Rateless Coding for Single-Source Networks with Common Information

by

Arghavan Modiri

A thesis submitted to the
Department of Electrical and Computer Engineering
in conformity with the requirements for
the degree of Master of Applied Science

Queen's University
Kingston, Ontario, Canada
June 2015

Copyright © Arghavan Modiri, 2015

Abstract

In this thesis, we consider the communication problem of a single source simultaneously transmitting to multiple receivers whose sets of requested messages overlap. For decades, one of the challenges in this broadcast setting has been decreasing the number of transmissions from the source to the terminals without increasing the system complexity. The multicast and broadcast problems with 'common' information have been mostly studied 'existentially': information-theoretical bounds on rates and capacities have been discussed in a number of previous works.

In this work, we take the contrasted 'constructive' viewpoint and attempt to design practical transmission protocols with low encoding and decoding complexities. Our approach is based on rateless fountain coding equipped with efficient belief propagation (BP) decoders. While previous network coding solutions require high-complexity Gaussian elimination decoding for optimality, fountain codes allow for much better performance-complexity trade-offs with BP decoders due to their sparse decoding Tanner graphs.

We provide insights and solutions for the 2-terminal setting as an example that show the extraordinary power and flexibility of fountain codes to address the conflicting challenges in the design of efficient transmission protocols. Our ideas can be extended to larger number of receivers and though we have focused on Luby-Transform

(LT) codes, other fountain coding distributions can be equally useful.

The proposed coding methods in this thesis can be applied to many practical systems such as wireless sensor networks (WSN) that have limited power, memory, and processing capabilities.

Acknowledgments

This thesis would not have been finished successfully without the help and support of many people around me.

I would like to begin by thanking my supervisor, Prof *Shahram Yousefi*, but I cannot find words to express my sincere gratitude toward him who patiently helped me to take the first steps of my research journey confidently. Master's program is the beginning of a long journey and Dr.Yousefi's valuable advice, support, and encouragement enlightened me on the enigmas of this starting point, along with allowing me the room for creativity.

I also wish to thank all of my colleagues and dear friends for always being there when I needed help. During this time, I was surrounded by their love and companionship that makes this period of time unforgettable for me.

I would like to thank the Electrical and Computer Department and School of Graduate Studies at Queen's University for their financial and academic supports during this time.

Last but not least, I would like to thank my parents, *Lida* and *Hassan*, and my beloved sister, *Solmaz*, to whom I will always be indebted. Their constant love, support, and belief in my abilities have always encouraged me to go forward in the face of every opposition. My today's achievements are all caused by their love and support.

Contents

List of Tables

List of Figures

Glossary

ACK Acknowledgement.

AWGN Additive White Gaussian Noise.

BEC Binary Erasure Channel.

BP Belief Propagation.

GE Gaussian Elimination.

ISD Ideal Soliton Distribution.

LDPC Low-Density Parity-Check.

LT Luby Transform.

MDS Maximum Distance Separable.

MFC Mixed Fountain Codes.

MFC-CS Mixed Fountain Codes-Central Sampling.

MFC-SUS Mixed Fountain Codes-Semi-Uniform Sampling.

ML Maximum Likelihood.

RS Reed-Solomon.

RSD Robust Soliton Distribution.

TDMA Time-Division Multiple Access.

List of Symbols

Symbol	Description
β	Normalized factor of RSD
γ_j^I	Probability of choosing degree j in mode E^I
ε	Channel erasure rate
η	Fraction of redundancy in code design
$\mu(i)$	Robust-Soliton degree distribution
$\rho(i)$	Ideal-Soliton degree distribution
$\tau(i)$	The ripple correction term for RSD
$\Omega^I(x)$	Degree distribution used at mode E^I
ℓ	length of each chunk of information
d_i	The picked degree distribution at time slot i
$d_{\max}^I$	Maximum possible degree to be chosen in mode E^I
E_i	The event of purely sampling just from X_i
E_{ij}	The event of sampling from X_i and X_j
E^i	The event of sampling from X_i with or without bits of X_0
e	Erased packets at the receiver
G_{ij}	Randomly picked binary value at time slot i corresponding with bit number j

k	Number of information bits
N_1	Average value of n_1
N_2	Average value of n_2
n_1	Number of transmission until ACK_1 is sent
n_2	Number of transmission until ACK_2 is sent
n	Number of encoded bits in fixed rate erasure codes
n_i	Number of chosen bits from X_0 in mode E^I at time slot i
$n_{X_0}^{T_1}$	the average number of information bits from X_0 used during a transmission session and observed at T_1
n_{X_1}	The average number of information bits from X_1 used during a transmission session and observed at T_1
M	Size of data information before packetizing
p_i	Occurrence probability for event E_i
p_{ij}	Occurrence probability for event E_{ij}
p^i	Occurrence probability for event E^i
p_{T_1}	Probability of producing packets allocated to T_1
p_{T_2}	Probability of producing packets allocated to T_2
q_0	Probability of $n_i = 0$ given d_i when $1 < d_i \leqslant m_1$
R	Expectation of the initial ripple size
$\hat{s}_i$	Recovered information bit at the receiver at time slot i
t_i	Received packet at the destination at time slot i
T_1	Terminal 1
T_2	Terminal 2
u_j	The bit number j of the source information bits

X_0	Common information requested by all terminals
X_1	Information source just needed by T_1
X_2	Information source just needed by T_2
x_{ij}	The bit number j of the information source of X_i

Chapter 1

Introduction

We are living in an era when the demands for trading information are at their highest and still increasing at a rapid pace. Many different devices connect to each other as nodes of a network to exchange their information. The largest network of this era, the Internet, emerged by connecting a number of computer science laboratories in early 1960s. Since then, it has continuously grown to the point that today it allows the devices around the world to connect to each other to exchange large amounts of information in a matter of seconds. Online gaming, video sharing, and social networking are some of the few examples of the Internet usage. Despite the constant growing rate of the Internet usage, its infrastructure has not dramatically developed to efficiently adapt with today's applications. Hence, the current networks should be reconfigured to keep up with the growing load of information.

On the other hand, the information that is passing through a channel from one node to the other can get corrupted or lost due to some phenomena like channel fading, noise or interference. They may also get lost because of the huge amount of traffic on the network edges. These phenomena are not limited to the Internet, but any communication channel to some extent suffers from data loss or errors. Therefore,

in all communication systems, the designers look for reliable data exchange in an acceptable range of cost and complexity at the nodes without too much sacrificing the resources such as bandwidth.

One way of increasing the communication reliability is through improving the physical characteristics of the channel which results in a huge cost! Changing the actual channel to a reliable environment that is free of noise or loss is too expensive that makes this approach impractical. Another way of improving the performance with reasonable cost is to accept the channel as an imperfect environment and use channel coding to combat its deficiency. There is a famous quote from Richard Blahut that emphasizes the importance of using coding for communication, "to build a communication channel as good as we can is a waste of money; use coding instead!" Channel coding can turn noisy channels into reliable environments at the costs of increasing complexity and reducing bandwidth efficiency.

The key idea of channel coding to increase the reliability of communication is to add 'redundancy'. Each packet is protected by sending a number of extra packets as its back-up, so if an error occurs or a packet gets lost, there is a possibility that the affected packet can be recovered at the destination by perfectly receiving back-up packets. The amount and the method of adding redundancy should be compatible with the probability of packets getting corrupted. When the probability of erroneous or lost packets in a channel is high, more redundant packets are needed for protection and in more reliable channels, less redundancy is required.

Any effective solution has a downside. As mentioned earlier, the solution proposed by channel coding to combat channel imperfections comes at the costs of incremental complexity at the source and destination nodes in addition to more bandwidth usage.

This trade-off should be considered in designing codes for different channels and applications.

The main focus of this thesis is designing and developing codes on the binary channels with erasures, channels that are affected by the phenomenon of packet loss. These channels are commonly used as models in communication systems like the Internet.

1.1 Erasure Channel Model

Erasure channels are one of the simplest channels to work with, so they are commonly used in coding and information theory. Data transmission over the Internet and wireless fading channels are modeled by erasure channels [1]. In general, They are mostly used as models for networks in which data are chopped into small pieces. In this thesis, the smallest piece of information transmitted in channel is called packet. For simplicity and without loss of generality, we represent packets with bits.

Packets sent over erasure channels are either received correctly or not received at all due to buffer overflows, excessive network delays, check-sum fails, and etc. In other words, the channel occasionally erases one of the transmitted packets. The simplest form of an erasure channel is the binary erasure channel (BEC). The inputs of such channels are binary and the produced outputs are ternary symbols (Figure 1.1) [2]. By probability of $(1-\varepsilon)$, each packet is received as transmitted and by probability of ε $(0<\varepsilon<1)$ it gets corrupted and received as an ambiguity or not received at all: we show this by symbol e (Figure 1.1).

The maximum possible rate for reliable transmission (capacity) over BEC is $(1-\varepsilon)$ bits/channel use. Intuitively speaking, since ε percent of the packets get lost, the

decoder cannot receive more than $(1-\varepsilon)$ percent of them. It is proven in [3] that the capacities of BEC with and without feedback are the same and equal to $(1-\varepsilon)$. This is a very important characteristic of BEC because mostly in practice, the source does not have reliable feedback on channel qualities: there is no penalty for this from the viewpoint of capacity [3].

In this thesis, we assume the channels are memoryless binary erasure channels without feedback. In memoryless channels, the output of the channel only depends on the channel input at the same instant. In other words, the inputs of the previous or future intervals do not affect the output of the channel at the present. These channels are easier to analyze.

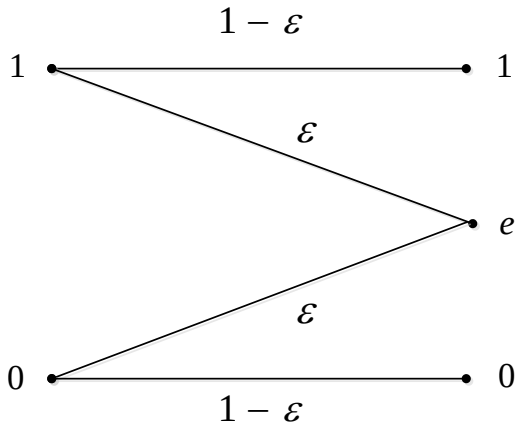

Figure 1.1: Transition diagram of binary erasure channel.

