2011

LT Code Equations

Stanley Truitt

LT Code Equations

Stanley Truitt

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
Electrical Engineering
Telecommunications

University of Mississippi

2011

Abstract

The LT Code equations describe the process of encoding input symbols into an error correcting code that requires no feedback from the receiver. The mathematical process involved the development of LT Codes is important.

This thesis address the issue of improving the understanding of the LT Code equations presented in the original paper. This task is accomplished by inserting the mathematical details when possible, providing graphical results to the equations, and comparing the equations results against random computer generated simulations results.

This thesis will improve the understanding of how LT Codes equations related to actual results.

Dedication

This thesis is dedicated to the faculty and staff of the School of Engineering for their support and the Graduate School for their assistant with formatting this document.

LIST OF ABBREVIATIONS AND SYMBOLS

k The number of input symbols

K The number of encoding symbols

d The degree of an encoding symbol

δ The allowable failure probability of the decoder to recover the data.

L The number of input symbols unprocessed.

i Degree of encoding symbol

e Erasure probability of the BEC channel.

P_b Bit erasure probability.

Acknowledgments

I express my deepest appreciate to my advisor, Dr.Lei Cao and my committee members, Dr. Paul M. Goggans, and Dr. John N. Dsigle. I could not have financed my studies without the assistantship provided by the Department of Electrical Engineering.

In addition, I thank Dr. John N. Daigle of the University of Mississippi for his guidance with understanding Probability Modeling.

Table of Contents

List of Figures

Chapter 1

Introduction

Michael Luby developed and presented LT codes at the 43 rd Annual IEEE Symposium on Foundations for Computer Science in 2002. They are the first class of erasure codes called universal erasure codes. The symbol length of the encoding symbols varies from 1 bit to l bit binary symbols. LT codes have the ability to produce a limitless number of encoding symbols from the input symbols. They can adjust the number of encoding symbols as needed so that LT codes are near optimal.

1.1 Application

LT codes offer dependability for transferring data between applications. The advantage of using LT codes in a one-to-one data delivery system is that the flow and congestion control mechanisms are designed independently of reliability, M. Adler (1997).

LT Codes offer advantages for the one-to-many data transfer problem, where the receiver needs to minimize the feedback to the sender. In this type of system retransmitting data that has already been received by some receivers is inefficiency as given in J. Nonnenmacher (1996), E. Schooler (1997). The benefit of using LT codes is that a single sender can be used to dependably transport data to a large number of coexisting receivers without feedback.

The problem of several senders transmitting the same data simultaneously from possible different locations to one receiver is an issue that can be addressed using LT codes. There is the possibility that the same data packets are received multiple times. This situation is well thought-out in J.W. Byers (1999).

1.2 Purpose

The purpose of this thesis is to examine the mathematical techniques used to develop the LT Code equations and to provide graphical displays of these equations when possible. It also provides computer simulations that support the LT code equations. The information presented is in the same order as given in Luby's origin paper.

Chapter 2

LT Codes Design

The lengths of the encoding symbols are selected as required for the situation but the process is more efficient for large values of code length. The encoder partition the data in k input symbols of length l. Luby describes the encoding process in Luby (2002). The encoder works as follows, the data has length of N is partitioned into $k = N/l$ input symbols, each input symbol is of length l. The process of generating an encoding symbol is conceptually very easy to describe:

- Randomly choose the degree d of the encoding symbol from a degree distribution.
- Choose uniformly at random d distinct input symbols as neighbors of the encoding symbol.
- The value of the encoding symbol is the exclusive-or of the d neighbors.

Definition 1 (Decoder recovery rule ,from Luby (2002):) *If there is at least one encoding symbol that has exactly one neighbor then the neighbor can be recovered immediately since it is a copy of the encoding symbol. The value of the recovered input symbol is exclusive-ored into any remaining encoding symbols that also have that input symbol as a neighbor, the recovered input symbol is removed as a neighbor from each of these encoding symbols and the degree of each such encoding symbol is decreased by one to reflect this removal.*

2.1 LT Process

The LT process describes the goal and assessment of the LT code degree distribution. Let K denote the number of encoding symbols required in the process. The process takes a broad view of the process of throwing balls into bins. The mean number of balls required to have at least one ball in each bin is $K = k \cdot \ ln(k/\delta)$ with probability $1 - \delta$. The balls represent encoding symbols and the bins represent the input symbols in the analysis of the LT process.

Definition 2 (LT process, from Luby (2002):) *All input symbols are initially uncovered. At the first step all encoding symbols with one neighbor are released to cover their unique neighbor. The set of covered input symbols that have not yet been processed is called the ripple, and thus at this point all covered input symbols are in the ripple. At each subsequent step one input symbol in the ripple is processed: it is removed as a neighbor from all encoding symbols which have it as a neighbor and all such encoding symbols that subsequently have exactly one remaining neighbor are released to cover their remaining neighbor. Some of these neighbors may have been previously uncovered, causing the ripple to grow, while others of these neighbors may have already been in the ripple, causing no growth in the ripple. The process ends when the ripple is empty at the end of some step. The process fails if there is at least one uncovered input symbol at the end. The process succeeds if all input symbols are covered by the end.*

The following figures are from Thomas Stockhammer (2009),present a graphical representation of the LT encoding process. The encoding symbols are denoted by Y_i and the input symbols are denoted by X_j as in figure 2.1. At the first step all encoding symbols with one neighbor are released to cover their unique neighbor. Encoding symbol Y_5 is released to cover its one neighbor input symbol X_5 as shown in figure 2.2. The set of covered input symbols that have not yet been processed is called the ripple, and input symbol X_5 is in the ripple. When symbol X_5 is processed, it is removed as a neighbor of encoding symbol Y_4, Y_5, and Y_6 as illustrated in figure 2.3. Encoding symbol Y_4 is released to cover

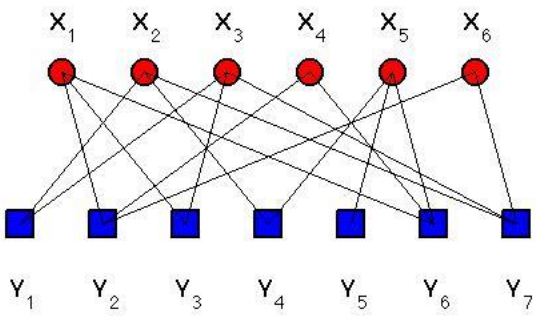

Figure 2.1. Encoding symbols and Input symbols

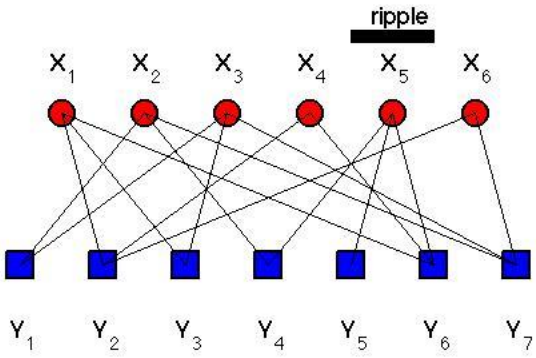

Figure 2.2. Input symbol in the ripple

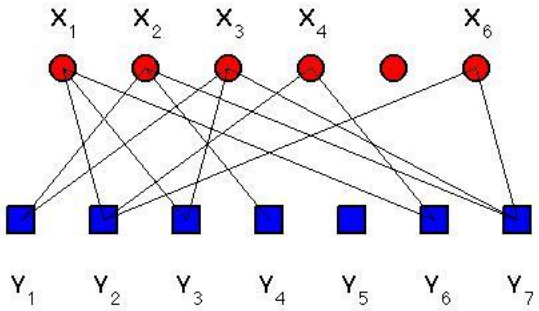

Figure 2.3. Input symbol is processed

its one neighbor input symbol X_2 as shown in figure 2.4; at this point X_2 is in the ripple. Input symbol X_2 is processed, it is removed as a neighbor of encoding symbol Y_1, Y_4, and Y_7 as illustrated in figure 2.5. The LT process is completed in figures 2.6, 2.7, 2.8, 2.9, 2.10, and 2.11. Note in figure 2.8 that the ripple increases to two input symbols. This process continues until the ripple is empty at the last step. The process fails if there is at least one uncovered input symbol at the end. The process succeeds if all input symbols are covered by the end.

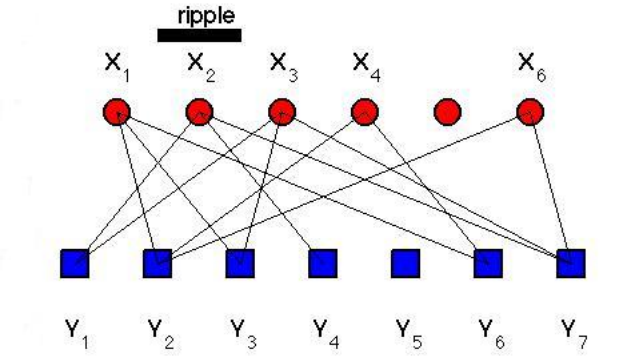

Figure 2.4. Input symbol X_2 is in the ripple

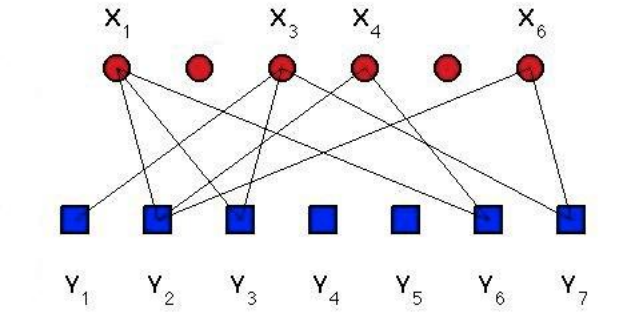

Figure 2.5. Input symbol is processed

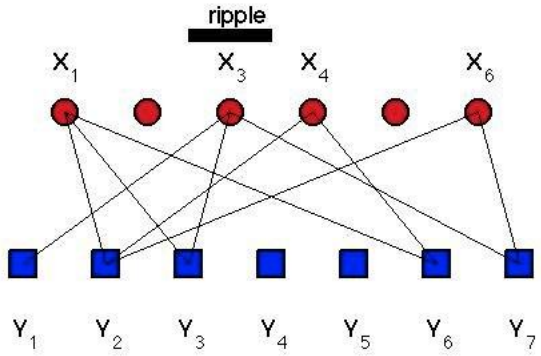

Figure 2.6. Input symbol three in in the ripple

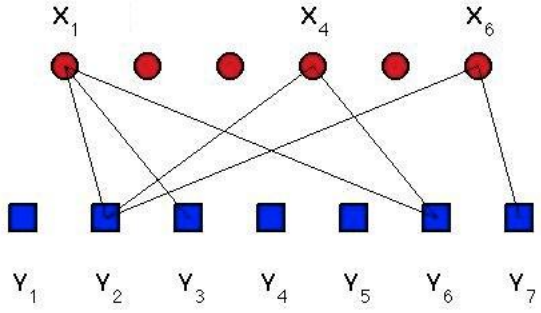

Figure 2.7. Input symbol three is processed

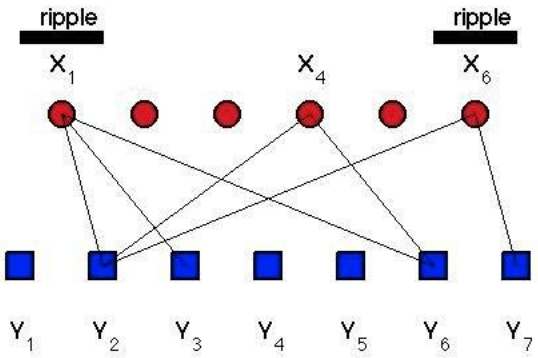

Figure 2.8. Input symbols one and six are in the ripple

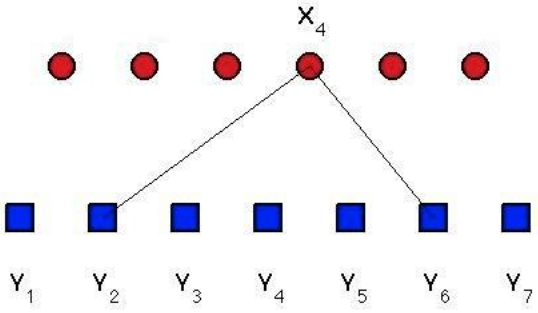

Figure 2.9. Input symbols are processed

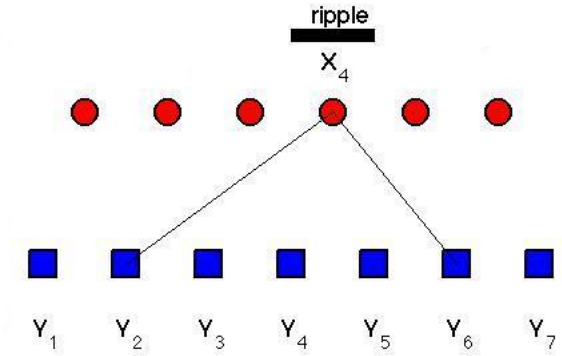

Figure 2.10. Input symbol four in the ripple

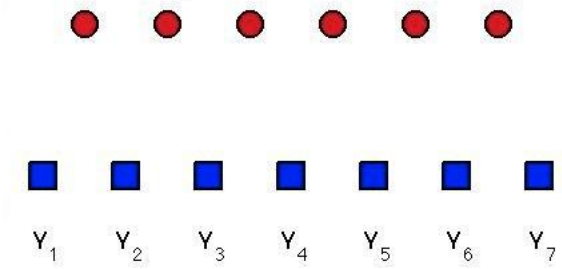

Figure 2.11. The LT Process is complete

Chapter 3

LT Degree Distribution

Each encoding symbol's degree is selected separately by the execution of a probability degree distribution formula.

