國立交通大學

網路工程研究所

碩 士 論 文

使用於多媒體串流之重疊無限制比率前向糾錯碼分析模型

An Analytic Model of Overlapped Rateless Codes for Multimedia Streaming

研 究 生：張玉書

指導教授：蕭旭峯　教授

中 華 民 國　九 十 八 年 九 月

使用於多媒體串流之重疊無限制比率前向糾錯碼分析模型
An Analytic Model of Overlapped Rateless Codes for Multimedia Streaming

研 究 生：張玉書　　Student：Yu-Shu Chang

指導教授：蕭旭峯　　Advisor：Hsu-Feng Hsiao

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

Sep 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年九月

使用於多媒體串流之重疊無限制比率前向糾錯碼分析模型

研究生：張玉書　　　　　　　　指導教授：蕭旭峯

國立交通大學網路工程研究所

摘要

多媒體串流的資料是有可能源源不絕的，而現有的 TCP 重新傳送機制來傳送封包，會使得時間會有延遲。使用前向糾錯碼來取代 TCP 傳送多媒體串流資料是一個很好的選擇。在傳送資料前，我們無法得知通道的封包遺失率，所以無限制比率前向糾錯碼很適合應用於此環境，再加上此前向糾錯碼的編碼與解碼速度相較快於 Reed-Solomon 碼，使用此前向糾錯碼來傳送多媒體串流的資料是有利的。對於多媒體串流資料，我們先提出一種分析模型於一小段時間內的資料上，來分析有權重的選擇編碼方式，可否能達到更低的解碼失敗率。在有此分析模型之後，放到整個多媒體串流來預估怎樣的權重序列，才會使得整體的解碼失敗率降低。最後用實驗來驗證有權重方式的多媒體串流會優於無權重方式。

An Analytic Model of Overlapped Rateless Codes for Multimedia Streaming

Student：Yu-Shu Chang Advisor：Hsu-Feng Hsiao

Department of Computer Science

National Chiao Tung University

Abstract

The amount of the multimedia streaming data could be endless. If we use TCP scheme to transmit, it may cause long delay due to possible retransmission. Forward Error Correction (FEC) codes are very suitable candidates. Before transmitting the encoded packets over the error prone channel, we do not know the packet loss rate. So the rateless codes could apply in this environment with its rateless feature, and the complexity of the rateless codes' encoding / decoding is much less than the well-known Reed-Solomon codes. For the multimedia streaming, we propose an analytic model of the unequal overlapped rateless codes on a short time data. From the simulation results, the weighted selection could achieve lower decoding failure probability. With this analytic model, better encoding strategy for the multimedia streaming could be formed to ensure low decoding failure probability across multiple sections. The experiment results show that the weighted selection is better than the uniform selection in the multimedia streaming.

Acknowledgement

首先要誠摯地感謝指導教授蕭旭峯老師，在無數次的挫折中，老師總能抽絲剝繭地帶領我慢慢找出問題的癥結，點醒應該要注意的方向。老師也教導了我許多求學上的正確態度，讓我獲益良多，最後終於完成這篇論文。其次要感謝所有口試委員，在百忙之中給予我許多寶貴的建議，使得本篇論文能夠更完整而嚴謹。

在交大的網路與訊號處理實驗室裡，不管是學術上的討論，或是生活點滴的關心，感謝所有的學長姐、同學、學弟妹，因爲你們讓研究生活變得多彩多姿。最後，感謝父母從小的養育與教誨，給予我良好的學習環境，讓我的求學生活無後顧之憂，並以此文獻給我摯愛的家人。

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Preface

Most communication in Internet Protocol (IP) networks is based on Transmission Control Protocol (TCP), which provides a reliable end-to-end data delivery service. TCP treats data as an ordered sequence of packets to transmit to the receivers. TCP uses retransmission on the lost packets to guarantee that the receivers will have the original file. However the retransmission scheme has its restrictions in a number of applications. Consider about the problem of distributing a file to several receivers, some of which may be short of bandwidth resource.

Rateless codes are new class of forward error correction (FEC) erasure codes. Digital Fountain codes [1] [2] [3], and Online codes [4] are examples of such codes with many desirable features. The idea behind the rateless codes is that every receiver continues collecting the encoded packets until the decoding process could be finished successfully. Unlike the traditional codes, rateless codes on erasure channels [5] do not assume any knowledge about the channel. Low Density Parity Check (LDPC) codes [6] [7] and Tornado codes [8] are also providing this idea, but the encoding complexity could be large for generating endless encoded packets for channel with large packet loss rate.

1.2 Motivation

While downloading and storing a complete multimedia file may not be suitable for Reed-Solomon codes [9] due to high encoding and decoding complexity, rateless codes could suitably be done with low encoding and decoding complexity. Because the major problem is the time delay, the multimedia streaming file which in the period of playback must be decoded from the received encoded packets before the multimedia streaming file begins to play as soon as possible.

Many solutions partition the multimedia streaming file into sections. The length of a section is determined by the application. When using the section-based system, the current section is being received, while an earlier section is played. Multimedia streaming application might impose some constraints on the received encoded packets. In a communication system that does not guarantee the correct order of packets, if the received encoded packet is far away from the playback, it might be dropped due to limited memory size. Similarly, if the received encoded packets contain in a section that has been played, these packets are useless.

In [10], the author proposed an innovative scheme for multimedia streaming by connecting one more sections to gradually provide the encoded packets. We use this scheme as the multimedia streaming structure of our proposed analytic model.

1.3 Research Objectives

The objectives of our framework could be summarized as the following:

- **Analyze the proposed unequal overlapped rateless codes for the single section to obtain the decoding failure probability.**

Based on [10] [11] [12], one section could be partitioned into two parts. The different part could have different decoding failure probability. We could use And-Or tree [13] to model and analyze the expected decoding failure probability for each part in the single section. This proposed analytic model will then be applied in the scenario of the multiple sections.

- **Find optimal parameter sequence in the multiple sections.**

After having the analytic model for the single section, we could use the local minimization approach and dynamic programming approach to find the weights sequence for the overall of the minimal decoding failure probability in the multimedia streaming.

1.4 Outline of the thesis

In Chapter 2, we discuss the background and related work of rateless codes. In Chapter 3, we propose the unequal overlapped rateless codes, and then an analytic model based on And-Or tree analysis is derived. Finally, we use the local minimization approach and dynamic programming approach to find the optimal sequence of weights for the overall of minimal decoding failure probability in the multimedia streaming with the multiple sections. Our experiment results in Chapter 4. Chapter 5, we give the conclusion for our proposed method.

Chapter 2 Background and Related Works

Before the proposed method, we describe how to partition the multimedia streaming into sections with rateless codes and introduce And-Or tree lemma as our main tool to analyze the model of the single section.

2.1 Multimedia Streaming in Section-based System

Normally, in order to avoid the unacceptable delay when playing the multimedia streaming, the entire multimedia streaming will be partitioned into the sections as shown in Figure 2-1.

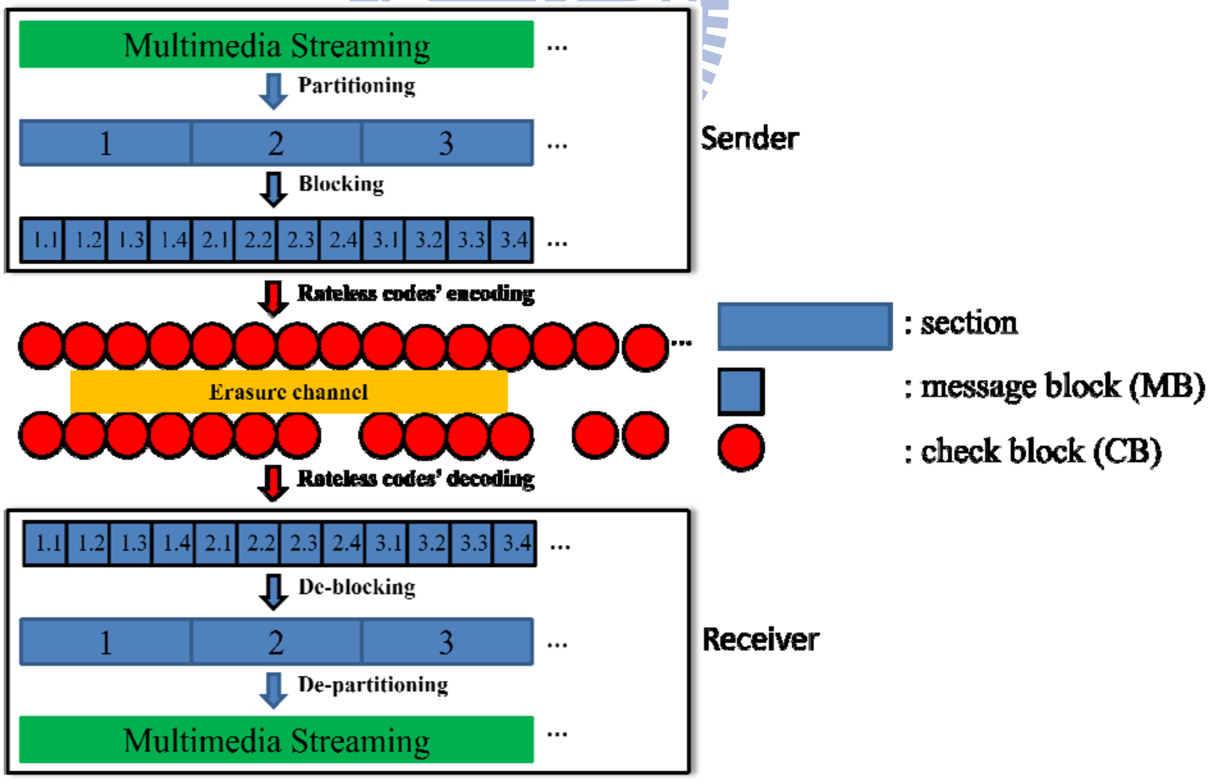


Figure 2-1: Multimedia streaming in the section-based system.

The entire multimedia streaming could be partitioned into the sections with equal length. The sender uses rateless codes to encode each section separately, and transmits each encoded block (also called check block) to the receiver on binary erasure channel [5]. After the receiver obtains sufficient check blocks, the decoder of rateless codes will recover the number of the message blocks as many as possible.

2.2 Luby Transform (LT) codes

In this section, we briefly review LT codes that are the well-known examples of rateless codes. LT codes were developed by M. Luby in a landmark paper in 2002 [1]. LT codes have low encoding and decoding complexity. Unlike the traditional codes, LT codes do not assume any knowledge about the channel erasure probability. It means that the rate of the codes does not need to be fixed before encoding. LT codes are universal that the length of the check block could be arbitrary from one-bit to any *i*-bit. The check blocks could be generated on the fly. LT codes only use exclusive or (XOR) operation to generate potentially infinite check blocks and the decoding process also only uses XOR operation to decode. LT codes are very efficient as the file length grows. Now, we could view the LT encoding and decoding process as the bipartite graph and explain the encoding / decoding process.

2.2.1 LT Encoding

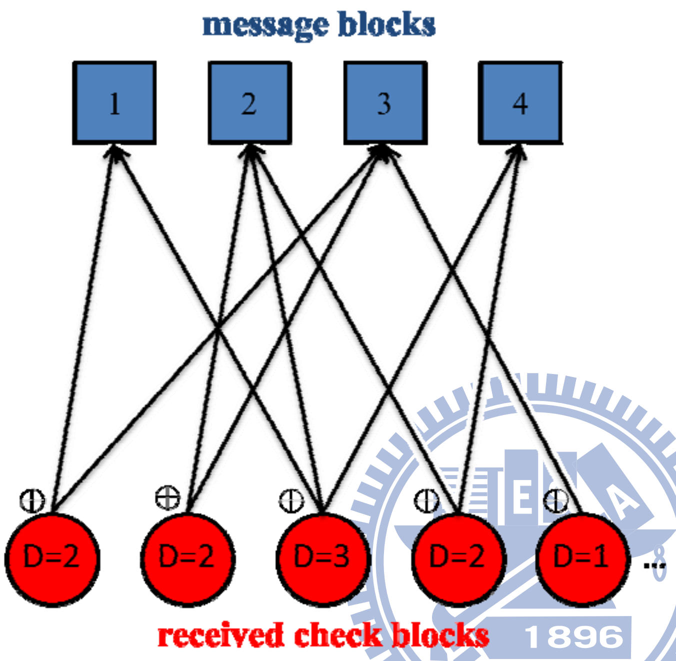
First, it divides the original file into the message blocks (MB) with fixed length. The encoding process is simple. Each check block is independently generated from the message blocks as follows:

1. Pick a degree d according to a degree distribution $\Omega(x)$.

2. Choose uniformly d distinct message blocks as the neighbors of this check block.

3. Set the value of this check block to be the XOR (⊕) of the *d* neighbors.

The degree distribution *Ω(x)* of LT codes is a probability distribution and Ω_d is the probability of generating a check block consisting of *d* message blocks. The encoding process is shown in Figure 2-2.



Degree Distribution Table

Degree	Probability
1	0.1
2	0.5
3	0.2
...	...
Max_Degree	0.01

Figure 2-2: LT encoding process.

In Figure 2-2, the value of the most left encoded block is XORed of the message block 1 and the message block 3.