1.2 Coding for Erasures

Erasure codes are used to transfer data reliably over the BEC. When k information bits are intended to go through a BEC to reach the destination, some of them get lost.

To make sure that destination can fully recover all of the transmitted bits, additional bits are required to be sent to provide redundancy. Thus, n bits must be transmitted instead of k bits ($n > k$). The ratio of k/n is called the 'code rate' and adjusted according to the channel parameters. In the following, we explain how the value of code rate is selected.

1.2.1 Traditional Erasure Codes

Traditionally, erasure codes are fixed-rate codes designed to protect data against getting lost [4]. The data of size M that is needed to be transmitted over an erasure channel is first divided into k pieces, each of size M/k, then these pieces are encoded to a longer message of size n [5]. At the receivers, the original message should be decoded from a subset of n transmitted packets.

If the original data can be fully recovered by using any k pieces, the code presents a notion of 'optimality' in the design of its redundancy. The famous and much celebrated class of Reed-Solomon (RS) codes are such 'any-k-out-of-n' optimal codes. Any k received bits guarantee the recovery of the original k information bits [6]. RS codes are non-binary Maximum Distance Separable (MDS) codes satisfying the Singleton bound with equality and hence providing optimal trade-off between rate and error control capability. RS codes are impractical for many cases due to high encoding and decoding complexities. During the past three decades, there has been much activity in the area of 'MDS-type' code design. The objection has been to relax the 'k-out-of-n' condition to gain reductions in complexity.

For instance, in non-optimal erasure codes, reconstructing data might be possible using $(1+\eta)k$ bits. If the value of η is small enough that only slightly more than k bits

are required to reconstruct the data, the codes are 'near optimal'. Tornado Codes are such a class of erasure codes [7]. They are a sub-class of LDPC codes described over sparse graphs. Although Tornado codes need more packets than RS codes to obtain enough information, they are faster (their encoding and decoding time is linear in n) [4].

On the other hand, both Tornado and Reed-Solomon codes have some drawbacks. The encoding and decoding time of Reed-Solomon is quadratic in message length making them impractical for settings with large k and n [4].

The other problem that applies to all traditional block codes goes back to their fixed rate nature that makes them harder ro adapt to changes in channel characteristics or demands. For example, in wireless networks when channel state information is not available or there is too much variability in channel conditions, fixing the code rate can decrease the bandwidth efficiency or the decodability. In other words, choosing large amount of n for BEC channels with low erasure results in sending far too many redundant packets and wasting the bandwidth, while choosing low values of n for the actual erasure rates encountered renders the received signal undecodable.

1.2.2 Fountain Codes

Fountain codes are the latest generation of erasure codes that address the drawbacks of the traditional codes. Using such codes, the server can generously produce an infinite number of packets and send them to terminals to the point that all the terminals decode the entire message. No prior information about the channel condition is needed. The output packets of the server are produced on the fly. On the other side of the channel, terminals keep receiving uncorrupted packets as much as they need

until the whole message is successfully decoded. This rateless method is similar to the procedure of filling a cup by a water fountain [7]. The cup should be filled no matter which drops of water are actually being used. As the packets are generated randomly without any order, new terminals can join or leave the networks at any time, however fountain coding is more effective when a massive number of receivers connect to a server roughly at the same time. The rateless nature of fountain codes makes them adaptable to the channels with unknown conditions.

The most famous classes of fountain codes are Luby-Transform (LT) codes and Raptor (Rapid Tornado) codes. LT codes were introduced by Luby in 2002 as the first universal erasure codes [4]. LT codes are universal since they are asymptotically optimal for any erasure channel. They are optimal in the sense of required redundancy and achieving channel capacity. As the size of the message grows, they become more efficient. The other important class of fountain codes are Raptor codes introduced by Shokrollahi as an improvement over LT codes. Raptor codes improve the per-information-bit encoding and decoding complexity of LT codes from logarithmic in k to constant. They are formed by concatenating a traditional high-rate pre-coder with an LT code [8].

Fountain codes are designed for single-hop BEC networks; they do not hold their optimality and universality features in other system models. Fountain codes are designed for single-hop networks where a source is broadcasting data directly to one or many receivers. It is also assumed that all terminals are requesting the exact same data from the server. For multi-hop settings where network coding at the intermediate nodes is required to increase bandwidth efficiency, there are a number of challenges. Liau et al. have provided the best known transmission protocols based on LT codes

for subset of such architectures [9] [10].

Most previous models are not able to describe the myriad of different applications in today's technologies where massive numbers of customers connect to the servers of an on-demand media provider to download different contents such as one or more videos. The sets of demanded messages by different terminals are not necessarily the same although there might be some overlaps among them.

To study and develop proper encoding schemes for different network structures, first we need to review some network fundamentals. In the next section, different methods of data transmission are discussed.

1.3 Network Transmission Modes

There are different methods of data transmission over the network: unicast, broadcast, and multicast. Depending on the application, one or more methods may be used in each network. In this section, we briefly describe each method and discuss some of their features.

In unicast method, each requested data piece is transmitted from a source to a destination node. In other words, each transmitted packet is destined for only one node, so if multiple destinations request the same data from the source node, separate messages are transmitted individually to each of them.

On the other hand, in broadcasting and multicasting, the target of messages transmitted from source node(s) are not just one particular destination. Although these two methods have some similarities, they should not be confused with each other. In broadcasting, the packets are sent to all recipients in the network or sub-network, while in multicasting, the target of each packet is a subset of the destinations (the

selective destinations). In computer networking, the term multicasting is also used for situation in which information is distributed within several sources and need to dissipate within several destinations.

Casting Types	**Association**
Unicast	One to One
Multicast	One/Many to Many
Broadcast	One to Many

Table 1.1: Different Casting Types

According to the above definition, examples of multicasting applications include video conferencing or video sharing when the targets are selected among all recipients. Multicasting protocol in comparison to broadcasting protocol has additional overhead, so broadcasting is preferred in some scenarios that the majority of nodes in the network form the multicasting set. This preference is an efficient choice under the condition that selecting and processing additional information at the destinations does not increase the cost beyond the tolerance [11].

In spite of the bandwidth efficiency brought to the network by multicasting and broadcasting methods, the Internet is mostly structured based on unicasting because the routers do not support the other methods. Due to the increasing demands for data over the Internet, researchers and the Internet providers have been evaluating the benefits of multicasting option more seriously [12] [11].

1.4 Motivation and Contribution

The fountain coding approach is designed specifically for broadcasting networks in which a single source is serving a number of receivers with the same demands. However, in many applications, different devices have different demands.

Wireless sensor network (WSN) is one of the examples of a scenario where the data collected in some nodes of the sensor network are needed at some other nodes, for instance in the 'data collector nodes' or 'junction hubs'. The information needed by sensors of different areas can be different, while there might be some overlap among them.

Dense wireless sensor networks are composed of a large number of unsophisticated and limited sensor units that are in rapid deployment for many applications such as military surveillance, home health care or assisted living and environmental science among many others. In most cases, the deployed sensors are limited in terms of power (e.g., battery life), memory, and processing capability. As such, it is paramount to invent transmission and reception technologies that conserve power, have low complexity, and require an acceptable level of memory and computation power [13]. Therefore, we propose a fountain coding approach toward solving such problem. Although, there are a large number of other applications that suit the proposed solutions.

In this work, we focus on the transmission protocol design for a source node S with multiple measurements (information sources) needing to multicast information to some other nodes with different but overlapping demands. Information-theoretically, this corresponds to some very interesting and yet quite challenging problems. For example, consider the rather simple formation of three nodes. A transmitter wishes

to send independent information to two receivers and common information to both receivers. The capacity region of this channel for the Gaussian case is implicitly characterized by El Gamal [14]. For most other cases, the rate and capacity regions are unknown. Our interest here is rather on the communication aspects of this problem. We intend to design practical and efficient transmission schemes equipped with fast decoders for the above-mentioned setting. We assume that the parallel links are all BEC with different erasure rates.

In this thesis, we propose a novel transmission protocol aiming to minimize the number of source transmissions required to satisfy both terminals. We make the following key contributions:

- Analyzing the performance of finite-length fountain codes over BEC by calculating the precise amount of recovery rate after receiving an arbitrary number of packets. The analysis is aimed for a belief propagation decoder.

- Proposing a new transmission protocol for a multicast setting composed of a single source and multiple receivers with arbitrary sets of overlapping demands. The proposed algorithm is aimed to decrease the number of transmissions for any arbitrary fountain degree distribution.

The ideas developed can be extended to other network formations and channel models.

The following publication is related to the material presented in this thesis:

- A. Modiri and S. Yousefi, "Single-Source Two-Terminal Multicast Networks with Overlapping Demands Over the Binary Erasure Channel," in Proc. of *Canadian Workshop on Information Theory (CWIT 2015)*, St. John's, NL, Canada, July 2015.

1.5 Organization of Thesis

The rest of this thesis is organized as follows. Chapter 2 discusses the fundamentals of fountain coding along with some related previous work. In Chapter 3, our proposed transmission protocol based on mixed fountain coding is presented. By considering a multicasting setting, the relation between sampling pattern from different messages and the mixing probabilities is studied. Then, the proposed algorithm is compared with related benchmarks through Monte-Carlo simulations. Finally in Chapter 4, we make the concluding remarks and recommend some areas for for future activity related to our work.

Chapter 2

Fundamentals and Prior Art

The capacity of a broadcast channel has been investigated for decades and been formulated for some specific cases such as the Gaussian or deterministic broadcast channels; however the problem still remains unsolved for the general case. Marton found the rate region of general broadcast channels in 1979 [15]. In his paper, he showed that the achieved rate region is the same as capacity region if the broadcast channel has one deterministic element, but the equality is not proved for other cases of broadcast channel [15][16].

According to Shannon theory, in order to guarantee 'reliable' communication, the rate of a broadcasting setting with common information is upper-bounded by the minimum channel capacity of the receivers, so the terminals with better channel have to wait more and downgrade to the performance of poor receiver [17]. Time-division multiple access (TDMA) can provide better solution in such cases. Cover shows that it is possible to do even better than TDMA with coding [18]. In his seminal work, he used superposition coding to achieve a higher rate for binary additive white gaussian noise (AWGN) and binary symmetric channel cases.