Definition 3 (Degree distribution, from Luby (2002):) *For all d, $\rho(d)$ is the probability that an encoding symbol has degree d.*

The performance of the LT process is influenced by following variables.

- The degree distribution $\rho(\cdot)$.
- The number of encoding symbols K.
- The number of input symbols k.

The goal is to design the degree distribution to comply with these goals.

- As few encoding symbols as possible on average are required to ensure success of the LT process.
- The average degree of the encoding symbols is as low as possible.

The LT process can be analyzed similar to solving the problem of throwing balls into bins.

Definition 4 (All-At-Once distribution, from Luby (2002):) $\rho(1) = 1$

The classical balls and bins process suggest that $k \cdot \ln(k/\delta)$ encoding symbols are required to encoded k input symbols with probability $1 - \delta$.

3.1 Some preliminary probabilistic analysis

In this section the task is to determine what is the possibility of an encoding symbol of degree i is released when there are L input symbols that remain unprocessed.

Definition 5 (Encoding symbol release, from Luby (2002)) *Let us say that an encoding symbol is released when L input symbols remain unprocessed if it is released by the processing of the $k - L^{th}$ input symbol, at which point the encoding symbol randomly covers one of the L unprocessed input symbols.*

Definition 6 (Degree release probability, from Luby (2002)) *Let $q(i, L)$ be the probability that an encoding symbol of degree i is released when L input symbols remain unprocessed.*

Proposition 7 (Degree release probability formula, from Luby (2002))

- $q(1, k) = 1$.
- *For $i = 2, \ldots, k$, for all $L = k - i + 1, \ldots, 1$*

$$q(i, L) = \frac{i(i-1) \cdot L \cdot \prod_{j=0}^{i-3}(k - (L+1) - j)}{\prod_{j=0}^{i-1} k - j}$$

- *For all other i and L, $q(i, L) = 0$.*

Proof. From Luby (2002), This is the probability that $i-2$ of the neighbors of the encoding symbol are among the first $k-(L+1)$ symbols processed, one neighbor is processed at step $k - L$, and the remaining neighbor is among the L unprocessed input symbols. ■

Definition 8 (Overall release probability, from Luby (2002)) *Let $r(i, L)$ be the probability that an encoding symbol is chosen to be of degree i and is released when L input symbols remain unprocessed, i.e. $r(i, L) = \rho(i) \cdot q(i, L)$. Let $r(L)$ be the overall probability that an encoding symbol is released when L input symbols remain unprocessed, i.e., $r(L) = \sum_i r(i, L)$.*

Figure 3.1 plots the degree release probability formula curves for encoding symbols of degree $i = 2, i = 3, i = 5, i = 10$, and $i = 15$ with $k = 1000$ inputs symbols.

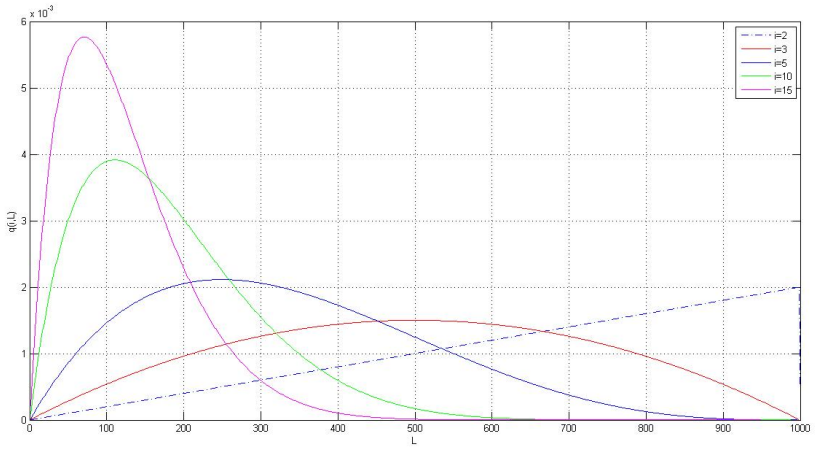

Figure 3.1. Degree release probability formula

3.2 The Ideal Soliton distribution

The preferred results for the distribution is to have the minority number of released encoding symbols cover an input symbol that is currently in the ripple. The purpose is to keep the number of input symbols in the ripple at the least possible number. The number of input symbols in the ripple should be maintained large enough to complete the LT process. The number input symbols should not be to a reasonable number.

The Ideal Soliton distribution performance is unacceptable when applied to a real system but it helps to explain the Robust Soliton distribution.

Definition 9 (Ideal Soliton Distribution, from Luby (2002)) *The Ideal Soliton distribution is* $\rho(1), \ldots, \rho(k)$*, where*

- $\rho(1) = 1/k$
- *For all* $i = 2, \ldots, k, \rho(i) = 1/i(i-1)$

Note that $\sum_i \rho(i) = 1$ as required for a probability distribution.

$$\sum_{i=1}^{k} \rho(i) = 1$$

$$\sum_{i=1}^{k} \rho(i) = \rho(i) + \sum_{i=2}^{k} \frac{1}{i(i-1)}$$

$$\sum_{i=1}^{k} \rho(i) = \frac{1}{k} + \sum_{i=2}^{k} \frac{1}{i(i-1)}$$

Proof by mathematical induction,

For $k = 2$

$$\sum_{i=1}^{2} \rho(i) = \frac{1}{2} + \sum_{i=2}^{2} \frac{1}{i(i-1)}$$

$$\sum_{i=1}^{2} \rho(i) = \frac{1}{2} + \frac{1}{2(2-1)}$$

$$\sum_{i=1}^{2} \rho(i) = \frac{1}{2} + \frac{1}{2} = 1$$

For the term,

$$\sum_{i=2}^{k} \frac{1}{i(i-1)}$$

$$\sum_{i=2}^{2} \frac{1}{i(i-1)} = \frac{1}{2(2-1)} = \frac{1}{2}$$

$$\sum_{i=2}^{3} \frac{1}{i(i-1)} = \frac{1}{2(2-1)} + \frac{1}{3(3-1)} = \frac{2}{3}$$

$$\sum_{i=2}^{4} \frac{1}{i(i-1)} = \frac{1}{2(2-1)} + \frac{1}{3(3-1)} + \frac{1}{4(4-1)} = \frac{3}{4}$$

The general pattern appears to be,

$$\sum_{i=2}^{k} \frac{1}{i(i-1)} = \frac{k-1}{k}$$

For $k = n$

$$\sum_{i=2}^{n} \frac{1}{i(i-1)} = \frac{n-1}{n}$$

For $k = n + 1$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{(n+1)-1}{n+1}$$

Induction steps,

$$\sum_{i=2}^{n} \frac{1}{i(i-1)} = \frac{n-1}{n}$$

$$\sum_{i=2}^{n} \frac{1}{i(i-1)} + \frac{1}{(n+1)((n+1)-1)} = \frac{n-1}{n} + \frac{1}{(n+1)((n+1)-1)}$$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{n-1}{n} + \frac{1}{(n+1)n}$$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{n-1}{n}\frac{(n+1)}{(n+1)} + \frac{1}{n(n+1)}$$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{n^2-1}{n(n+1)} + \frac{1}{n(n+1)}$$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{n^2-1+1}{n(n+1)}$$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{n^2}{n(n+1)}$$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{n}{(n+1)}$$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{n+1-1}{(n+1)}$$

$$\sum_{i=2}^{n+1} \frac{1}{i(i-1)} = \frac{(n+1)-1}{(n+1)}$$

Figure 3.2 is a plot of the points 1 to 50 of the Ideal soliton distribution.

A method for selecting a sample from the distribution is to generate a random real value $v \in (0, 1]$, and then for $i = 2, \ldots, k$, let the sample value be i if $1/i < v \leq 1/i - 1$, and

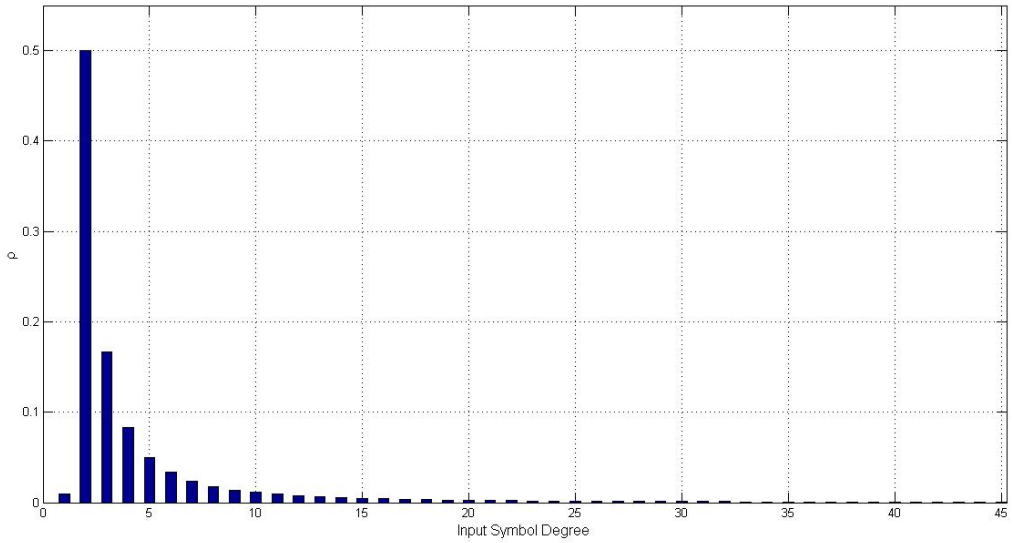

Figure 3.2. Ideal Soliton for $k = 100$

let the sample value be 1 if $0 < v \leq 1/k$. The expected degree of an encoding symbol is $\sum_{i=1}^{k} i/i(i-1) = H(k)$, where $H(k) \approx \ln(k)$ is the harmonic sum up to k. This method was derived by Luby, Luby (2002). Figure 3.3 displays a plot of sample select equations for k=100 input symbols and figure 3.4 displays the first 40 points.

Proposition 10 (Uniform release Probability, from Luby (2002)) *For the Ideal Soliton distribution,* $r(L) = 1/k$ *for all* $L = k, \ldots, 1$

Proof. From Luby (2002), for $L = k$ all encoding symbols of degree one are released, and an encoding symbol is of degree one with probability $1/k$, and thus the statement is true

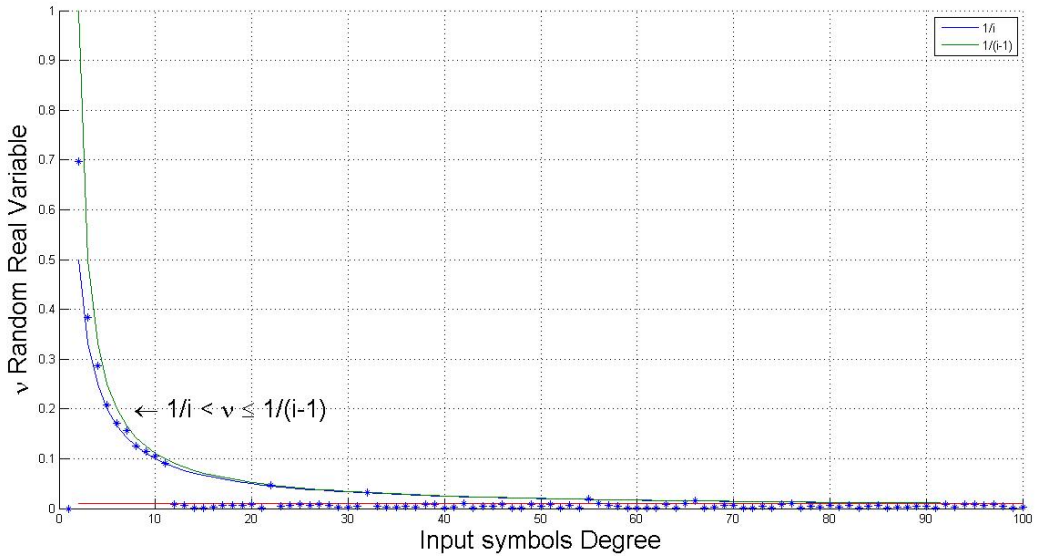

Figure 3.3. Sample selection for $k = 100$

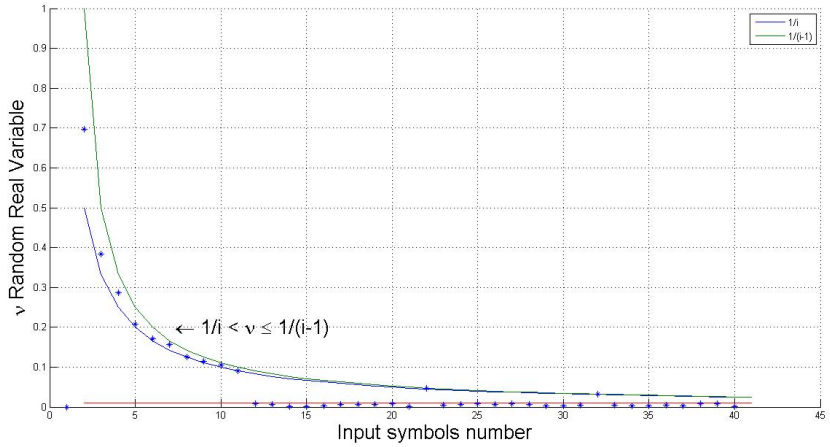

Figure 3.4. Sample selection for $k = 100$

for $L = k$.