2.2.2 LT Decoding

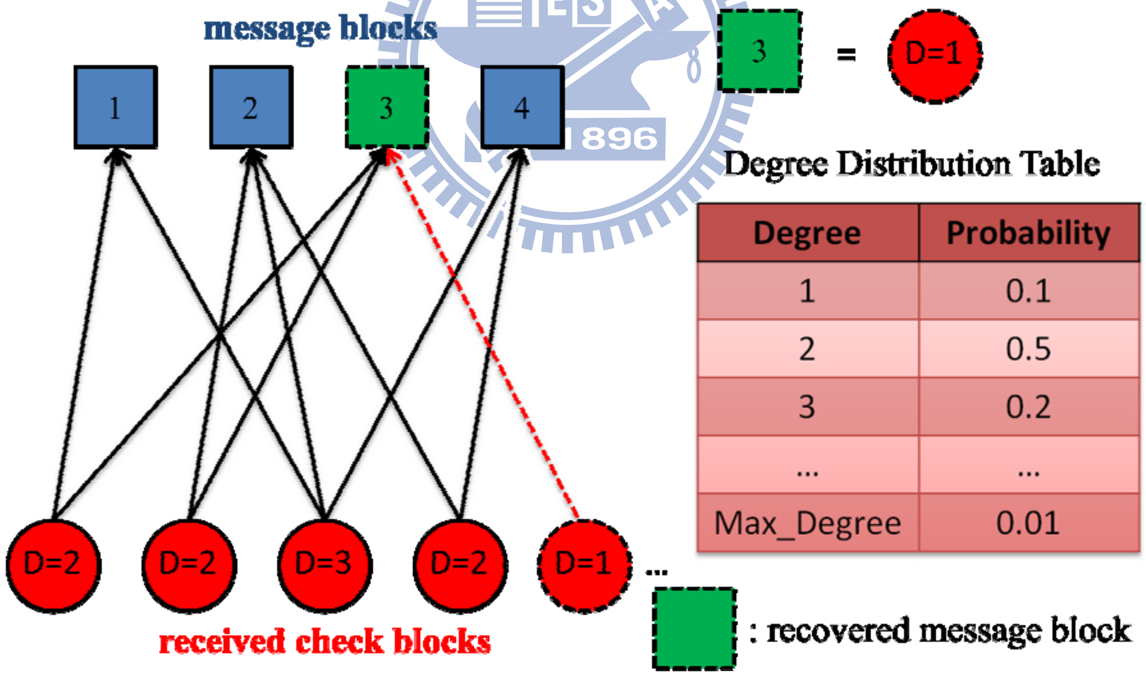
The decoding process needs the information of the degree value of each received check block and the information of which message blocks are XORed together in each check block. There are many ways of communicating this information to the decoder. For example, the encoder and the decoder use the same key or seed number to a random number generator that generates the same degree value and the neighbor list for each received block. The reader may refer to [1] for more details. Follow [2] and [4], we could think the decoding graph as the bipartite graph and the decoding process could be called the iterative decoding or belief propagation decoding. The decoding process could be described as follows:

1. Find a check block *c*, all of whose message blocks are recovered, except for one. If it exists, we could call this message block *m*.

2. Set $m = c \oplus m_1 \oplus \ldots \oplus m_{d-1}$, where $m_1, \ldots, m_{d-1}$ are the recovered message blocks that are adjacent to *c*. Then, we set the message block *m* is recovered.

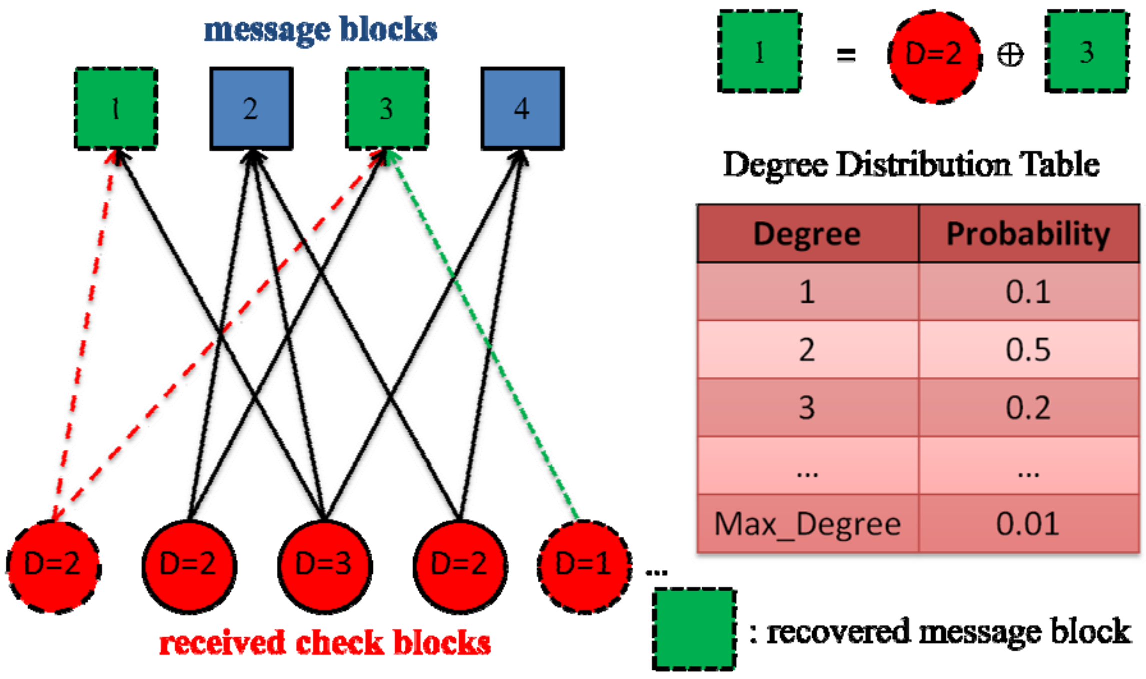
3. Repeat step 1 and step 2 until we could not find such *c*.

The decoding succeeds if all message blocks are recovered, else the decoding fails. The decoding process is shown in Figure 2-3 (a), Figure 2-3 (b).



Degree	Probability
1	0.1
2	0.5
3	0.2
…	…
Max_Degree	0.01

Figure 2-3 (a): We could find the most right check block for recovering message block 3.



Degree	Probability
1	0.1
2	0.5
3	0.2
...	...
Max_Degree	0.01

Figure 2-3 (b): The most left check block could recover message block 1, and continues.

Figure 2-3: LT decoding process.

After recovering the message block 3, we could recover the message block 1. And then we could recover message block 2; finally, we could recover message block 4 to complete the decoding process. We know that LT codes have simple encoding and decoding complexity, because the encoding and decoding complexity depends on the edges in the entire bipartite graph. And it turns out that the degree probability distribution is a critical part of the design. Good asymptotical degree distributions for them were also developed in [1] [2]. For finite-length LT codes, good degree distributions have been proposed in [14] [15].

LT codes are also used to protect the subsets of the message blocks of different importance. We will introduce in section 2-4.

2.3 And-Or Tree Analysis

LT codes could be analyzed by And-Or tree [13] [16] [17]. This section explains the structure of an And-Or tree and the proof of And-Or tree Lemma.

2.3.1 Construction of And-Or Tree

An And-Or tree T_L is a randomly generated tree of depth $2L$ (L is a constant). The root of this tree is at depth 0. Its children are at depth 1, and their children are at depth 2, so on and so forth. Each leaf-node at depth $2L$ will be assigned the value 0 or 1 independently. Each node at depth $0, 2, 4, \cdots, 2L-2$ will be labeled with OR-nodes (and it is evaluated by the "OR" operation of its children), and each node at depth $1, 3, 5, \cdots, 2L-1$ will be labeled with AND-nodes (and it is evaluated by the "AND" operation of its children). The tree will be generated from top to bottom, starting with the root node. Each node will independently choose how many children to have. Let $(\alpha_0, \alpha_1, \cdots, \alpha_A)$ and $(\beta_0, \beta_1, \cdots, \beta_B)$ be the probability distribution and $\sum_{i=0}^{A} \alpha_i = 1, \sum_{j=0}^{B} \beta_j = 1$. Each OR-node is chosen to have i children with probability α_i, and each AND-node is chosen to have j children with probability β_j. The OR-node with no children is assumed to have a value of 0, and the AND-node with no children is assumed to have a value of 1. We are going to be interested in the probability that the root node is evaluated as 0. An example of And-Or tree is shown in Figure 2-4.

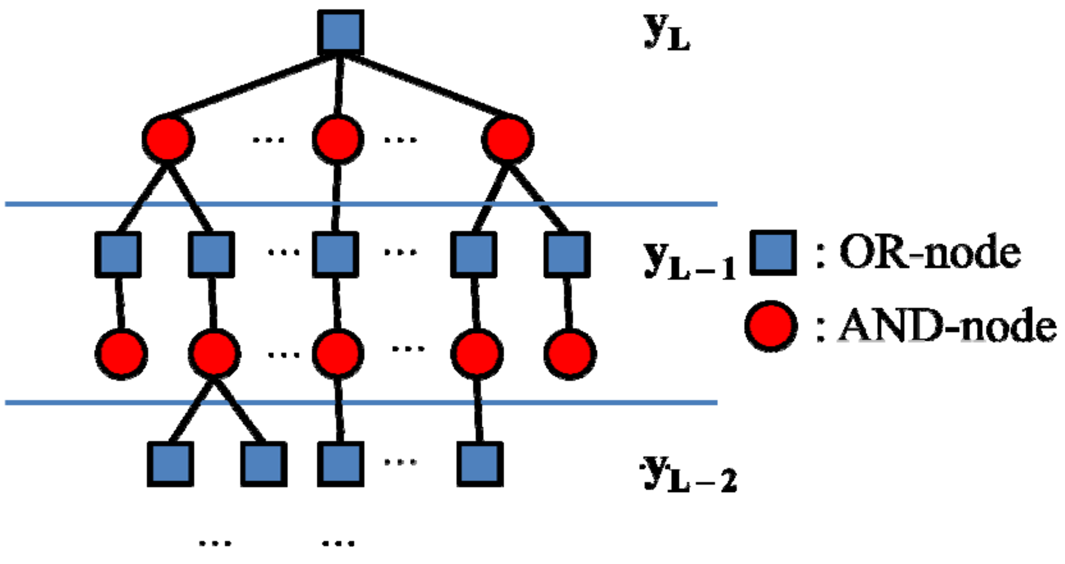


Figure 2-4: An example of And-Or tree.

Our goal is to compute y_L – the probability that T_L's root is evaluated as 0. Because the OR-nodes at depth 2 in T_L form the T_{L-1} And-Or trees, we could compute y_L recursively as a function of y_{L-1}. y_{L-1} is the probability that the root of a And-Or tree T_{L-1} is evaluated as 0. We could use the following lemma from [13]:

The And-Or Tree Lemma: The probability y_L that the root node of a T_L And-Or tree is evaluated as 0 is $y_L = f(y_{L-1})$, where y_{L-1} is the probability that the root node of a And-Or tree T_{L-1} is evaluated as 0. y_0 is the probability that it is 0.

$$f(x) = \alpha\big(1-\beta(1-x)\big), \text{for}$$
$$\alpha(x) = \sum_{i=0}^{A} \alpha_i x^i \; and \; \beta(x) = \sum_{j=0}^{B} \beta_j x^j. \tag{1}$$

2.3.2 Proof of And-Or Tree Lemma

In order to model the LT decoding process via this And-Or tree based on T_L, we need to use the probability distribution on the number of children of OR-nodes and AND-nodes. λ_d is the probability that a uniformly chosen edge is attached to an OR-node of degree d. ρ_d is the probability that a uniformly chosen edge is attached to an AND-node of degree d. From [13], we have

$$\lambda_d = \frac{d \cdot \alpha_d}{\sum_{j=1}^{A} j \cdot \alpha_j} \; and \; \rho_d = \frac{d \cdot \beta_d}{\sum_{j=1}^{B} \beta_j}. \tag{2}$$

Further, λ_d is the probability that the edge is connected to the OR-node that has d–1 children; ρ_d is the probability that the edge is connected to the AND-node that has d–1 children.

The decoding process could be analyzed by the same way as the belief propagation [18] [19]. At each round, the edge will send a message to the nodes. The messages are sent along the edges from AND-nodes to OR-nodes, and then from OR-nodes to AND-nodes in each round. Let u_i be the probability that the message sent from OR-node to AND-node at round i of the algorithm is 0. h_i is the probability that the message sent from AND-node to OR-node at round i of the algorithm is 0. For the event that the OR-node is of degree d, we have

$$u_{i+1} = h_i^{d-1}. \tag{3}$$

Indeed, a message from an OR-node v to an AND-node w is 0 if and only if v was labeled 0 and all the messages coming from the neighboring AND-node other than w are 0 in previous round, where h_i^{d-1} is obtained under the independence assumption. For the event that the AND-node is of degree d, we have

$$h_i = 1 - (1 - u_i)^{d-1}. \tag{4}$$

Because we could know the AND-node w sends a message 1 to the OR-node v if and only if all the neighboring OR-nodes except for v send a message 1 to w, the probability is

$$(1 - u_i)^{d-1}. \tag{5}$$

These recursions are not in a usable form yet since their condition is on the degree of the OR-nodes and AND-nodes. We set

$$\lambda(x) = \sum_{d=1} \lambda_d x^{d-1} \ and \ \rho(x) = \sum_{d=1} \rho_d x^{d-1}. \tag{6}$$

Then

$$u_{i+1} = \lambda\big(1 - \rho(1 - x)\big). \tag{7}$$

Let us analyze the probability y_L that the root of this And-Or tree T_L is 0. According to And-Or tree lemma, we would like that y_L is decreasing to δ as L grows.

$$\begin{gathered} \lambda\big(1 - \rho(1 - x)\big) < x, for\ x \in [\delta, 1], for \\ \lambda(x) = \sum_{d=1} \lambda_d x^{d-1} \ and \ \rho(x) = \sum_{d=1} \rho_d x^{d-1}. \end{gathered} \tag{8}$$

If L is a constant, then the number of nodes in the tree T_L is also a constant.