Later in 1990s, fountain codes emerged claiming to reach the Shannon capacity in

memoryless binary erasure channel for every receiver. In fact, the receivers are able to decode the demanded message by receiving packets with a number just slightly larger than the message length. Fountain codes are particularly beneficial when access to reliable and timely channel state information at the transmitter side is not possible. Fountain codes are *rateless*: the code rate is not known *a priori* and the number of encoding symbols that can be generated from the data is potentially limitless. Furthermore, encoding symbols can be generated on the fly, as few or as many as needed. The number of transmissions required and hence the overall rate of the system is determined according to the actual states of the underlying channels.

The capacity of the fountain codes is not discussed in detail in the original works [4][8]. Later, it is investigated by Shamai et al. for any arbitrary channels: with or without memory [17]. As described in the paper, the most important difference between fountain codes and all the works done previously is the concept of rate. The rate of fountain codes should be defined to emphasize its receiver-based feature, while in Shannon Theory, the rate is stated from the transmitter perspective. The rate of fountain codes can be defined as the ratio of actual gained information bits to the received symbols from the channel, then the fountain capacity is defined according to the rate definition as the maximum rate in which terminals can reliably receive information [17]. The Shannon and fountain capacities are equal for stationary memoryless channels, but in the other forms, fountain capacity is upper-bounded by Shannon capacity.

In this chapter, we continue with an introduction to fountain coding.

2.1 Random Linear Fountain Codes

Random linear fountain code is the simplest class of fountain codes. The packets, as the smallest elements for data transmission over the networks, are formed by parsing information bits or symbols[1] into chunks of length ℓ where ℓ is in the range of a few hundreds to 65,535 bytes. Transmitted packets are random linear combinations of k information packets parsed. k information packets considered together form a 'generation'. Assume a given generation of data information at the source shown by $u_1,\ u_2,\ \ldots u_k$. Then, the transmitted packet t_i at time slot i can simply be described by the linear equation:

$$t_i = \sum_{j=1}^{k} G_{ij} u_j \ . \tag{2.1}$$

When working with the binary fiels, G_{ij} is a binary value picked randomly by the encoder at time slot i.

The number of received packets at the decoder must be equal or more than k so that the terminal can successfully decode all the k packets of the message generation. After receiving enough packets, the decoder can recover the information via :

$$\hat{s}_i = \sum_{j=1}^{k} G_{ij}^{-1} t_j \ , \tag{2.2}$$

if the $[G_{ij}]$ is invertible.

The biggest drawback of this system is the decoding complexity in (2.2) which is generally cubic for GE decoder [19]. In the next section, we introduce LT codes as a practical realization of linear fountain codes that can be adjusted by a very simple decoder.

[1]Octal or bytes are the most common form of symbol used in this context.

2.2 LT Codes

Luby Transform (LT) codes are the first practical realization of the digital fountain paradigm where by transferring the adaptivity from the transmitter to the receiver of a telecom system, better channel frequency and time usage is achieved while avoiding outage and minimizing error rates [4]. These are all achieved at impressively low encoding and decoding complexities along with much flexibility and scalability for the network. LT codes based on the robust soliton distribution (RSD) are universal and optimal for the BEC: their proposed decoders can recover the data from nearly minimal number of encoding symbols possible irrespective of the channel erasure probabilities. They also perform well over noisy channels and many other architectures [9].

2.2.1 Encoding

The encoder of a binary LT code[2] takes the following procedure to generate a new packet at each iteration:

- picking a degree d_i based on a degree distribution $\Omega(k)$,
- choosing d_i distinct symbols uniformly at random from all the available data symbols, $u_1,\ u_2,\ \ldots u_k$,
- setting the transmitted packet t_i as the bitwise summation of the d_i selected symbols.

Although the encoding steps look very simple, there are two important concepts embedded in them; the choices of degree distribution and the sampling protocol for the

[2]Nonbinary fountain codes can be similarly defined using larger fields.

symbols. A careful design should decrease the complexity of the system dramatically while it increases the system performance via good degree distributions and suitable source sampling. We will talk about the degree distribution introduced by Luby after reviewing some decoding concepts.

2.2.2 Decoding

The key feature of LT codes is that one can use sub-optimal belief propagation (BP) decoders instead of the optimal maximum-likelihood (ML) Gaussian Elimination decoders to extract the k source bits from n (which is just slightly larger than k) coded bits. This option reduces the complexity of the decoding dramatically compared to GE.

Belief propagation decoding is formulated to work on graphs. The bipartite graph is used as a graphical model to describe the structure of the code. The vertices of a bipartite graph are grouped in two disjoint sets such that no two nodes within each group are adjacent. We refer to these groups as left and right hand side nodes. The nodes on the right and left hand sides of the graph correspond to information symbols and encoded symbols, respectively. In the BEC, the packets are either received at the decoder perfectly or not received at all. This simplicity in the structure of channel transitions for packets results in a simple decoding algorithm provided below:

- Find an output packet t_i with degree equal to one. If no such packet is available, the decoder fails.

- Recover the neighbor node (u_j) of the t_i by setting its value as equal to t_i and remove the connecting edge between them.

- XOR the value of u_j with all the connected neighbors and remove all the connected edges to u_j.
- Repeat the above procedure until all the k symbols are successfully decoded.

If XORing one symbol with another or copying one symbol value to another symbol are each counted as one symbol operation, then the BP needs on average $O(k.\ln(k/\delta))$ operations to recover all k symbols with probability of $(1-\delta)$ while GE requires cubic operations (in k).

2.2.3 Degree Distribution Design

The first degree distributions of practical fountain codes introduced by Luby in 1988 are the Ideal-Soliton and robust-Soliton degree distributions. These distributions are designed to be compatible with BP decoding. Here, we briefly describe the innovative idea shaping the design of the degree distribution.

The decoding process starts via degree-one packets. By processing each of them, potentially some new degree-one nodes are produced. The set of all non-processed degree-one nodes on the right is called the *ripple*. At each decoding iteration, one of the elements in the ripple is processed as described in section 2.2.2 and removed from the ripple while new degree-one nodes are added to the ripple when values are added from left to right and edges are removed. The decoding procedure continues as long as the ripple size is greater than zero unless all the information symbols are decoded.

If the ripple set vanishes before recovering all the demanded symbols, a failure in decoding procedure is occurred. This shows the critical role of the ripple size in decoding performance. Therefore, the degree distribution should be designed such that the ripple set survives to the end of decoding without being too large. The

number of low-degree packets should be kept as small as possible to avoid generating redundant packets which will hurt the error and erasure performance. How the code design responds to this trade-off has a large impact on its performance.

Ideal-Soliton degree distribution (ISD) attempts to keep the ripple size as low as one at each iteration which seems ideal. At each iteration, a degree-one node is processed and leaves the set, while a new degree-one node is added to the ripple. ISD is given by [4]:

$$\rho(i) = \begin{cases} \frac{1}{k} & , \quad \text{for } i = 1 \\ \frac{1}{i(i-1)} & , \quad \text{for } i = 2, ..., k. \end{cases} \tag{2.3}$$

Despite the theoretical statement, the ISD described above performs poorly in practice. As the expected ripple size is one, it is very likely to vanish soon and cause the decoding to terminate unsuccessfully. Luby improved the performance by increasing the expected size of ripple from 1 to $\ln(k/\delta)\sqrt{k}$. In his paper, he proved that the probability of deviation of ripple size from its expected value by more than $\ln(k/\delta)\sqrt{k}$ is less than δ. The Robust-Soliton distribution is defined (RSD) as [4]:

$$\mu(i) = (\rho(i) + \tau(i)) / \beta \quad \text{for } i = 1, ..., k, \tag{2.4}$$

where β is the normalization factor $\beta = \sum_{i=1}^{k} \rho(i) + \tau(i)$. $\tau(i)$ is defined as follows:

$$\tau(i) = \begin{cases} R/ik & , \quad \text{for } i = 1, ..., k/R - 1, \\ R\ln(R/\delta)/k & , \quad \text{for } i = k/R, \\ 0 & , \quad \text{for all } i = k/R + 1, ..., k. \end{cases} \tag{2.5}$$

where $R = c\ln(k/\delta)\sqrt{k}$ for $c > 0$.

The spike added at $i = k/R$ is intended to make sure all the symbols are covered by output packets.

2.2.4 Source Sampling Effect

In the encoding procedure described in Section 2.2.1, we mentioned that d_i distinct symbols should be selected from the data symbols. In traditional LT codes, the symbols are selected uniformly and without prioritizing. This sampling pattern causes the degree distribution of the left hand sided nodes of the bipartite graph to be Poisson [4]. Uniform sampling combined with RSD works asymptotically to be universally optimal for broadcasting information to multiple receivers with identical sets of demands.

However, in other applications and in finite regime, the system can benefit by non-uniform or memory-based sampling. As an example, for most finite-length cases, by sampling from the source symbols with highest left degree when $d_i = 1$, the system performance improves due to enhancing the graph connectivity [20]. Similarly, in applications where some parts of the message have priority to the others, prioritized LT codes are preferred to uniform LT codes as they provide better performance-complexity trade-offs. By this choice, high priority messages either have more chance of getting selected for more protection or they are more likely to be combined in lower degree packets for faster recovering [21] [22].

The same rule applies to the system model used in this thesis. The network model of this thesis is composed of a single source serving multiple terminals with diverse sets of demands. The system model and the approaches toward solving the problem

is explained in detail in the next chapter. We just emphasize here that the traditional RSD and uniformly sampling are not the best choice for this problem.

Chapter 3

Coding for Single-Source Networks

A wireless sensor network (WSN) is made of numerous connected autonomous sensors that are distributed in designated areas to monitor and report the status of some variables. The information available at the sensors are transmitted through the network to the central nodes or demanding areas.

The wireless sensor nodes in the network are mostly small microelectronic devices with limited sources of power. In many applications, the sensors are installed in remote places where replacing the power supplements is almost impossible or costs too much. This limitation enforces the designers to use the optimized methods of data processing with minimum energy consumption in the sensors, so that the sensors lifetimes increase.

To address this issue, fountain coding approach seems to be a good candidate. Achieving capacity universally in BEC channels, no requirement to have prior information on the channel erasure rates at the encoder, and having encoder/decoder with low complexity are some of the interesting features of fountain codes. These unique features motivate us to take a fountain coding approach towards solving the WSN and similar content delivery problems.