$$\begin{aligned}
\rho(1) &= \frac{1}{k} \text{ given} \\
q(1, L) &= 1 \text{ given} \\
r(L) &= \sum_{i=1}^{1} r(i, L) \\
&= \sum_{i=1}^{1} \rho(i) \cdot q(i, L) \\
&= \rho(1) \cdot q(1, L) \\
&= (\frac{1}{k})(1)
\end{aligned}$$

$$r(L) = \frac{1}{k} \tag{3.1}$$

For all other values of L,

$$r(i, L) = \rho(i) \cdot q(i, L)$$

$$\rho(i) = \frac{1}{i(i-1)}$$

$$q(i, L) = \frac{i(i-1) \cdot L \cdot \prod_{j=0}^{i-3}(k - (L+1) - j)}{\prod_{j=0}^{i-1}(k-j)}$$

$$r(i, L) = \frac{1}{i(i-1)} \frac{i(i-1) \cdot L \cdot \prod_{j=0}^{i-3}(k - (L+1) - j)}{\prod_{j=0}^{i-1}(k-j)}$$

$$r(i, L) = \frac{L \cdot \prod_{j=0}^{i-3}(k - (L+1) - j)}{\prod_{j=0}^{i-1}(k-j)}$$

$$r(L) = \sum_{i} r(i, L)$$

$$r(i, L) = \sum_{i=2}^{k} \frac{L \cdot \prod_{j=0}^{i-3}(k - (L+1) - j)}{\prod_{j=0}^{i-1}(k-j)}$$

It can be verified that $k \cdot r(i, L)$ can be interpreted as the probability that when throwing balls uniformly at random among $k-1$ bins, eliminating each bin as it is covered by a ball, that it is the $(i-1)^{\text{rst}}$ ball thrown that lands in one of L designated bins. These events are mutually exclusive for different values of i, and since $i = 2, \ldots, k-L+1$ covers all the possible outcomes,

$$k \cdot r(i, L) = k \sum_{i=1}^{k-L+1} r(i, L) = 1$$

■

The Ideal soliton distribution behavior is not the actual behavior as note by Luby. The Ideal Soliton distribution performs disappointingly when applied to an actual system because the ripple size of one is too small.

3.3 Robust Solition Distribution

The Ideal Soliton distribution performs inadequately when implemented in an actual system because there is only one input symbol in the ripple. Any change in the ripple could cause the encoding process to fail. The Robust Soliton distribution increases the number of input symbols in the ripple therefore reducing the probability that it will vanish before the process is complete.

The inspiration for the design of the Robust Soliton distribution is to maintain the mean number of input symbols in the ripple at approximately $\ln(k/\delta)\sqrt{k}$ during the process.

Definition 11 (Robust Soliton distribution, from Luby (2002)) *The Robust Soliton distribution is $\mu(\cdot)$ define as follows. Let $R = c \cdot \ln(k/\delta)\sqrt{k}$ for some suitable constant $c > 0$. Define*

$$\tau(i) = \begin{cases} R/ik & \text{for } i = 1, \ldots, k/R - 1 \\ R\ln(R/\delta)/k & \text{for } i = k/R \\ 0 & \text{for } i = k/R + 1, \ldots, k \end{cases}$$

Add the Ideal Soliton distribution $\rho(\cdot)$ to $\tau(\cdot)$ and normalize to obtain $\mu(\cdot)$:

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

The variable $\tau(\cdot)$ makes sure that the ripple's initial size is adequate. Consider the process in the middle. Suppose that an input symbol is processed and L input symbols remain unprocessed. Each time an input symbol is processed that symbol is removed from the ripple; another input symbol should be added to replace the processed one. Let R denote the size of the ripple then $(L-R)/L$ is the probability that an encoding symbol adds to the ripple. This implies that it requires $L/(L-R)$ released encoding symbols on average to add one to the ripple.

From Proposition 7 it is possible to verify that the release rate of encoding symbols of degree i for i within a constant factor of k/L make up a constant portion of the release rate when L input symbols remain unprocessed. Thus, if the ripple size is to be maintained at approximately R, then the density of encoding symbols with degree $i = k/L$ should be proportional to

$$\rho(i) \cdot \frac{L}{(L-R)} = \frac{L}{i(i-1) \cdot (L-R)}$$

$$\frac{L}{i(i-1) \cdot (L-R)} = \frac{k/i}{i(i-1) \cdot (k/i - R)}$$

$$\frac{L}{i(i-1) \cdot (L-R)} = \frac{k/i}{i(i-1) \cdot (k/i - R)}$$

$$\frac{L}{i(i-1) \cdot (L-R)} = \frac{k/i}{i(i-1) \cdot ((k-iR)/i)}$$

$$\frac{L}{i(i-1) \cdot (L-R)} = \frac{k/i}{i(i-1)\frac{(k-iR)}{i}}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{k/i}{(i-1)\cdot(k-iR)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{k}{i(i-1)\cdot(k-iR)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{k}{i(i-1)\cdot(k-iR)} + \frac{1}{i(i-1)} - \frac{1}{i(i-1)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{1}{i(i-1)} + \frac{k}{i(i-1)\cdot(k-iR)} - \frac{1}{i(i-1)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{1}{i(i-1)} + \frac{k}{i(i-1)\cdot(k-iR)} - \frac{1}{i(i-1)}\cdot\frac{(k-iR)}{(k-iR)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{1}{i(i-1)} + \frac{k}{i(i-1)\cdot(k-iR)} - \frac{(k-iR)}{i(i-1)\cdot(k-iR)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{1}{i(i-1)} + \frac{k-(k-iR)}{i(i-1)\cdot(k-iR)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{1}{i(i-1)} + \frac{k-k+iR}{i(i-1)\cdot(k-iR)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{1}{i(i-1)} + \frac{iR}{i(i-1)\cdot(k-iR)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} = \frac{1}{i(i-1)} + \frac{R}{(i-1)\cdot(k-iR)}$$

$$\frac{L}{i(i-1)\cdot(L-R)} \approx \rho(i) + \tau(i) \tag{3.2}$$

for $i = 2, \ldots, k/R - 1$, this equation is derived in Luby (2002).

Figure 3.5 and figure 3.6 are plots of τ, figure 3.7 and figure 3.8 are plots of ρ, and figures 3.9 and figure 3.10 are plots of μ. Make note of the spikes at τ/k, a topic Luby discuses. The final spike $\tau(k/R)$ ensures that all the input symbols unprocessed when $L = R$ are all covered. This is similar to simultaneously releasing $R\ln(R/\delta)$ encoding symbols when R input symbols remain unprocessed to cover them all at once as given in Luby (2002).