2.4 Expanding Window Fountain Codes

In [20], the author proposed an idea to deal with the subsets of message blocks of different importance. If there have k message blocks that we want to encode with different importance. We assume that the numbers s_1, s_2,⋯, s_r, such that s_1+s_2+…+s_r = k. s_i is the subset of the message blocks and $Z_i = \sum_{j=1}^{i} s_j$. The window i consists of the message blocks in Z_i. The example is as shown in Figure 2-5:

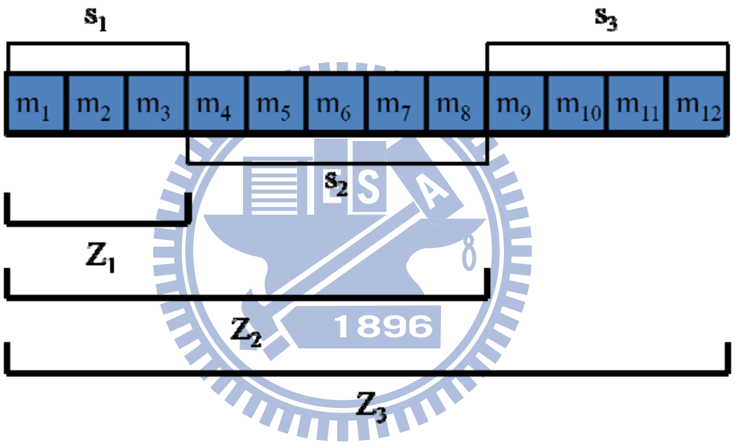


Figure 2-5: The example of Expanding Window Fountain Codes.

The encoding process is using LT codes. The difference from the traditional LT codes is choosing the message blocks non-uniformly. Before generating a check block, we need to decide which window we select, and then using standard LT codes’ encoding process to encode. Γ_i is the probability that the window i is chosen. If some of message blocks appear in more windows, it stands for these blocks being selected probability are larger than other message blocks. We could know that the most important message blocks are contained in more windows, so the most important subset is s_1. The author uses the windowing method to achieve the subset of message blocks of different importance. This method could also apply in multicast environment [21] by the same author.

2.5 Sliding-Window Digital Fountain Codes

In [10], the author proposed an idea by applying the sliding window scheme on the multimedia streaming to virtually extend the number of message block, and therefore to enhance the performance of rateless codes by reducing the decoding overhead and decreasing unrecovered message blocks. In a traditional section-based system with rateless codes, each section won't have any relation; but in a sliding window scheme, each section could related to neighboring sections. We set the length of one section equal to the window size in traditional section-based, as shown in Figure 2-6.

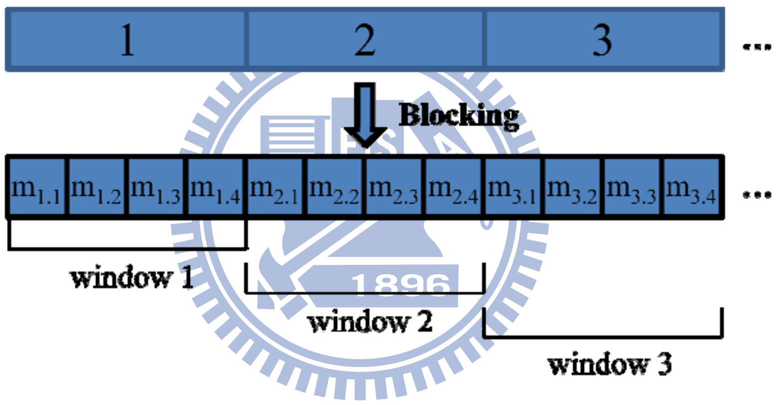

1
2
3
...
Blocking
$m_{1.1}$ $m_{1.2}$ $m_{1.3}$ $m_{1.4}$ $m_{2.1}$ $m_{2.2}$ $m_{2.3}$ $m_{2.4}$ $m_{3.1}$ $m_{3.2}$ $m_{3.3}$ $m_{3.4}$
...
window 1
window 2
window 3

Figure 2-6: Traditional section-based system.

In Figure 2-6, the rateless codes will encode each window in the traditional section-based system, and each window won't have any relation.

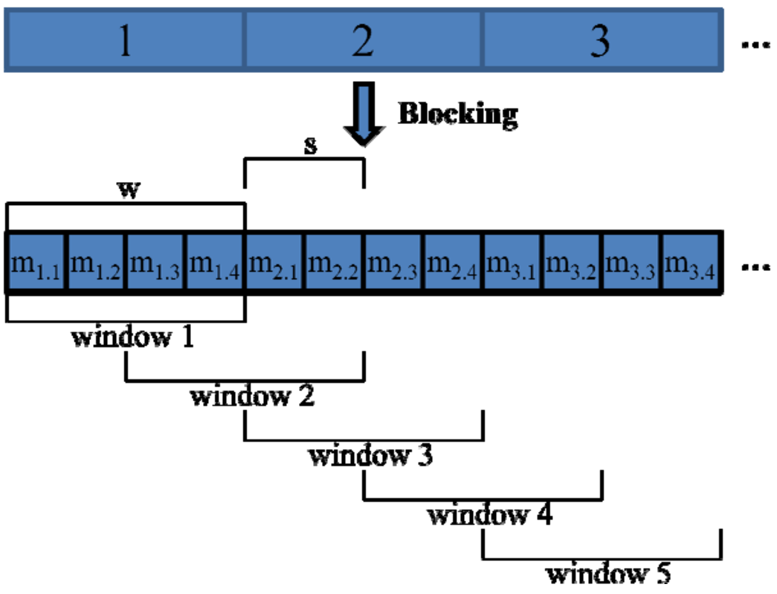


Figure 2-7: Sliding window scheme.

In Figure 2-7, we could know about the sliding window scheme. The rateless codes will encode each sliding window, and each sliding window has the relation of encoding and decoding process.

The sliding window size w should be chosen carefully. w should be as large as possible to achieve a small overhead by the asymptotic performance; however, w should be as small as possible to avoid the unacceptable delay. The sliding window movement s should also be chosen carefully, which equals the number of old message blocks discarded and new message blocks considered when the sliding window shifts to the successive sliding window. The sliding window movement s also determines the proportion of the overlapped message blocks between the preceding and the successive sliding windows. If s decreases, the same portion of the message blocks will be encoded into more successive sliding windows. For example, we could see the Figure 2-7. If the sliding window movement s is 2, and then the message blocks $m_{1.3}$ and $m_{1.4}$ will be encoded in sliding window 1 and sliding window 2. If the sliding window movement s is 1, the message block $m_{1.3}$ and $m_{1.4}$ will be encoded in sliding window 1, sliding window 2 and sliding window 3. This change will cause the decoder of the rateless codes virtually processes on a larger portion of message blocks. For example, we could also see the Figure 2-7. The decoder of the rateless codes in sliding window 1 will be in charge of decoding $m_{1.1}$, $m_{1.2}$, $m_{1.3}$, $m_{1.4}$. In sliding window 2, the decoder will be in charge of decoding $m_{1.1}$, $m_{1.2}$, $m_{1.3}$, $m_{1.4}$, $m_{1.5}$, $m_{1.6}$. If the buffer size of the receiver is unlimited, the following sliding window will be in charge of decoding more and more message blocks.

The total number N_s of sliding window of size w in the entire multimedia streaming with k message blocks in sliding window scheme is as follows:

$$N_s = \frac{k-w}{s} + 1. \tag{9}$$

In order to have a fair comparison of the traditional section-based system and the sliding window scheme, the numbers of check blocks for each method should be equal. The decoder of the rateless codes with decoding overhead γ decodes from n check blocks to obtain k message blocks. Thus,

$$\frac{n}{k} = \gamma. \tag{10}$$

The sliding window scheme will encode each message block more than once. We know the message blocks are processed in w/s successive sliding windows and the original number of message blocks will be virtually enlarged to k':

$$k' = \frac{w}{s} \cdot k. \tag{11}$$

Thus, with the same overhead γ, the number of check blocks per virtual message block is:

$$\frac{n}{k'} = \frac{\gamma k}{\frac{w}{s} k} = \gamma \frac{s}{w}. \tag{12}$$

Every sliding window has w message blocks, and each sliding window have n_w check blocks.

$$n_w = \frac{n}{k'} \cdot w = \gamma \frac{s}{w} \cdot w = \gamma s. \tag{13}$$

The sliding window scheme receives the number of check blocks as many as possible and no need to know which sliding window is being decoded. The decoder of the rateless codes just receives and continues to decode the number of the message blocks as many as possible. When the sliding window shifts to successive sliding window, it will have two situations. One is the message blocks which are in the successive sliding window were all recovered; another is the message blocks which are in the successive sliding window were just few ones recovered. But the first situation is rare to happen for the reasonable overhead γ. The normal situation is that still a lot of check blocks could be used for recovering the message blocks in the preceding and the successive sliding windows.

Chapter 3 Proposed Method

In this chapter, we propose the unequal overlapped rateless codes for the multimedia streaming in the section-based and the corresponding analytic model. We use the sliding window scheme as the way to transmit multimedia streaming. When the sliding window shifts, we could observe one situation that we want to analyze. First, we need to set the evaluation criteria on the multimedia streaming, and then analyze those situations with the And-Or tree lemma.

3.1 Multimedia Streaming Evaluation Criteria

By using the sliding window scheme, there are two kinds of models that we could analyze. The single section and the multiple sections are the evaluation criteria in the multimedia streaming. At first, we analyze the decoding failure probability of the single section, and then use the result of proposed analytic model on the model of the multiple sections.

3.1.1 Single Section in The Multimedia Streaming

When the sliding window shifts to the successive sliding window, each sliding window contains two parts: the overlapped part and the non-overlapped part. The overlapped part is the part in current sliding window and may have portion of recovered message blocks. The non-overlapped part is the part which new message blocks are included in current sliding window. We could see the example in Figure 3-1. When sliding window 1 shifts to sliding window 2, the $m_{1,3}$ and $m_{1,4}$ are in the overlapped part, $m_{2.1}$ and $m_{2.2}$ are in the non-overlapped part. Each sliding window forms a single section that contains the overlapped part and non-overlapped part.

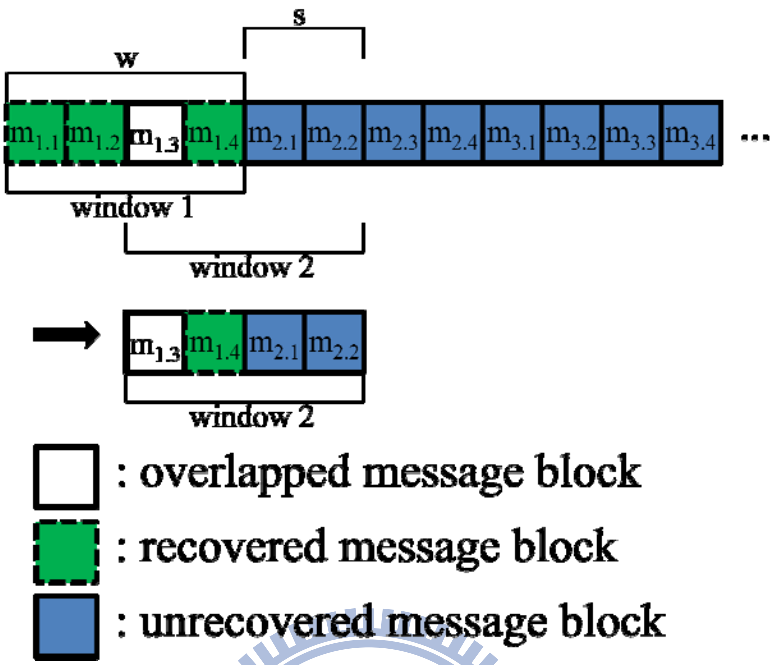


Figure 3-1: The example of overlapped and non-overlapped part in the single section.

We could use the proposed analytic model with unequal overlapped rateless codes on the single section to find the decoding failure probability in each part, and then applies the results on the model of the multiple sections. Let $p_{L,1}$ be the decoding failure probability of the overlapped part and $p_{L,2}$ be the decoding failure probability of the non-overlapped part. L is constant that we will introduce later and the decoding failure probability is the proportion of the number of the unrecovered message blocks over the sliding window size.

3.1.2 Multiple Sections in The Multimedia Streaming

After calculating the decoding failure probability $p_{L,1}$ and $p_{L,2}$ in the single section by the proposed analytic model, we could calculate the decoding failure probability of the multiple sections with the proposed unequal overlapped rateless codes. Following the above example shown in Figure 3-1, we could use the decoding failure probability of the non-overlapped part in sliding window 1 as the initial decoding failure probability of the overlapped part in sliding window 2. Then, we could calculate the decoding failure probability $p_{L,1}$ and $p_{L,2}$ in sliding window 2 and continue on this procedure to the end of the multimedia streaming. The example of sliding window shifts to the successive sliding window is shown in Figure 3-2.

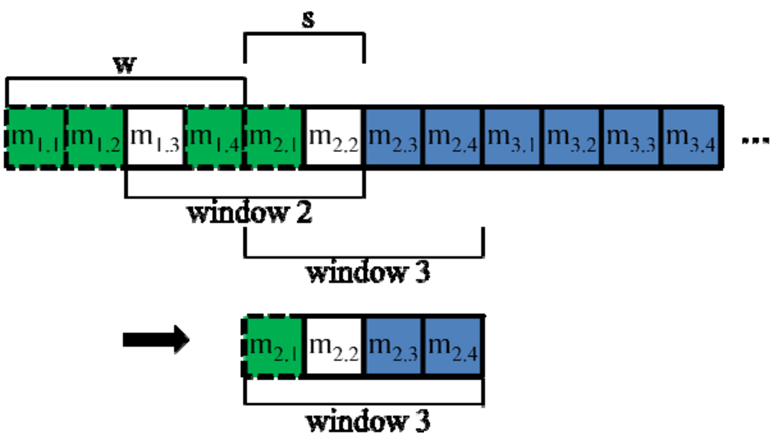


Figure 3-2: The example of sliding window shifts to the successive sliding window.

