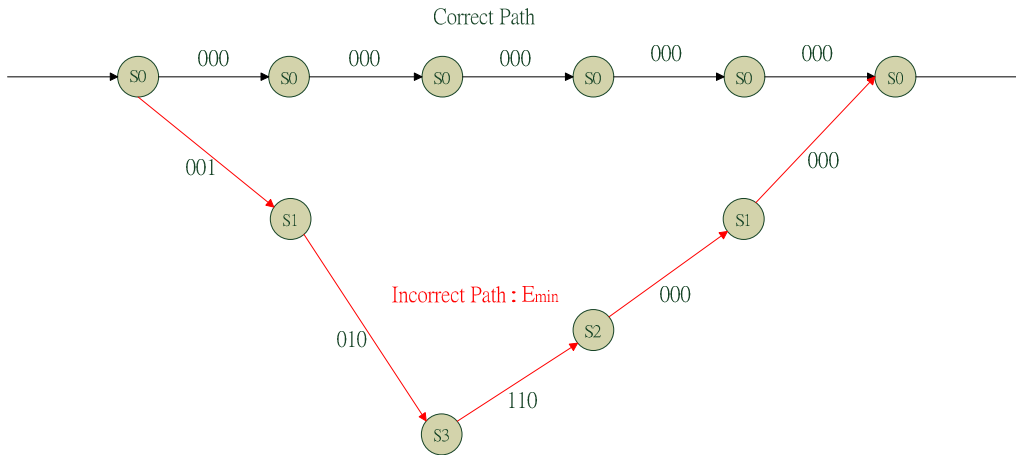


Figure. 14 : Minimum Weight of Fundamental Path in Viterbi Decoding Process

with length of E_{\min} : $l = 5(\text{branch}) = 25(\text{bit})$ and the weight of E_{\min} : $w = 4$. ■

Unfortunately the erased positions after LT decoding is too concentrate. [Figure. 15 : Error Distances in Postcoder Output Sequence] gives a simple example shows the error distance behavior before and after doing block interleaving, with 100 LT code blocks and rateless overhead 0.11, the interleaver block size is 0.1M bit. It is clear that the system with interleaver has smaller average distance of errors, and cases for continuous errors and cases with small error distance are decreased after interleaving.

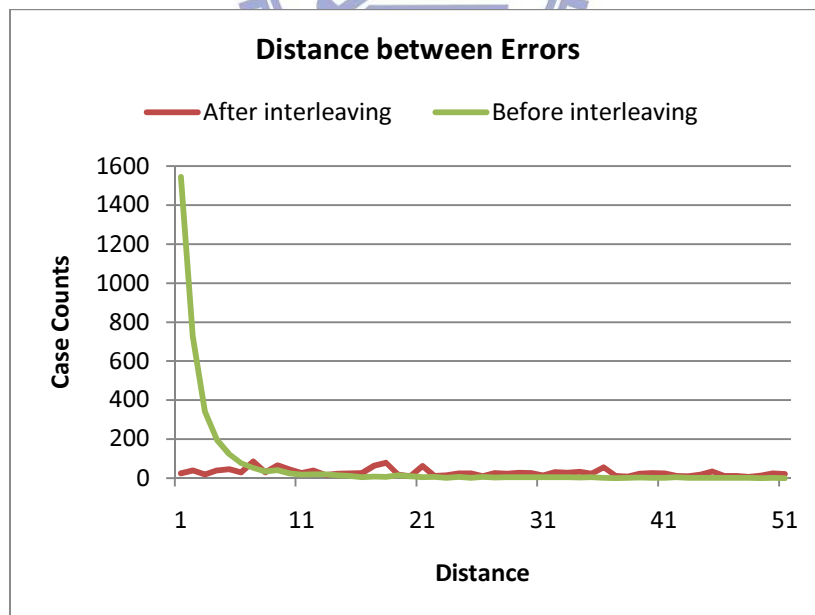


Figure. 15 : Error Distances in Postcoder Output Sequence

The thing that our interleaver need to do is trying to avoid more than w erased positions gather within a section of path with length l but spread over the whole output codeword sequence, because when there exist a section of length l with more than w erased positions, the error event may occur.

To approach the goal, we have to know how erasure positions distribute over the output codeword sequence, we apply the *Sliding Window*, sliding from the first position of output codeword sequentially.

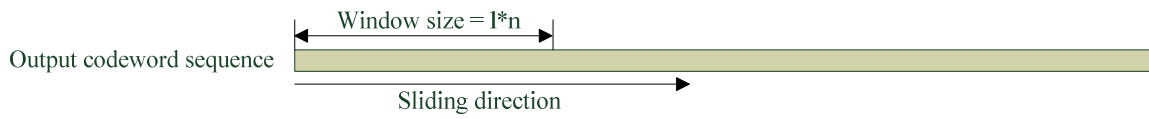


Figure. 16 : Sliding Window for Error Ratio Estimation

We called *Error Ratio* is the density of erasure counts in the sliding window that

$$\text{Error Ratio} := \frac{\text{\#error in sliding windows}}{\text{sliding window size}}$$

And the sliding window index is the index of first symbol in the sliding window. As [Figure. 17 : Error Ratio before interleaving] and [Figure. 18 : Error Ratio before interleaving] , it is clear that how errors distributed over the output codeword sequence before and after interleaving.

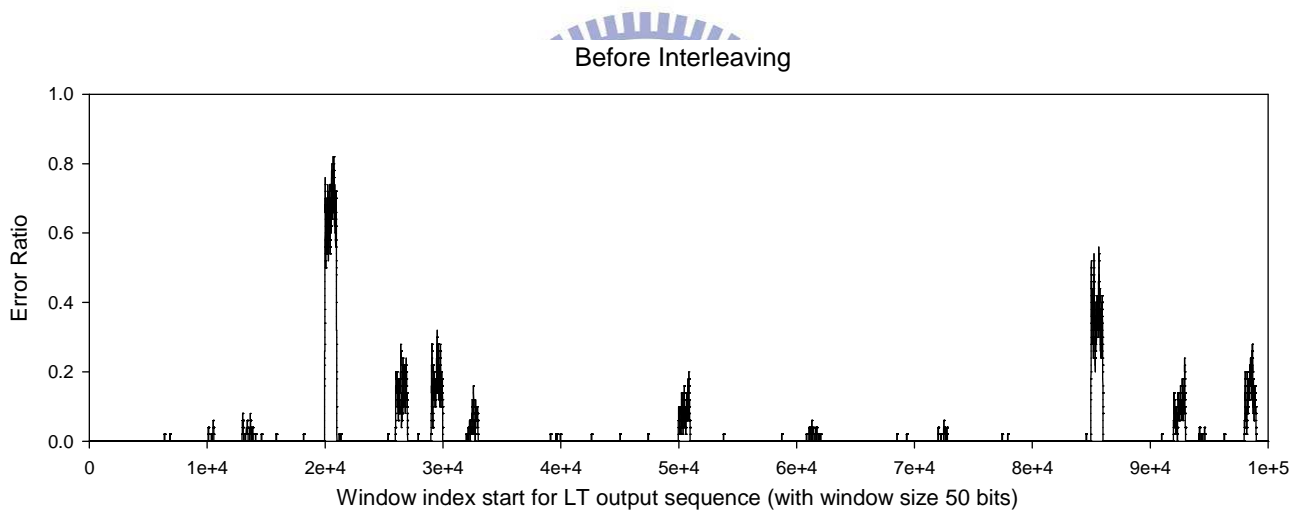


Figure. 17 : Error Ratio before interleaving

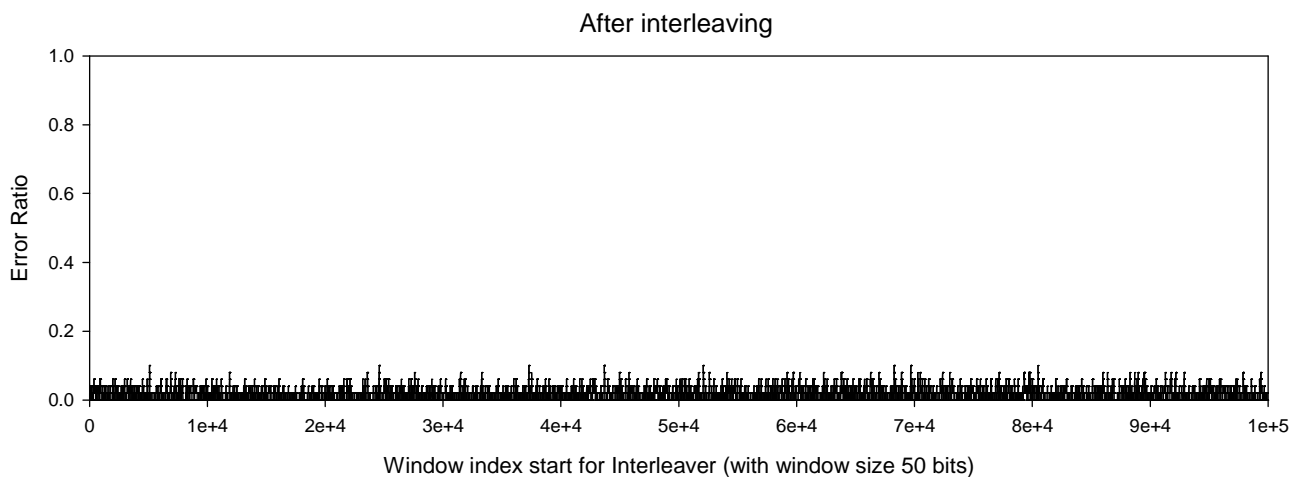


Figure. 18 : Error Ratio before interleaving

Another reason we have to eliminate effects of error burst event is to keep our UEP ability work after performing LT decoding, because

$$\begin{aligned} & \text{Overall Decoding Failure Probability} \\ &= \Pr\{\text{LTburst}\} \cdot \Pr\{\text{ConvolutionalDecoderFailure}|\text{LTburst}\} \\ &+ \Pr\{\text{LTnonburst}\} \cdot \Pr\{\text{ConvolutionalDecoderFailure}|\text{LTnonburst}\} \end{aligned}$$

To a specific rateless overhead, $\Pr\{\text{LTburst}\}$ and $\Pr\{\text{LTnonburst}\}$ is the same to different UEP streaming, at high rateless overhead, $\Pr\{\text{LTnonburst}\}$ is close to 1 and $\Pr\{\text{LTburst}\}$ is close to zero, so the overall decoding failure probability is approximately equal to $\Pr\{\text{ConvolutionalDecoderFailure}\}$. But in the low rateless overhead, $\Pr\{\text{LTburst}\}$ may not smaller than $\Pr\{\text{LTnonburst}\}$, to the stronger protection layer, the term $\Pr\{\text{ConvolutionalDecoderFailure}|\text{LTnonburst}\}$ is quite smaller than the weaker one, it implies the stronger protection ability in a layer, the $\Pr\{\text{ConvolutionalDecoderFailure}|\text{LTburst}\}$ dominates the overall decoding failure probability, it eliminate the UEP ability.