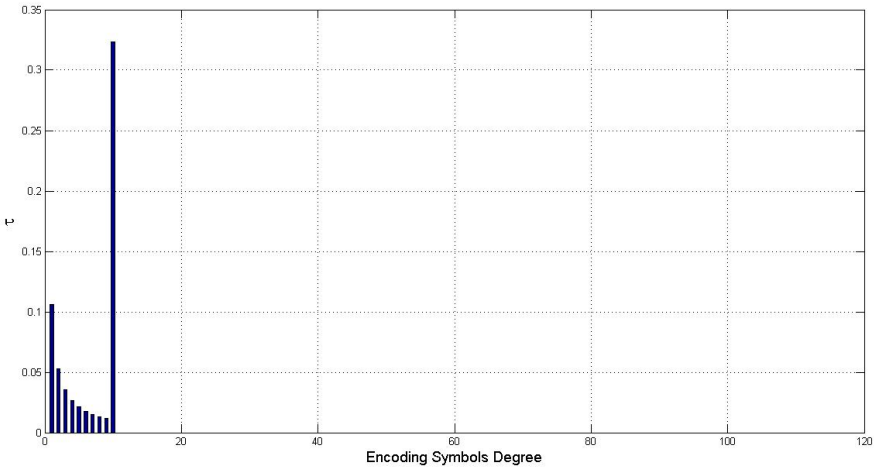

Figure 3.5. $\delta = 0.5, c = 0.2$ **and** $k = 100$

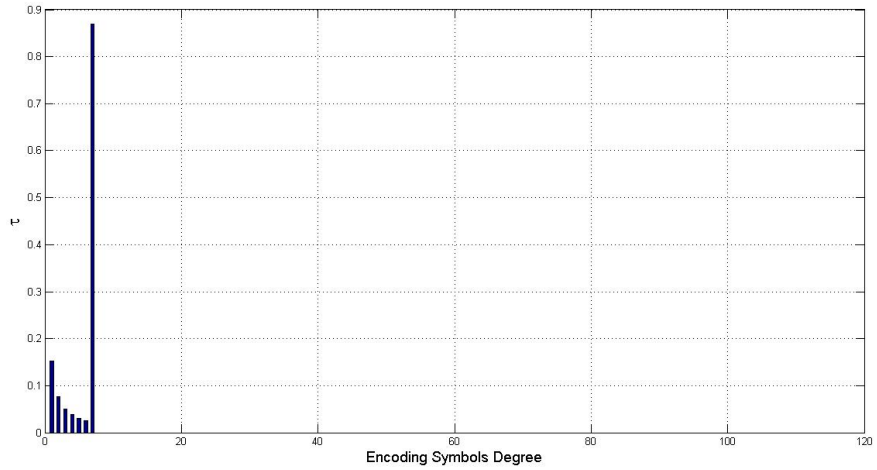

Figure 3.6. $\delta = 0.05, c = 0.2$ **and** $k = 100$

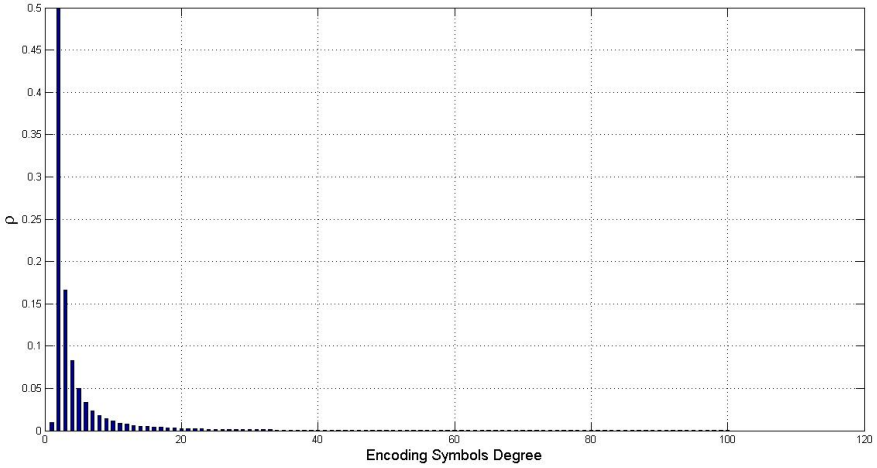

Figure 3.7. $\delta = 0.5, c = 0.2$ **and** $k = 100$

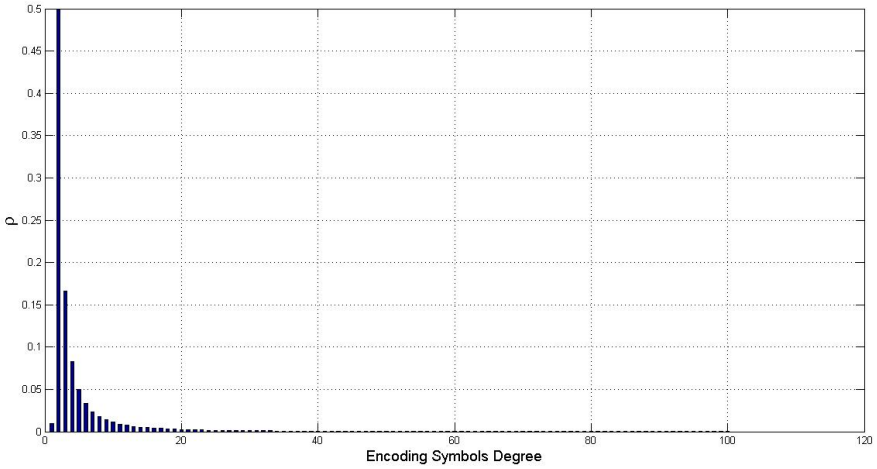

Figure 3.8. $\delta = 0.05, c = 0.2$ **and** $k = 100$

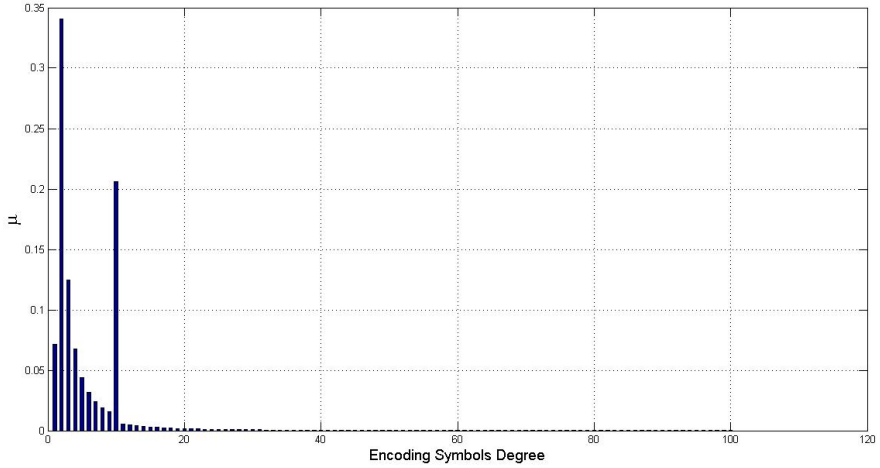

Figure 3.9. $\delta = 0.5, c = 0.2$**, and** $k = 100$

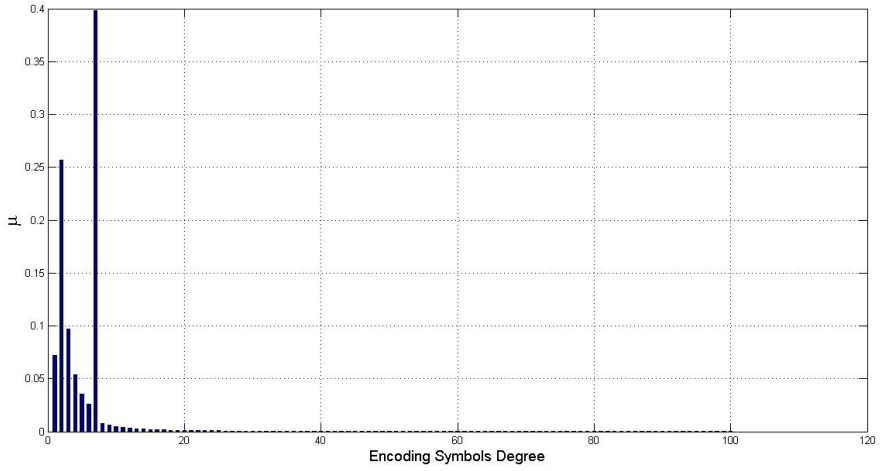

Figure 3.10. $\delta = 0.05, c = 0.2$ **and** $k = 100$

The number of encoding symbols is placed to

$$K = k\beta \tag{3.3}$$

which means that $k \cdot (\rho(i) + \tau(i))$ is the expected number of encoding symbols of degree i.

Figure 3.11 is an actual experiment performed by David MacKay and included in his textbook MacKay (2003). In practice, LT codes can be tuned so that a file of original size $K \approx 10,000$ packets is recovered with an overhead of about 5%. Figure 3.11 shows histograms of the actual number of packets required for a couple of settings of the parameters c and δ.

Figure 3.12, displays the results of using equation (3.3) ,from Luby (2002), to estimate the number of encoding symbols to be approximately $K = 10,104$, for $k = 10,000, c = 0.01$ and $\delta = 0.5$. The estimated number of encoding symbols is near the actual number required when compared to the top histogram in figure 3.11. The estimated number of encoding symbols required to recover a file of size $k = 10,000$ input packets is $K = 10,311$ for $c = 0.03$ and $\delta = 0.5$,figure 3.13 and figure 3.11 middle histogram, and $K = 11,037$ for $c = 0.1$ and $\delta = 0.5$,figure 3.14 and figure 3.11 bottom histogram.

3.4 Analysis of Robust Solition Distribution

This section gives some properties of the Robust Soliton distribution.

Theorem 12 (number of encoding symbols, from Luby (2002)) *The number of encoding symbols is* $K = k + O(\sqrt{k} \cdot \ln^2(k/\delta))$.

Proof. From Luby (2002)

$$\sum_{i=1}^{k} \rho(i) = 1$$

$$\tau(i) = \begin{cases} \frac{R}{ik} & \text{For } i = 1, \ldots, \frac{k}{R} - 1 \\ \frac{R \ln(R/\delta)}{k} & \text{For } i = \frac{k}{R} \\ 0 & \text{for } i = \frac{k}{R} + 1, \ldots, k \end{cases}$$

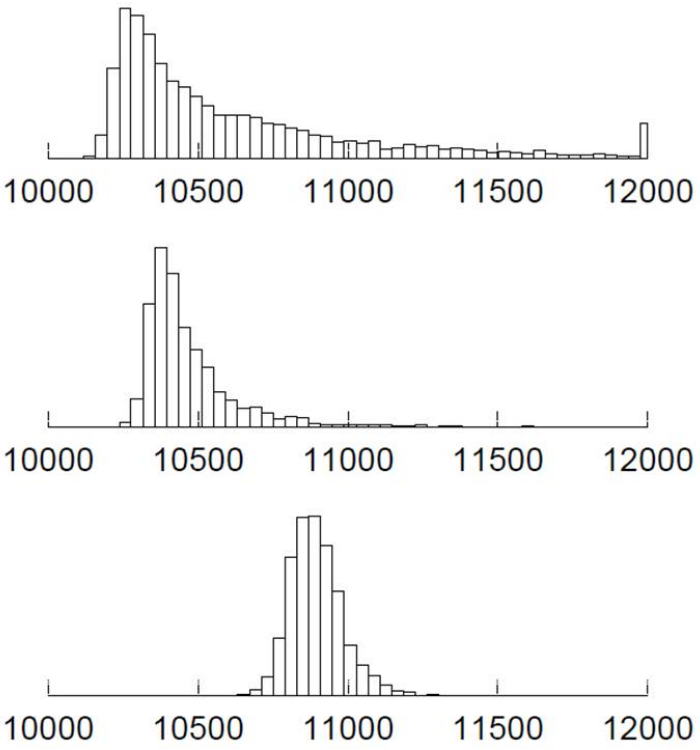

Figure 3.11. Histograms of the actual number of packets N required in order to recover a file of size $K = 10,000$ packets. The parameters were as follows: top histogram: $c = 0.01, \delta = 0.5$, middle: $c = 0.03, \delta = 0.5$, bottom: $c = 0.1, \delta = 0.5$

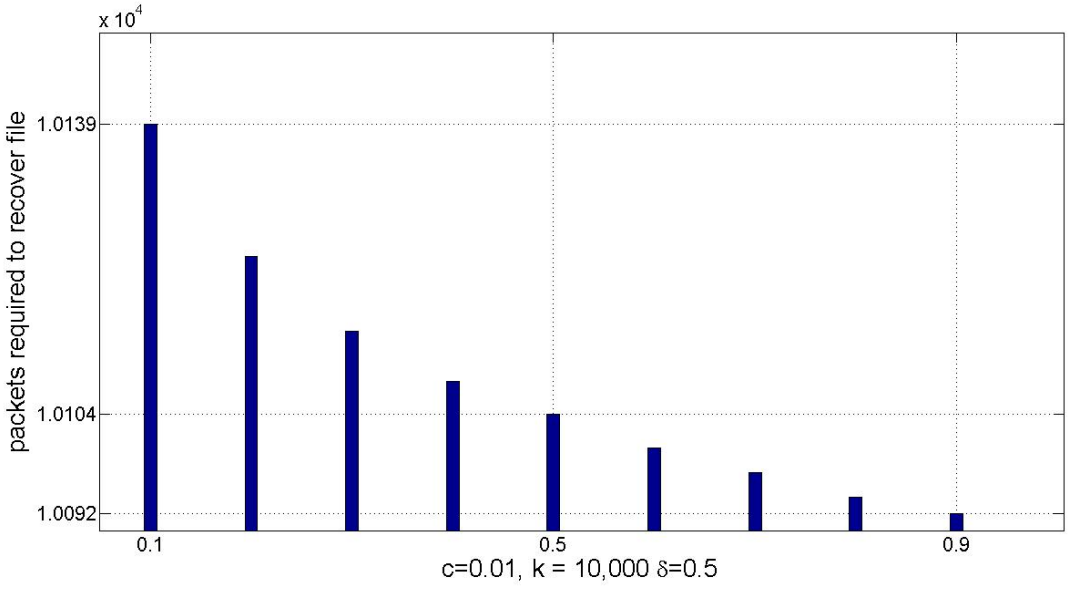

Figure 3.12. $c = 0.01$

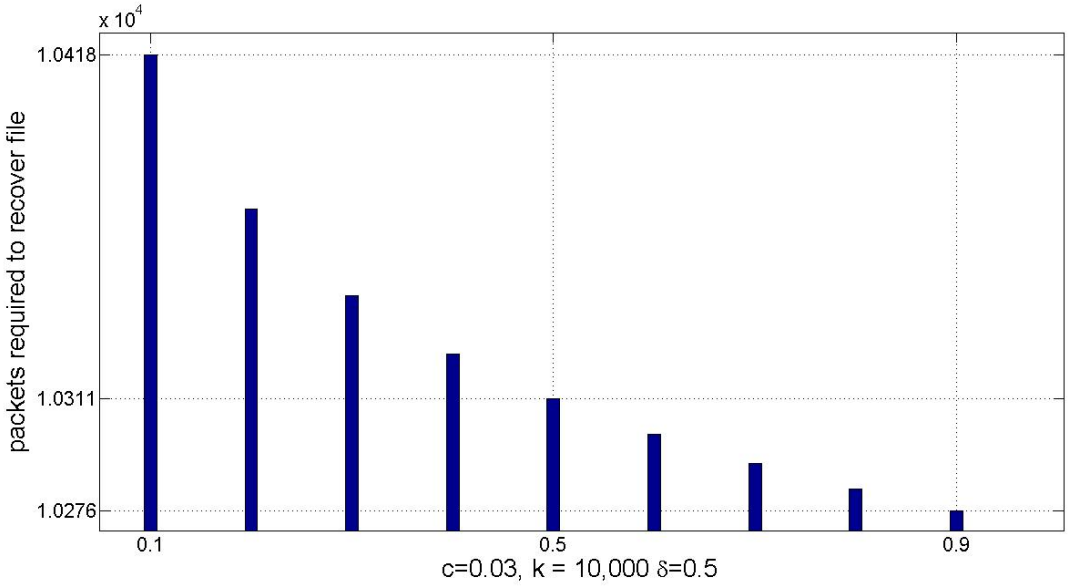

Figure 3.13. $c = 0.03$

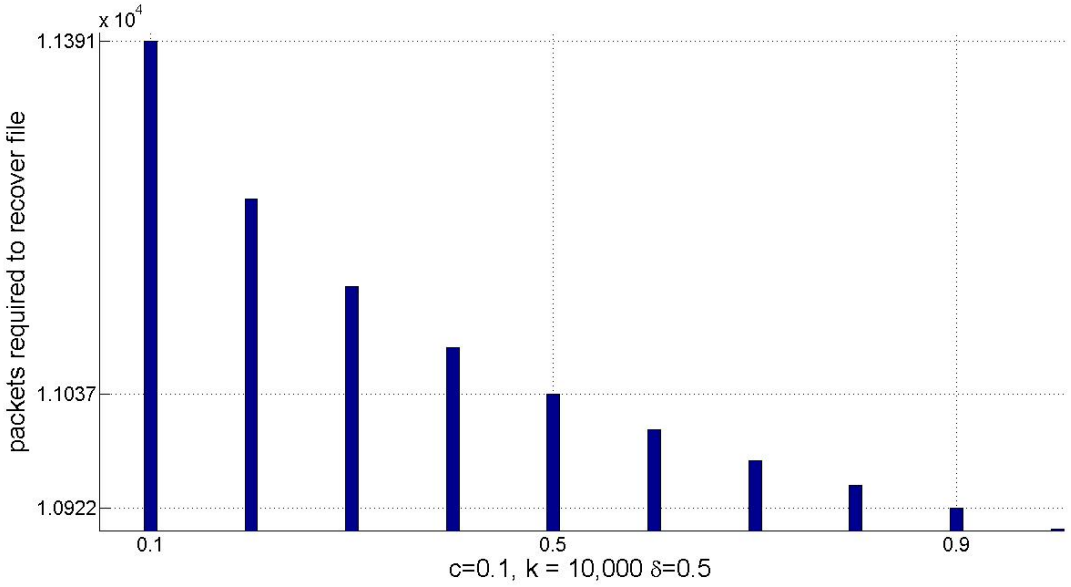

Figure 3.14. $c = 0.1$

$$\beta = \sum_{i=1}^{k} \rho(i) + \tau(i)$$

$$\begin{aligned}
K &= k\beta \\
&= k \cdot \left(\sum_{i=1}^{k} \rho(i) + \tau(i) \right) \\
&= k \cdot \left(1 + \sum_{i=1}^{k/R-1} \tau(i) + \tau(k/R) \right) \\
&= k \cdot \left(1 + \sum_{i=1}^{k/R-1} \frac{R}{ik} + \frac{R \ln(R/\delta)}{k} \right) \\
&= k + \sum_{i=1}^{k/R-1} \frac{R}{i} + R \ln(R/\delta) \\
&\leq k + R \cdot H(k/R) + R \cdot \ln(R/\delta)
\end{aligned}$$

■

Theorem 13 (average degree of an encoding symbol, from Luby (2002)) *The average degree of an encoding symbol is* $D = O(\ln(k/\delta))$.

Proof. From Luby (2002)

$$\mu(i) = \frac{(\rho(i) + \tau(i))}{\beta}$$

$$H(k) = \sum_{i=1}^{k} \frac{i}{i(i-1)}$$

$$
\begin{aligned}
D &= E[\mu(i)] \\
&= \sum_i i\mu(i) \\
&= \sum_i i\frac{(\rho(i)+\tau(i))}{\beta} \\
&\leq \sum_i i(\rho(i)+\tau(i)) \\
&= i\left(\sum_i^{k+1} \frac{1}{i(i-1)} + \sum_{i=1}^{k/R-1} \frac{R}{ik} + R\ln(R/\delta)\right) \\
&= \sum_{i=2}^{k+1} \frac{1}{(i-1)} + \sum_{i=1}^{k/R-1} \frac{R}{k} + R\ln(R/\delta) \\
&\leq H(k) + 1 + \ln(R/\delta)
\end{aligned}
$$

■

The following propositions are used in the proof of the theorem below that the LT process succeeds with high probability.