When sliding window 2 shifts to the successive sliding window, the overlapped part is $m_{2.1}$ and $m_{2.2}$, and the non-overlapped part is $m_{2.3}$ and $m_{2.4}$ in the successive sliding window as shown in Figure 3-2. Now, we set the overlapped parameter α is the proportion of the number of old message blocks discarded or new message blocks considered.

$$\alpha = \frac{window\ movement}{window\ size} = \frac{s}{w}. \tag{14}$$

Now, we only consider $\alpha = 1/2$ in the following description, and also assume the overlapped part has the portion of recovered message blocks. We could use the local minimization approach (LMA) and dynamic programming approach (DPA) to find the parameter sequence for the overall of the minimal decoding failure probability of the multiple sections.

3.2 The Analytic Model for Single Section

For finding the minimal decoding failure probability in a single section, we could modify the And-Or tree lemma to analyze our unequal overlapped rateless codes to determine the decoding failure probability.

3.2.1 Rateless Codes in And-Or Tree

This section explains how to transform rateless codes to And-Or tree. The transformation process could be found in [13]. First, we could think the decoding paradigm of rateless codes as a bipartite graph, as shown in Figure 3-3.

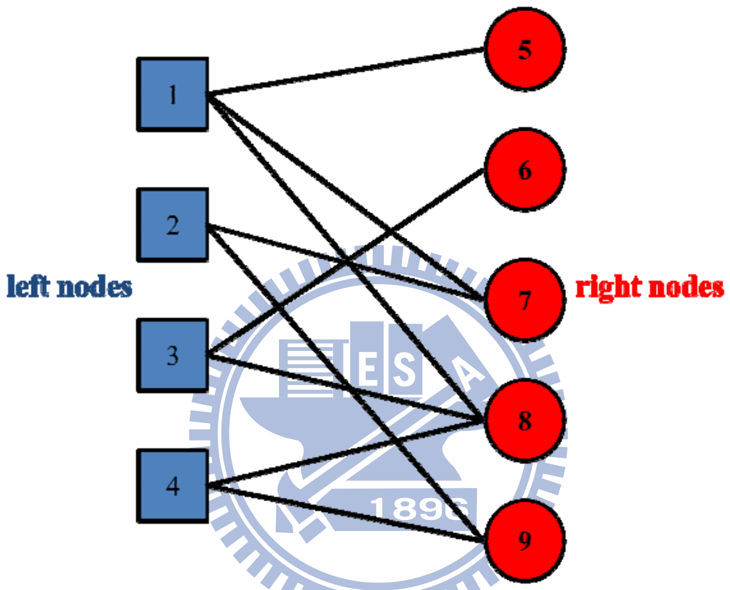


Figure 3-3: Bipartite graph G.

Let G be a bipartite graph with k nodes on the left side, n nodes on the right side, and e edges in total between the nodes on the left nodes and the right nodes. We could set each left node corresponding to a message block and each right node corresponding to a check block. If the left node is evaluated as 1, it stands for recovering the corresponding message block. If the right node is evaluated as 1, it stands for the situation that it could help to recover the neighbor.

We assume the receiver could reconstruct this bipartite (for example: using random number generator with the same random seed). All left nodes are unrecovered (evaluated as 0) initially. If the left node could be recovered as the progress of LT decoding, we set this left node as 1. Following the paper [13], let (f_0, f_1,⋯, f_n) and (g_0, g_1,⋯, g_k) be the probability distribution that each left node is chosen to have degree d with probability f_d, and each right node is chosen to have degree d with probability g_d, where all choices are made independently. We could calculate the total edges e in this random bipartite graph.

$$e = k \cdot \sum_{d=0}^{n} d f_d = n \cdot \sum_{d=0}^{k} d g_d. \tag{15}$$

We are going to look at a random subgraph G_L (where L is a constant) of G, which is chosen as follows:

1. Choose an edge (v, w) of G randomly and uniformly, and call v the root of G_L.
2. Remove (v, w) from G.
3. G_L will consist of the left node v, all neighbors nodes of v within $2L$ hops from v, and all edges of G that any two of these nodes.

The G_L is shown in Figure 3-4.

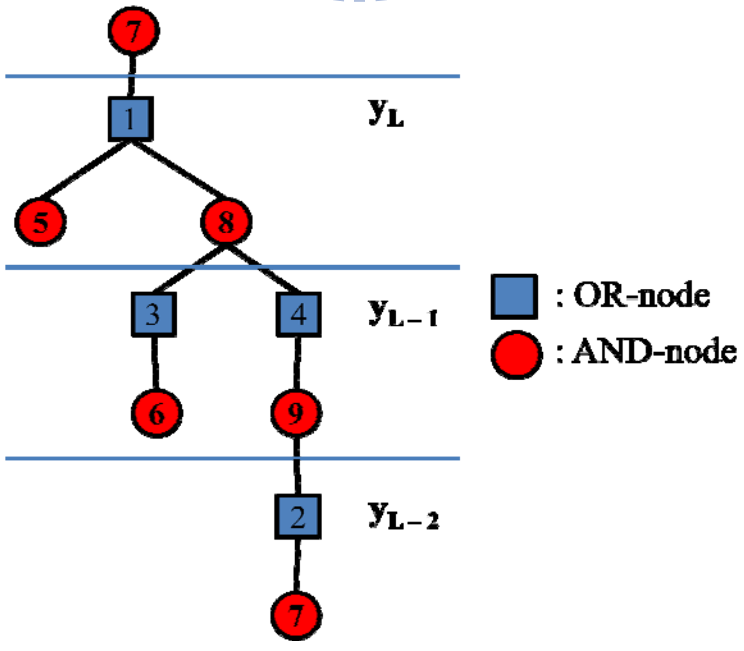


Figure 3-4: Example of G_L.

Here, we claim that G_L is tree-shaped. G_L is created by first selecting and revealing a random edge (v, w), and then reveal v's neighbors, and so forth for $2L$ hops. The probability that G_L is not a tree (there is a cycle in G_L) will be proportional to $1/k$. Therefore, as k grows large enough, G_L is a tree with high probability.

Formally, we think G_L as an And-Or tree in the following way. The left nodes of G_L map to the OR-nodes of the tree and the right nodes of G_L map to AND-nodes. An OR-node is assigned the value 1 if we recover the value of the corresponding message block, which could happen when at least one of its children (AND-nodes) is evaluated as 1. It is just like the OR-node to do the OR operation of its children. An AND-node is assigned the value 1, either if it has no children in the tree, just like it has the degree 1 in G and could immediately to recover its neighbor of the left node, or if all of children of the OR-node are evaluated as 1. Let y_L be the probability that the root of G_L that is evaluated as 0. Each time AND-node will pass a message to the OR-node, and then OR-node will pass a message to the AND-node. It could be computed by the And-Or tree lemma. If v is evaluated as 1, we could know that v also is evaluated as 1 in the bipartite graph G.

The number of the left nodes which are evaluated as 0 is close to the expected value when And-Or tree lemma holds. If we choose another message node as the root node, the same result will be obtained according to the standard edge-exposure martingale [22]. After the transformation process, we could find the decoding failure probability by using And-Or tree lemma. The decoding failure probability in LT decoding process is equal to the proportion of the number of the left nodes which are evaluated as 0.

3.2.2 And-Or Tree with Unequal Weight

The rateless codes with weights could be initially found in [11]. Now, we modify the construction of the And-Or tree to the case that OR-nodes may be unlike each other and contains portion of recovered message blocks. We could know that when the sliding window shifts, it will produce such situation, as shown in Figure 3-5:

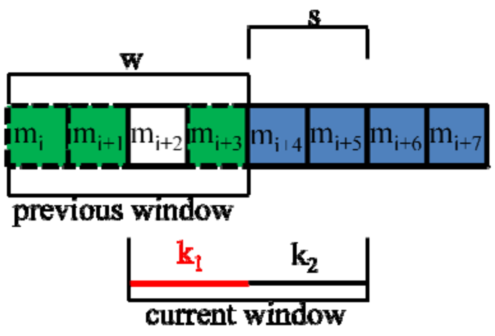


Figure 3-5: Situation of sliding window movement.

In here, we want to construct a generalized And-Or tree for each part of the message blocks to find out the decoding failure probability in a single section, as shown in Figure 3-6.

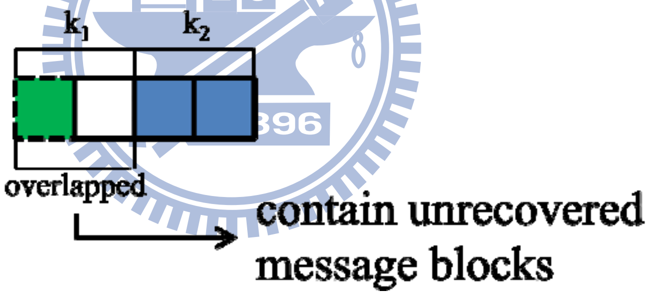


Figure 3-6: Analytic section.

The root of the generalized And-Or tree $GT_{L,i}$ stands for an OR-node in part k_i, and the depth of this tree is $2L$. w_1, w_2 are the probabilities to select a node from k_1, k_2. The construction of $GT_{L,i}$ is the same as G_L except the root node is in part k_i. We set the probability that an edge is connected to an OR-node in part k_j of degree d is $R_{d,j}$ and each OR-node in part k_1 has the decoding failure probability δ initially. The probability that an edge is connected to an AND-node of degree d is A_d. However, unlike the original And-Or tree, each child of an AND-node is independently calculated by using the Wallenius' Distribution, and we want to see if the unequal selection is better than the uniform selection. Follow the definition of

And-Or tree, the OR-nodes with no children are assumed to be evaluated as 0, whereas the AND-nodes with no children are assumed to be evaluated as 1. We could calculate the $y_{L,i}$ that is the probability that the root of $GT_{L,i}$ is evaluated as 0. $p_{i,j}$ is the probability that the message passed from the OR-node in part k_j to the AND-node at round i is 0. q_i: the probability that the message passed from the AND-node to the OR-node at round i is 0. To obtain q_i, we need to calculate the probability of each degree pair consisting of d_1 nodes $\in k_1$ and d_2 nodes $\in k_2$ by using the Wallenius' Distribution. For the event that AND-node has degree d+1 at round i, and we know the AND-node sends a message 1 to the OR-node if and only if all the children of AND-nodes send a message 1. See Figure 3-7:

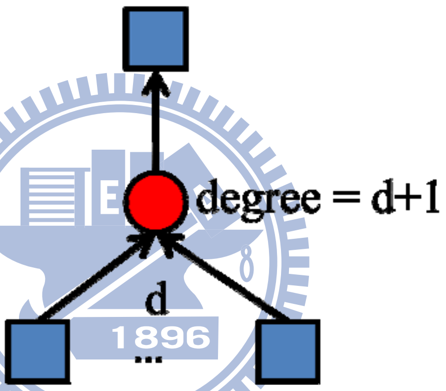


Figure 3-7: Conditional on the AND-node of degree d+1.

We could calculate the probability that this AND-node sends message 0 to OR-node by considering all the combination of the degree pairs. Let d_1+d_2=d. We could get the probability of the AND-node of degree d+1 being sending the message is evaluated as 0:

$$1-\left(1-y_{i,1}\right)^{d_1}\left(1-y_{i,2}\right)^{d_2} \qquad (16)$$

From (16), the q_i is all the combination of the degree pairs:

$$q_i=\sum_{d=1}^{Max\_D-1} A_{d+1}\cdot\sum_{j=0}^{d}\left(WDP(j,d-j)*\left(1-\left(1-y_{i,1}\right)^{j}\left(1-y_{i,2}\right)^{d-j}\right)\right) \qquad (17)$$

And, we could know that the OR-node in part k_j will send a message 0 to the AND-node if all the messages coming from the children AND-nodes are 0 at round *i*. The condition that the OR-node is degree *d*+1 at round *i*+1, as shown in Figure 3-8:

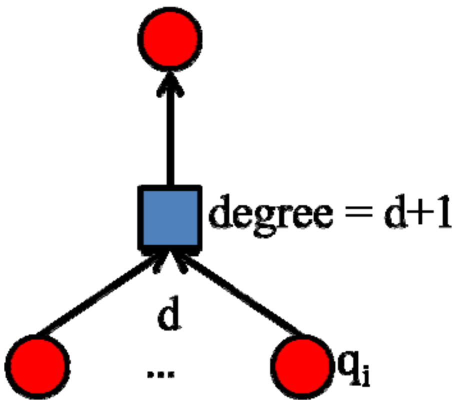


Figure 3-8: Conditional on the OR-node is degree *d*+1.

We could calculate the probability that this OR-node sends message 0 to the AND-node by the number of its children.

$$\begin{cases} \delta \cdot q_i^d & \text{if this OR-node} \in k_1 \\ q_i^d & \text{if this OR-node} \in k_2 \end{cases} \tag{18}$$

For simplicity, we define the polynomial distribution:

$$R_j(x) = \sum_{d=1}^{Max\_D} R_{d,j} \cdot x^{d-1} \tag{19}$$

From (18) and (19), we get $p_{i+1,j}$ is

$$\begin{cases} p_{i+1,1} = \delta \cdot \sum_{d=1}^{Max\_D} R_{d,1} \cdot q_i^{d-1} \\ p_{i+1,2} = \sum_{d=1}^{Max\_D} R_{d,2} \cdot q_i^{d-1} \end{cases} \tag{20}$$

We obtain the following recursion using the $R_j(x)$ for expected probability:

$$\begin{cases} p_{i+1,1} = \delta \cdot R_1(q_i) = y_{i+1,1} \\ p_{i+1,2} = R_2(q_i) = y_{i+1,2} \end{cases} \tag{21}$$

We get the probability $y_{L,i}$ for part k_1 and k_2 by using edge degree probability distribution to calculate in (21). Next, we make the connection between the edge's degree probability distribution and the block's degree probability distribution.