However, traditional fountain codes are not properly designed for problems under our consideration. Fountain codes are tailored for broadcast settings where a single server is transmitting information to multiple destinations or terminals. In the afore mentioned system model, all terminals are interested in the same data available at the source. On the other hand, the problems encountered in WSNs are more generalized than this. Each sensor in a WSNs, depending on its application, is interested in some of the measurements available at the other nodes, so there is no need that all of the sensors request the same data. As a model for such networks, we consider a system composed of a server with multiple measurements (information sources) that is required to multicast information to some other nodes with different but overlapping demands.

We intend to design practical and efficient transmission schemes equipped with fast decoders by taking the fountain codes approach. More precisely, our main goal is to minimize the total number of transmissions from the server to all terminals, so that the source can switch off as soon as possible and save the battery. As mentioned in the previous chapter, we can adjust the traditional fountain codes to our new system model by modifying the sampling protocol or degree distribution. In this chapter, we propose a transmission protocol design that uses a different method of sampling. The degree distribution is also modified to match with the new protocol.

It is to be noted that the optimal designed of a transmission protocol in terms of both the degree distribution and sampling is a prohibitively complex task, one that is not done in any other system for the finite regime.

The single transmitter considered here broadcasts a single packet, denoted by a single bit for brevity, to all terminals. As a stepping stone, we consider only two

terminals in this work. Each sent packet can be a function of three information sources at any given time: X_0 is needed at both terminals while X_1 and X_2 are only needed at terminals T_1 and T_2, respectively.

3.1 System Model

The network selected in this thesis is composed of two terminals connecting to a source via multicast channels with independent erasures. All information is in the hands of the source and there is no communication between the terminals. Both terminals can receive all the encoded packets transmitted by the server and collect their desired ones (if not erased). Each link transports one packet/bit per use. At each terminal, an online BP decoder recovers the messages continually and sends back an ACK (acknowledgment) signal when all the demanded messages are recovered. Assume the information available at the server is the collection of information sources $X_1,\ X_2,$ and X_3. In each session of transmission, it is intended to transmit a block of m_i symbols of message X_i. The information at each session can be written as $(x_{i1}, x_{i2}, \ldots, x_{im_i}) \in \{0,1\}^{m_i}$. The demands of terminal one and two known as *Want* set can be represented as $W_1 = \{X_0, X_1\}$ and $W_2 = \{X_0, X_2\}$, respectively. The links to T_1 and T_2 are BEC links with independent erasures of ε_1 and ε_2, respectively.

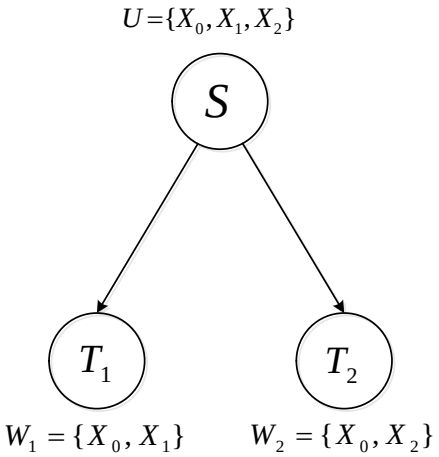

Figure 3.1: System model of a single-source two-terminal broadcast system

The packets transmitted from the server towards the terminals can be associated with one of the following three distinct events:

- E_0 : When the packets are purely made up of X_0 bits. This event has the occurrence probability of p_0.
- E^I : When the packets are made up of X_1 with or without X_0 bits. This event has the occurrence probability of p^I.
- E^{II} : When the packets are made up of X_2 with or without X_0 bits. This event has the occurrence probability of p^{II}.

At any time slot, one of the above events occurs, so it is clear that

$$p_0 + p^I + p^{II} = 1. \tag{3.1}$$

The scenarios that include mixing bits of both X_1 and X_2 are intentionally avoided

here due to their ineffectiveness. According to the system model, each terminal drops the packets that contain any bit from messages out of its Want set. As the packets including both X_1 and X_2 are not considered by any terminals, the server does not perform such mixing.

It is to be noted that mixing or coding is proven to be indispensable if optimality of the system in terms of min-cut max-flow is targeted [23]. This fundamental result of the original network coding (NC) paper dictates that if all server-receiver pairs are to enjoy their maximum rates, one must mix whenever possible.

3.2 Mixed Fountain Coding Solution

In what follows, we propose a transmission protocol to be used at each server. The solution is essentially a mixed fountain and network coding architecture. Ideally, the fountain distributions and mixing and sampling methods are to be designed and optimized simultaneously. This is a seemingly impossible feat. We turn our attention to a step-wise approach. We start by fixing the fountain distribution and attempt to find simple and efficient mixing and sampling techniques.

As described earlier, each transmitted packet is generated by XORing some of the information bits at the source. The packets are generated on the fly until all terminals are satisfied. At the time that each terminal decodes successfully the entire messages of its Want set, it sends back an ACK to the server to terminate its connection. In the system model described in this thesis, the server should keep transmitting the packets until two acknowledgment notices from T_1 and T_2 are received. Let ACK_1 and ACK_2 be the acknowledgments received from T_1 and T_2 after transmission of n_1 and n_2 number of packets, respectively. Then, the server can switch off after N

transmissions where

$$N = \max(N_1, N_2). \tag{3.2}$$

Our goal in this thesis is to design a transmission protocol that minimizes the total number of transmissions N and correspondingly the power consumption of the source. It is trivial that the minimum amount of N is produced when $N_1 = N_2$. Thus, we set

$$N = N_1 = N_2. \tag{3.3}$$

In fact, N_1 and N_2 are expectation values of two random variables:

$$N_1 = E[n_1] = \text{average number of transmission until receiving at least } m_0 + m_1 \text{ packets at } T_1. \tag{3.4}$$

$$N_2 = E[n_2] = \text{average number of transmission until receiving at least } m_0 + m_2 \text{ packets at } T_2. \tag{3.5}$$

The above random variables have the negative binomial distribution, so their averages are calculated as follows:

$$N_1 = \frac{m_0 + m_1}{(1-\varepsilon_1)(p_0 + p^I)} = \frac{m_0 + m_1}{(1-\varepsilon_1)(1 - p^{II})}, \tag{3.6}$$

$$N_2 = \frac{m_0 + m_2}{(1-\varepsilon_2)(p_0 + p^{II})} = \frac{m_0 + m_2}{(1-\varepsilon_2)(1 - p^{I})}. \tag{3.7}$$

In the above equations, it is optimistically assumed that all the arrived packets at the receivers are informative and no redundant packet is generated. Although this assumption seems unrealistic, it does not affect our analysis dramatically because the equations are only purposed to calculate the probabilities of p_0, p^I, and p^{II}.

Using equation (3.3), a direct relation between p^I and p^{II} appears. Minimizing one leads to minimizing the other and consequently minimizing the number of transmissions (N). If we assume that $\varepsilon_1 = \varepsilon_2 = \varepsilon$, then

$$N = N_1 = N_2 = \frac{m_0 + m_1}{(1 - p^{II})} = \frac{m_0 + m_2}{(1 - p^I)} \tag{3.8}$$

leading to:

$$p^I = \frac{m_0 + m_2}{m_0 + m_1} p^{II} + \frac{m_1 - m_2}{m_0 + m_1}. \tag{3.9}$$

The values assigned to N_1 and N_2 ensure that both terminals have enough packets to potentially recover all the messages in their Want sets, however, a couple of other conditions are required to guarantee the availability of enough information for the recovery of each individual element in the Want sets:

$$N.p^{II} \geqslant m_2, \tag{3.10}$$

$$N.p^I \geqslant m_1. \tag{3.11}$$

The three above equations lead to

$$p^{II} \geqslant \frac{m_2}{m_0 + m_1 + m_2}. \tag{3.12}$$

By choosing the minimum value of p^{II} and accordingly p^{I}, the number of transmissions is minimized[1] , so

$$(p_0, p^I, p^{II}) = (\frac{m_0}{m_0+m_1+m_2}, \frac{m_1}{m_0+m_1+m_2}, \frac{m_2}{m_0+m_1+m_2}) \tag{3.13}$$

leading to:

$$N_{\min} = (m_0+m_1+m_2)/(1-\varepsilon). \tag{3.14}$$

3.2.1 Mixing Modes of the Protocol

In the previous section, we found a suitable set of occurrence probabilities for all possible events, E_0, E^I, and E^{II}, however our definitions are not complete. Each of the events E^I and E^{II} break down into two different events: E_1 and E_{01} for E^I and E_2 and E_{02} for E^{II}:

- E_1: When the packets are purely made by sampling X_1. This event has the occurrence probability of p_1.
- E_2: When the packets are purely made by sampling X_2. This event has the occurrence probability of p_2.
- E_{01}: When the packets are formed by sampling both X_1 AND X_0. This event has the occurrence probability of p_{01}.
- E_{02}: When the packets are formed by sampling both X_2 AND X_0. This event has the occurrence probability of p_{02}.

[1] We do not mean a mathematical minimum in true sense. Rather, this is a reduction within the context of the problem with the afore mentioned set of assumptions.

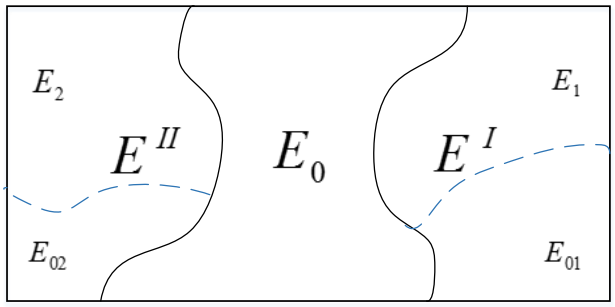

Figure 3.2: The classified outcomes of the source with different messages

The analysis of this section aims to calculate p_1 and p_{01}. The corresponding analysis can also be applied to find p_2 and p_{02}.

The event E^I occurs by probability of p^I. In this mode, the server picks a degree d_i based on the degree distribution

$$\Omega^I(x) = \sum_{j=1}^{d^I_{\max}} \gamma^I_j x^j \quad \text{that} \quad d^I_{\max} = m_0 + m_1. \tag{3.15}$$

and the coefficient γ^I_j is the probability of choosing degree j.