5.2.2 Objective Function for Interleaver Design

In the previous section, we knew that the dominate term of overall decoding failure probability at low rateless overhead is $\Pr\{\text{ConvolutionalDecoderFailure}|\text{LTburst}\}$ because at low rateless overhead we cannot neglect $\Pr\{\text{LTburst}\}$.

$$\text{Decoding results} \left\{ \begin{array}{l} \text{LT codes with burst error event} \left\{ \begin{array}{l} \text{Burst error event does not be resolved} \\ \text{Burst error event is resolved by interleaver} \end{array} \right. \\ \text{LT codes without burst error event} \end{array} \right.$$

If we add an Interleaver, the overall decoding failure probability can be rewrite as

$$\begin{aligned} & \text{Overall Decoding Failure Probability} \\ &= \Pr\{\text{LTburst}\} \cdot \\ & \{ \Pr\{\text{BurstUnresolved}|\text{LTburst}\} \cdot \Pr\{\text{ConvolutionalDecoderFailure}|\text{LTburst}, \text{BurstUnresolved}\} \\ & \quad + \Pr\{\text{BurstResolved}|\text{LTburst}\} \\ & \quad \cdot \Pr\{\text{ConvolutionalDecoderFailure}|\text{LTburst}, \text{BurstResolved}\} \} \\ & + \Pr\{\text{LTnonburst}\} \cdot \Pr\{\text{ConvolutionalDecoderFailure}|\text{LTnonburst}\} \end{aligned}$$

Therefore,

Overall Decoding Failure Probability

$$= \Pr\{\text{LTburst, BurstUnresolved}\} \cdot \Pr\{\text{ConvolutionalDecoderFailure}|\text{LTburst, BurstUnresolved}\} \\ + \Pr\{\text{LTnonburst} \cup (\text{LTburst, BurstResolved})\} \cdot \Pr\{\text{ConvolutionalDecoderFailure}|\text{LTnonburst} \\ \cup (\text{LTburst, BurstResolved})\}$$

This guide us to the goal of interleaver design: To decrease the $\Pr\{\text{LTburst, BurstUnresolved}\}$, the following section will describe how to estimate the probability.

We define the error burst is the event that may cause error event in the Viterbi decoding process by following parameters:

k/n	Be the input to output ratio of Convolutional codes.
$E_{\min,i}$	Be the fundamental path with minimum Hamming weight of UEP layer i in Convolutional coder.
w_i	Be the Hamming weights of $E_{\min,i}$.
l_i	Be the length of $E_{\min,i}$, in branches.

Table. 3 : Parameters in Viterbi decoding process

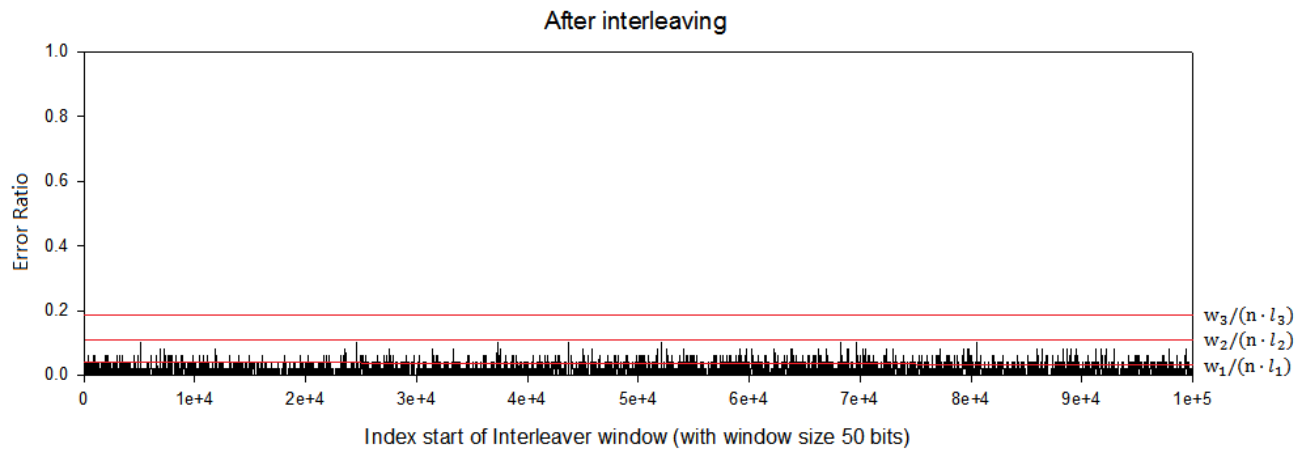
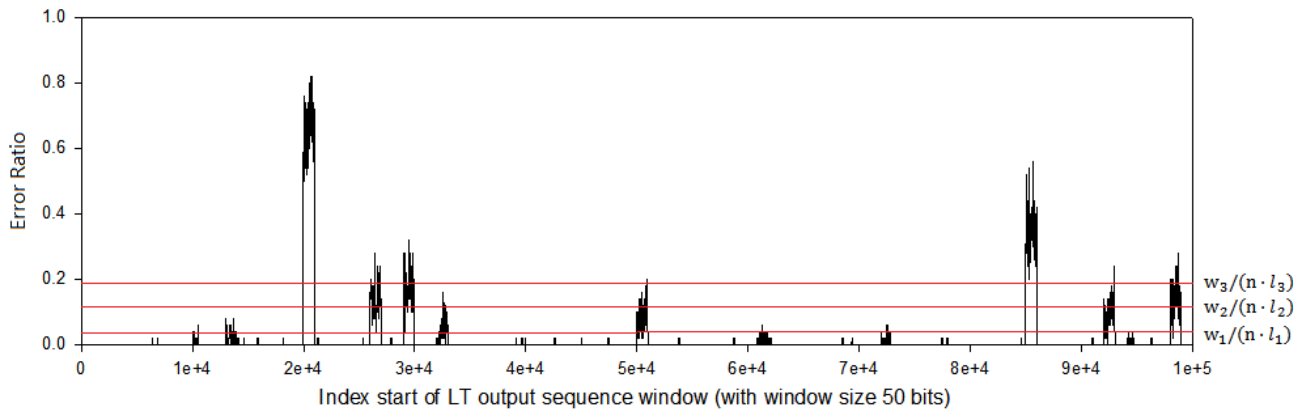


Figure. 19 : Error Ratio boundary for different UEP layer

[Figure. 19 : Error Ratio boundary for different UEP layer] shows the Error Ratio value beyond the value $w_i/(n \cdot l_i)$ may cause the error event, we called the section error burst. On the other side, the area with density value beneath the line is a region without error burst. Thus, we think our objective is trying to minimize the probability that Error Ratio over the cordon.

5.2.3 Realization/Application

In the section, we will describe how we realize our interleaver by guidelines and design principles mentioned in the previous article.

(A) Block Interleaver

One of realizations is block interleaver, and the block interleaver is able to use following parameters to adjust for better performance:

B	Be the average length of LT error burst, the size in bits.
L	Be the block size of interleaver, in bits.

Table. 4 : Parameters used in block interleaver design

If our information is written column by column and read row by row, in order to break up our error burst and stretch out the distance between errors, we have to let column size of block interleaver to be B and row size of interleaver to be $\lfloor L/B \rfloor$. The design is simple but actually these parameters are independent with convolutional Precoder, thus we provide another interleaver design, it is more complex and more customize to Convolutional Precoder and LT Postcoder design.

(B) S-random Interleaver

Another realization is modified by a classic S-random Interleaver, adjusting it by some parameters of the Convolutional coder and LT Postcoder. The S-random interleaver is able to use following parameters to adjust for better performance:

k/n	Be the input to output ratio of Convolutional codes.
B	Be the maximum length of LT error burst, the size in bits.
L	Be the block size of interleaver, in bits.
$E_{\min,i}$	Be the fundamental path with minimum Hamming weight of layer i in Convolutional coder.
w_i	Be the Hamming weights of $E_{\min,i}$.

l_i	Be the length of $E_{\min,i}$, in branches.
-------	--

Table. 5 : Parameters used in S-random interleaver design

When realizing our interleaver, we need to find the minimum Hamming weight of the fundamental path in Convolutional codes, which is $E_{\min,i}$, we also need to know its weight (w_i) of the , length (l_i), and LT block length. The steps of deciding an index value in the interleaver are

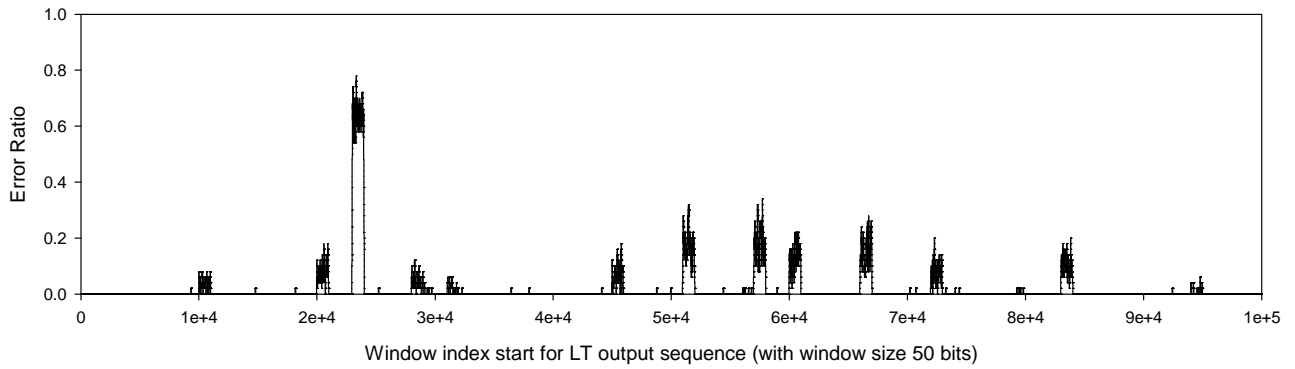
Step.1	Randomly choose an integer smaller than the interleaver size.
Step.2	Compare with the previous $\lfloor \frac{B}{w_i} \rfloor$ indices, if it is equal to any of $\lfloor \frac{B}{w_i} \rfloor$ previous selection within a distance of $\pm \lfloor \frac{l_i \cdot n}{2} \rfloor$, repeat Step.1.

Each randomly selected index will compare with the $\lfloor \frac{B}{w_i} \rfloor$ previous selected indices. If the current selection is equal to any of $\lfloor \frac{B}{w_i} \rfloor$ previous selection within a distance of $\pm (l_i \cdot n)/2$, then the current selection will be rejected.

