CODING FOR WIRELESS AD-HOC AND SENSOR NETWORKS: UNEQUAL ERROR PROTECTION AND EFFICIENT DATA BROADCASTING

A Thesis
Presented to
The Academic Faculty

by

Nazanin Rahnavard

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
December 2007

CODING FOR WIRELESS AD-HOC AND SENSOR NETWORKS: UNEQUAL ERROR PROTECTION AND EFFICIENT DATA BROADCASTING

Approved by:

Professor Faramarz Fekri, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Ian F. Akyildiz
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Steven W. McLaughlin
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor James H. McClellan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Christopher E. Heil
School of Mathematics
Georgia Institute of Technology

Date Approved: 25 July 2007

To my lovely parents, sister, and husband:

Ahmad, Farideh, Asal Rahnavard, and Reza

ACKNOWLEDGEMENTS

There are so many names and faces that come to my mind when I think of people who have had an impact in my professional life and success, those who taught me, encouraged me, and supported me. This long journey would not have been possible at all if I did not have so many great people in my life. I am sure I will leave out the names of many of them here, but all of them will be remembered forever.

I start with my great advisor, Professor Faramarz Fekri. I feel very fortunate to have him as my thesis advisor, and I thank him a lot for his guidance, encouragement, and support. Not only was he a great mentor in my research, but he was also a role model of hard work and dedication for me. I thank him for being beside me in all the ups and downs that a PhD student must surely face.

I would like to extend my special thanks to Professor Ian F. Akyildiz. I got the chance to learn about wireless sensor networks from him, a pioneer in this field, and to be inspired by his knowledge and personality. I thank him deeply for being a great mentor and support to me.

I also would like to thank my other thesis committee members: Professor Steven W. McLaughlin, and Professor Christopher E. Heil, whom I had also the opportunity to take their classes and to learn a lot from them, and Professor James H. McClellan, whom I feel lucky to have had the chance to know him through my proposal exam and to benefit from his valuable insight. I thank all of these outstanding professors for their inspiration, support, and guidance on my thesis.

I have so many friends and colleagues whom I would also like to thank. My special thanks to Hossein Pishro-Nik and Badri N. Vellambi, with whom I had the most collaboration and discussions. Chapter 6 of this thesis is the result of my

collaboration with Badri. I also thank Farshid Delgosha, Mina Sartipi, Kevin Chan, Ramanan Subramanian, Majid Fozounbal, Raviv Raich, Omid Momtahan, Tommaso Melodia, Marco Siniscalchi, Dario Pompili, and Kurt Belgum, for their friendships and helpful discussions. Our paths have crossed during the past few years, and we shared a lot with each other.

I would like to extend my gratitude to the students, faculty, and staff at the Center for Signal and Image Processing (CSIP), who provided a great, thriving, and friendly work environment. I thank all the faculty and staff of the School of Electrical and Computer Engineering and also the whole Georgia Tech community. I would also like to show my appreciation for those people who made Georgia Tech a nice and safe home for students. With their non-stop effort, I was able to work after hours and leave late, knowing that many people care about our safety, and they work hard to smooth our paths to success.

I would like to thank my great and dear family. My dear mom, Farideh Haj Hosseinkhan, and my dear dad, Ahmad Rahnavard, who taught me to love learning and search for it. They taught me to be strong and determined. Thank you very much for all your love and support, and dedication and sacrifices that you made in your lives to make my life a better and more fruitful one. Hearing your voices from miles away has been a great source of energy for me to keep going. I also would like to thank my dear sister, Asal Rahnavard, for her loving care, support, and encouragement. I would not be this happy and comforted if I did not have you and your lovely son, Alireza, in my life. I also thank my unforgettable grandparents.

Last but not least, I thank my best friend and dear husband, Reza. We started this journey 12 years ago together, and his presence has made a significant difference in both my professional and personal life. Thank you for always being there for me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

This thesis investigates both theoretical and practical aspects of the design and analysis of modern error-control coding schemes, namely low-density parity-check (LDPC) codes and rateless codes for unequal error protection (UEP). It also studies the application of modern error-control codes in efficient data dissemination in wireless ad-hoc and sensor networks.