Similar to traditional fountain coding, when a degree d is picked from $\Omega^I(x)$, d distinct bits should be chosen among X_0 and X_1. Here, we propose to deviate from the uniform sampling done previously. We aim to design the sampling pattern such that both messages wanted at T_1 get recovered as fast as possible. In this section, we will present good values of p_1 and p_{01} as functions of the sampling pattern for a given degree distribution.

Let's say that for any degree d_i picked at time slot i of mode E^I, n_i and accordingly $(d_i - n_i)$ bits are chosen among bits of X_0 and X_1, respectively. It should be noted

that it is required to select at least one bit from X_1 for any d_i in mode E^I. Thus, we have:

$$0 \leqslant n_i \leqslant d_i - 1 \quad \text{and} \quad n_i \leqslant m_0, \tag{3.16}$$

$$1 \leqslant d_i - n_i \leqslant d_i \quad \text{and} \quad d_i - n_i \leqslant m_1, \tag{3.17}$$

or briefly, we present

$$\max(0, d_i - m_1) \leqslant n_i \leqslant \min(m_0, d_i - 1). \tag{3.18}$$

By the notation we introduced here, we can rewrite p_1 and p_{01} as:

$$p_1 = \sum_{d_i=1}^{m_1} p(n_i = 0 \mid d_i)\, p(d_i)\, p^I, \tag{3.19}$$

$$p_{01} = \sum_{d_i=1}^{m_0+m_1} p(n_i > 0 \mid d_i)\, p(d_i)\, p^I\,. \tag{3.20}$$

In general for any d_i, we can say that

$$\sum_{n_i=0}^{d_i} p(n_i \mid d_i) = 1 \quad \text{for} \quad 1 \leqslant d_i \leqslant m_0 + m_1, \tag{3.21}$$

where

$$p(n_i = d_i \mid d_i) = 0\ . \tag{3.22}$$

It is clear that the summation in (3.19) cannot go beyond m_1 because it is impossible to select more than m_1 distinct bits from X_1, thus

$$p(n_i = 0 \mid d_i) = 0 \quad \text{for} \quad d_i > m_1. \tag{3.23}$$

The probability of sampling purely from X_1 in this mode, that is $p(n_i = 0 \mid d_i)$ for any d_i, has a large impact on the proper values of p_1 and p_{01} and accordingly on the system performance. Let N be the total number of packets transmitted from the server. On the average, $\left(p_0 + p^I\right).N$ of them that are relevant to T_1. Using the original analogy of a water fountain and a glass to fill, we should design the sampling and mixing process for any degree distribution such that the glasses for X_0 and X_1 get filled of the same rate. Clearly, p_0, p_2, p_{01}, and the distribution of degrees between X_0 and X_1 in the E_{01} mode are jointly controlling the above mentioned rate.

Thus, to improve the efficiency of the transmission protocol, we require the ratio of the average number of received droplets using X_0 and X_1 to be equal to the ratio of m_0 and m_1. This in turn, demands the proportional use of X_0 and X_1 information bits during the encoding process. This 'information rate proportionality' is supported by information-theoretical arguments presented by El Gamal and Marton [14][15]. We set:

$$\frac{n_{X_0}^{T_1}}{m_0} = \frac{n_{X_1}}{m_1} \tag{3.24}$$

where $n_{X_0}^{T_1}$ and n_{X_1} show the average number of information bits from X_0 and X_1 used during a transmission session and observed at T_1. Accordingly, the statement

also applies to T_2 where we demand:

$$\frac{n_{X_0}^{T_2}}{m_0} = \frac{n_{X_2}}{m_1}. \tag{3.25}$$

where similarly $n_{X_0}^{T_2}$ and n_{X_2} denote the average number of information bits from X_0 and X_2 used and observed at T_2.

To estimate $n_{X_0}^{T_1}$, we need to consider all the packets transmitted in E_0 and E^I modes because the outcomes of these two events are relevant to T_1:

$$n_{X_0}^{T_1} = \sum_{d_i=1}^{m_0} p_0 d_i \, \gamma_{d_i}^0(d_i) + \sum_{d_i=1}^{m_0+m_1} n_i \, p(n_i \mid d_i) \, \gamma_{d_i}^I \, p^I \tag{3.26}$$

where $\gamma_{d_i}^0$ is the probability that degree d_i is picked from $\Omega^0(x)$ in mode E_0. Also, n_{X_1} is calculated as:

$$n_{X_1} = \sum_{d_i=1}^{m_0} \sum_{n_i=1}^{d_i-1} (d_i - n_i) p(n_i \mid d_i) \, \gamma_{d_i}^I \, p^I. \tag{3.27}$$

Using the above expressions, Equation (3.24) and simple derivations, the following relation among the system parameters is derived:

$$\sum_{d_i=1}^{m_1} \gamma_{d_i}^I \sum_{n_i=1}^{d_i-1} n_i p(n_i \mid d_i) = \Bigg(p^I \sum_{d_i=1}^{m_0+m_1} d_i \, \gamma_{d_i}^I - \frac{m_1}{m_0} p_0 \sum_{d_i=1}^{m_0} d_i \, \gamma_{d_i}^0 - p^I \frac{m_0+m_1}{m_0} \sum_{d_i=m_1+1}^{m_0+m_1} \gamma_{d_i}^I \sum_{n_i=1}^{d_i-1} n_i p(n_i \mid d_i) \Bigg) \Bigg/ \left(p^I \frac{m_0+m_1}{m_0} \right). \tag{3.28}$$

Our extensive simulations show that (3.19), (3.20), and (3.28) lead to improvement in system performance in all cases studied. In what follows, we investigate the role

of sampling method in the overall number of transmissions required.

3.3 The Transmission Protocol

Equation (3.28) shows the relation among the important variables for the system, but falls short of explicitly identifying values for $p(n_i \mid d_i)$. Therefore, more degrees of freedom are available in terms of sampling the information bits.

For simplicity, let us assume that in mode E^I, the probability of generating packets purely made from X_1 is independent of the chosen degree d_i, so

$$p(n_i = 0 \mid d_i) = \begin{cases} 1 & , \quad \text{for } d_i = 1 \\ q_0 = \text{const.} & , \quad \text{for } 1 < d_i \leqslant m_1 \\ 0 & , \quad \text{for } m_1 < d_i \leqslant m_0 + m_1. \end{cases} \tag{3.29}$$

and accordingly,

$$\sum_{j=1}^{d_i-1} p(n_i = j \mid d_i) = \begin{cases} 0 & , \quad \text{for } d_i = 1 \\ 1 - q_0 = \text{const.} & , \quad \text{for } 1 < d_i \leqslant m_1 \\ 1 & , \quad \text{for } m_1 < d_i \leqslant m_0 + m_1. \end{cases} \tag{3.30}$$

Using Equation (3.19), we can find the relation among q_0, p_1 and p^I as:

$$\sum_{d_i=1}^{m_1} p(n_i = 0|d_i)\gamma_{d_i} p^I = p_1 \rightarrow \gamma_1^I p^I + q_0\, p^I \sum_{d_i=2}^{m_1} \gamma_{d_i} = p_1 \rightarrow \tag{3.31}$$

evaluating to:

$$q_0 = \frac{p_1 - \gamma_1^I p^I}{p^I \sum_{d_i=2}^{m_1} \gamma_{d_i}}. \tag{3.32}$$

The value of $p(n_i = 0 \mid d_i)$ is calculated, and we still need to determine the value of $p(n_i \mid d_i)$ for $n_i > 0$. Here, we choose to take two opposite approaches for information sampling and compare their performances.

The first approach is referred to as the *Semi-Uniform* sampling method. This approach attempts to select d_i distinct packets as uniform as possible among bits of X_0 and X_1 while keeping the constraints satisfied. One of the constraints here is to ensure that at least one bit from X_0 and one bit from X_1 are selected. Moreover, the value of n_i should not go beyond the range specified by Equation (3.18). Thus:

$$p(n_i = j \mid d_i) = \begin{cases} \frac{\binom{m_0}{1}\binom{m_1}{1}\binom{m_0-1}{j-1}\binom{m_1-1}{d_i-j-1}}{\binom{m_0}{1}\binom{m_1}{1}\binom{m_0+m_1-2}{d_i-2}}(1-q_0) & , \text{ for } \begin{cases} 1 < d_i \leqslant m_1 \\ 1 \leqslant j \leqslant \min(m_0, d_i - 1) \end{cases} \\ \\ \frac{\binom{m_1}{1}\binom{m_0}{j}\binom{m_1-1}{d_i-j-1}}{\binom{m_1}{1}\binom{m_0+m_1-1}{d_i-1}} & , \text{ for } \begin{cases} m_1 < d_i \leqslant m_0 + m_1 \\ d_i - m_1 \leqslant j \leqslant \min(m_0, d_i - 1). \end{cases} \end{cases} \tag{3.33}$$

To make sure that both X_0 and X_1 are included in the generated packets when $d_i \leqslant m_1$, first the server should select one bit from X_0 and another bit from X_1, then $(d_i - 2)$ bits are picked uniformly at random from X_1 and X_0 without replacement. This is shown in the first row of Equation (3.33). However, if $m_1 < d_i \leqslant m_0 + m_1$, there is no need to be concerned about X_0 because at least one bit of it is certainly selected.

As such, all parameters of Equation (3.28) are derived. Further, we can evaluate

the value of q_0 as

$$q_0 = \left(\frac{m_1}{m_0} p_0 \sum_{d_i=1}^{m_0} d_i \, \gamma_{d_i}^0 - p^I \sum_{d_i=1}^{m_0+m_1} d_i \, \gamma_{d_i}^I + p^I \frac{m_0+m_1}{m_0} \Big[\sum_{d_i=2}^{m_1} \gamma_{d_i}^I \, A + \sum_{d_i=m_1+1}^{m_0+m_1} \gamma_{d_i}^I \, B \Big] \right) \Big/ \Big(p^I \frac{m_0+m_1}{m_0} \sum_{d_i=2}^{m_1} \gamma_{d_i}^I \, A \Big) \quad (3.34)$$

where,

$$A = \sum_{j=1}^{\min(m_0, d_i-1)} j \frac{\binom{m_0-1}{j-1}\binom{m_1-1}{d_i-j-1}}{\binom{m_0+m_1-2}{d_i-2}}, \quad (3.35)$$

$$B = \sum_{j=d_i-m_1}^{\min(m_0, d_i-1)} j \frac{\binom{m_0}{j}\binom{m_1-1}{d_i-j-1}}{\binom{m_0+m_1-1}{d_i-1}}. \quad (3.36)$$

The encoding algorithm for the semi-uniform approach is refereed to as *mixed fountain coding-semi-uniform sampling* (MFC-SUS) algorithm (Algorithm 1).