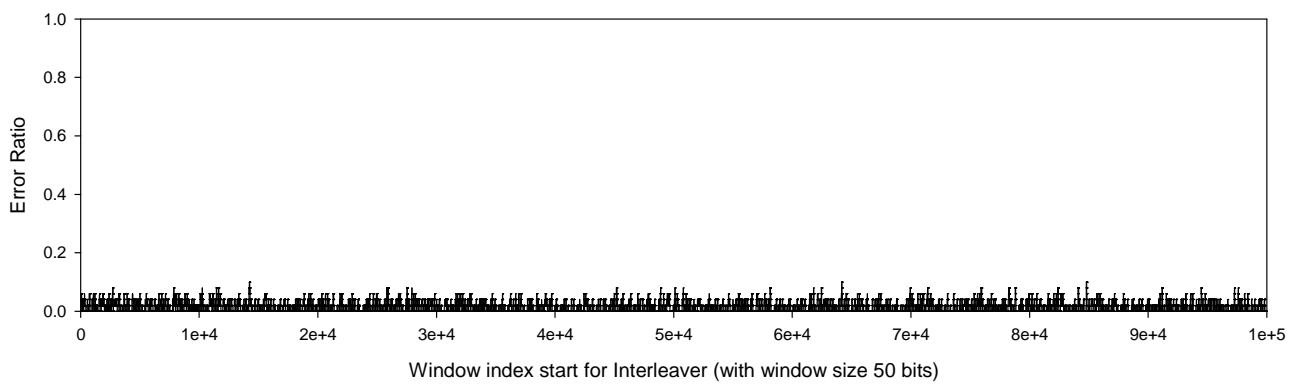
It is clear that there are more parameters about Convolutional Precoder and LT Postcoder that we can adjust in S-random interleaver than traditional block interleaver, and by experiment results, the UEP performance of S-random interleaver is truly better than block interleaver.

[Figure. 20 : Error distribution for Interleaver size 100Kbit] to [Figure. 25 : Bit error rate for Interleaver size 10Kbit] compare between traditional block interleaver and our modified s-random interleaver with different interleaver size. As you can see, there is not obvious difference between block interleaver and s-random interleaver when interleaver size is equal to 100kbit, but as we decrease the size of interleaver, the performance of s-random interleaver will be more outstanding. Because our interleaver sizes in all experiments are equal to or larger than 100kbits, thus we can directly use the block interleaver to reduce the complexity.

Before Interleaving



S-random Interleaver



Block Interleaver

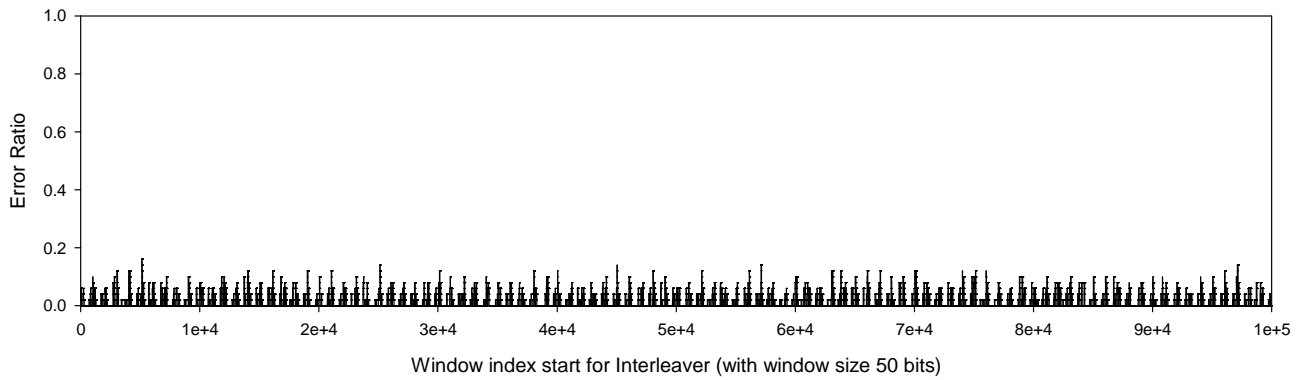
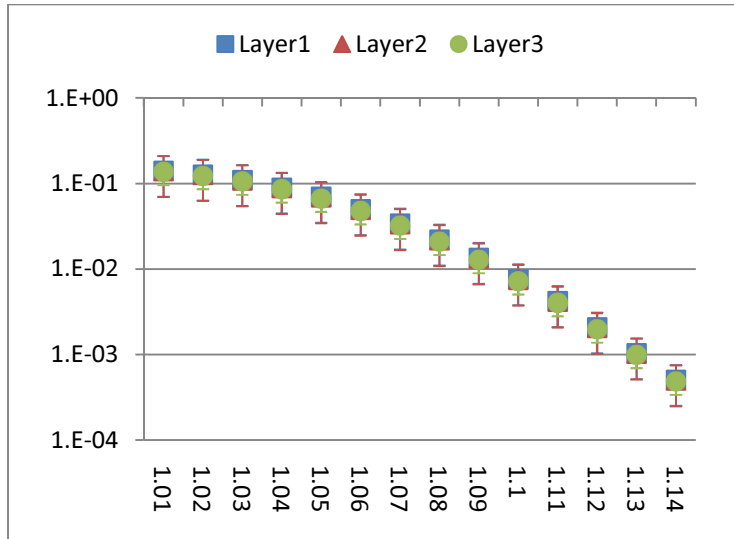
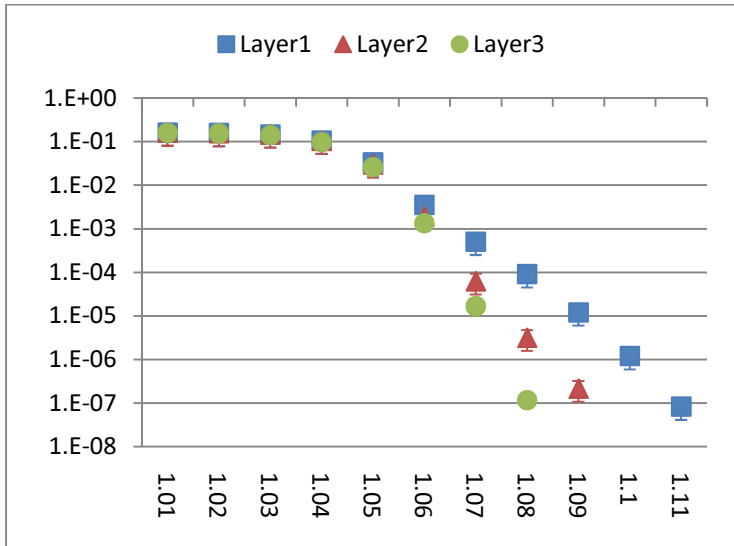


Figure. 20 : Error distribution for Interleaver size 100Kbit

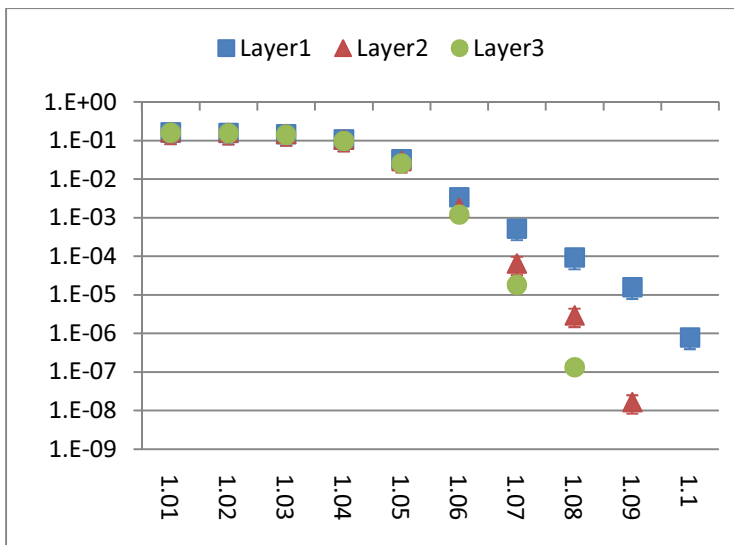
The following results are real bit error rate of system, with respect to block interleaver, s-random interleaver and without interleaver, x-axis represents rateless overhead and y-axis represents bit error rate.