3.2.3 And-Or Tree with Unequal Weight to Bipartite Graph

Let $\Omega(x) = \sum_{d=1}^{n} \Omega_d x^d$ be the polynomial generator corresponding to the probability distribution of the degrees of check blocks in rateless codes. We partition the k message blocks into two parts k_1 and k_2 of sizes αk and $(1 - \alpha)k$. Let p_i be the probability that an edge is connected to a particular message block in k_i, for $i = 1, 2$. And total check blocks γk are involved in the decoding process, we call γ the overhead. We set μ the average degree of the check block:

$$\mu = \sum_{d=1}^{k} d\Omega_d = \Omega'(1), \tag{22}$$

where $\Omega'(x)$ is the derivative of $\Omega(x)$ with respect to x.

Consider k message blocks encoded by the proposed unequal overlapped rateless codes in the single section with parameter $\Omega(x)$, k, α, p_1, p_2 and γ. We could calculate the probability of the degree of message blocks in k_i, for $i = 1, 2$. The probability $\lambda_{d,i}$ that the message block in k_i has a degree d is

$$\lambda_{d,i} = \binom{\mu\gamma k}{d} p_i^d (1 - p_i)^{\mu\gamma k - d}. \tag{23}$$

When k is large enough and p_i is small enough. Asymptotically, (23) approaches to

$$\lambda_{d,i} = \frac{e^{-\mu\gamma k p_i} (\mu\gamma k p_i)^d}{d!} \tag{24}$$

which is a Poisson distribution with the mean $\mu\gamma k p_i$.

Now, we could get

$$A_d = \frac{d \cdot \Omega_d}{\Omega'(1)} \tag{25}$$

and

$$R_{d,i} = \frac{d \cdot \lambda_{d,i}}{p_i \mu \gamma k} \tag{26}$$

then

$$R_i(x) = e^{p_i \mu \gamma k (x-1)}. \tag{27}$$

We could get p_1 and p_2.

$$\begin{cases} p_1 = \dfrac{w_1}{w_1 + w_2} \cdot \dfrac{1}{\alpha \cdot k} \\ p_2 = \dfrac{w_2}{w_1 + w_2} \cdot \dfrac{1}{(1-\alpha) \cdot k} \end{cases}. \tag{28}$$

3.3 Wallenius' Noncentral Hypergeometric Distribution

After we propose our analytic model for unequal overlapped rateless codes, we need to know about the noncentral hypergeometric distribution for unequal selection. In probability theory, Wallenius' noncentral hypergeometric distribution (WD) [23] [24] is a generalization of the hypergeometric distribution which describes the items that are sampled with bias in a finite population without replacement. The distribution could be illustrated as an urn model with bias. For example, an urn contains m_1 white balls and m_2 black balls. Each white ball has the weight w_1 and each black ball has the weight w_2. Now we take n balls, one by one without replacement, in such way that the probability of taking a particular ball at a particular draw is dependent on its proportion of the total weight of all balls that remains in the urn at that moment. If we want to take x_1 white balls and x_2 black balls, the most reliable calculation method is recursive calculation from [25] [26].

The following recursion formula is useful to calculate the probabilities:

$$\begin{aligned} & WDP(x;n,m,N,w) \\ & = WDP(x-1;n-1,m,N,w)\cdot\frac{(m-x+1)w}{(m-x+1)w+N-n-m+x} \\ & \quad + WDP(x;n-1,m,N,w)\cdot\frac{N-n-m+x+1}{(m-x)w+N-n-m+x+1}, \end{aligned} \tag{29}$$

where $x = x_1$, $N = m_1+m_2$, $m = m_1$, $w = w_1/w_2$.

3.4 The Analytic Model for Multiple Sections

By using the proposed analytic model with unequal overlapped rateless codes, we could calculate the decoding failure probability $p_{L,1}$ and $p_{L,2}$ in the single section. We also could calculate the decoding failure probability of the multiple sections by using the same analytic model with the proposed unequal overlapped rateless codes. We provide local minimization approach (LMA) and dynamic programming approach (DPA) to calculate the overall of minimal decoding failure probability in the multiple sections.

3.4.1 Local Minimization Approach (LMA)

Assume giving $\alpha\gamma w$ check blocks to each sliding window. We need to find the weights $\overline{w_i} = (w_{i,1}, w_{i,2})$ that gives the decoding failure probability in sliding window i with the given initial decoding failure probability in k_1 (that is, the decoding failure probability of k_2 in the previous sliding window). We could continue to calculate the weights in each sliding window for getting the overall of minimal decoding failure probability. As shown in Figure 3-9:

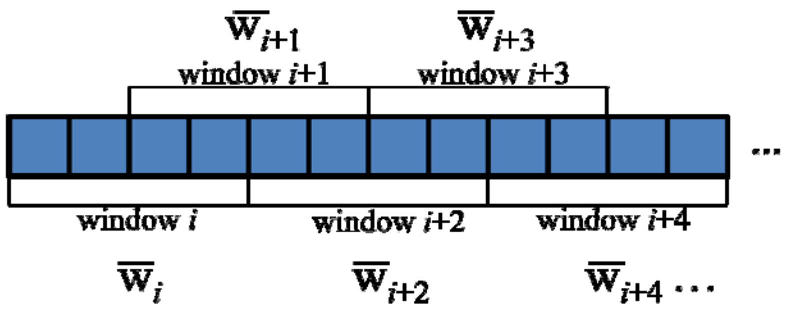


Figure 3-9: Local Minimization Approach.

According to the decoding failure probability of sliding window i, we could use the decoding failure probability of k_2 in sliding window i to be the initial decoding failure probability δ in sliding window i+1. We could get the overall of minimal decoding failure probability based on the sequence of the chosen weights.

3.4.2 Dynamic Programming Approach (DPA)

Local minimization approach is based on the estimated minimal decoding failure probability of each sliding window. The dynamic programming approach is based on the selected weights with respect to the preceding cumulated decoding failure probability and the number of weights sequences that are more than one.

We have $F_{i,a}$ and $F_{i,b}$ as the decoding failure probability of k_1 and k_2 based on the given weights at the sliding window i. Each estimation step is according to the preceding cumulated decoding failure probability. The cumulated decoding failure probability in the sliding window i is

$$\left(\sum_{j=1}^{i} F_{j,a}\right) + F_{i,b} \tag{30}$$

The example of DPA is shown in Figure 3-10. After giving different weights in sliding window 1, we could choose the minimal decoding failure probability. And then using the proposed model with weights to calculate next weights for sliding window 2, and forth.

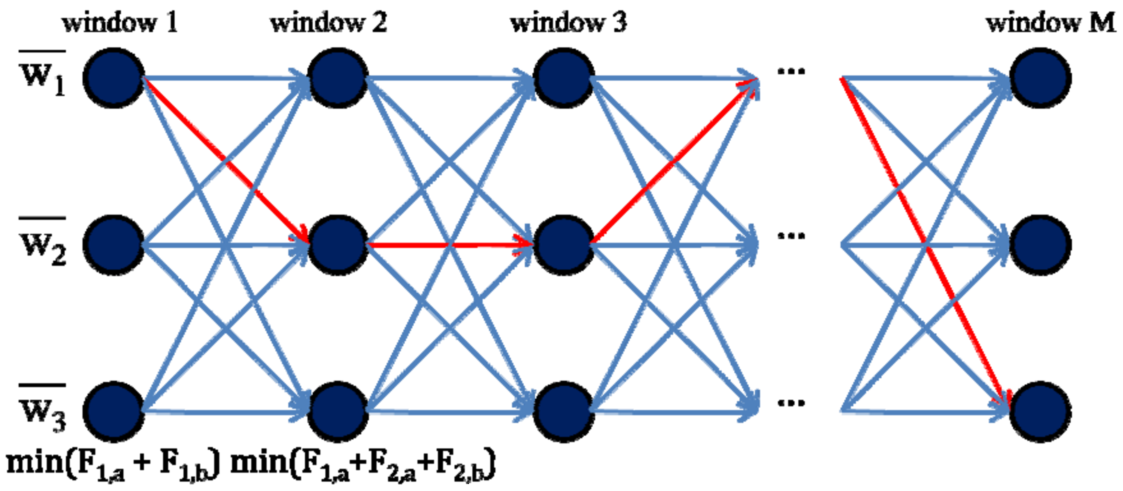


Figure 3-10: Dynamic Programming Approach.

3.4.3 Time Complexity of LMA and DPA

To get the overall of minimal decoding failure probability in LMA, we only need to calculate each weights in each sliding window. So the time complexity is linear to sliding window number N_s in the multimedia streaming. In DPA, we need to consider the number of weights sequences.

If there are M sliding windows in the multimedia streaming, A is the number of weights sequences considered and B is the number of all possible weights. We could have the time complexity:

$$LMA: O(BM)$$
$$\text{DPA:} \begin{cases} O(\text{BAM}), & \text{if A} < B \\ O(\text{B}^2\text{M}), & \text{if A } \geq \text{B}^{\cdot} \end{cases} \tag{31}$$

3.5 Buffer Mode

LMA and DPA are proposed in the multiple sections based on the analytic model of the single section. If we impose some delay between the playback section id and the receiving section, the decoding performance could be enhanced. It means the decoding could be interactive between each sliding window. Because some of the message blocks are recovered in sliding window *i* initially and we know that the LT decoding process is finding the check block which all of whose message blocks are recovered except for one is unrecovered. The LT decoding could use the recovered message blocks in window *i*–1 to help increasing the chance that check blocks in window *i* which connect to only one message block that is unrecovered. The interactive relation could reduce the probability of the condition that there are still having a lot of unrecovered message blocks. In here, we could have the buffer mode to realize this method. As shown in Figure 3-11.

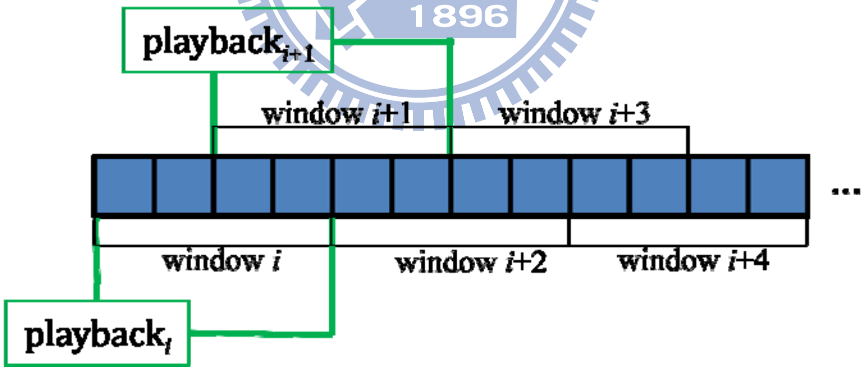


Figure 3-11: Buffer size is one sliding window.

Figure 3-11 illustrates the buffer size is one sliding window (SW). The decoder could receive and decode sliding window *i* at the same time.

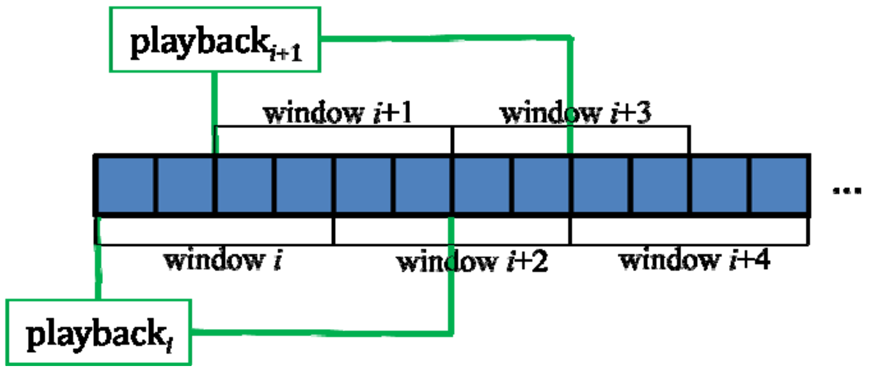


Figure 3-12: Buffer size is two sliding windows.

Figure 3-12 describes the situation when the buffer size is two sliding windows. For the playback$_i$, the decoder could receive and decode sliding window i and sliding window i+1 at the same time. Next, we show the experiment results to see the decoding performance.

Chapter 4 Experiment Results

We propose an analytic model of the unequal overlapped structure. First, we compare the estimated decoding failure probability based on the analytic model for the single section and simulations, and then show the simulation results in the multiple sections.

4.1 Simulation Environment

- 5 sections, 9 sections
- Section size = 2,000 message blocks
- Number of the sliding window = 9, 19
- Overhead = 1.04 ~ 1.20
- Results are the average of 100 independent simulations
- Degree distribution is from Raptor codes [2]

Table 4-1: The degree distribution table of Raptor codes.

Degree	Probability	Degree	Probability
1	0.007969	8	0.056058
2	0.493570	9	0.037229
3	0.166220	19	0.055590
4	0.072646	64	0.025023
5	0.082558	66	0.003135

Table 4-2: The decoding failure probability after giving γk check blocks, k=1000.

Overhead γ	**Decoding Failure Probability (δ)**
1.04	0.608039
1.06	0.460122
1.08	0.332582
1.10	0.263896
1.12	0.175325
1.14	0.144782
1.16	0.114760
1.18	0.071047
1.20	0.055560

Table 4-3: The decoding failure probability after giving γk check blocks, k=2000.

Overhead γ	**Decoding Failure Probability (δ)**
1.04	0.393454
1.06	0.195176
1.08	0.107732
1.10	0.058554
1.12	0.032648
1.14	0.020570
1.16	0.012885
1.18	0.008360
1.20	0.003477

4.2 Single Section

Assume the overlapped part k_1 has δ portion of unrecovered message blocks. We give $\alpha\gamma k$ check blocks to the single section, and then observe the portion of unrecovered message blocks and the estimated decoding failure probability. The δ is from Table 4-2. According to different weights, the simulation values and estimated values are shown in Figure 4-1, 4-2, 4-3 and 4-4.