Proposition 14 (robust uniform release probability, from Luby (2002)) *For all* $L = k-1, \ldots, R$,

$$K \cdot r(L) \geq \frac{L}{(L-\theta R)}$$

for a suitable constant $\theta \geq 0$, *excluding the contribution of* $\tau(k/R)$.

Proof. This proof uses the contributions of $\tau(2), \ldots, \tau(k/R-1)$ and the Ideal Soliton distribution $\rho(\cdot)$. For $L = k/2, \ldots, k-1$ using Proposition 7 and Proposition 10, and the number of encoding symbols ,$K = k\beta$.

This is displayed once more to make the process clearer, Proposition 7(degree release probability formula)

- $q(1,k) = 1$

- *For* $i = 2, \ldots, k$, *for all* $L = k - i + 1, \ldots, 1$

$$q(i, L) = \frac{i(i-1) \cdot L \cdot \prod_{j=0}^{i-3} k - (L+1) - j}{\prod_{j=0}^{i-1} k - j}$$

- For all other i and L,$q(i, L) = 0$

Definition 8

$$r(i, L) = \rho(i) \cdot q(i, L)$$

$$r(L) = \sum_i r(i, L)$$

This is displayed once more to make the process clearer, Proposition 10 (uniform release probability)

$$r(L) = \frac{1}{k}, \quad L = k, \ldots, 1$$

$$r(i, L) = \frac{L \cdot \prod_{j=0}^{i-3} k - (L+1) - j}{\prod_{j=0}^{i-1} k - j}$$

$$k \cdot r(L) = k \cdot \sum_{i=1}^{k-L+1} r(i, L) = 1$$

This is displayed once more to make the process clearer, Proposition 14 Proof (robust uniform release probability)

$L = k/2, \ldots, k - 1$

$$K \cdot r(L) \geq K \cdot \left(\frac{\left(\sum_i \frac{1}{i(i-1)} \cdot q(i, L) \right) + \tau(2) \cdot q(2, L)}{\beta} \right)$$

$$K \cdot r(L) \geq K \cdot \left(\frac{\left(\sum_i \frac{1}{i(i-1)} \cdot \frac{i(i-1) \cdot L \cdot \prod_{j=0}^{i-3} k - (L+1) - j}{\prod_{j=0}^{i-1} k - j} \right) + \tau(2) \cdot q(2, L)}{\beta} \right)$$

$$K \cdot r(L) \geq K \cdot \left(\frac{\left(\sum_i \frac{L \cdot \prod_{j=0}^{i-3} k-(L+1)-j}{\prod_{j=0}^{i-1} k-j} \right) + \tau(2) \cdot q(2,L)}{\beta} \right)$$

$$K \cdot r(L) \geq K \cdot \left(\frac{\left(\sum_i r(i,L) \right) + \tau(2) \cdot q(2,L)}{\beta} \right)$$

$$K \cdot r(L) \geq K \cdot \left(\frac{r(L) + \tau(2) \cdot q(2,L)}{\beta} \right)$$

$$K \cdot r(L) \geq k\beta \cdot \left(\frac{r(L) + \tau(2) \cdot q(2,L)}{\beta} \right)$$

$$K \cdot r(L) \geq k \cdot (r(L) + \tau(2) \cdot q(2,L))$$

$$K \cdot r(L) \geq kr(L) + k\tau(2) \cdot q(2,L)$$

By;

$$k \cdot r(L) = k \cdot \sum_{i=1}^{k-L+1} r(i,L) = 1$$

Then

$$K \cdot r(L) \geq 1 + k\tau(2) \cdot q(2,L)$$

By;

$$\tau(i) = \frac{R}{ik} \quad for \;\; i = 1, \ldots, k/R - 1$$

$$\tau(2) = \frac{R}{2k}$$

$$q(i,L) = \frac{i(i-1) \cdot L \cdot \prod_{j=0}^{i-3} k - (L+1) - j}{\prod_{j=0}^{i-1} k - j}$$

$$q(2,L) = \frac{2(2-1) \cdot L \cdot \prod_{j=0}^{2-3} k - (L+1) - j}{\prod_{j=0}^{2-1} k - j}$$

$$q(2,L) = \frac{2(1)\cdot L \cdot \prod_{j=0}^{-1} k-(L+1)-j}{\prod_{j=0}^{1} k-j}$$

$$q(2,L) = \frac{2\cdot L\cdot(1)}{(k-0)\cdot(k-1)} = \frac{2\cdot L}{k\cdot(k-1)}$$

Then

$$K\cdot r(L) \geq 1 + k\tau(2)\cdot q(2,L)$$

$$K\cdot r(L) \geq 1 + k\frac{R}{2k}\cdot\frac{2\cdot L}{k\cdot(k-1)}$$

$$K\cdot r(L) \geq 1 + \frac{R\cdot L}{k\cdot(k-1)} \geq \frac{L}{L-R/6}$$

More generally, for $L \geq R$,

$$K\cdot r(L) \geq 1 + k\cdot\sum_{d=k/2L}^{k/L}\tau(d)\cdot q(d,L)$$

Then, using Proposition 7 and Proposition 10

$$\tau(d) = \frac{R}{dk}$$

$$q(d,L) = \frac{d(d-1)\cdot L\cdot\prod_{j=0}^{d-3} k-(L+1)-j}{\prod_{j=0}^{d-1} k-j}$$

$$k\cdot\sum_{d=k/2L}^{k/L}\tau(d)\cdot q(d,L) =$$

$$= k\cdot\sum_{d=k/2L}^{k/L}\frac{R}{d\cdot k}\cdot\frac{d(d-1)\cdot L\cdot\prod_{j=0}^{d-3} k-(L+1)-j}{\prod_{j=0}^{d-1} k-j}$$

$$= \sum_{d=k/2L}^{k/L} R\cdot\frac{(d-1)\cdot L\cdot\prod_{j=0}^{d-3} k-(L+1)-j}{\prod_{j=0}^{d-1} k-j}$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1) \cdot \prod_{j=0}^{d-3} k-(L+1)-j}{\prod_{j=0}^{d-1} k-j}$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \frac{\cdot \prod_{j=0}^{d-3} k-(L+1)-j}{\prod_{j=0}^{d-3} k-j-2}$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \frac{\cdot \prod_{j=0}^{d-3} k-(L+1)-j}{\prod_{j=0}^{d-3} k-(j+2)}$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \cdot \prod_{j=0}^{d-3} \left(\frac{k-(L+1)-j}{k-(j+2)} \right)$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \cdot \prod_{j=0}^{d-3} \left(\frac{k-(L+1)-j+(k-(j+2))-(k-(j+2))}{k-(j+2)} \right)$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \cdot \prod_{j=0}^{d-3} \left(\frac{(k-(j+2))+k-(L+1)-j-(k-(j+2))}{k-(j+2)} \right)$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \cdot \prod_{j=0}^{d-3} \left(\frac{k-(j+2)}{k-(j+2)} + \frac{k-(L+1)-j-(k-(j+2))}{k-(j+2)} \right)$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \cdot \prod_{j=0}^{d-3} \left(\frac{k-(j+2)}{k-(j+2)} + \frac{k-L-1-j-k+j+2}{k-(j+2)} \right)$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \cdot \prod_{j=0}^{d-3} \left(\frac{k-(j+2)}{k-(j+2)} + \frac{-L-1+2}{k-(j+2)} \right)$$

$$= \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \cdot \prod_{j=0}^{d-3} \left(\frac{k-(j+2)}{k-(j+2)} + \frac{-L+1}{k-(j+2)} \right)$$

$$k \cdot \sum_{d=k/2L}^{k/L} \tau(d) \cdot q(d,L) = \sum_{d=k/2L}^{k/L} \frac{RL(d-1)}{k(k-1)} \cdot \prod_{j=0}^{d-3} \left(1 - \frac{L-1}{k-j-2} \right)$$

For all $v = k/2L, \ldots, k/L$,

$$\prod_{j=0}^{d-3}\left(1-\frac{L-1}{k-j-2}\right) \geq \left(1-\frac{L}{k(1-\frac{1}{L}-\frac{2}{k})}\right)^{\frac{k}{L}-3} \approx 1/\mathrm{e}$$

Thus,

$$k \cdot \sum_{d=k/2L}^{k/L} \tau(d) \cdot q(d,L) \gtrapprox \frac{R}{8eL}$$

Putting this together yields

$$K \cdot r(L) \gtrapprox 1 + \frac{R}{8\mathrm{e}L} \geq \frac{L}{L-\theta R}$$

for $\theta = \frac{1}{16\mathrm{e}}$. ■

Proposition 15 (robust release at end probability) *Using only the contribution of* $\tau(k/R)$

$$K \cdot \sum_{L=r}^{2R} r(L) \geq \gamma \cdot R \cdot \ln(R/\delta)$$

for a suitable constant $\gamma > 0$.

Proof. Fix L between $2R$ and R. It is not hard to show that

$$K \cdot \frac{\tau(k/R)}{\beta} \cdot q(k/R, L) \geq \gamma \cdot \ln(R/\delta)$$

for an appropriate constant $\gamma > 0$. ■

Chapter 4

Simulation

The computer simulation program used to generate the following graphs was developed by Filler & Fridrich (2010) using MatLab. The graphs plotted in figure 4.1 and 4.2, display the results of the values for $c = 0.2, c = 0.4, c = 0.6$, and $c = 0.8$ for all four plots $\delta = 0.05$ and $k = 8000$ input symbols. I executed the simulation program more than once to determine if the program produce consistence results. The output plots displayed in figure 4.1 and figure 4.2 are the results of varying the parameter c and the value of the other variables are constant within this simulation. The graph demonstrates that by increasing the magnitude from $c = 0.2$ to $c = 0.8$ improves the performance of the channel. The improved performance can be seen in figure 4.1 by holding $P_b \approx 0.1$, for $c = 0.8$, $e \approx 0.085$,and for $c = 0.6$ $e \approx 0.165$,and for $c = 0.4$, $e \approx 0.21$, and for $c = 0.2$,$e \approx 0.34$.

The plots of figure 4.3 and figure 4.4 display the results of varying the value of δ and holding the other variable constant. The values of the constant variables are $c = 0.2$ and $k = 8000$ input symbols and the values for δ are $0.00005, 0.0005, 005$, and 0.05. These figures demonstrate that as the value of delta increases the performance of the channel decreases.

The performance of LT Codes become more efficient by reducing the bit erasure probability as the number of input symbols is increased. The variables $c = 0.2$ and $\delta = 0.005$ are held constant while k is increased from $2,000$ to $10,000$. This is demonstrated in figure 4.5, figure 4.6, figure 4.7, figure 4.8, figure 4.9, figure 4.10, figure 4.11, figure 4.12,figure 4.13, and figure 4.14.