An alternative approach towards solving the sampling problem is using our proposed *Central Sampling*. In this method, the server will choose the average value of the specified range in (3.18) for any selected degree d_i. That is,

$$p(n_i = j \mid d_i) = \begin{cases} (1-q_0) & , \quad 1 < d_i \leqslant m_1, \; j = \Big\lfloor \dfrac{1+\min(m_0, d_i-1)}{2} \Big\rfloor \\ 0 & , \quad 1 < d_i \leqslant m_1, \; j \neq \Big\lfloor \dfrac{1+\min(m_0, d_i-1)}{2} \Big\rfloor \\ \\ 1 & , \quad m_1 < d_i \leqslant m_0+m_1, \; j = \Big\lfloor \dfrac{d_i-m_1+\min(m_0, d_i-1)}{2} \Big\rfloor \\ 0 & , \quad m_1 < d_i \leqslant m_0+m_1, \; j \neq \Big\lfloor \dfrac{d_i-m_1+\min(m_0, d_i-1)}{2} \Big\rfloor. \end{cases} \quad (3.37)$$

For each section, all the probabilities of $p(n_i > 0 \mid d_i)$ are associated with a particular n_i which is the average value in the range and the probability for all the other values of n_i is simply set to zero. The corresponding q_0 for such a pattern is calculated as:

$$q_0 = \left(\frac{m_1}{m_0} p_0 \sum_{d_i=1}^{m_0} d_i \, \gamma_{d_i}^0 - p^I \sum_{d_i=1}^{m_0+m_1} d_i \, \gamma_{d_i}^I + p^I \frac{m_0+m_1}{m_0} \Big[\sum_{d_i=2}^{m_1} \gamma_{d_i}^I \, a_{d_i} + \sum_{d_i=m_1+1}^{m_0+m_1} \gamma_{d_i}^I \, b_{d_i} \Big] \right) \Big/ \left(p^I \frac{m_0+m_1}{m_0} \sum_{d_i=2}^{m_1} \gamma_{d_i}^I \, a_{d_i} \right) \quad (3.38)$$

where

$$a_{d_i} = \Big\lfloor \frac{1 + \min(m_0, d_i - 1)}{2} \Big\rfloor \quad (3.39)$$

$$b_{d_i} = \Big\lfloor \frac{d_i - m_1 + \min(m_0, d_i - 1)}{2} \Big\rfloor \quad (3.40)$$

The algorithm for central sampling approach is referred to as *mixed fountain coding-central sampling* (MFC-CS). This algorithm is presented in full in algorithm 2.

Algorithm 1 MFC-SUS

$p \leftarrow rand(0 \text{ to } 1)$
if $p < m_0/(m_0 + m_1 + m_2)$ **then**
  $d \sim RSD(c, \delta, m_0 k)$
  choose d distinct symbols from X_0 uniformly
else if $p < (m_0 + m_1)/(m_0 + m_1 + m_2)$ **then**
  $d \sim RSD(c, \delta, (m_0 + m_1)k)$
  $q \leftarrow rand(0 \text{ to } 1)$
  if $d > m_1$ **then**
    choose 1 symbol from X_0 uniformly
    choose $d - 1$ distinct symbols from the rest of X_0 or X_1 uniformly
  else if $q < q_0$ **then**
    choose d symbol from X_1 uniformly
  else
    choose 1 symbol from X_0 uniformly
    choose 1 symbol from X_2 uniformly
    choose $d - 2$ distinct symbols from the rest of X_0 or X_1 uniformly
  end if
else
  $d \sim RSD(c, \delta, (m_0 + m_2)k)$
  $q \leftarrow rand(0 \text{ to } 1)$
  if $d > m_2$ **then**
    choose 1 symbol from X_0 uniformly
    choose $d - 1$ distinct symbols from the rest of X_0 or X_2 uniformly
  else if $q < q_0'$ **then**
    choose d symbols from X_2 uniformly
  else
    choose 1 symbol from X_0 uniformly
    choose 1 symbol from X_2 uniformly
    choose $d - 2$ distinct symbols from the rest of X_0 or X_2 uniformly
  end if
end if

Algorithm 2 MFC-CS

$p \leftarrow rand(0 \text{ to } 1)$
if $p < m_0/(m_0 + m_1 + m_2)$ **then**
  $d \sim RSD(c, \delta, m_0 k)$
  choose d distinct symbols from X_0 uniformly
else if $p < (m_0 + m_1)/(m_0 + m_1 + m_2)$ **then**
  $d \sim RSD(c, \delta, (m_0 + m_1)k)$
  $q \leftarrow rand(0 \text{ to } 1)$
  if $d > m_1$ **then**
    choose $d' = [(d - m_1 + \min(m_0, d - 1))/2]$ symbols from X_0 uniformly
    choose $d - d'$ distinct symbols from the X_1 uniformly
  else if $q < q_0$ **then**
    choose d symbols from X_1 uniformly
  else
    choose $d' = [(1 + \min(m_0, d - 1))/2]$ symbols from X_0 uniformly
    choose $d - d'$ distinct symbols from the X_1 uniformly
  end if
else
  $d \sim RSD(c, \delta, (m_0 + m_2)k)$
  $q \leftarrow rand(0 \text{ to } 1)$
  if $d > m_2$ **then**
    choose $d' = [(d - m_2 + \min(m_0, d - 1))/2]$ symbols from X_0 uniformly
    choose $d - d'$ distinct symbols from X_2 uniformly
  else if $q < q_0'$ **then**
    choose d symbols from X_1 uniformly
  else
    choose $d' = [(1 + \min(m_0, d - 1))/2]$ symbols from X_0 uniformly
    choose $d - d'$ distinct symbols from the X_2 uniformly
  end if
end if

3.4 Simulation Results and Conclusion

In this section, the two different transmissions protocols introduced in Section 3.3 are compared with the benchmarks through Monte-Carlo simulations. In all the simulations, we assume that RSD is used at the server, however, there is no obligation on the choice of distribution. The difference in performance illustrated here is caused by different sampling patterns. As the benchmarks for this problem, we consider the conventional LT codes combined with time-division multiple access (TDMA) protocol. Two different combinations are observed: *RSD-TDMA-Message-Based* and *RSD-TDMA-Client-Based*.

The RSD-TDMA-Message-Based mentioned in the figures refers to a protocol in which no packets are generated by mixing the bits of different messages. In other words, each transmitted packet just belongs to one of the requested messages whose degree is chosen based on RSD. The term message-based used in the name also emphasizes the fact that the transmission opportunities are shared among the messages at different time slots. The probabilities of transmitting pure packets of different messages are based on the arguments in Section 3.2.

$$(p_0, p_1, p_2) = \left(\frac{m_0}{m_0 + m_1 + m_2}, \frac{m_1}{m_0 + m_1 + m_2}, \frac{m_2}{m_0 + m_1 + m_2} \right) \tag{3.41}$$

The other benchmark scheme considered is another TDMA-based protocol where the transmission opportunities at different time slots are shared among the terminals and according to the corresponding Want sets. This protocol is referred to as RSD-TDMA-Client-Based protocol. Here, the Want sets of individual terminals point to a uniform sampling of the 'composite' message. This method does not take into account

the fact that the demands of the terminals overlap. As such, it is expected to work poorly when the size of the joint message m_0 increases in relation to m_1 and m_2. If the probabilities of switching between T_1 and T_2 are shown by p_{T_1} and p_{T_2} respectively, then we have:

$$(p_{T_1}, p_{T_2}) = \left(\frac{m_0 + m_1}{m_0 + m_1 + m_2}, \frac{m_0 + m_2}{m_0 + m_1 + m_2} \right). \tag{3.42}$$

Our proposed schemes outperform both of these protocols in terms of minimizing the number of transmissions. In the following figures, we consider three different scenarios, $m_0 = m_1 = m_2 = k$, $2m_0 = 2m_1 = m_2 = 2k$, and $6m_0 = 3m_1 = 2m_2 = 6k$ and show the effect of message length and channel erasure rates on the four schemes. It is noteworthy that two proposed protocols perform very similarly. In all the cases studied, the difference between our two proposed methods has been negligible. This means that for the ranges of m_0, m_1, and m_2 tested for WSN applications, it suffices to satisfy Equation (3.24) and mix. The level of mixing of the sources does not play a significant role.

In addition to evaluating the system performance versus message length, we consider the effect of channel erasure rates for the scenario $m_0 = m_1 = m_2 = k$. The result is illustrated in Figure 3.4. For other two scenarios, the patterns would be the same. We also evaluate the proposed protocol performance in comparison to the benchmarks by sending a fix percentage of overhead and considering success rate and percentage of erasure bits. The results demonstrated in Figures 3.5 and 3.6 show that TDMA-Message-Based algorithm outperforms the TDMA-Client-Based algorithm in less overhead percentage, while it falls behind TDMA-Client-Based as the overhead percentage increases. However, our proposed algorithm outperforms both methods in every stages of transmission.