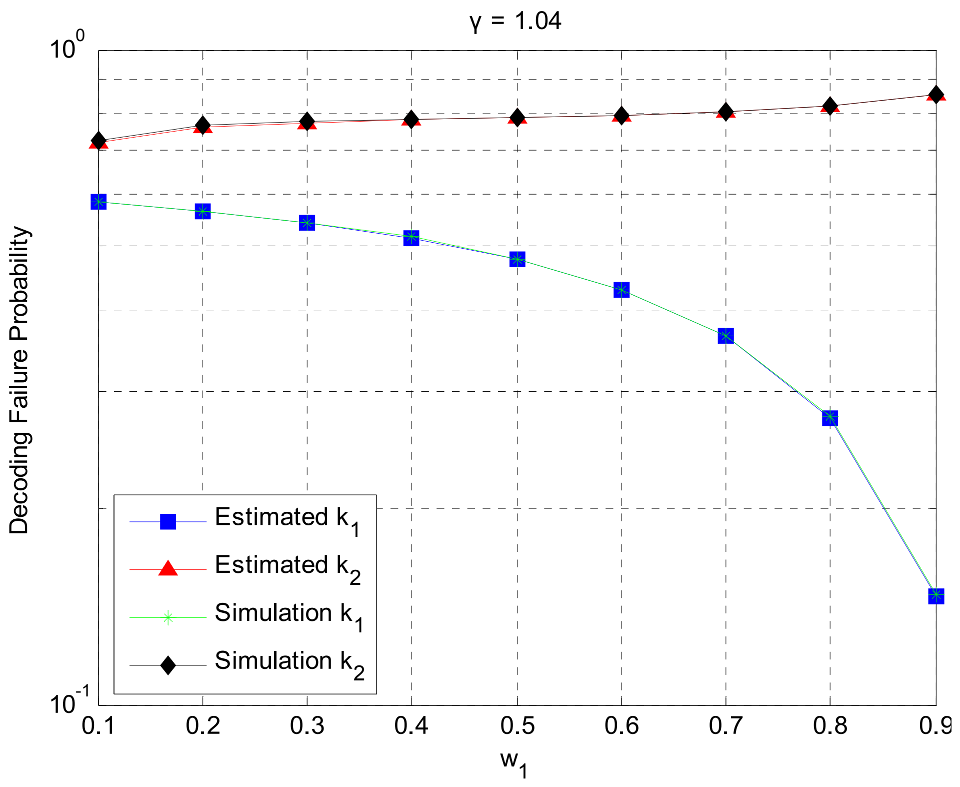


Figure 4-1: γ is 1.04

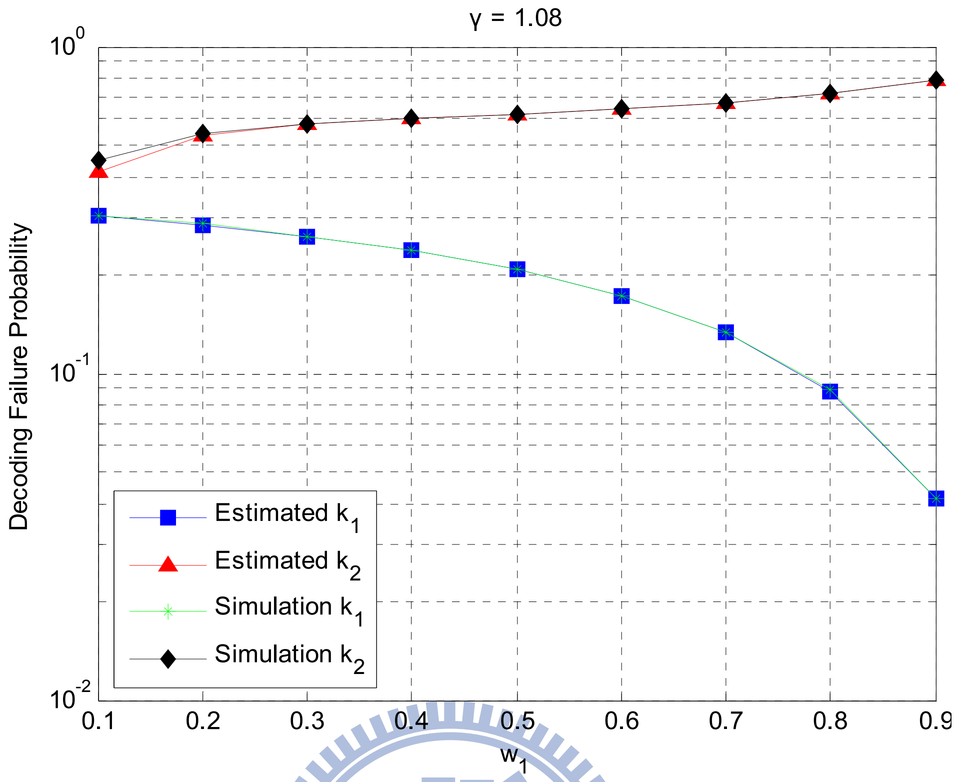


Figure 4-2: γ is 1.08

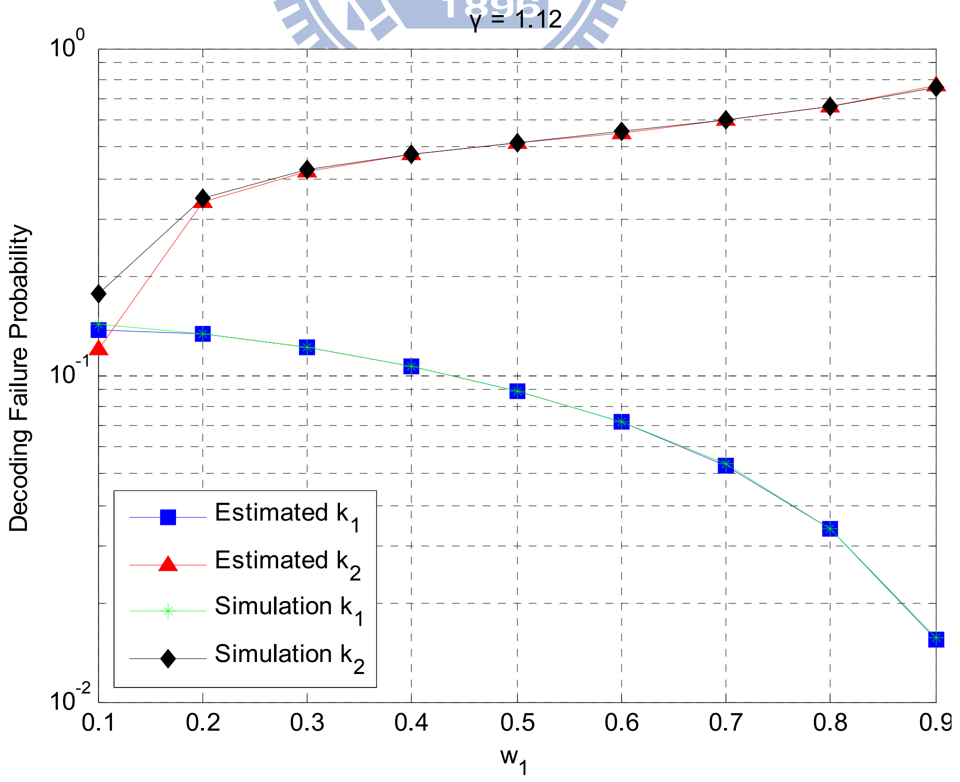


Figure 4-3: γ is 1.12

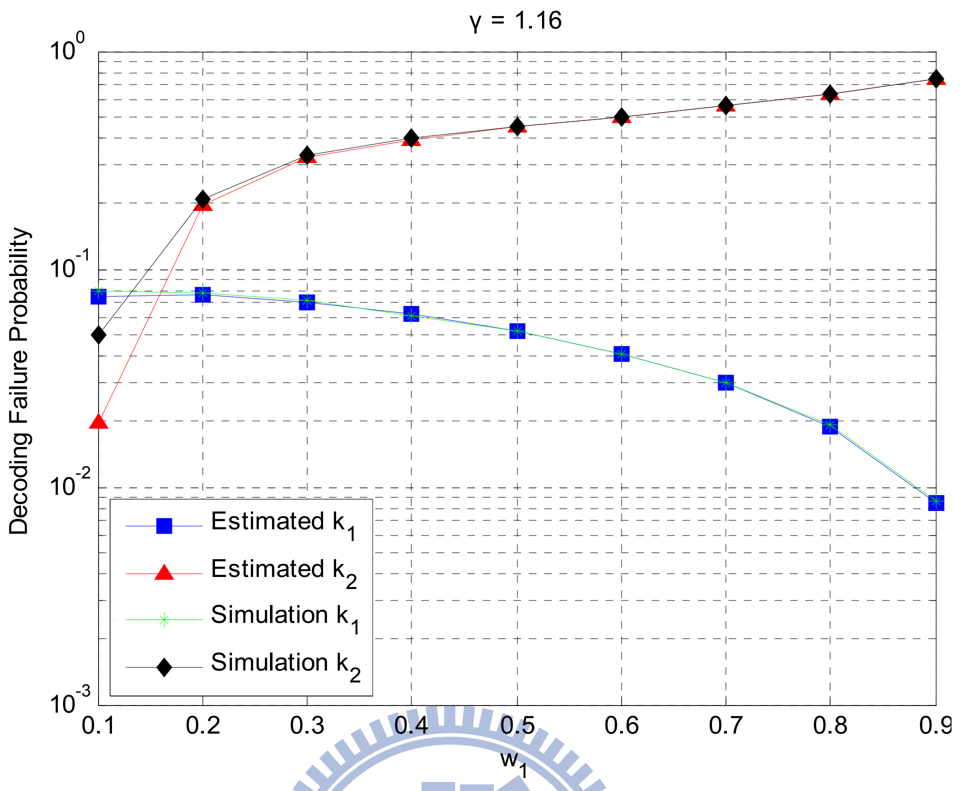


Figure 4-4: γ is 1.16

When the overhead is low, the analytic model could provide good estimated values at k=2000. Next, we estimate whole sliding windows by local minimization approach and dynamic programming approach in the successive section. Then, we could find that the weights sequences to have better performance than the equal weights sequence.

4.3 Multiple Sections

After having the estimated decoding failure probability of the single section, we use the local minimization approach and dynamic programming approach to find good weights sequence for the multiple sections. “Equal” approach is the sliding window scheme with equal weights sequence for k_1 and k_2 in [10].

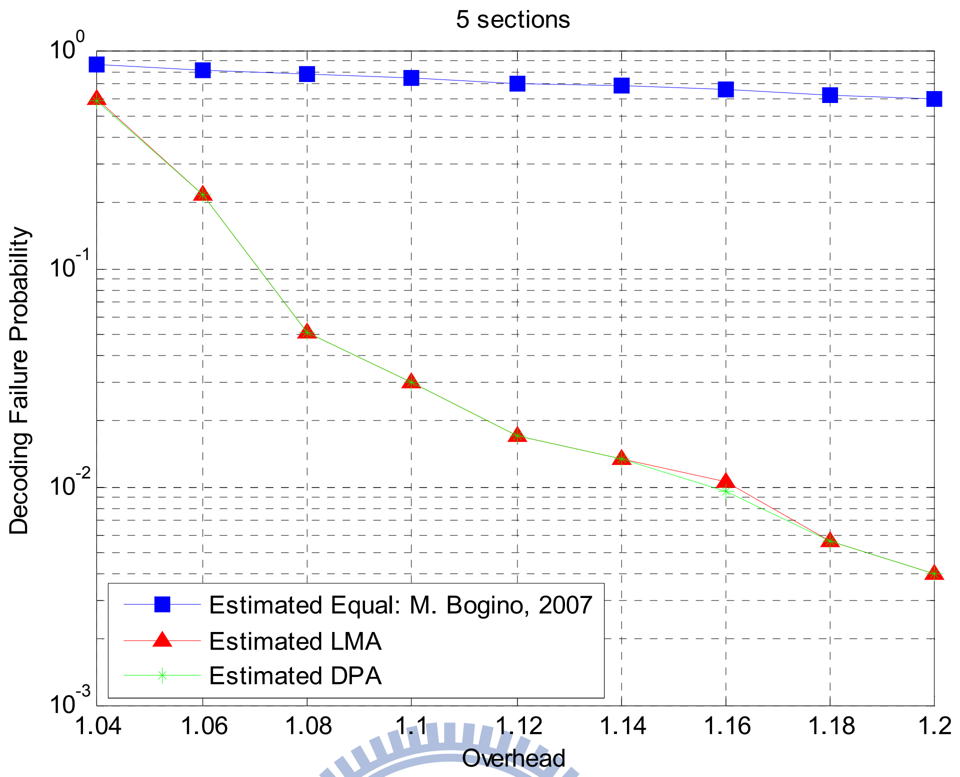


Figure 4-5: Estimated values of Equal, LMA and DPA in 5 sections.