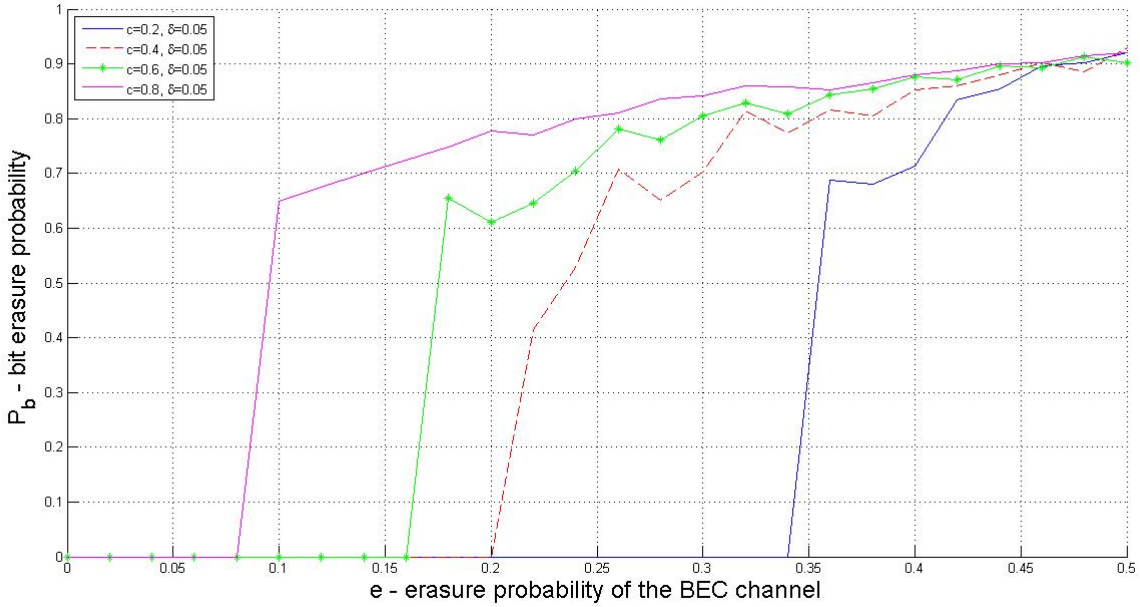

Figure 4.1. Varying the value of c first execution

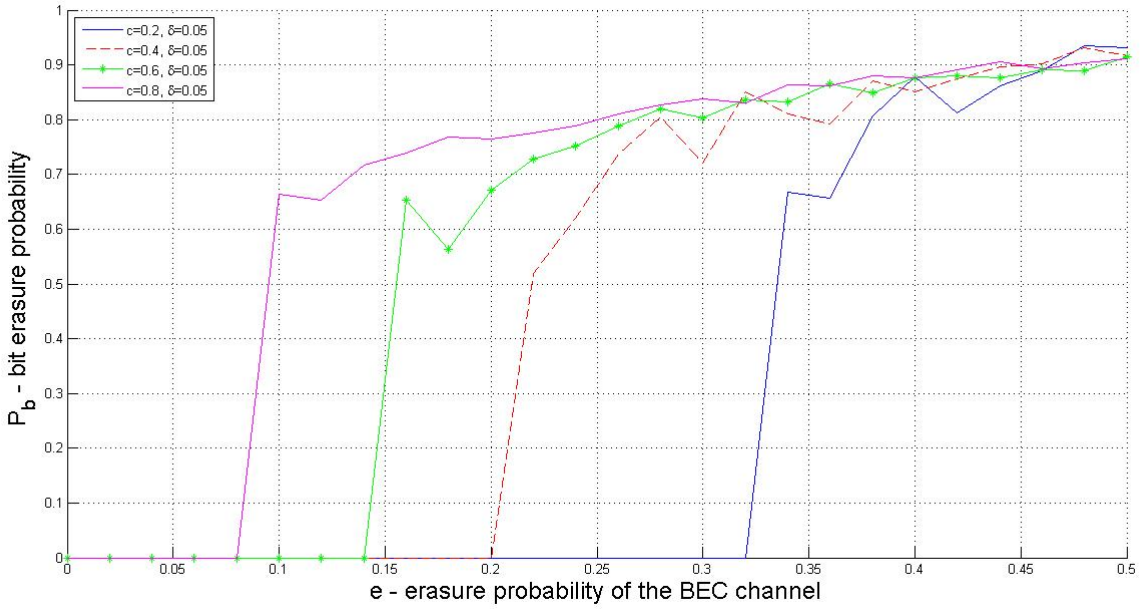

Figure 4.2. Vary the value of c second execution

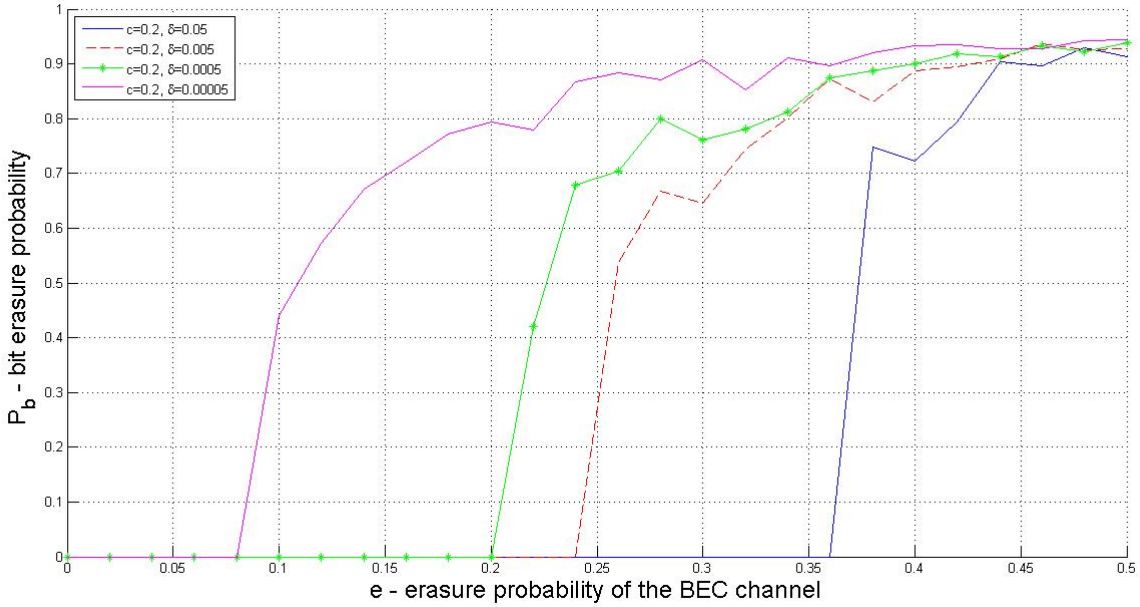

Figure 4.3. Varying the value of δ first execution

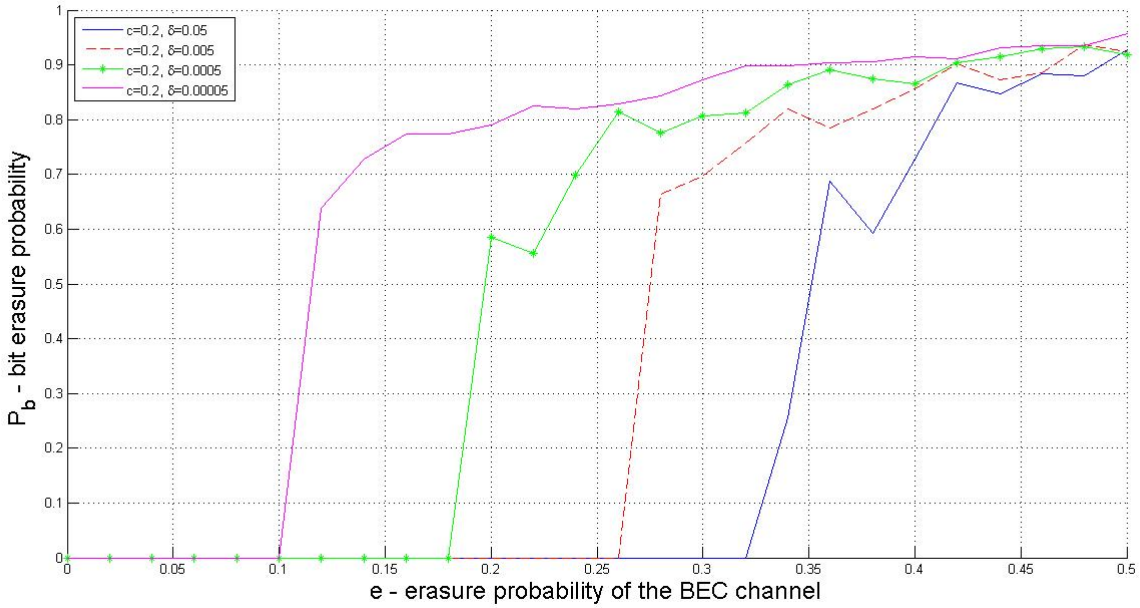

Figure 4.4. Varying the value of δ second execution

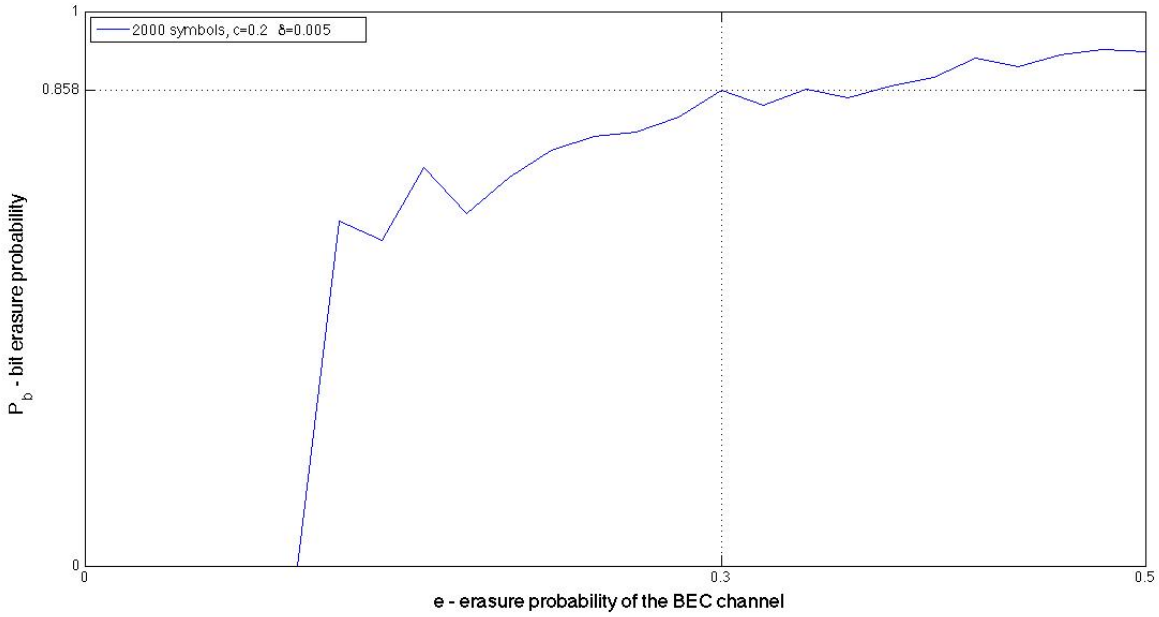

Figure 4.5. 2,000 input symbols, first execution

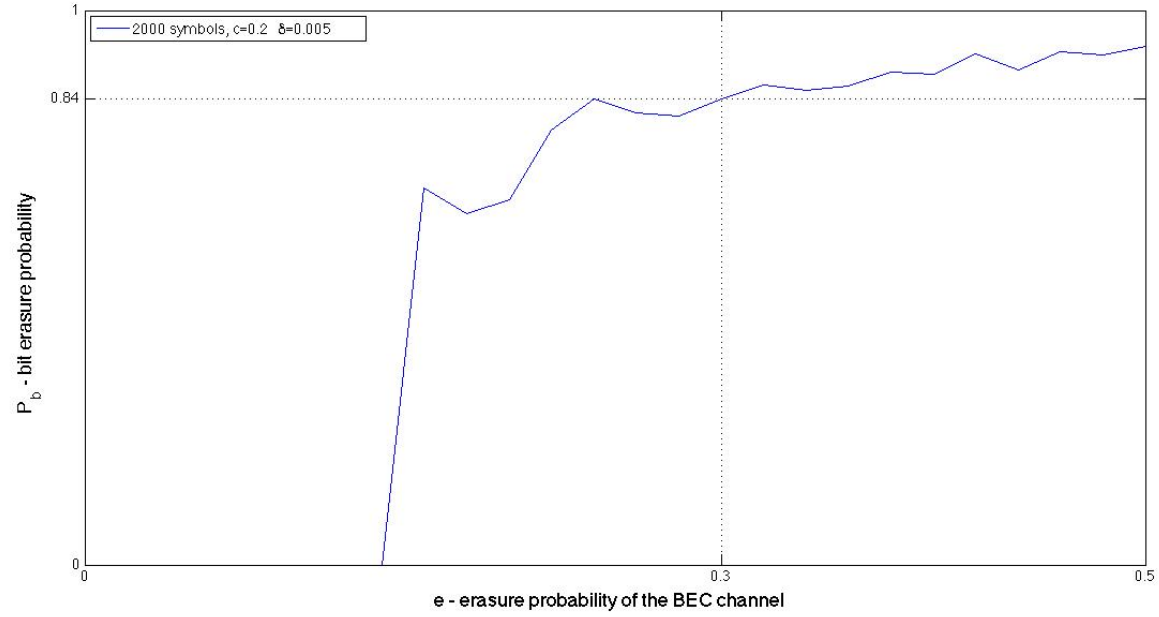

Figure 4.6. 2,000 input symbols, second execution

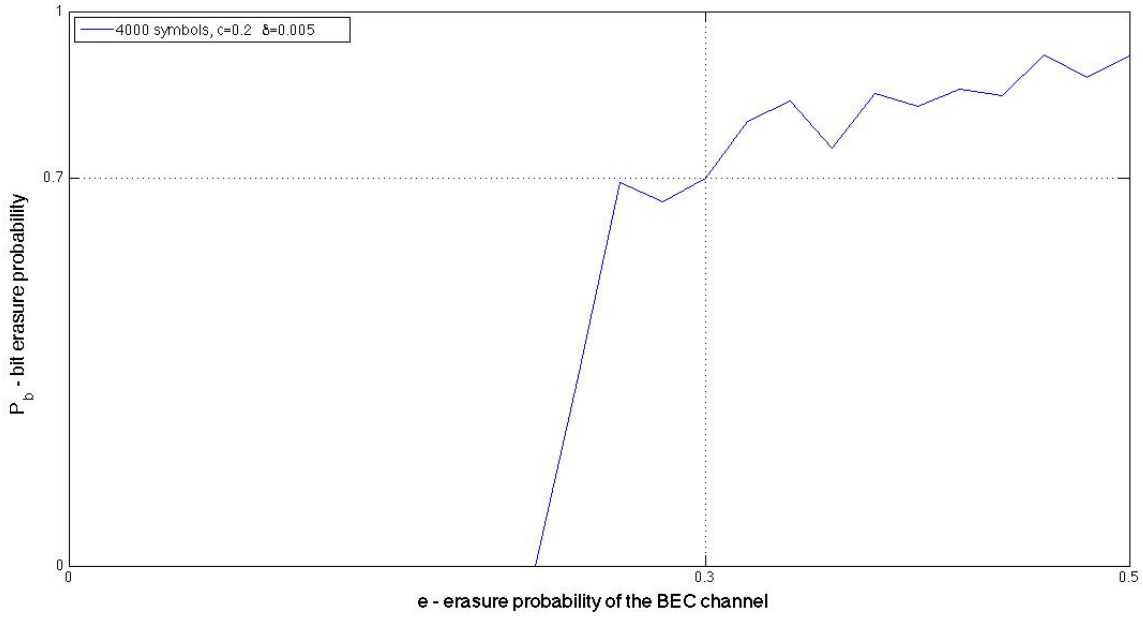

Figure 4.7. 4,000 input symbols, first execution

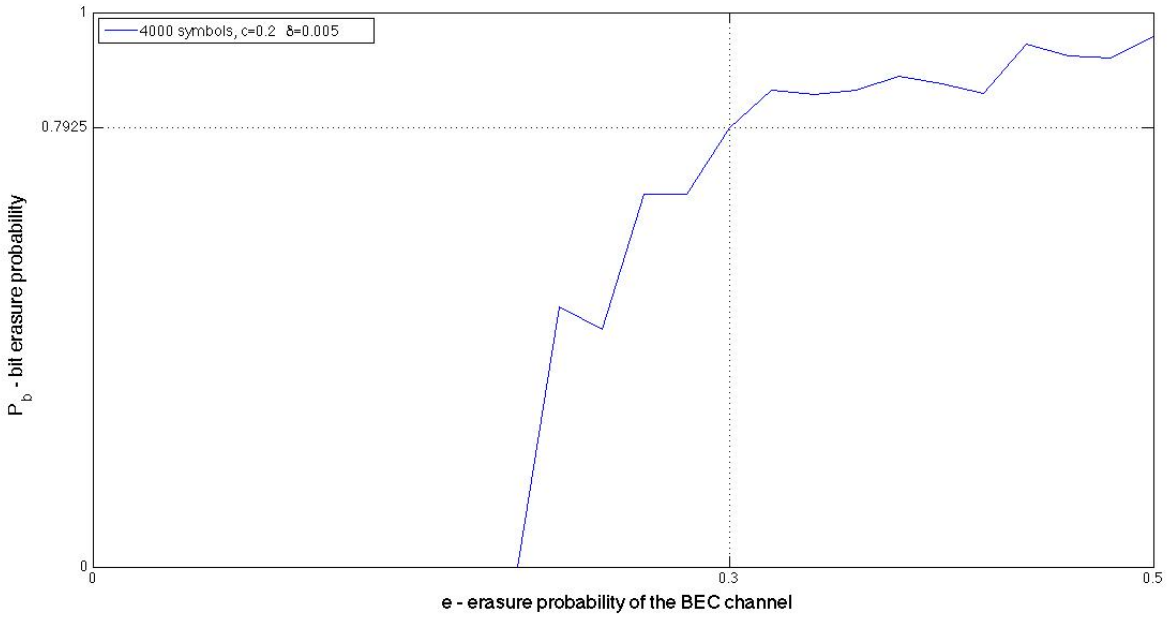

Figure 4.8. 4,000 input symbols, second execution

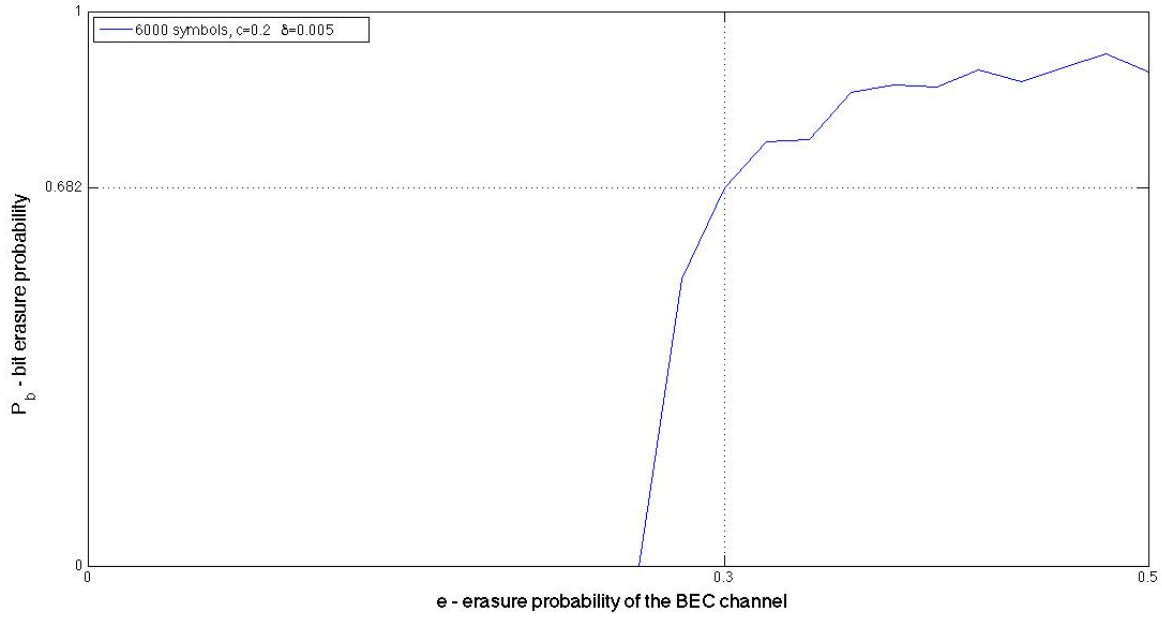

Figure 4.9. 6,000 input symbols, first execution

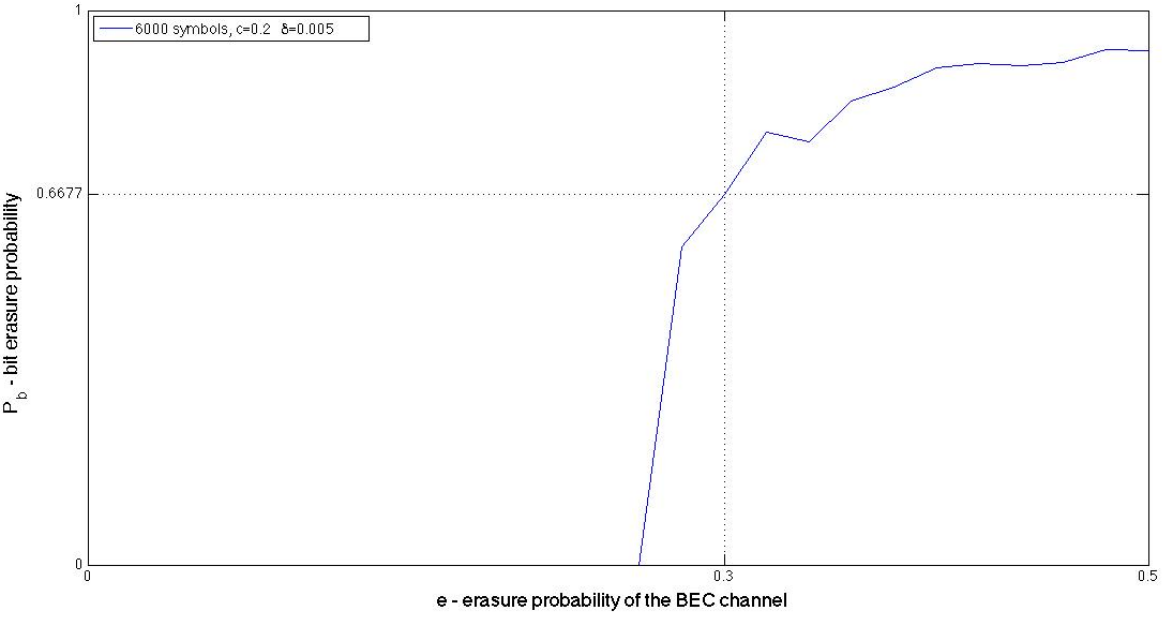

Figure 4.10. 6,000 input symbols, first execution

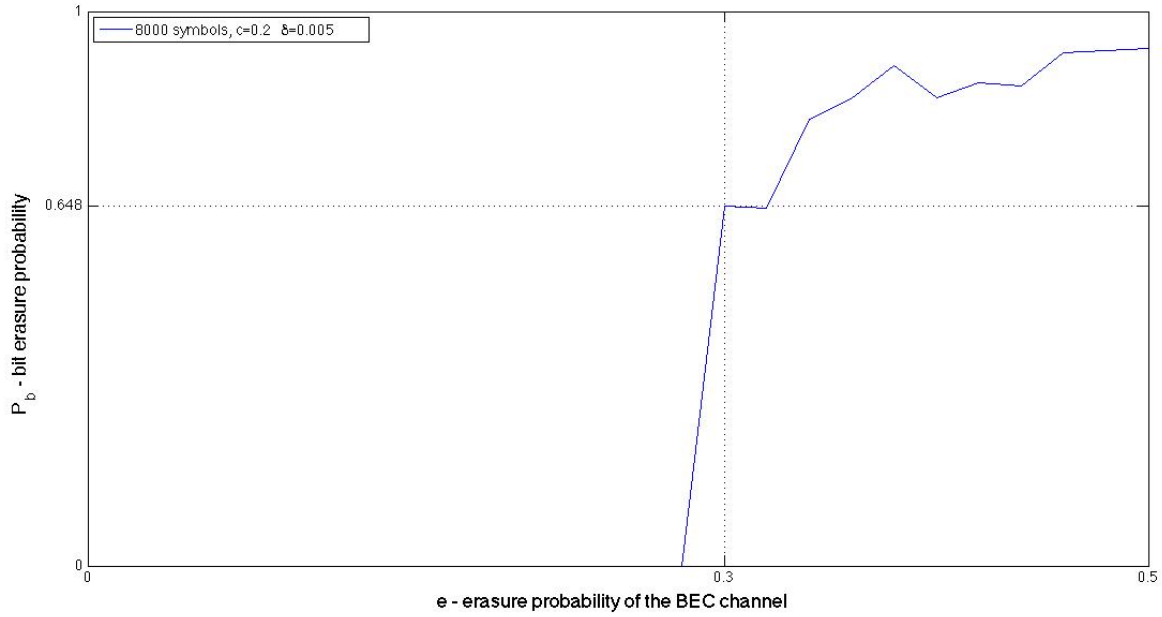

Figure 4.11. 8,000 input symbols, first execution

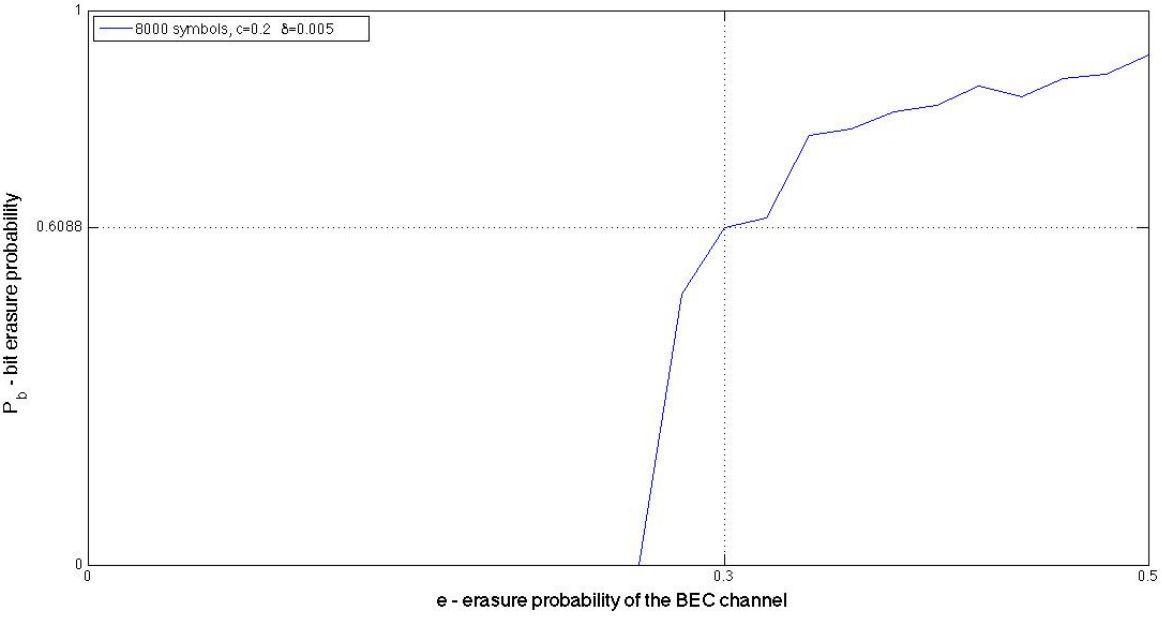

Figure 4.12. 8,000 input symbols, second execution

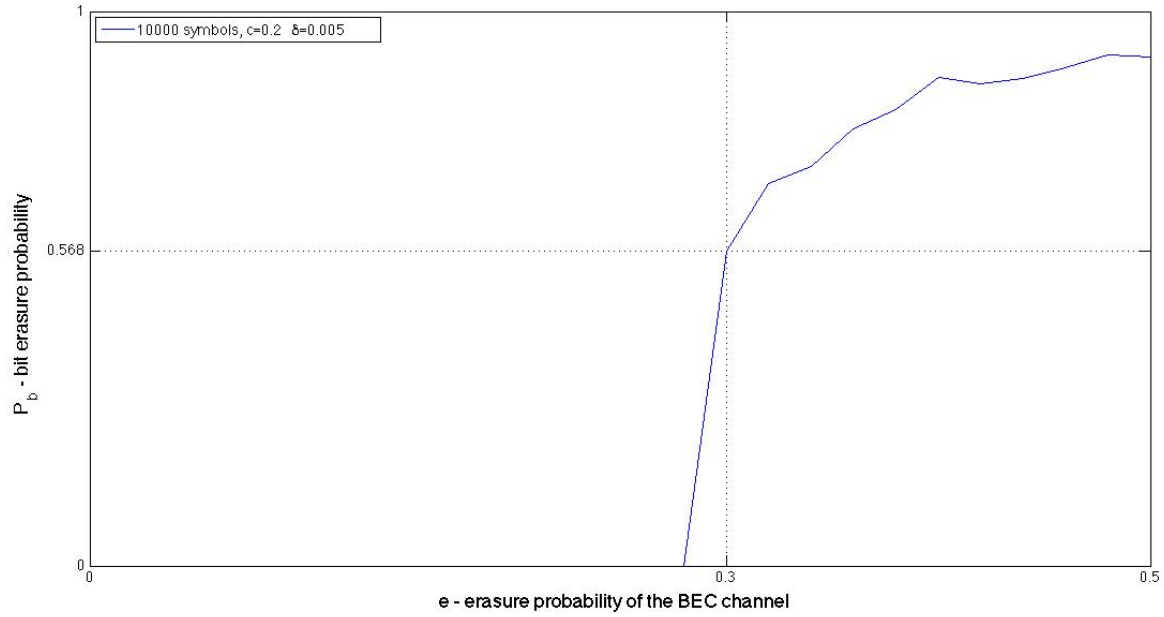

Figure 4.13. 10,000 input symbols, first execution

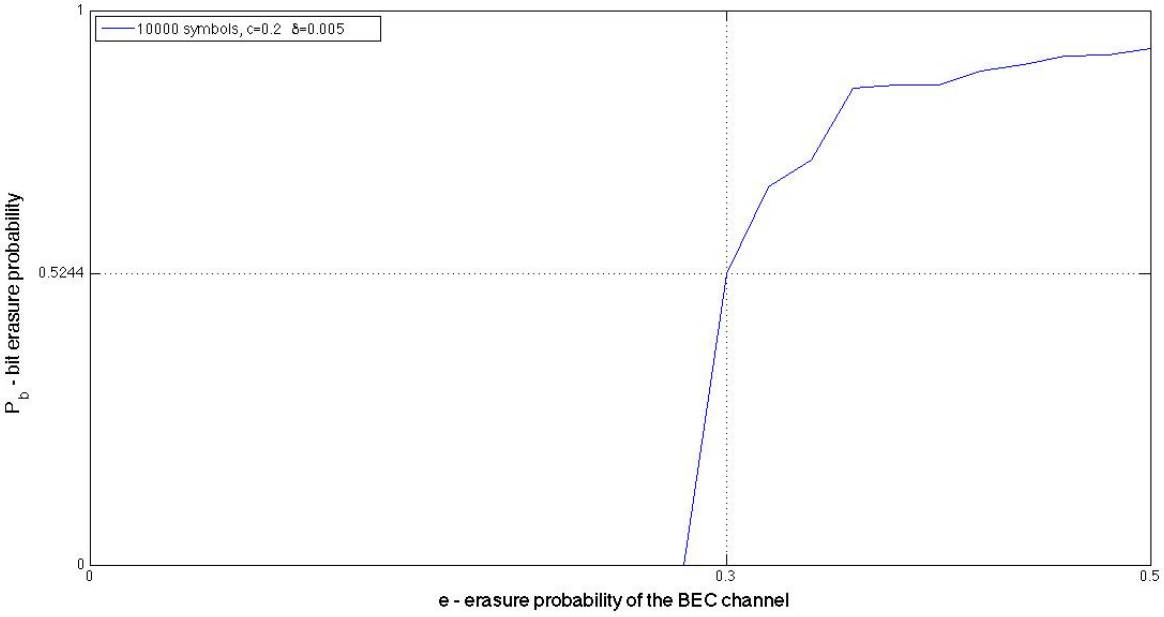

Figure 4.14. 10,000 input symbols, second execution

Chapter 5

Conclusion

The LT Code equations analytical results are very similar to the randomly generated computer simulations results. The insertion of the mathematical techniques used to develop the equations may provide an insight in to their development.

The goal is that this thesis helps to provide an improved understanding of the mathematical foundation and the performance of LT Codes.

Bibliography

Bibliography

Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; & C, S. (2001) "Introduction to Algorithms." Technical report.

E. Schooler, J. G. (1997) "Using multicast FEC to solve the midnight madness problem." *Microsoft Research Technical Report MS-TR-97-25*, Technical note.

Filler, T. & Fridrich, J. (2010) *EECE 580B Modern Coding Theory.* Binghamton University, State University of New York, rev a edition.

J. Nonnenmacher, E. B. (1996) "Reliable Multicast:Where to Use Forward Error Correction." Proceedings of *IFIP 5th Intl Workshop on Protocols for High Speed Networks*, Technical note.

J.W. Byers, M. Luby, M. M. (1999) "Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speed Up Downloads." Proceedings of *IEEE INFOCOM 99, New York, NY*, Technical Note 275-283.