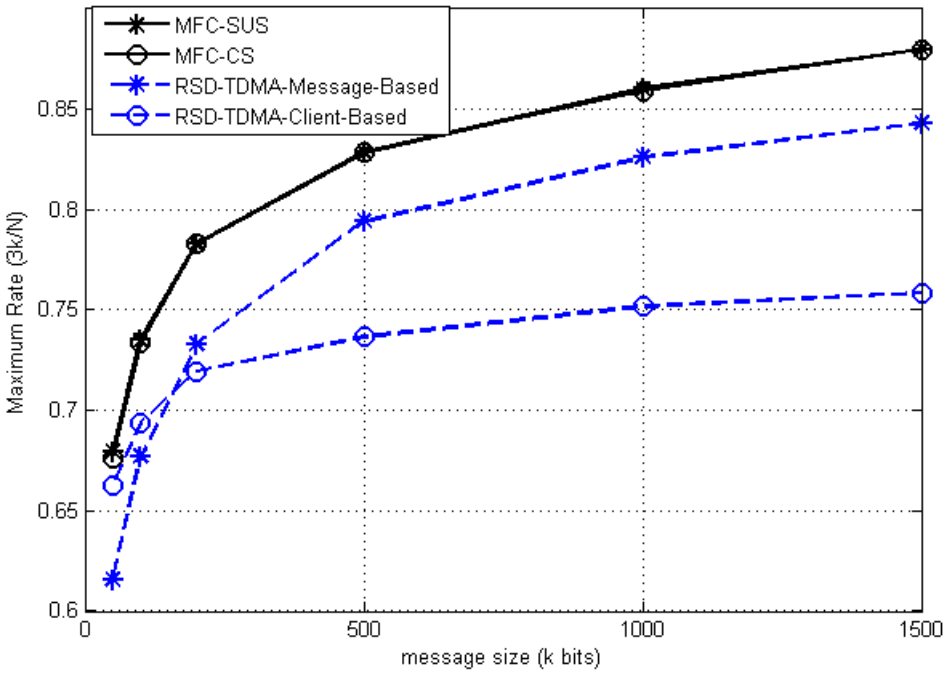

Figure 3.3: Ratio of the number of transmissions (N bits) to the total number of information bits (3k bits) versus message length for scenarios with $m_0 = m_1 = m_2 = k$

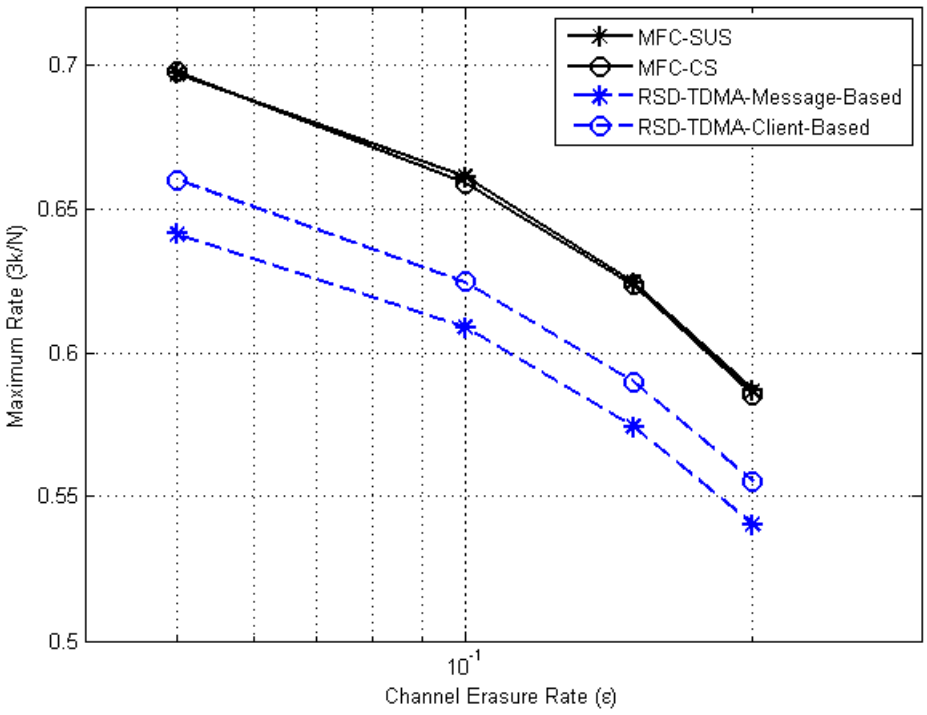

Figure 3.4: Ratio of the number of transmissions (N bits) to the total number of information bits (3k bits) versus channel erasure rate for scenarios with $m_0 = m_1 = m_2 = k$

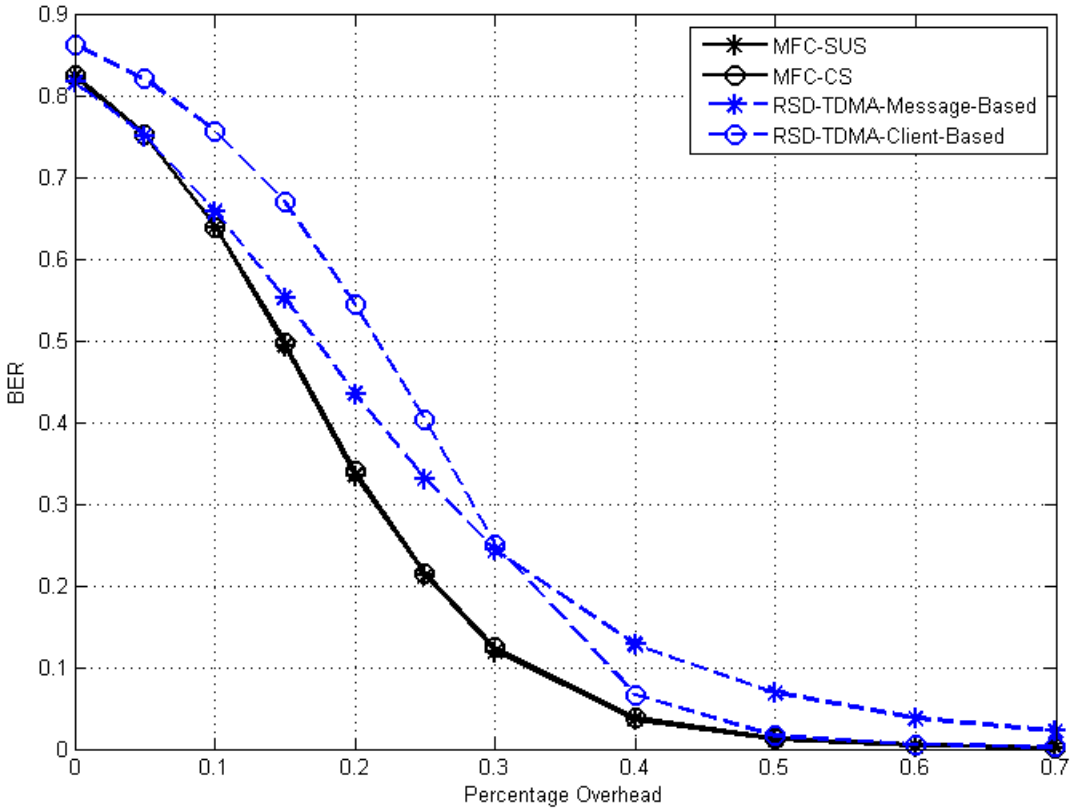

Figure 3.5: Bit rate versus overhead percentage when $m_0 = m_1 = m_2 = 150$ bits

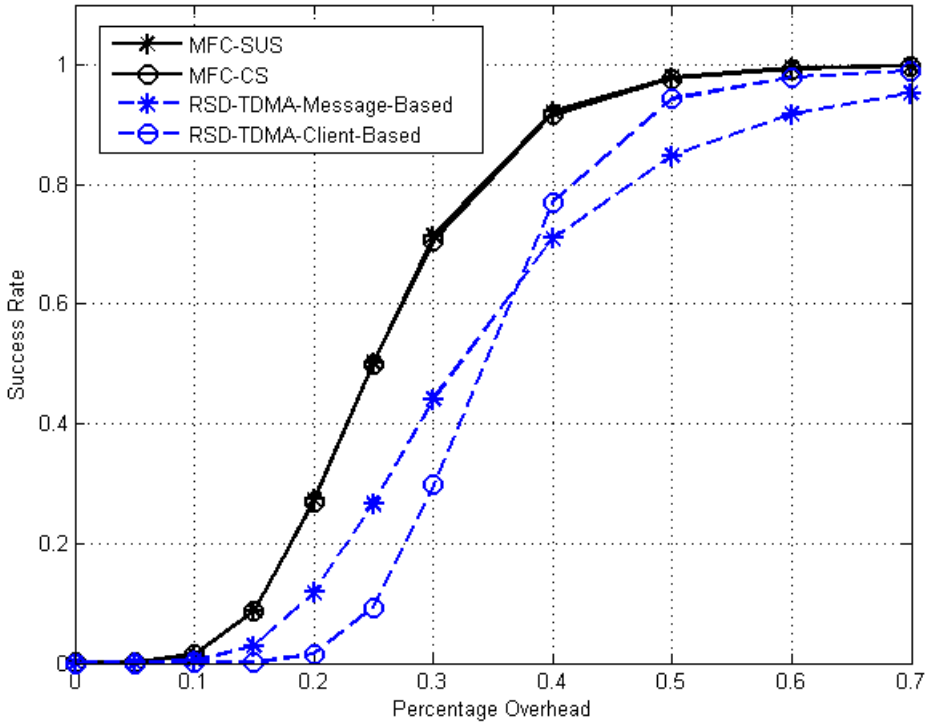

Figure 3.6: Success rate versus overhead percentage when $m_0 = m_1 = m_2 = 150$ bits

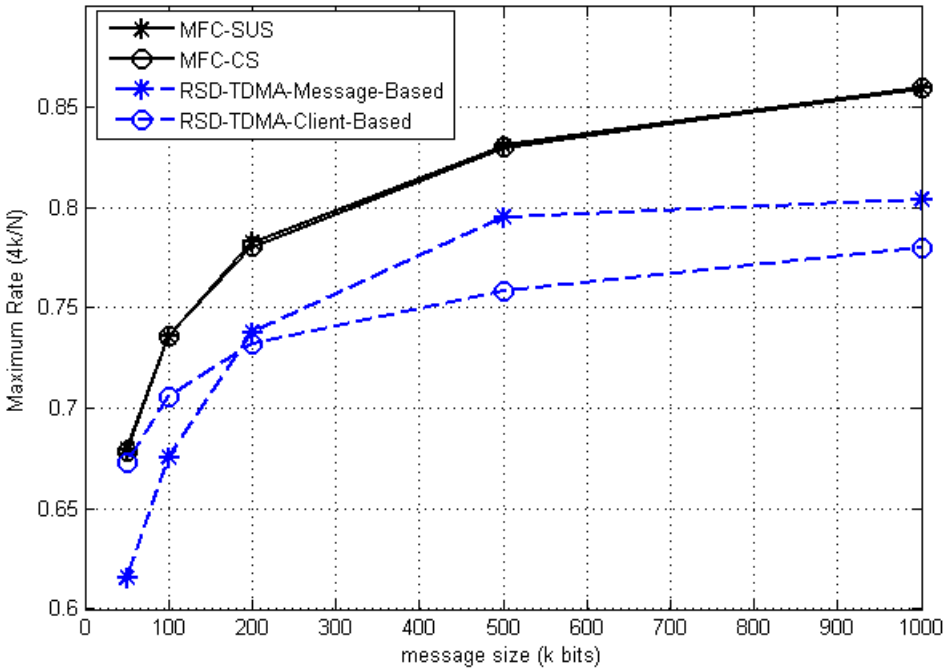

Figure 3.7: Ratio of the number of transmissions (N bits) to the total number of information bits (4k bits) versus message length for scenarios with $2m_0 = 2m_1 = m_2 = 2k$