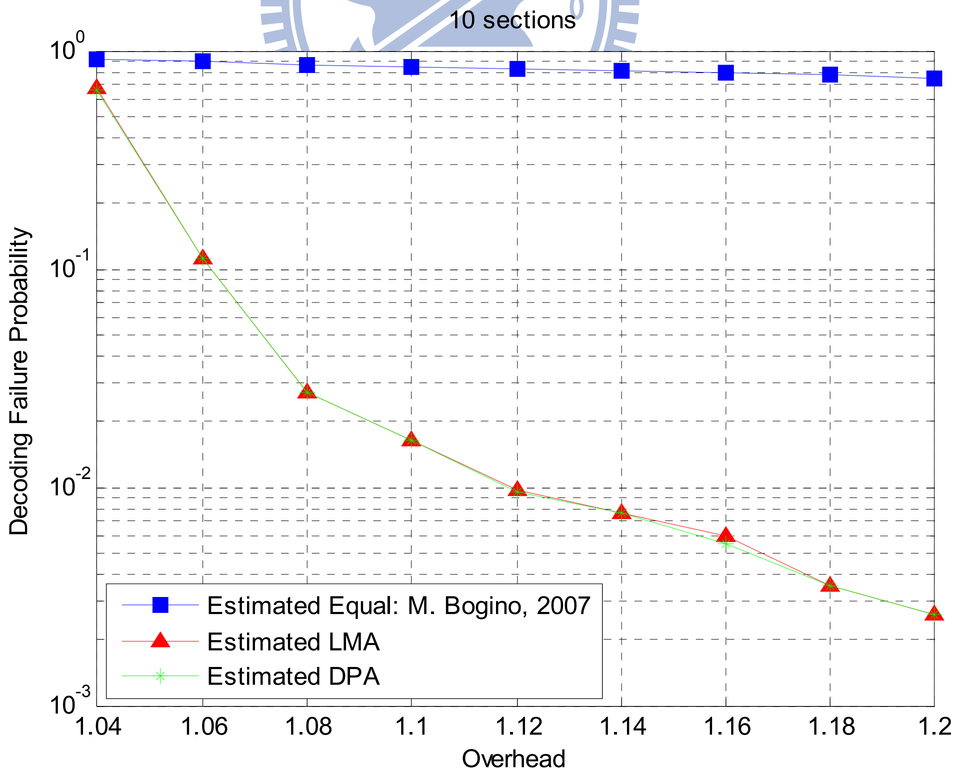


Figure 4-6: Estimated values of Equal, LMA and DPA in 10 sections.

In Figure 4-5 and 4-6, we could observe the estimated values of LMA and DPA are closely. There are 20 weights considering in LMA and DPA. We only hold back 9 weights sequence in DPA to find the overall of the minimal decoding failure probability in the multiple sections. After having the weights sequence, we could observe the LMA and DPA having similar estimated decoding failure probability. In here, we only simulate LMA for the low complexity of computation. “Fix” approach is using LT codes to encode and decode each section independently. The values are from Table 4-3.

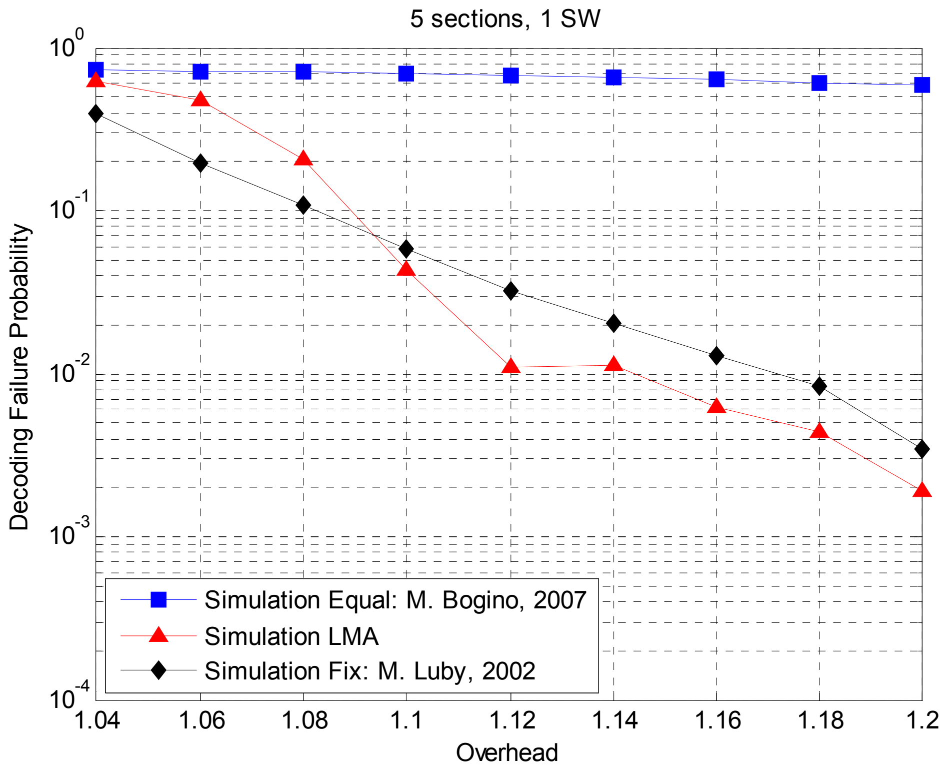


Figure 4-7: Simulation values of Equal and LMA in 5 sections with 1 SW.

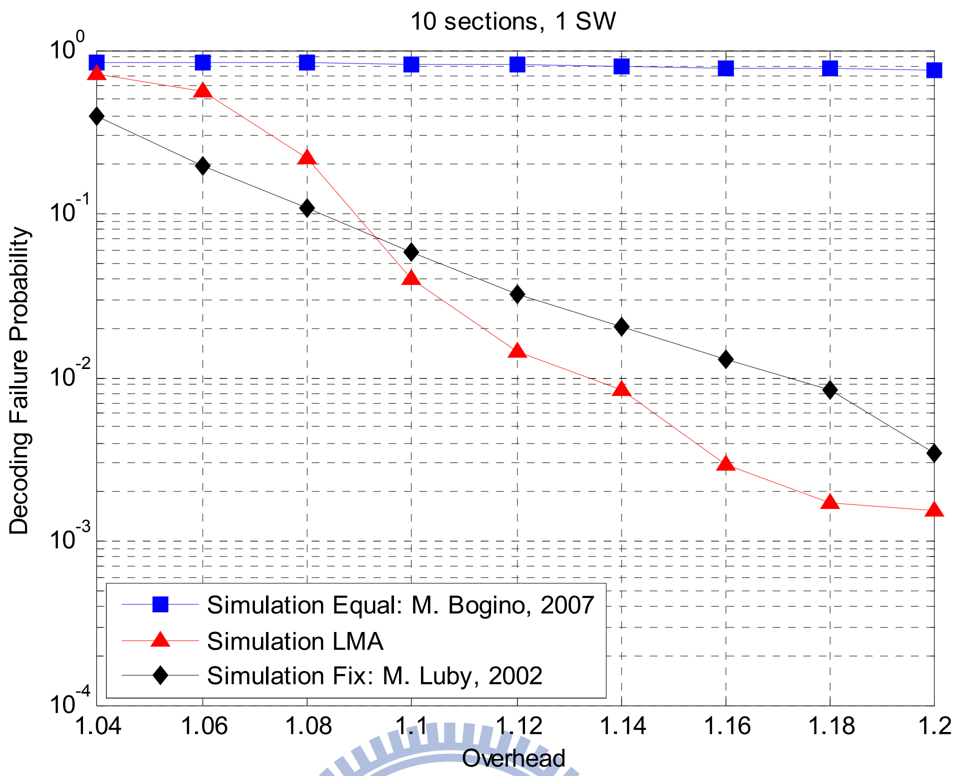


Figure 4-8: Simulation values of Equal and LMA in 10 sections with 1 SW.

The two figures above are setting the buffer size of one sliding window. Next, we set the buffer size of 2, 3 and whole SW.

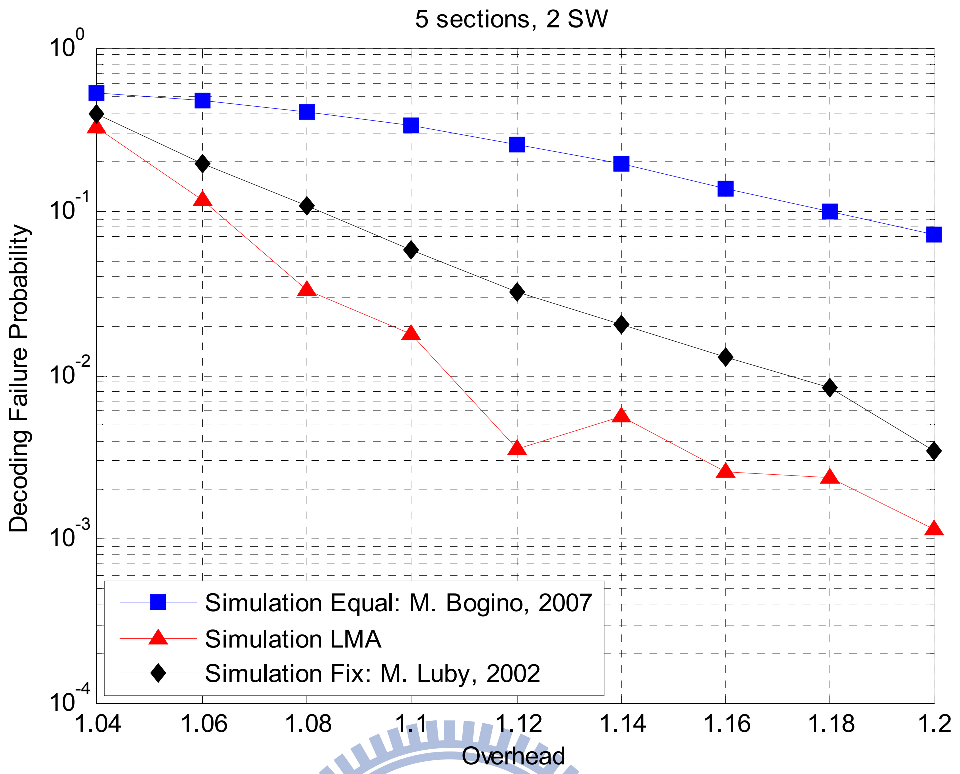


Figure 4-9: Simulation values of Equal and LMA in 5 sections with 2 SW.

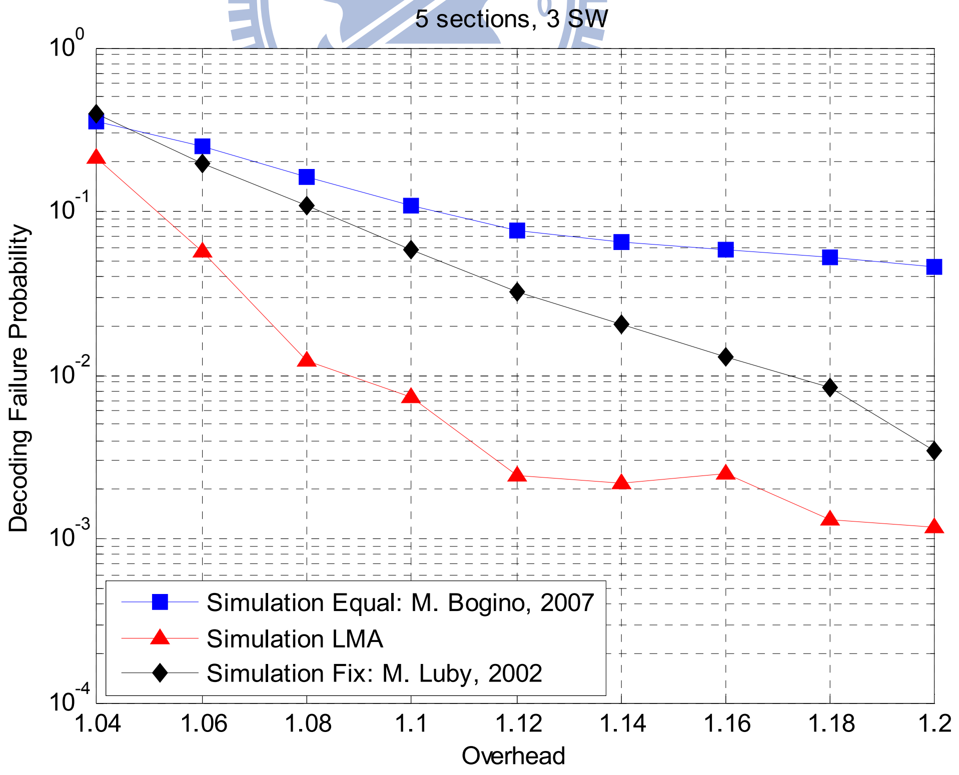


Figure 4-10: Simulation values of Equal and LMA in 5 sections with 3 SW.

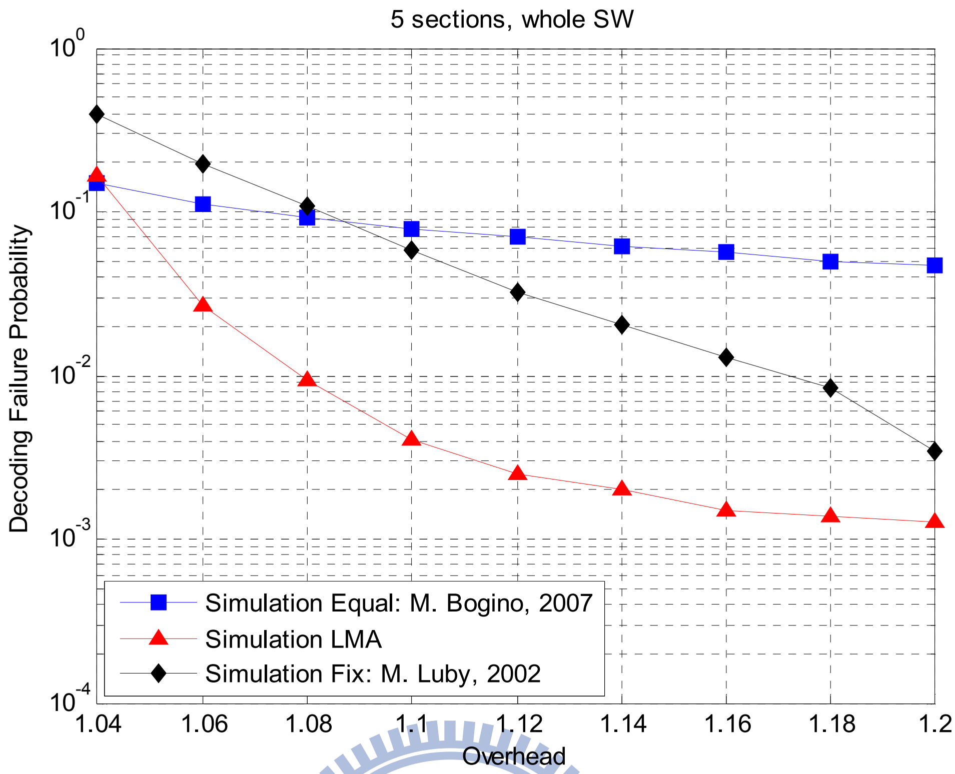


Figure 4-11: Simulation values of Equal and LMA in 5 sections with whole SW.