Without Interleaver



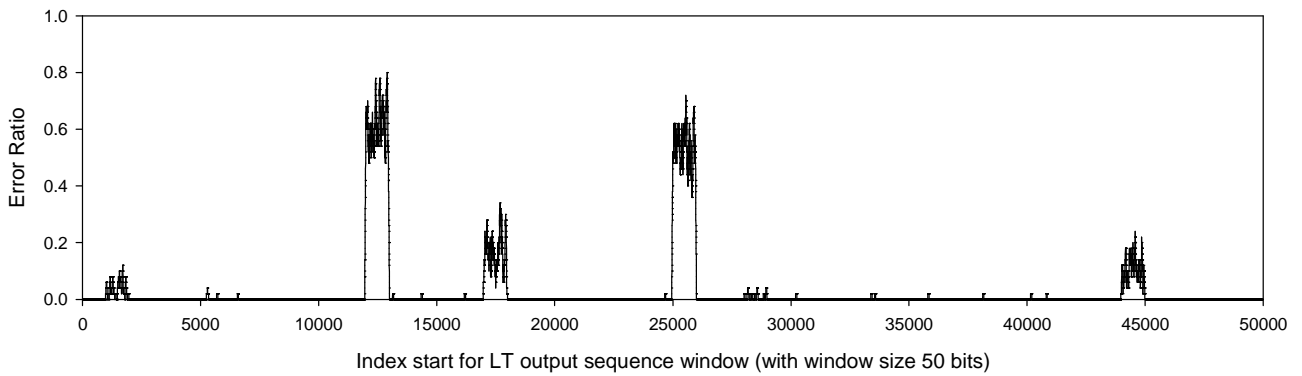
S-random Interleaver, 100k



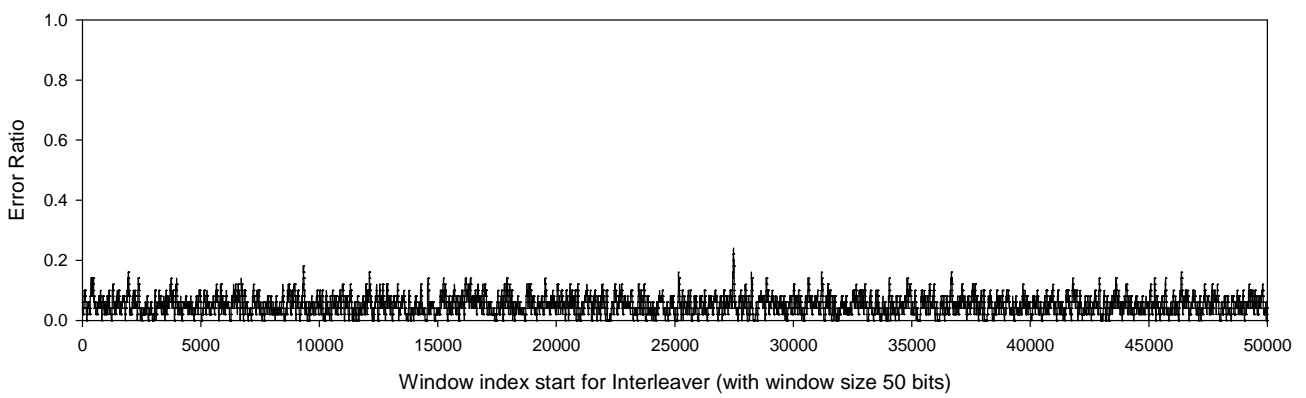
Block Interleaver, 100k

Figure. 21 : Bit error rate for Interleaver size 100Kbit

Before Interleaving



S-random Interleaver



Block Interleaver

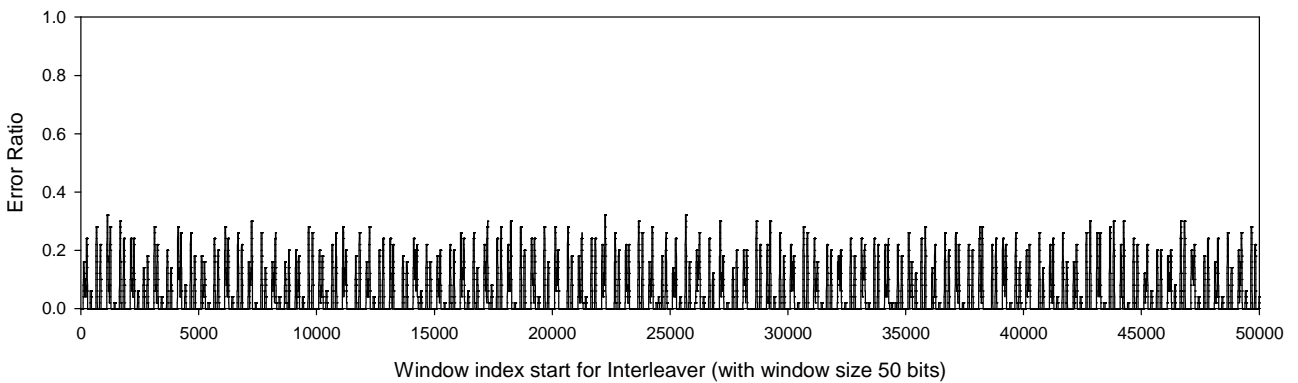
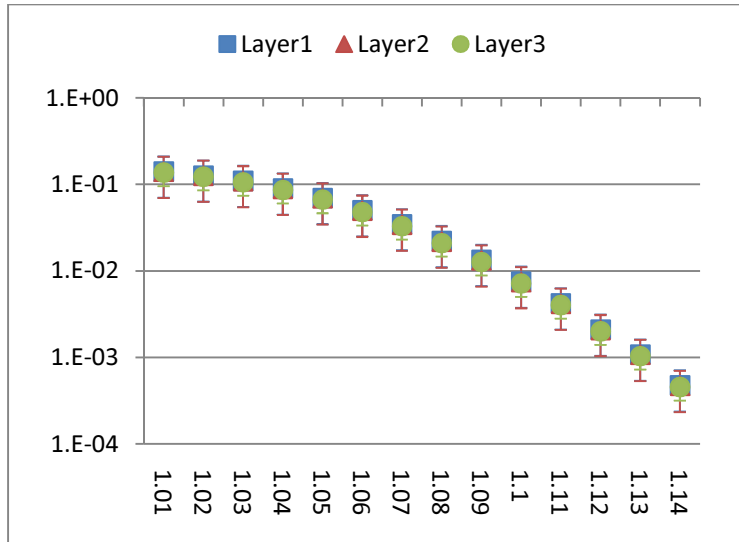
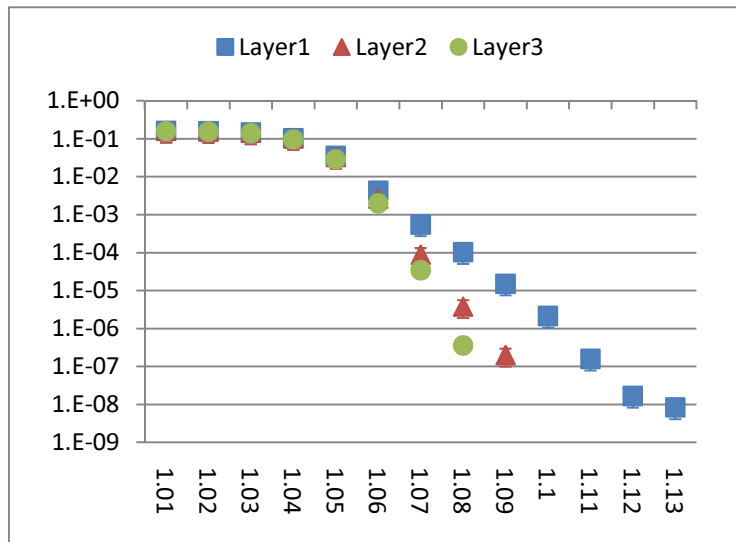


Figure. 22 : Error distribution for Interleaver size 50Kbit

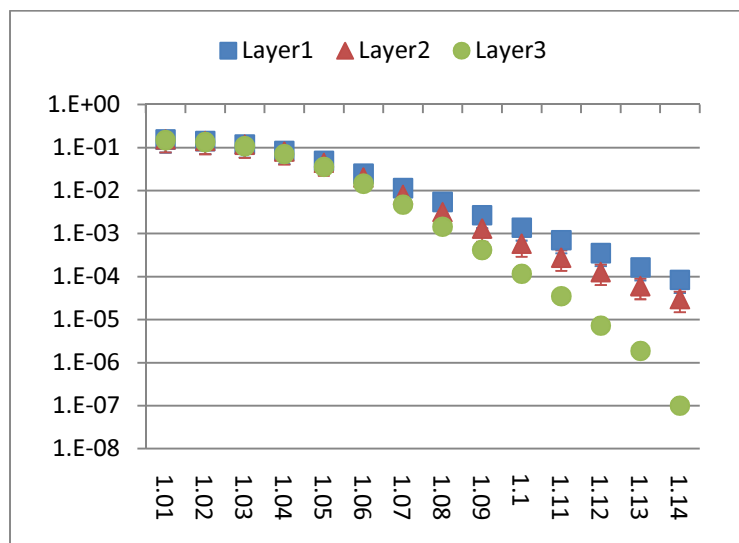
The following results are real bit error rate of system, with respect to block interleaver, s-random interleaver and without interleaver, x-axis represents rateless overhead and y-axis represents bit error rate.