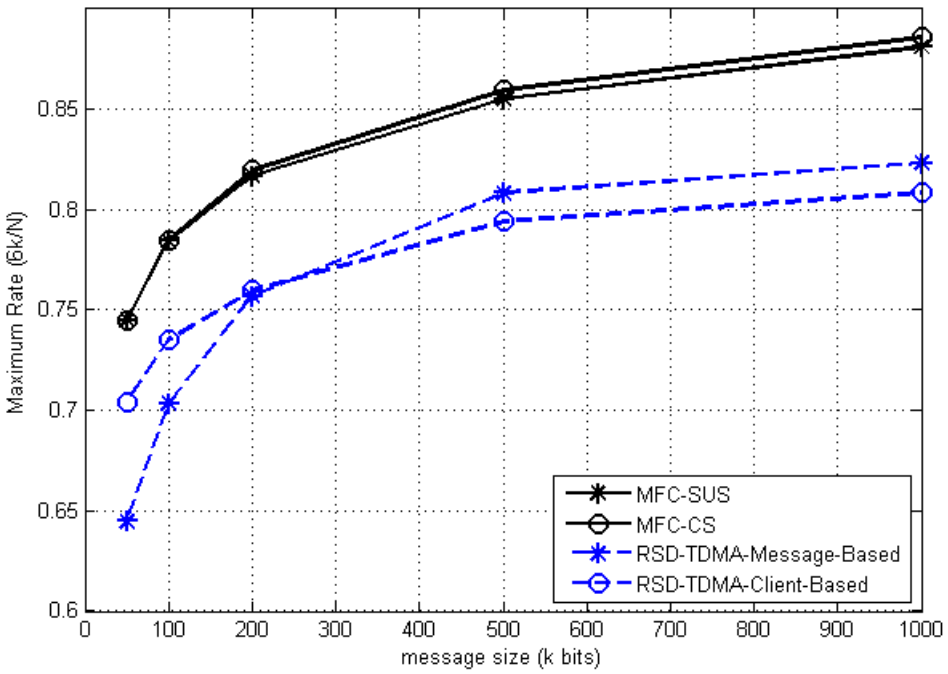

Figure 3.8: Ratio of the number of transmissions (N bits) to total number of information bits (6k bits) versus message length for scenarios that $6m_0 = 3m_1 = 2m_2 = 6k$

Chapter 4

Summary and Conclusions

4.1 Summary

Fountain codes have shown to be promising approach in erasure channels due to their outstanding features. The system models used to describe this kind of codes is a broadcast setting with single-source multiple-terminals with the same demands. Fountain codes asymptotically offer capacity-achieving universal solutions for such models that along with being compatible with low encoder/decoder, they are one of the most powerful erasure codes in coding theory. However, fountain codes are not structured for more general cases of broadcasting in addition to not showing good performance in finite range message sizes.

In this thesis, we investigated the performance of fountain codes for practical use and in a more generalized setting. We consider single source multiple-destinations systems in which each destination requests an arbitrary set of demands from the server. Our proposed fountain-coding-based transmission protocol decreases the number of transmissions from the server to the clients. The protocol introduced in this thesis

improves the number of transmissions by a new sampling scheme that works properly for a given degree distribution. For any distribution, we have found the exact probabilities of mixing the different messages. These probabilities are functions of the message size, number of requests for each message, and of course the degree distribution. The interesting point in this procedure is that some degrees of freedom for sampling is still left. In this thesis we introduce two different approaches: MFC-SUS and MFC-CS. Both of these sampling approaches can merged by any desired degree distribution, so there is no obligation on the choice of degree distribution

In chapter 3, we take Robust-Soliton as the given degree distribution and apply it to our sampling protocol. Then, we compare our results to other coding approaches for this problem through Monte-Carlo simulations. Both approaches introduced in this thesis beat the available schemes.

4.2 Future Directions

Although the results presented in this thesis have demonstrated the better performance of the MFC approach, it is not proved to be the optimal scheme, so there is room to go further and develop the optimized answer for the system.

In addition to investigating the optimal solution, there are numerous directions to expand this work, a list of which is provided below.

- The problem considered in this thesis is a stepping stone for a much bigger and sophisticated systems with multiple sources with higher number of receivers. Expanding this work for more generalized structures and ultimately for a random network structures is very valuable in terms of covering more practical applications.

- It is also informative to use different channels models like noisy or fading channels to investigate the performance [24][25][26].

- In the considered system model, it is assumed that there is no intermediate feedback from the receivers to inform the server about their current status. However, we can consider a system in which terminals send different ACKs to the server after recovering each individual messages. In the systems with multiple feedback, the probabilities values should get updated after each time the status of the system changes [27].

- As described in Chapter 3, there are different sampling patterns that can be applied to MFC approach. We have chosen two specific approaches in this thesis, semi-uniform and central sampling. Although there is a huge difference between these two methods, the performances of both are close to each other. This phenomenon opens up a discussion to study whether there is a prominent justification behind that or not.

Bibliography

[1] S. Puducheri, J. Kliewer, and T. E. Fuja, "The design and performance of distributed LT codes," *IEEE Transactions on Information Theory*, vol. 53, no. 10, pp. 3740–3754, Oct. 2007.

[2] P. Elias, "Coding for two noisy channels," in *Proc. 3rd London Symp. Information Theory*, London, U.K., 1956, pp. 61–76.

[3] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* John Wiley & Sons, Inc., 1991, ch. 7, p. 188.

[4] M. Luby, "LT codes," in *Proc. 43rd Annual IEEE Symposium on Foundations of Computer Science*, Vancouver, BC, Canada, Nov. 2002, pp. 271–280.

[5] H. Weatherspoon and J. D. Kubiatowicz, "Erasure coding vs. replication: a quantitiative comparison," in *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, Cambridge, MA, USA, Mar. 2002, pp. 328–338.

[6] G. S. I. S. Reed, "Polynomial codes over certain finite fields," *J. Soc. Indust. Appl. Math*, vol. 8, pp. 300–304, 1960.

[7] M. Mitzenmacher, "Digital fountains: A survey and look forward." in *Information Theory Workshop*, San Antonio, TX, USA, Oct. 2004, pp. 271 – 276.

[8] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.

[9] A. Liau, S. Yousefi, and I.-M. Kim, "Binary soliton-like rateless coding for the y-network," *IEEE Transactions on Communications*, vol. 59, no. 12, pp. 3217–3222, Dec. 2011.

[10] A. Liau, I.-M. Kim, and S. Yousefi, "Improved low-complexity soliton-like network coding for a resource-limited relay," *IEEE Transactions on Communications*, vol. 61, no. 8, pp. 3327–3335, Aug. 2013.

[11] B. Karaoglu and W. Heinzelman, "Multicasting vs. broadcasting: What are the trade-offs?" in *IEEE Global Telecommunications Conference (GLOBECOM)*, London, U.K., Dec. 2010, pp. 1–5.

[12] L. K. Law, S. V. Krishnamurthy, and M. Faloutsos, "Understanding and exploiting the trade-offs between broadcasting and multicasting in mobile ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 3, pp. 264–279, 2007.

[13] I. F. Akyildiz, S. Weilian, Y. Sankarasubramaniam, , and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[14] A. E. Gamal, "Capacity of the product and sum of two unmatched broadcast channels," *Problem of Information Transmission (English Translation of Problemi Peredachi Informatsii)*, vol. 16, no. 1, pp. 3–23, Jan.-Mar. 1980.

[15] K. Marton, "A coding theorem for the discrete memoryless broadcast channel," *IEEE Transactions on Information Theory*, vol. 25, no. 3, pp. 306–311, May 1979.

[16] T. M. Cover, "Comments on broadcast channels," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2524–2530, Oct. 1998.

[17] S. Shamai, I. E. Telatar, and S. Verdu, "Fountain capacity," *IEEE Transactions on Information Theory*, vol. 53, no. 11, pp. 4372–4376, Nov. 2007.

[18] T. M. Cover, "Broadcast channels," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 2–14, Jan. 1972.

[19] D. J. C. MacKay, "Fountain codes," *IEE Proc. Communications*, vol. 152, no. 6, pp. 1062–1068, Dec. 2005.

[20] K. F. Hayajneh, S. Yousefi, and M. Valipour, "Left degree distribution shaping for LT codes over the binary erasure channel," in *27th Biennial Symposium on Communications (QBSC)*, Kingston, ON, Canada, June 2014, pp. 198–202.

[21] S. S. Woo and M. K. Cheng, "Prioritized LT codes," in *Proc. 42nd Annual Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, Mar. 2008, pp. 568–573.

[22] A. Albanese, J. Blomer, J. Edmonds, M. Luby, and M. Sudan, "Priority encoding transmission," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1737–1744, Nov. 1996.

[23] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

[24] R. Palanki and J. S. Yedidia, "Rateless codes on noisy channels," in *Proc. International Symposium on Information Theory (ISIT)*, Chicago, IL, USA, July 2004, p. 37.

[25] X. Liu and T. J. Lim, "Fountain codes over fading relay channels," *IEEE Transactions on Wireless Communications*, vol. 8, no. 6, pp. 3278–3287, June 2009.

[26] B. Yang, R. Carrasco, and A. Adams, "Implementation of fountain codes over fading channels," in *International Conference on Wireless, Mobile and Multimedia Networks*, hangzhou, China, Nov. 2006, pp. 1–4.

[27] A. Hagedorn, S. Agarwal, D. Starobinski, and A. Trachtenberg, "Rateless coding with feedback," in *IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 2009, pp. 1791–1799.

[28] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 569–584, Feb. 2001.

[29] R. Karp, M. Luby, and A. Shokrollahi, "Finite length analysis of LT codes," in *Proc. International Symposium on Information Theory (ISIT)*, Chicago, IL, USA, July 2004, p. 39.