Lathi, B. (1998) *Modern Digital and Analog Communication System.* Oxford University Press, 3 edition.

Luby, M. (2002) "LT Codes." In *Proceedings of the* 43rd Symposium on Foundations of Computer Science, page 271, Washington, D.C. IEEE Computer Society.

M. Adler, Y. Bartal, J. B. M. L. D. R. (1997) "Modular Analysis of Network Transmission Protocols Proceedings." *Fifth Israeli Symposium on Theory of Computing and Systems*, Technical note.

MacKay, D. (2003) *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press, fourth edition.

Ross, S. (2010) *Introduction to Probability Models.* Academic Press, tenth edition.

Thomas Stockhammer, Amin Shokrollahi, M. W. M. L. T. G. (2009) *Application Layer Forward Error Correction for Mobile Multimedia Broadcasting. Digital Fountain Incorporated a Qualcomm Company*, rev a edition.

List of Appendices

Appendix:A

Binary Erasure Channel

The binary erasure channel (BEC) is displayed in figure A.1 is from Lathi (1998). In the

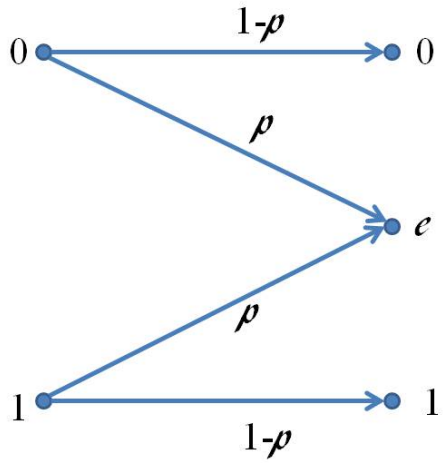

Figure A.1. Binary Erasure Channel

BEC e is the probability of a bit erasure to indicate the fact that nothing is known about the bit that was deleted. The capacity of the BEC is

$$C_{BEC} = 1 - p$$

, figure A.2 plots e vs. C_{BEC}. The conditional probabilities of the channel are:

$P\{\tilde{y} = 0|\tilde{x} = 0\} = 1 - p$

$P\{\tilde{y} = e|\tilde{x} = 0\} = p$

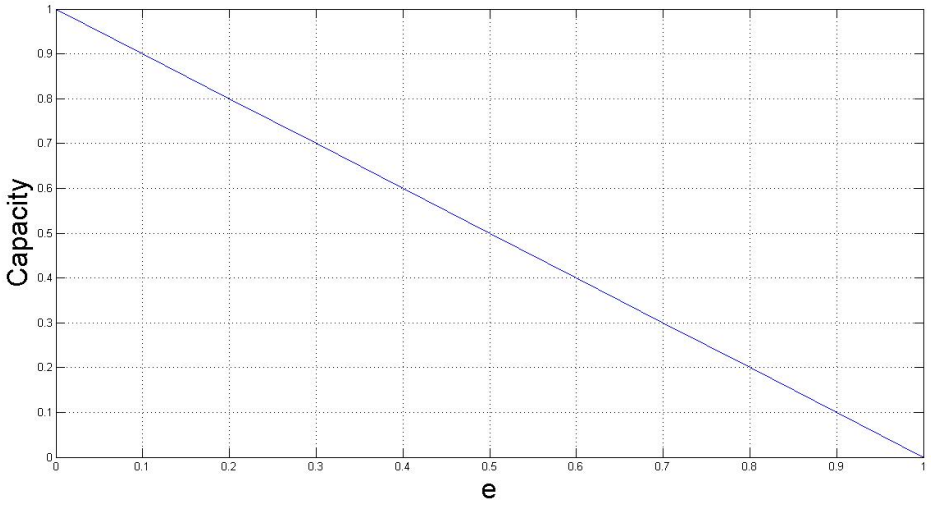

Figure A.2. Capacity

$P\{\tilde{y} = 0|\tilde{x} = 1\} = 0$

$P\{\tilde{y} = 1|\tilde{x} = 0\} = 0$

$P\{\tilde{y} = e|\tilde{x} = 1\} = p$

$P\{\tilde{y} = 1|\tilde{x} = 1\} = 1 - p$

Appendix:B

Balls and Bins

The classical process of randomly tossing balls into bins is taken from Cormen *et al.* (2001). Let's assume that the balls are identical and are numbered $1, 2, \cdots, b$ and there are b empty bins available. The tossing of one ball is independent of the tossing of any of the other $b-1$ balls. The probability that a ball lands in a particular bin is the same for all the b bins and has probability $1/b$. This is a Bernoulli experiment where a success is when a ball lands in the chosen bin.

The objective is to determine how many balls must be tossed until every bin contains at least one ball. Define a hit as the event that a ball lands in an empty bin. There must be at least one ball in each of the b bins so there is a need to determine the number of n tosses needed to obtain b hits.