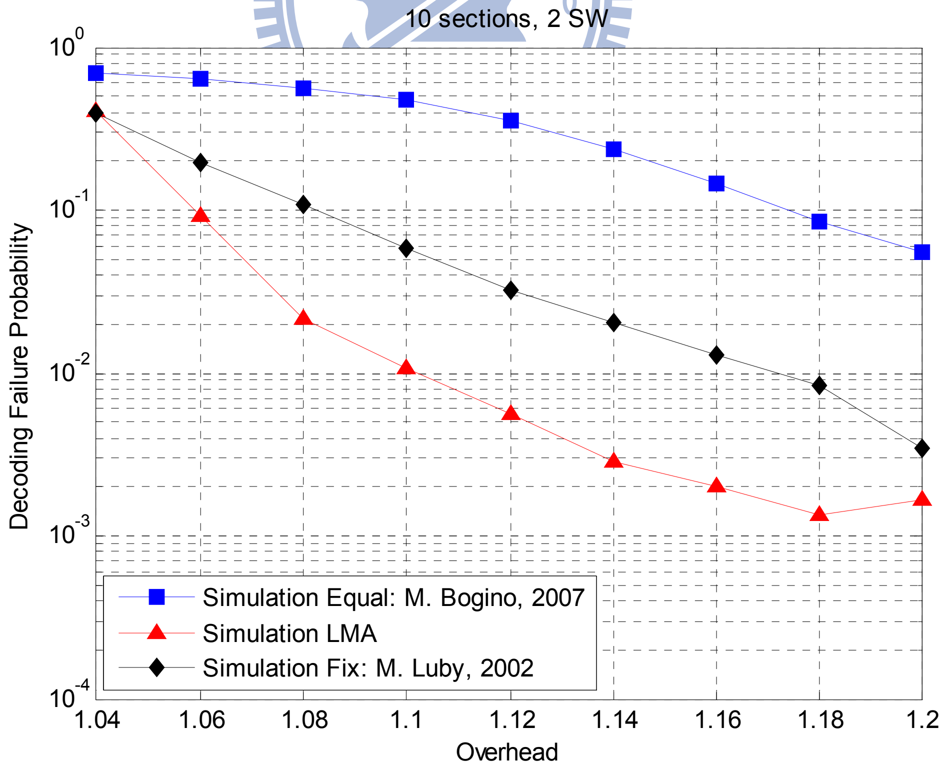


Figure 4-12: Simulation values of Equal and LMA in 10 sections with 2 SW.

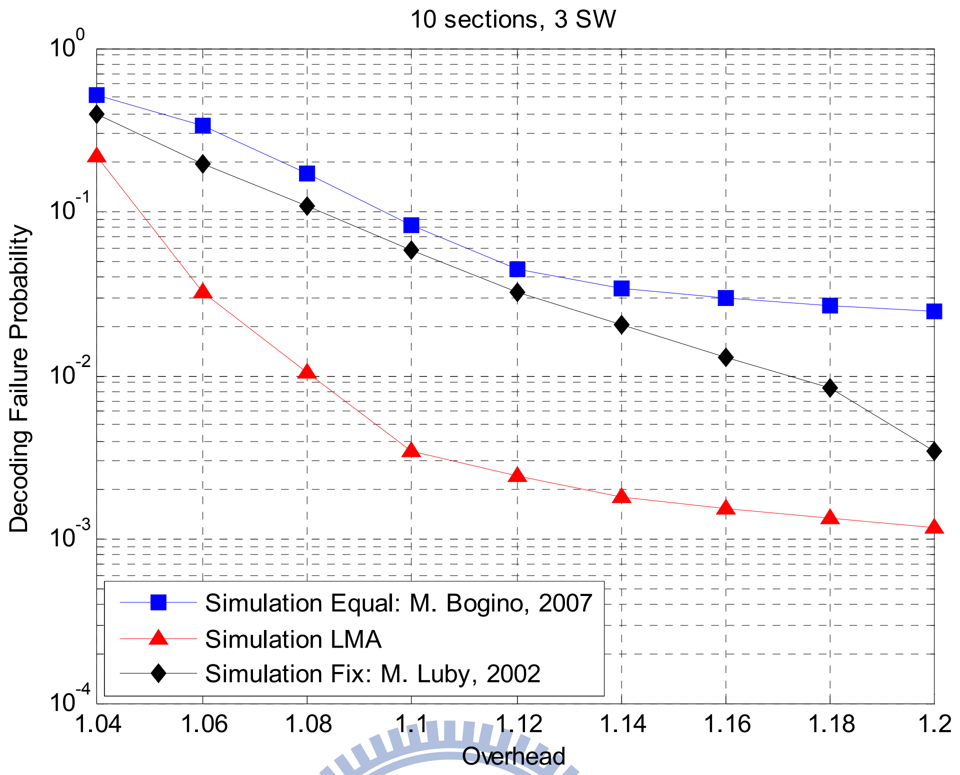


Figure 4-13: Simulation values of Equal and LMA in 10 sections with 3 SW.

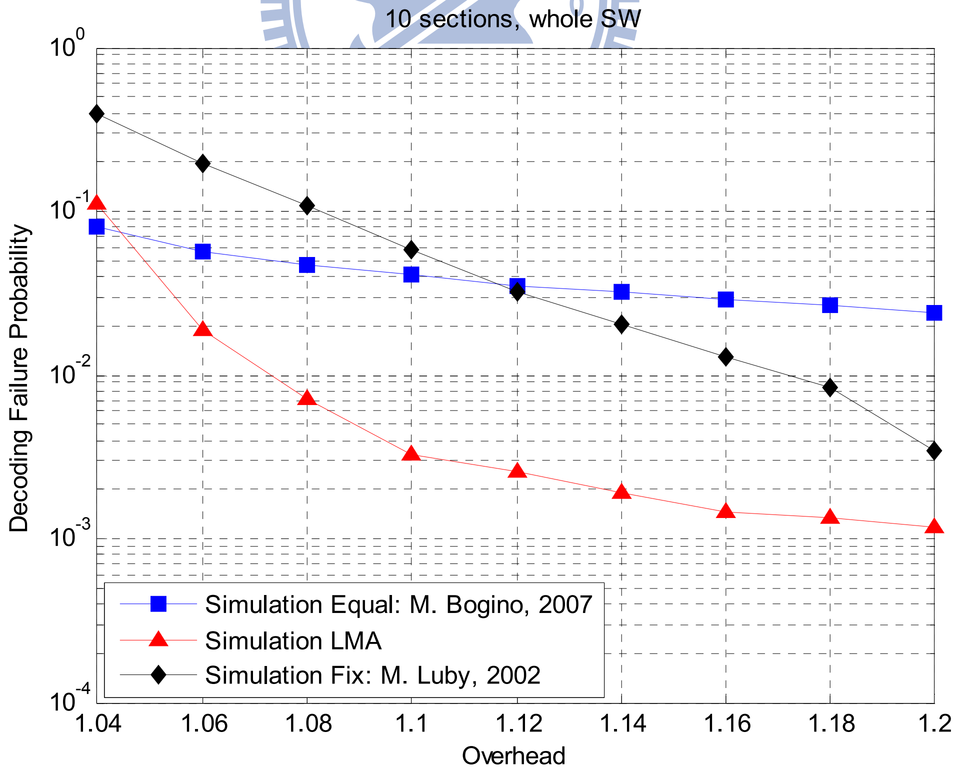


Figure 4-14: Simulation values of Equal and LMA in 10 sections with whole SW.

According to the figures above, we know the buffer size doesn’t need to be large to have low decoding failure probability. The decoding failure probability in buffer size of 3 SW could compare with buffer size of whole SW. Based on the simulation values and estimated values; we could observe the weighted selection is better than uniform selection with our proposed unequal overlapped rateless codes.

Chapter 5 Conclusion

Transmitting entire multimedia streaming file could cause long time delay. Section-based system is a good solution for shortening time delay and forward error correction codes are good choices in this real-time situation and rateless codes have lower encoding and decoding complexity than Reed-Solomon codes. Rateless codes are more suitable than Reed-Solomon in the multimedia streaming.

We could observe that the sliding window scheme will cause a situation: a portion of recovered message blocks in the overlapped part and no portion of recovered message blocks in the non-overlapped part. If we give the two parts different weights to encode, we could get different decoding failure probability in each part with our proposed analytic model. The analytic model of unequal overlapped rateless codes could provide a good expected decoding failure probability in a single section. We could use the weighted selection to provide a lower decoding failure probability than uniform selection.

LMA and DPA could find the weights sequence of weights to provide lower decoding failure probability than uniform selection scheme in the multimedia streaming. Low decoding failure probability in the multimedia streaming could provide good quality of the multimedia streaming contents. We could also have low decoding failure probability in limited buffer size. If the buffer size is larger, we could have lower decoding failure probability.

Chapter 6 Reference

[1] M. Luby, "LT codes," in *Proc. 43rd Annu. IEEE Symp. Foundations of Computer Science (FOCS)*, Vancouver, BC, Canada, Nov. 2002, pp. 271–280.

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

[3] M. Mitzenmacher, "Digital fountains: A survey and look forward," in *Proc. Inf. Theory Workshop*, Oct. 2004, pp. 271–276.

[4] P. Maymounkov, "Online codes," *NYU Technical Report TR2003-883*, 2002.

[5] P. Elias, "Coding for two noisy channels," in *Proc. 3rd London Symp. Information Theory*, London, U.K., pp. 61–76, 1955.

[6] R. G. Gallager, "Low-density parity-check codes," *IEEE Transactions on Information Theory*, 8(1), January 1962.

[7] J. S. Plank and M. G. Thomason, "On the practical use of LDPC erasure codes for distributed storage applications", *Technical Report UT-CS-0-510*, University of Tennessee, September 23, 2003.

[8] M. Luby, M. Mitzenmacher, A. Shokrollahi and D. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 569–584, 2001.

[9] Wicker, "Error control systems for digital communication and storage", Prentice-Hall, 1995.

[10] M. Bogino, P. Cataldi, M. Grangetto, E. Magli, and G. Olmo, "Sliding-window digital

fountain codes for streaming of multimedia contents," *IEEE International Symposium on Circuits and Systems (ISCAS)*, New Orleans, USA, May 2007.

[11] N. Rahnavard, B. Vellambi and F. Fekri, "Rateless codes with unequal error pProtection property," *IEEE Trans. Inf. Theory*, vol. 53, pp. 1521–1532, April 2007.

[12] N. Rahnavard and F. Fekri, "Generalization of rateless codes for unequal error protection and recovery time: Asymptotic analysis," in *Proc. IEEE Int. Symp. Inf. Theory*, Seattle, WA, Jul. 2006, pp. 523–527.

[13] M. Luby, M. Mitzenmacher, and A. Shokrallahi, "Analysis of random processes via and-or tree evaluation," in *Proc. 9th Annu. ACM-SIAM Symp. Discrete Algorithms*, 1998, pp. 364–373.

[14] E. Hyytiä, T. Tirronen, J. Virtamo, "Optimizing the degree distribution of LT codes with an importance sampling approach," in *RESIM 2006, 6th International Workshop on Rare Event Simulation*, Bamberg, Germany, 2006.

[15] E. Hyytiä, T. Tirronen, J. Virtamo, "Optimizing degree distribution for LT codes with small message length," in *IEEE Infocom mini-symposium*, pp. 2576–2580, Anchorage, Alaska, 2007.

[16] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proc. 29th Annu. ACM Symp. Theory Computing (STOC)*, 1997, pp. 150–159.

[17] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *Proc. ACM SIGCOMM,* Vancouver, BC, Canada, Aug. 1998, pp. 56–67.

[18] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Analysis of low density

codes and improved designs using irregular graphs," *IEEE Trans. Inform. Theory*, vol. 47, pp. 585–598, 2001.

[19] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, 2001.

[20] D. Sejdinović, D. Vukobratović, A. Doufexi, V. Šenk, and R. Piechocki, "Expanding window fountain codes for unequal error protection," *preprint, submitted for publication, 41st Annual Asilomar Conference on Signals, Systems and Computers 2007*.

[21] D. Sejdinović, D. Vukobratović, A. Doufexi, V. Šenk, and R. Piechocki, "Scalable data multicast using expanding window fountain codes," in *Proceedings of the 45th Annual Allerton 2007 Conf. on Communication, Control and Computing, Monticello*, IL, USA, September 2007.

[22] Noga Alon, Joel Spencer. *The probabilistic method*. John Wiley & Sons, 2000.

[23] Wallenius, K. T. "Biased sampling: The non-central hypergeometric probability distribution." Ph.D. Thesis, Stanford University, Department of Statistics, 1963.

[24] Johnson, N. L., Kemp, A. W. Kotz, S. *Univariate discrete distributions*. Hoboken, New Jersey: Wiley and Sons, 2005.

[25] Fog, A. "Sampling methods for Wallenius' and Fisher's noncentral hypergeometric distributions," *Commun. Statist. Simul. Computat.* 37:241–257, 2008.

[26] Fog, A. "Calculation methods for Wallenius' noncentral hypergeometric distribution", *Communications in statistics, Simulation and Computation* 37(2): 258–273, 2008.