Without Interleaver



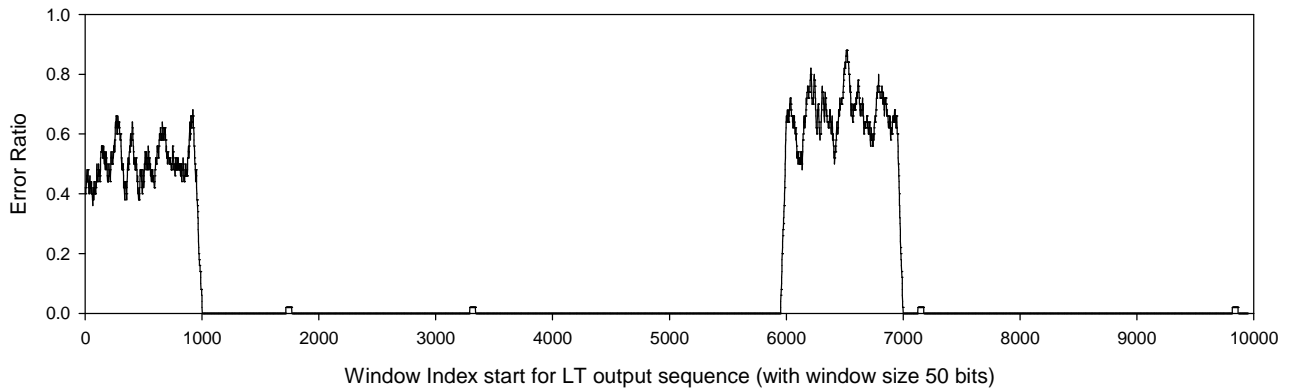
S-random Interleaver, 50k



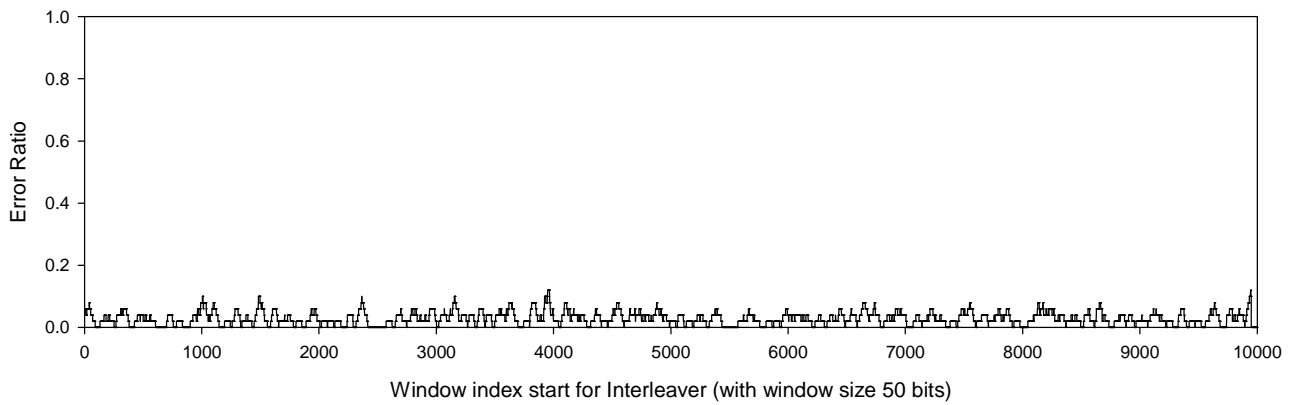
Block Interleaver, 50k

Figure. 23 : Bit error rate for Interleaver size 50Kbit

Before Interleaving



S-random Interleaver



Block Interleaver

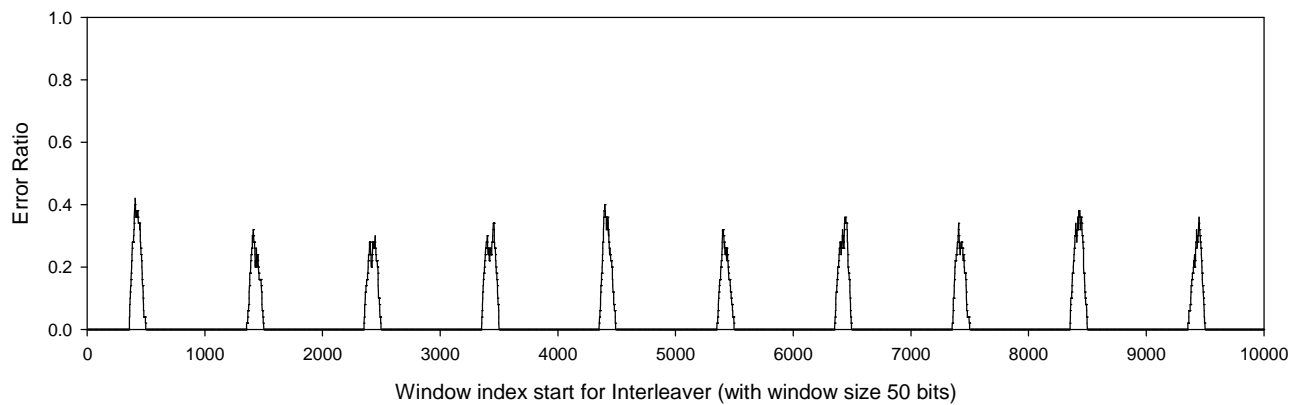
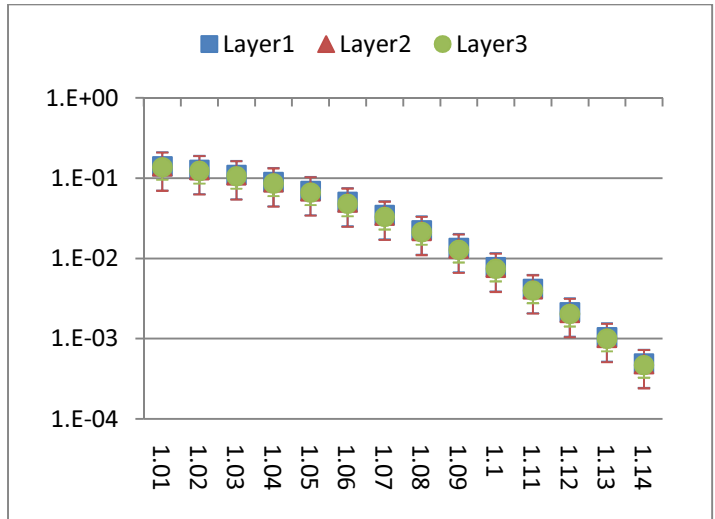
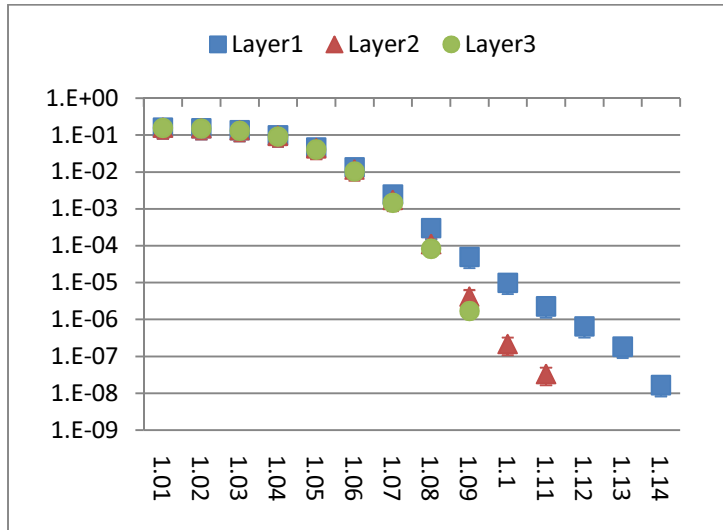


Figure. 24 : Error distribution for Interleaver size 10Kbit

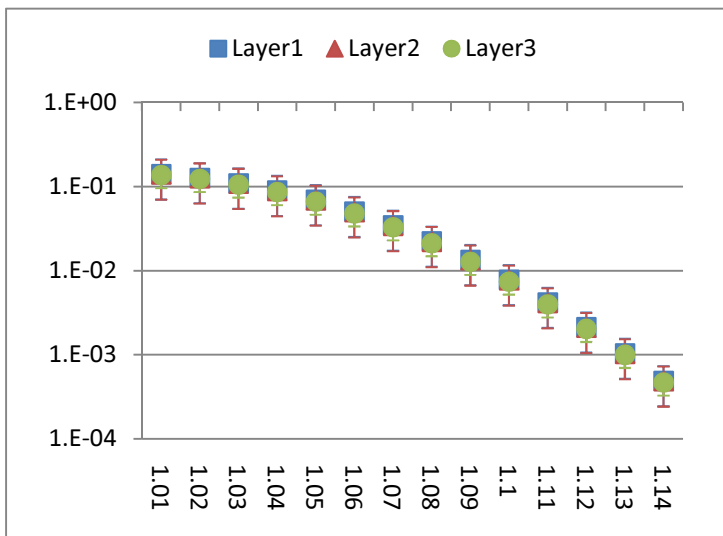
The following results are real bit error rate of system, with respect to block interleaver, s-random interleaver and without interleaver, x-axis represents rateless overhead and y-axis represents bit error rate.



Without Interleaver



S-random Interleaver, 10k



Block Interleaver, 10k

Figure. 25 : Bit error rate for Interleaver size 10Kbit

Chapter 6. Realization of Rateless Cascaded UEP Codec

In this chapter, we will cascade our channel coding system with the source coding system, to make sure our design for each part does truly improve the overall decoding performance. All experiments would be separated into three series, differences between them are at the Postcoder part. The first coding system is composed by a UEP Convolutional coder with a traditional Reed-Solomon coder, compared with the equal erasure protected Reed-Solomon coder. The second coding system is composed by the same UEP Convolutional Precoder with an equal erasure protected LT coder, the third coding system is composed by the same UEP Convolutional Precoder, but with an unequal erasure protected LT coder. We will see how different of their decoding performances as following.

6.1 Convolutional Precoder + Reed-Solomon Postcoder

In the first experiment series, we cascade our UEP Convolutional Codes with a tradition Reed-Solomon Codes, Our UEP encoder [Figure. 26 : System flow chart for UEP Conv. PreCoder cascaded with Reed-Solomon Code] consists of three components: (1) a three-level convolutional (N, k, M) UEP pre-coder with $N = 5, k = 3$ and $M = 7$ (2) a two-dimensional block interleaver with horizontal and vertical displacements of $(10, 1)$ and (3) a Reed-Solomon (N, k) post-coder with $N = 63$ and $k = 50$.

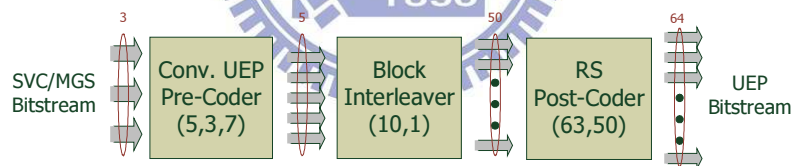


Figure. 26 : System flow chart for UEP Conv. PreCoder cascaded with Reed-Solomon Coder

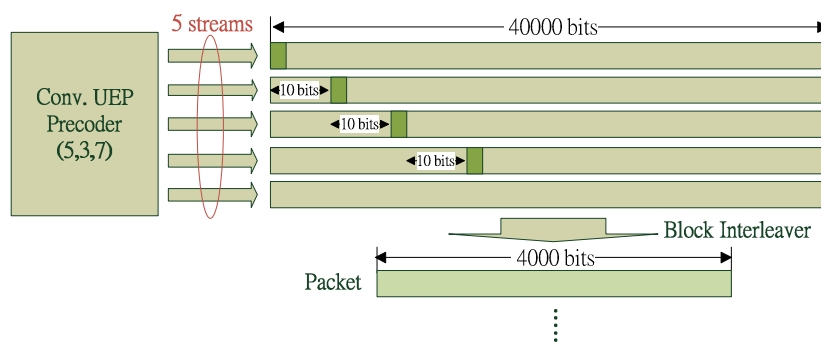


Figure. 27 : (10, 1)-Block interleaving to Reed-Solomon Coder

When transmitting, we separate our UEP bit stream into two parts, transmitting over the primary channel and secondary channel respectively, the primary channel contains the coding information from the UEP Convolutional Codes and secondary channel contains information encoded from the Reed-Solomon Encoder.

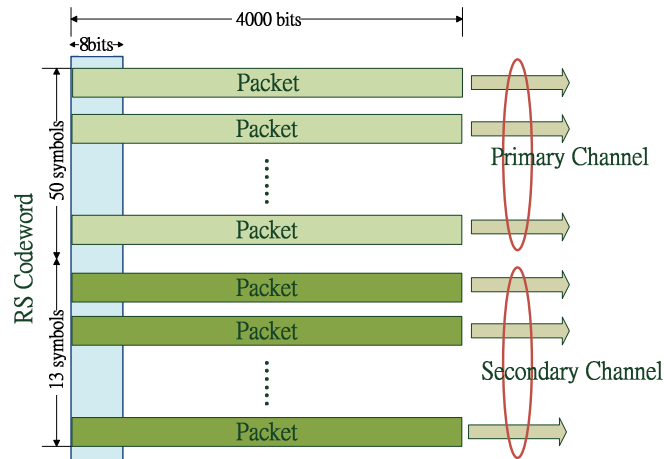
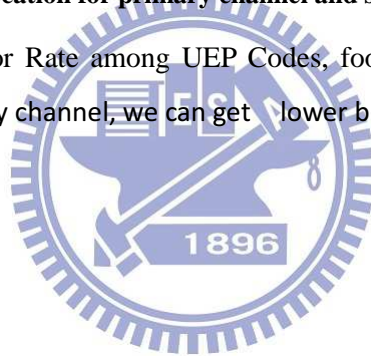


Figure. 28 : Allocation for primary channel and secondary channel

[Figure. 29 : Comparison of Bit Error Rate among UEP Codes, football] show that as we listening more redundant packets from the secondary channel, we can get lower bit error rate.