The hits are used to partition the tosses into n stages. The ith stage contains all the tosses after the $(i-1)$st hit. The first toss is guaranteed to hit one of the empty bins so the first stage is the first toss. During the ith stage each time a ball is tossed, there are $i-1$ bins that contain balls and $b-i+1$ empty bins. Each time a ball is tossed during the ith stage, the probability of a success is $(b-i+1)/b$.

Let n_i denote the number of tosses in the ith stage. Therefore, the number of tosses required to obtain b hits is $n = \sum_{i=1}^{b} n_i$. The random variable n_i has a geometric distribution with probability of success $(b-i+1)/b$. The expectation of the geometric random variable with a success of p is from Ross (2010):

$$E[n_i] = \sum_{n=1}^{\infty} np(1-p)^{n-1}$$

$$\text{Let } q = 1 - p$$

$$E[n_i] = p\sum_{n=1}^{\infty} nq^{n-1}$$

$$E[n_i] = p\sum_{n=1}^{\infty} \frac{d}{dq}(q^n)$$

$$E[n_i] = p\frac{d}{dq}\left(\sum_{n=1}^{\infty} q^n\right)$$

$$E[n_i] = p\frac{d}{dq}\left(q\sum_{n=1}^{\infty} q^{n-1}\right)$$

$$E[n_i] = p\frac{d}{dq}\left(\frac{q}{1-q}\right)$$

$$E[n_i] = p\left(\frac{\frac{d}{dq}(q)(1-q) - q\frac{d}{dq}(1-q)}{(1-q)^2}\right)$$

$$E[n_i] = p\left(\frac{(1-q) - q(-1)}{(1-q)^2}\right)$$

$$E[n_i] = p\left(\frac{1-q+q}{(1-q)^2}\right)$$

$$E[n_i] = p\frac{1}{(1-q)^2}$$

$$E[n_i] = \frac{p}{(1-(1-p))^2}$$

$$E[n_i] = \frac{p}{(1-1+p))^2}$$

$$E[n_i] = \frac{p}{p^2}$$

$$E[n_i] = \frac{1}{p}$$

$$E[n_i] = \frac{1}{(b-i+1)/b} = \frac{b}{b-i+1}$$

Then,

$$E[n] = E\left[\sum_{i=1}^{b} n_i\right]$$

$$E[n] = \sum_{i=1}^{b} E[n_i]$$

$$E[n] = \sum_{i=1}^{b} \frac{b}{b-i+1}$$

Example let $b = 5$, then

$$\sum_{i=1}^{5} \frac{5}{5-i+1} = 5\left(\frac{1}{5} + \frac{1}{4} + \frac{1}{3} + \frac{1}{2} + \frac{1}{1}\right)$$

$$E[n] = b\sum_{i=1}^{b} \frac{1}{i}$$

The function $f(i) = \frac{1}{i}$ is a is a monotonically decreasing function that can be approximated by:

$$\int_{m}^{n+1} f(x)dx \leq \sum_{k=m}^{n} f(k) \leq \int_{m-1}^{n} f(x)dx$$

The lower bound:

$$\sum_{k=1}^{n} \frac{1}{k} \geq \int_{1}^{n+1} \frac{dx}{x}$$

$$= \ln(n+1)$$

The upper bound:

$$\sum_{k=2}^{n} \frac{1}{k} \leq \int_{1}^{n} \frac{dx}{x}$$

$$= \ln n$$

Then it takes $E[n] = b\ln(b)$ balls for each bin to contain at least one ball.

VITA

Objective:

Doctorate of Electrical Engineering

Education:

Bachelor of Science, Electrical Engineering, University of Alabama at Birmingham, Birmingham, AL, 1993

Bachelor of Science, Mining Engineering, West Virginia University, Morgantown, WV, 1984

Skills and Qualifications:

- C++ and MatLab
- Microsoft Office