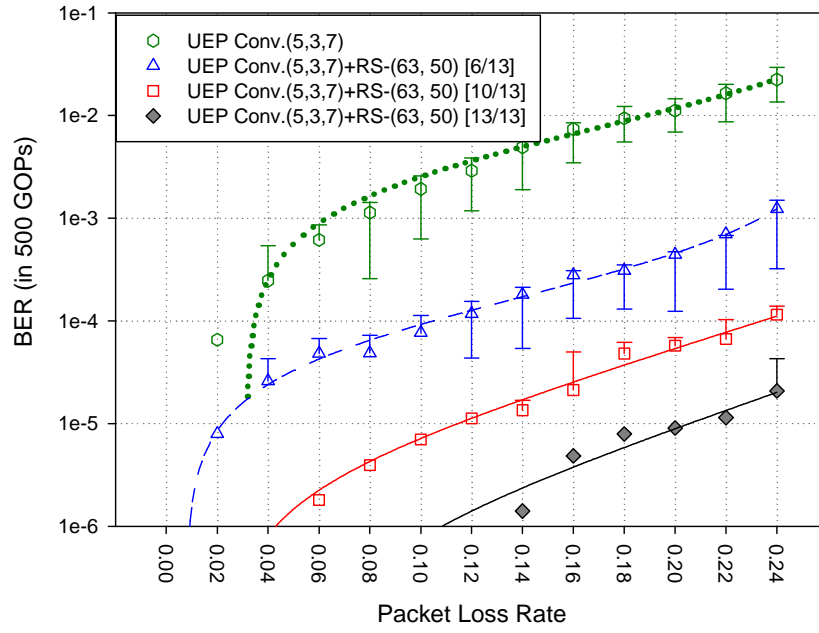


Figure. 29 : Comparison of Bit Error Rate among UEP Codes, football

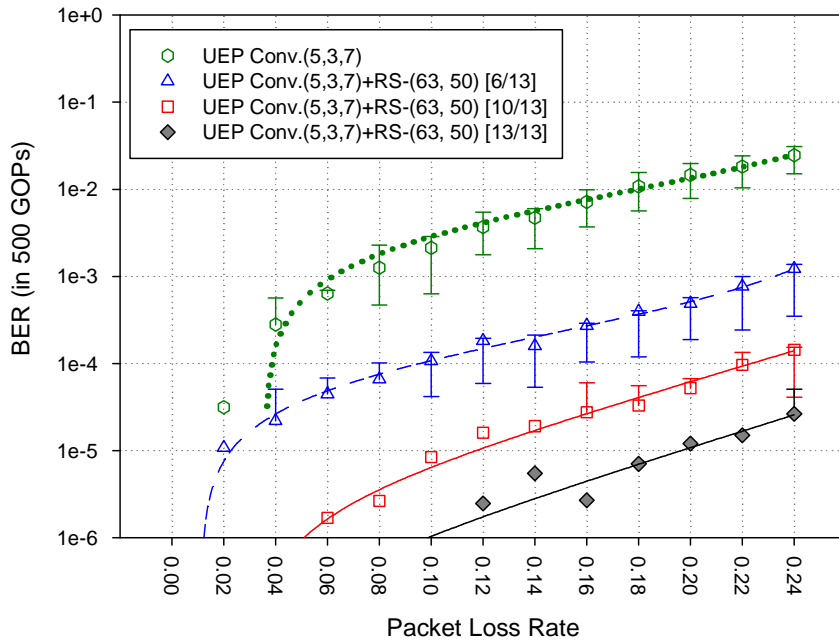


Figure. 30 : Comparison of Bit Error Rate among UEP Codes, foreman

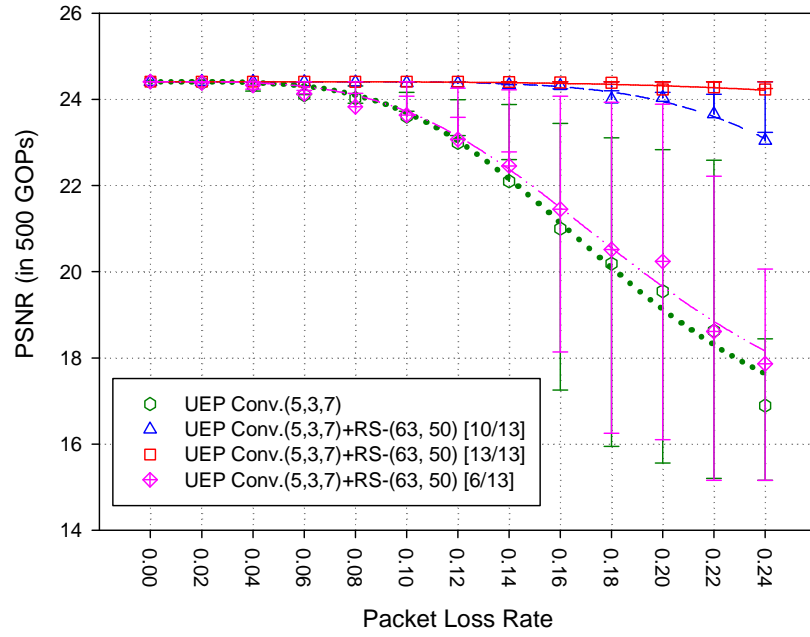


Figure. 31 : Latency-Distortion Tradeoffs of UEP Codes, football

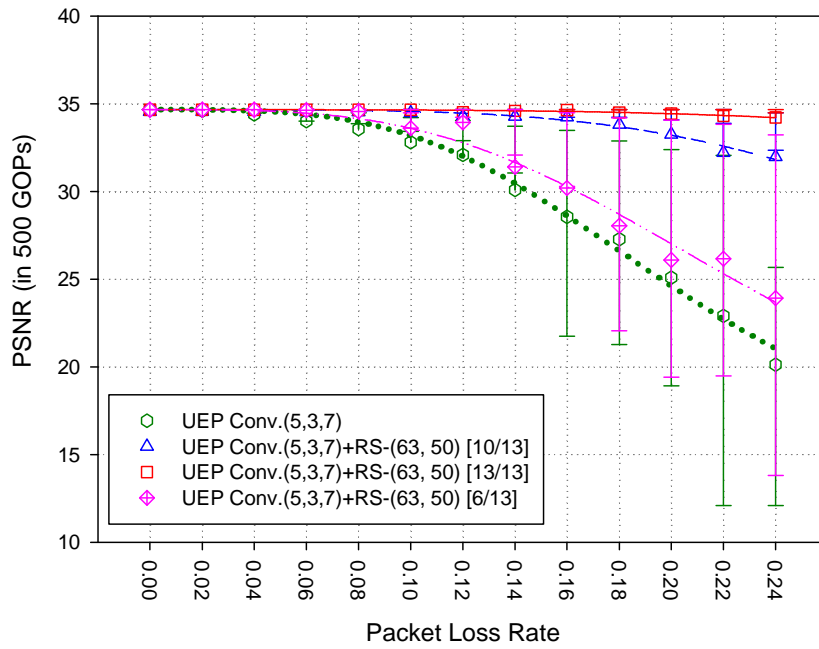


Figure. 32 : Latency-Distortion Tradeoffs of UEP Codes, foreman

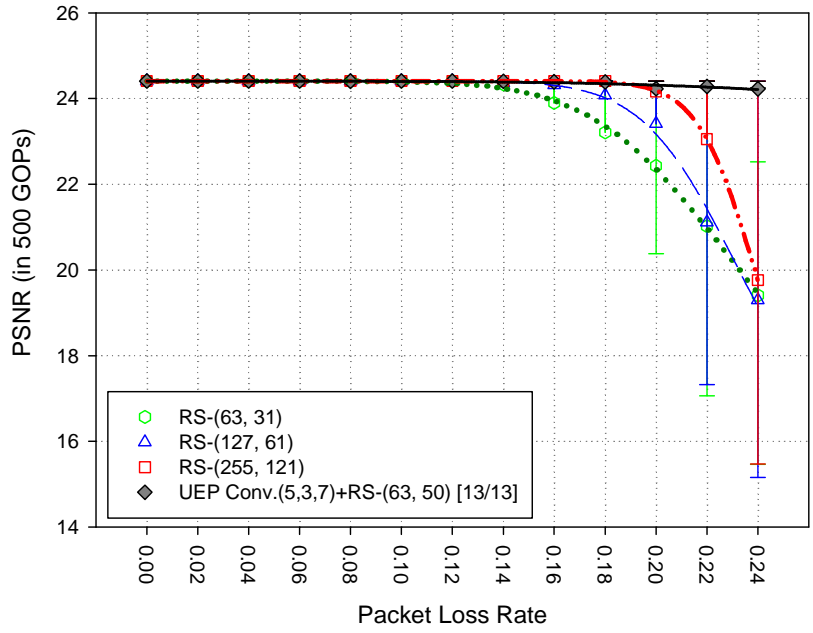


Figure. 33 : Comparison between Max. Latency UEP and RS Codes, football

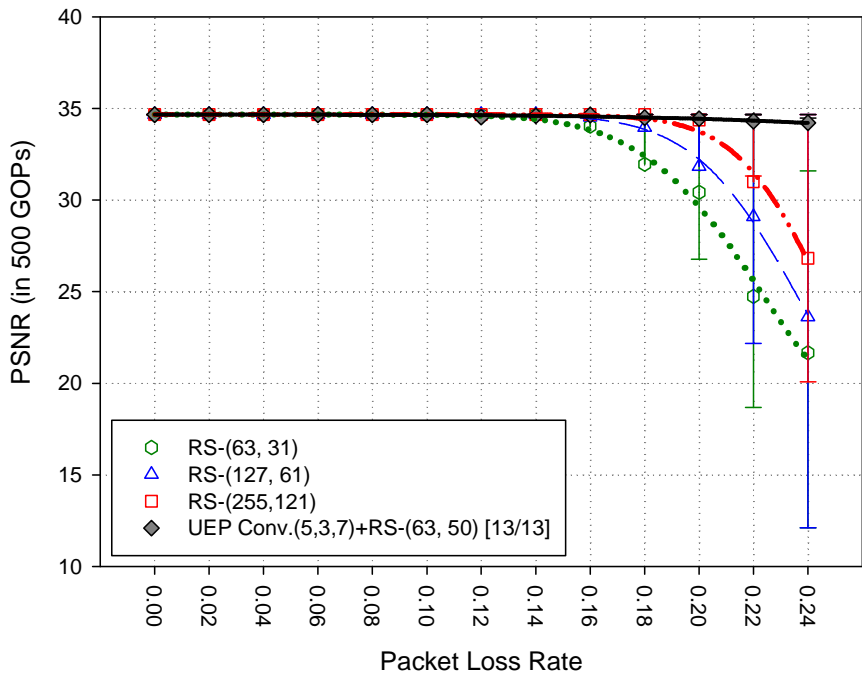


Figure. 34 : Comparison between Max. Latency UEP and RS Codes, foreman

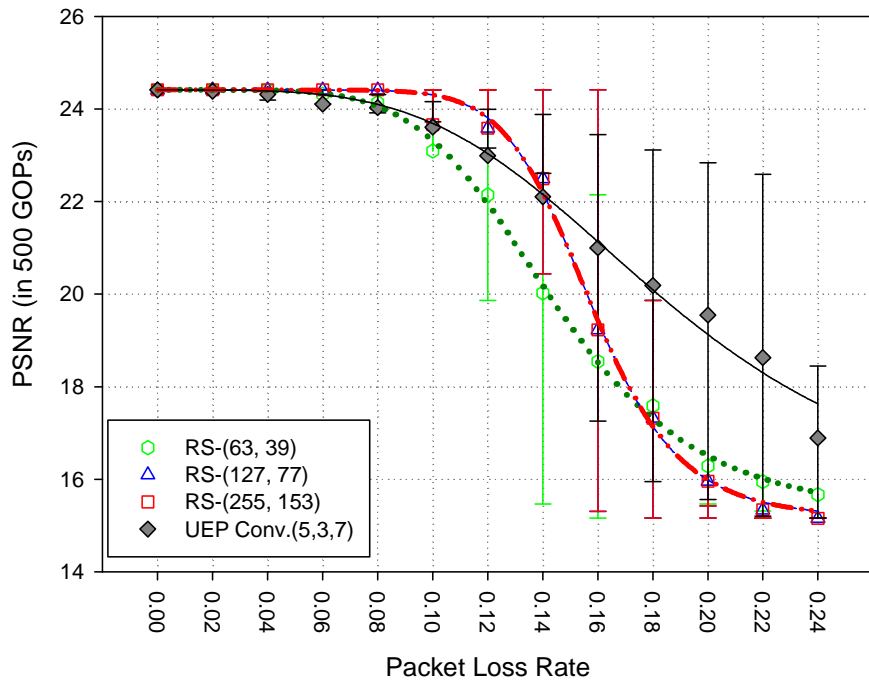


Figure. 35 : Comparison between Min. Latency UEP and RS Codes, football

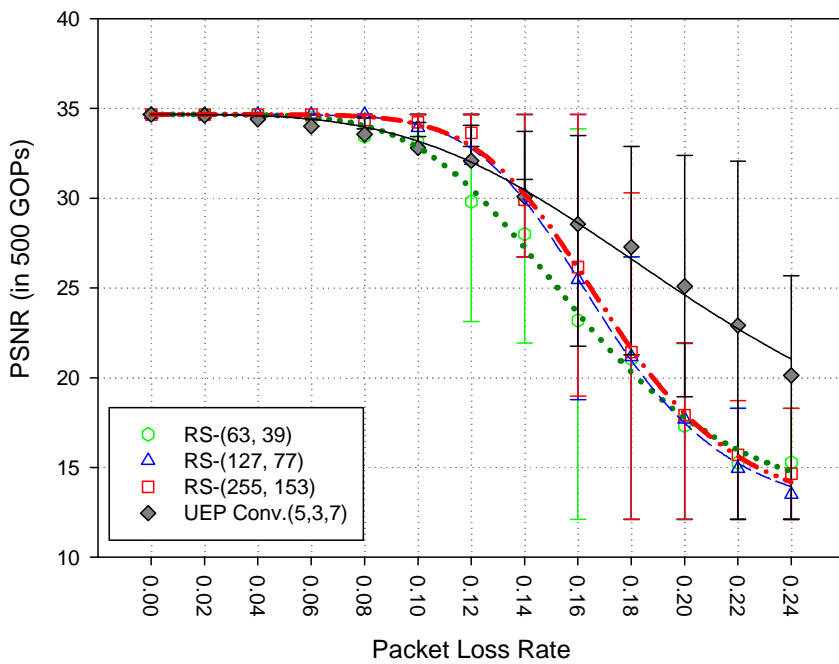


Figure. 36 : Comparison between Min. Latency UEP and RS Codes, foreman

6.2 Convolutional Codes + Equal Erasure Protected LT Codes

In the second series experiments, finite length LT codes is used in place of the Reed-Solomon Coder, our UEP encoder [Figure. 37 : System flow chart for UEP Conv. Precoder cascaded with LT rateless coder] is similar to the previous one, consisting of three components: (1) a three-level convolutional (N, k, M) UEP pre-coder with $N = 5, k = 3$ and $M = 7$ (2) a two-dimensional block interleaver and (3) a LT rateless coder (n, k) post-coder with $n = k \cdot (1 + \epsilon)$, ϵ is rateless overhead.

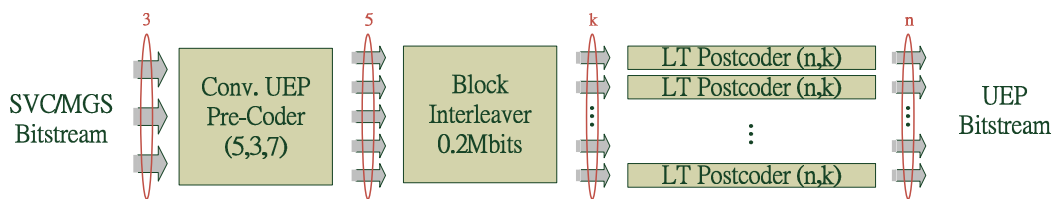


Figure. 37 : System flow chart for UEP Conv. Precoder cascaded with LT rateless coder

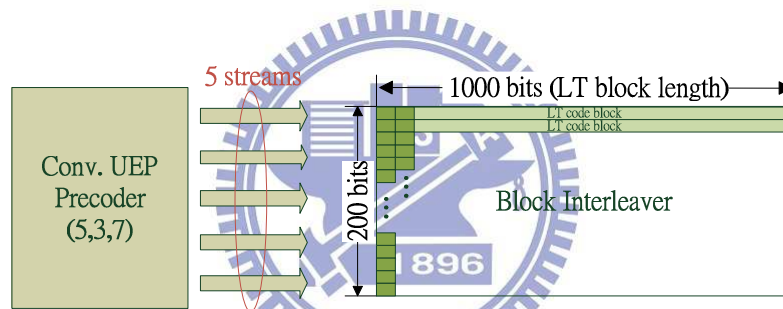


Figure. 38 : Block interleaver used in the Conv. UEP Precoder cascaded with LT rateless coder system

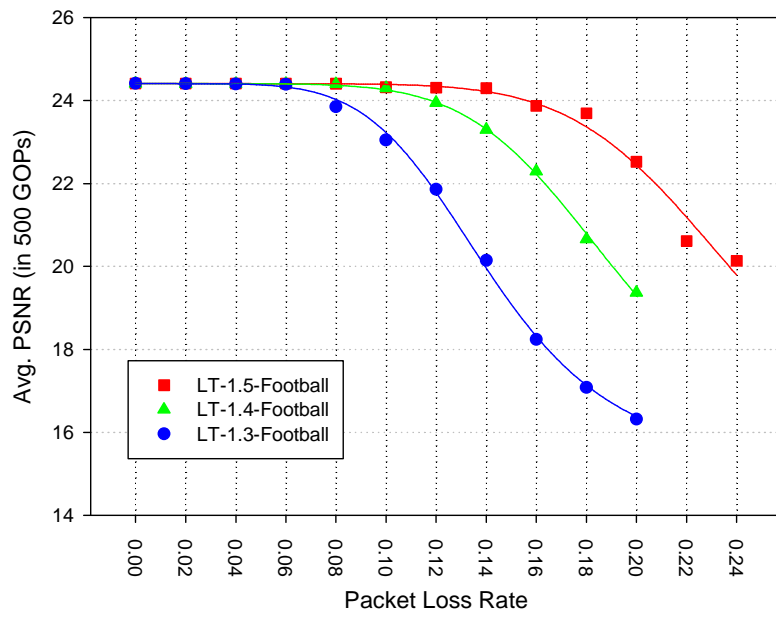


Figure. 39 : Latency-Distortion Tradeoffs of LT Codes, football

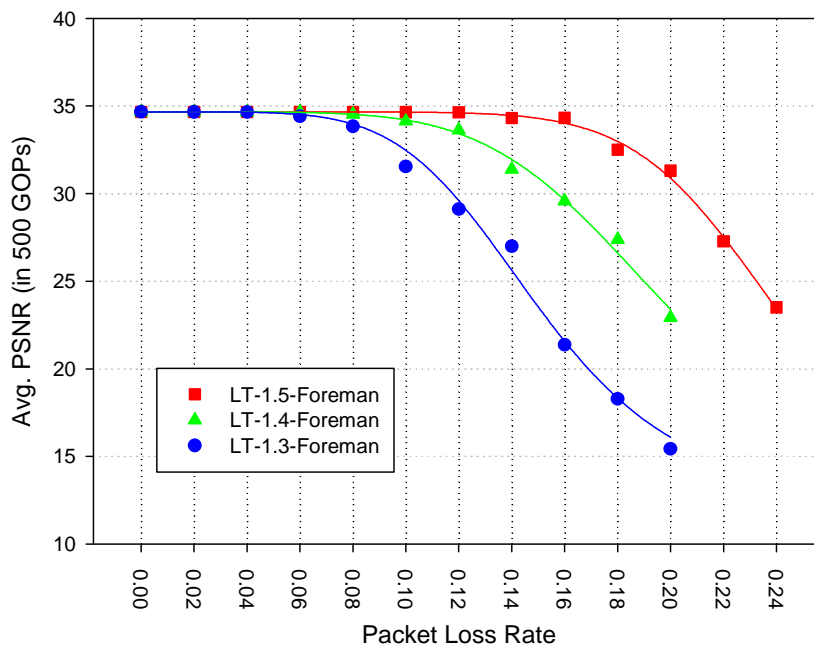


Figure. 40 : Latency-Distortion Tradeoffs of LT Codes, foreman

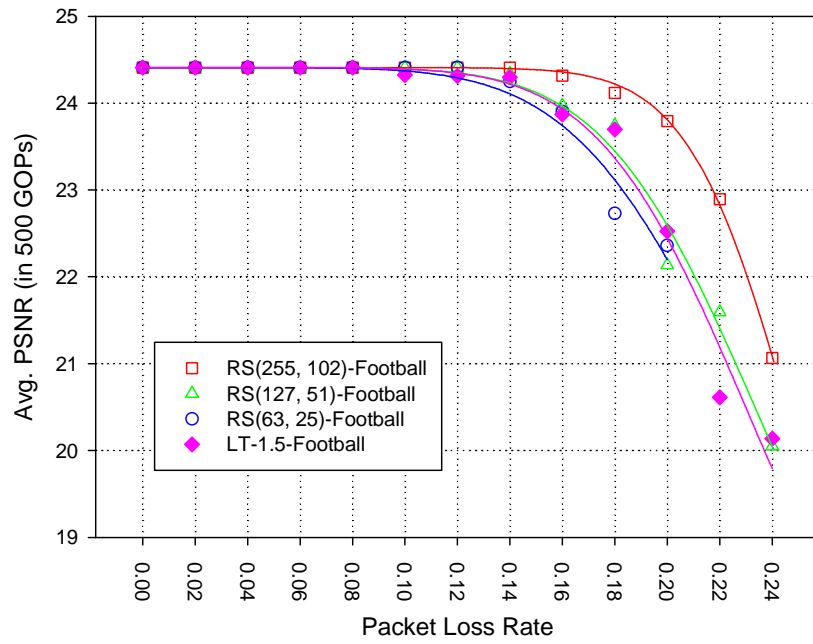


Figure. 41 : Comparison between Max. Latency LT code and RS Codes, football

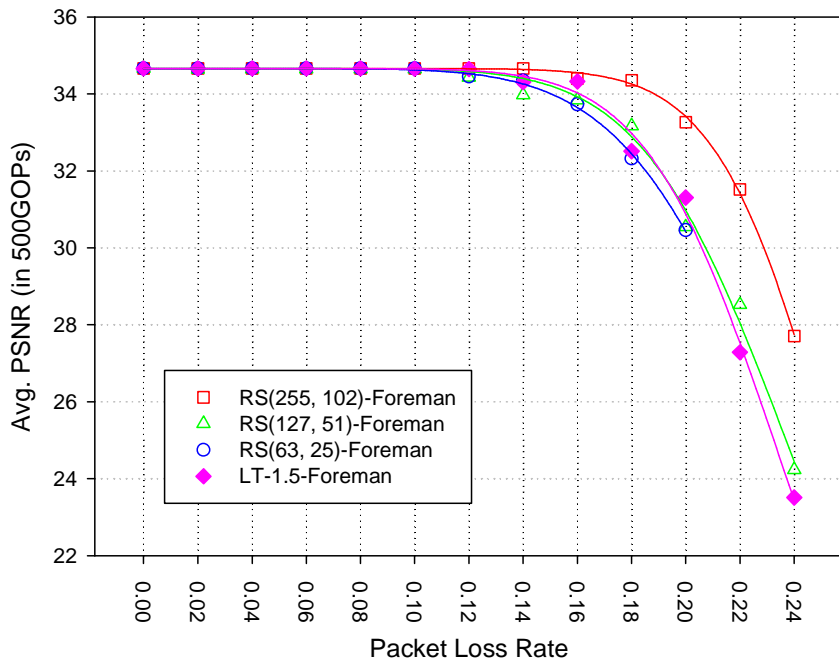


Figure. 42 : Comparison between Max. Latency LT code and RS Codes, foreman

6.3 Convolutional Codes + Unequal Erasure Protected LT Codes

In the third experiments series, we are trying to find out whether the LT codes with UEP ability will provide more layers as cascading with an UEP Precoder. The block size of LT Postcoder is 1000 and we compare performances for three different interleavers with different block size.

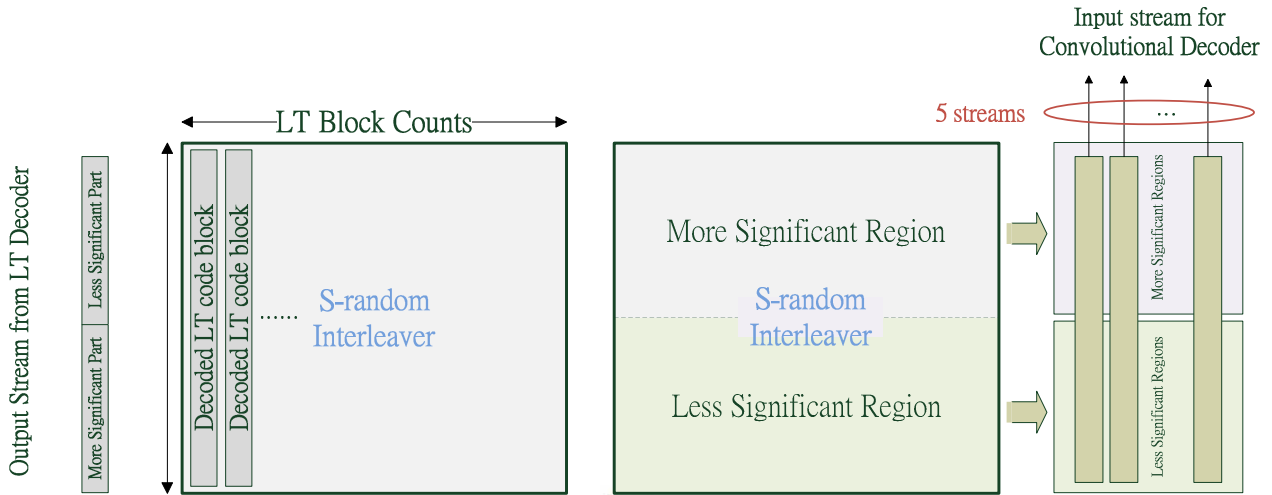


Figure. 43 : Decoding flow chart for UEP Precoder + UEP LT Postcoder

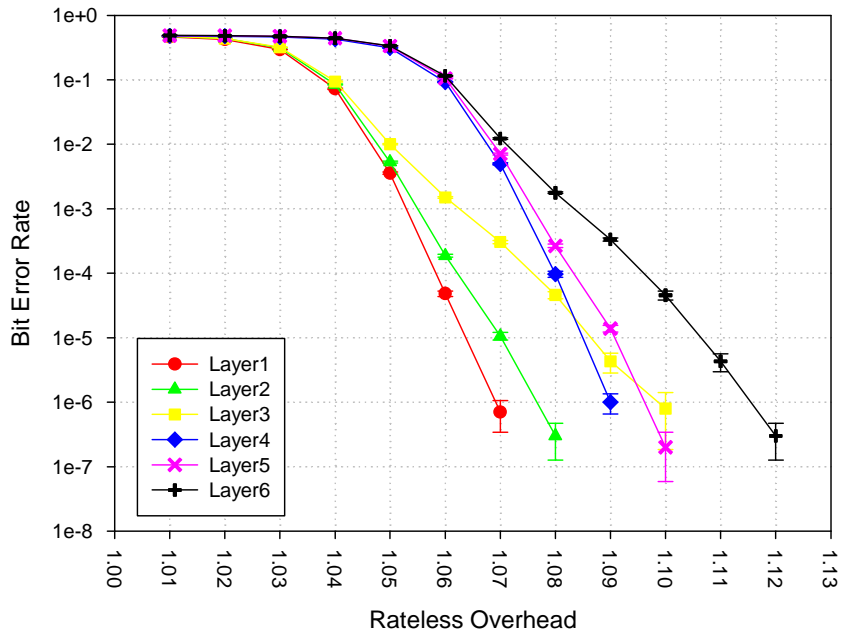


Figure. 44 : UEP Precoder + UEP LT Postcoder with interleaver size 100Kb

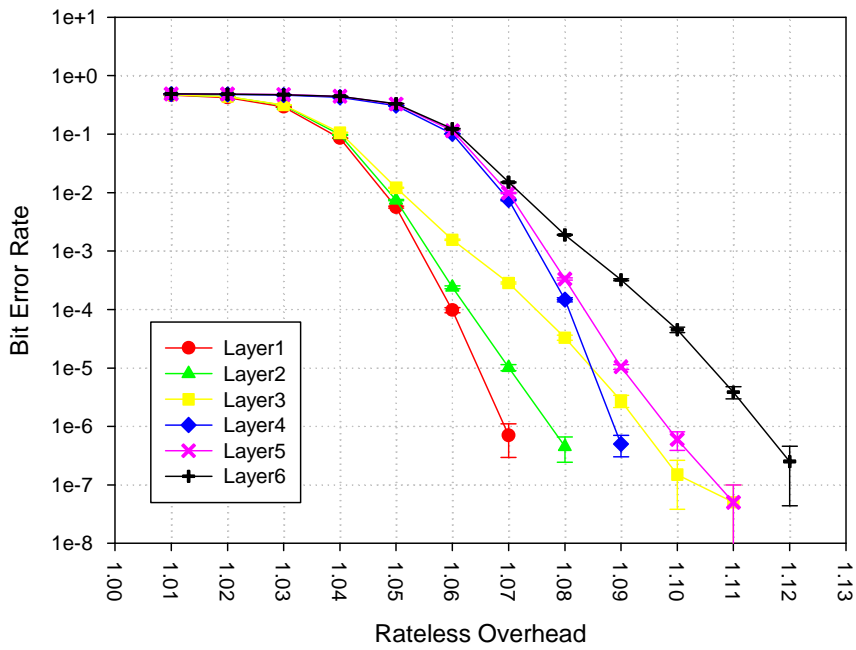


Figure. 45 : UEP Precoder + UEP LT Postcoder with interleaver size 50Kb

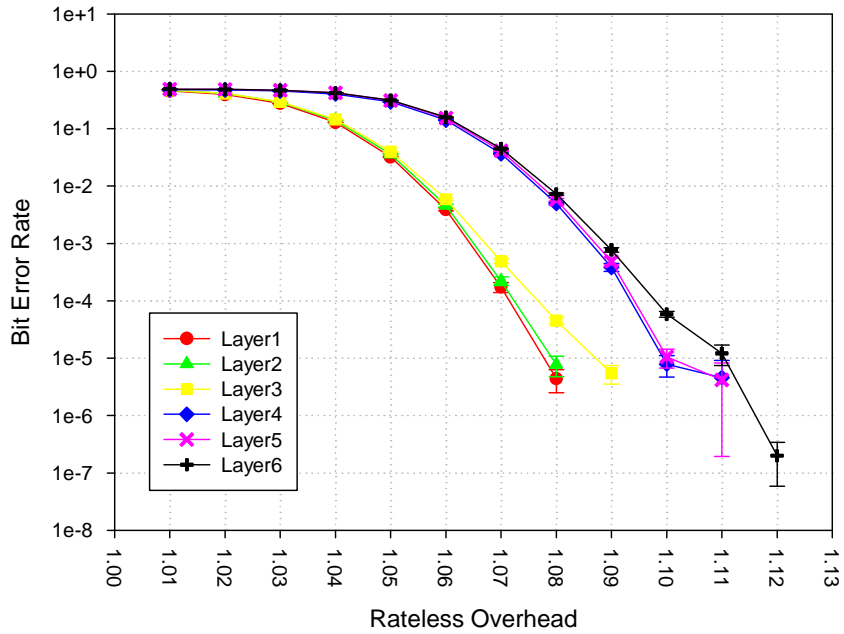


Figure. 46 : UEP Precoder + UEP LT Postcoder with interleaver size 10Kb

Chapter 7. Conclusions

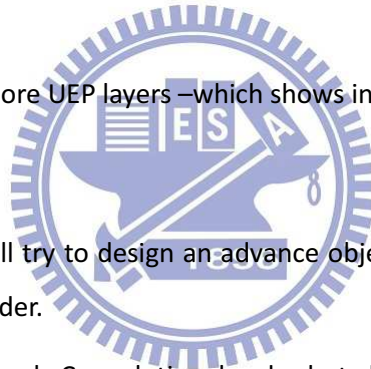
7.1 Accomplishment

We propose a rateless cascade channel coding system with Unequal Erasure Protection capability using on H.264/SVC transport for real time transport and design principles for realization. We claim that we have done following works,

1. Provides a novel method to optimize performance of LT codes –which show in **Chapter 4**.
2. Provides design principles for Multiplexer, in order to adjust our UEP performance for SVC coding scheme. –which shows in the **Chapter 5**.
3. Provides design principles for Interleaver, in order to eliminate burst error event in LT codes using S-random Interleaver and block interleaver. –which shows in the **Chapter 5**.
4. Provides a coding scheme with good UEP capability – which shows in experiments **Chapter 6.1** and **Chapter 6.2**,
5. Provides a coding scheme with more UEP layers –which shows in experiments in **Chapter 6.3**

7.2 Future Work

- ❖ To LT codes optimization, we shall try to design an advance objective function using in CMS-ES, and try different block size of rateless coder.
- ❖ To Precoder, we will consider not only Convolutional codes but also block codes or others,
- ❖ To Interleaver, we will try to figure out how to realize in other kinds of interleaver.



Reference

Bibliography

- [1] H.C. Huang, W.H. Peng, T. Chiang, and H.M. Hang, “Advances in the Scalable Amendment of H.264/SVC,” IEEE Communications Magazine, vol. 45, pp. 68 – 76, 2007.
- [2] H. Schwarz, D. Marpe, and T. Wiegand, “Overview of the Scalable Video Coding Extension of the H.264/AVC Standard,” IEEE International Conference on Image Processing (ICIP), October 2006.
- [3] T. Wiegand, G. Sullivan, J. Reichel, H. Schwarz, and M. Wien, “Joint Draft ITUT Rec. H.264 – ISO/IEC 14496-10/Amd.3 Scalable Video Coding,” ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, JVT-X201, July 2007.
- [4] Byers, J.W., Luby, M., Mitzenmacher, M., Rege, A. “A digital fountain approach to reliable distribution of bulk data”. In Proceedings of ACM SIGCOMM '98 (Vancouver, British Columbia, Canada, August 31 - September 04, 1998).
- [5] Byers, J.W., Luby, M., Mitzenmacher, M, “A Digital Fountain Approach to Asynchronous Reliable Multicast”. In IEEE Journal on Selected Areas of Communication (JSAC) 20(8), October 2002.
- [6] Chung-Hsuan Wang, Mao-Ching Chiu, Chi-chao Chao, “On Unequal Error Protection of Convolutional Codes From an Algebraic Perspective”, IEEE Transactions on Information Theory, 296-315, January, 2010
- [7] Nazanin Rahnavard, *Member, IEEE*, Badri N. Vellambi, *Student Member, IEEE*, and Faramarz Fekri, *Senior Member, IEEE*, “Rateless Codes With Unequal Error Protection Property”, IEEE Transactions on Information Theory, Vol 53, April, 2007