Two methodologies for the design and analysis of UEP-LDPC codes are proposed. For these proposed ensembles, density evolution formulas over the binary erasure channel are derived and used to optimize the degree distribution of the codes. Furthermore, for the first time, rateless codes that can provide UEP are developed. In addition to providing UEP, the proposed codes can be used in applications for which unequal recovery time is desirable, i.e., when more important parts of data are required to be recovered faster than less important parts. Asymptotic behavior of the UEP-rateless codes under the iterative decoding is investigated. In addition, the performance of the proposed codes is examined under the maximum-likelihood decoding, when the codes have short to moderate lengths. Results show that UEP-rateless codes are able to provide very low error rates for more important bits with only a subtle loss in the performance of less important bits. Moreover, it is shown that given a target bit error rate, different parts of the information symbols can be decoded after receiving different numbers of encoded symbols. This implies that information can be recovered in a progressive manner, which is of interest in many practical applications such as media-on-demand systems.

This work also explores fundamental research problems related to applying error-control coding such as rateless coding to the problem of reliable and energy-efficient

broadcasting in multihop wireless ad-hoc sensor networks. The proposed research touches on the four very large fields of wireless networking, coding theory, graph theory, and percolation theory. Based on the level of information that each node has about the network topology, several reliable and energy-efficient schemes are proposed, all of which are distributed and have low complexity of implementation. The first protocol does not require any information about the network topology. Another protocol, which is more energy efficient, assumes each node has local information about the network topology. In addition, this work proposes a distributed scheme for finding low-cost broadcast trees in wireless networks. This scheme takes into account various parameters such as distances between nodes and link losses. This protocol is then extended to find low-cost multicast trees. Several schemes are extensively simulated and are compared.

CHAPTER I

INTRODUCTION

Many communication channels suffer from noise, interference, or distortion because of hardware imperfections or physical limitations. The goal of *error-control coding* (ECC) is to encode information in such a way that even if the channel introduces errors, the receiver can correct the errors and recover the original transmitted information.

Error-control coding is pervasive in many aspects of human life. It has strongly influenced not only the development of wireless systems, CDs, DVDs, and data storage, but also computer networks, satellites, optical communication systems, mobile phones, and of course the Internet.

After the seminal work of Shannon in 1948 [66], which defined the theoretical limits on noiseless data communications, the first ECC scheme was proposed by Hamming [19]. After that, many other coding schemes were proposed. However, no one really knew how to achieve the theoretical performance limits promised by Shannon. In 1963, Galleger invented *low-density parity-check* (LDPC) codes [14]. The true power of these codes was, however, overlooked, partly because of the lack of sufficient computing power for their implementation at the time.

Loosely speaking, error-control coding history can be divided into pre-turbo code and post-turbo code. *Turbo codes* [5] and their ingenious iterative decoder, invented in 1993 by Claude Berrou and Alain Glavieux, revolutionized the area. After turbo codes started to gain attention, coding theorists turned their research efforts toward the area of soft decision iterative decoders and toward the search for lower complexity codes. These efforts led to the rediscovery of LDPC codes [35]. Today, the codes

coming closest to the Shannon bound are LDPC codes, and much work has recently focused on their construction and analysis.

While LDPC codes are among the most promising codes that are designed for fixed rates, a new class of codes, called *rateless* or *Fountain codes*, has recently been introduced. As the name suggests, rateless codes do not have any rate, and their design for lossy channel applications is independent of the value of channel loss. These codes were introduced primarily for the purpose of scalable and fault-tolerant data distribution over computer networks. Examples of such codes are *LT codes* [31], *Raptor codes* [68], and *Online codes* [39]. The fact that the design of rateless codes is independent of the channel parameter makes them very interesting for applications such as transmitting data over lossy multicast channels, nonuniform channels, and time-varying channels.

This work investigates both the theoretical and practical aspects of the design and analysis of LDPC codes and rateless codes. In Chapters 3, we propose two schemes for designing LDPC codes that provide *unequal error protection* (UEP). UEP is very important when certain parts of the information may need a higher level of protection against error than other parts. For example, in a packet sent through a network, header bits are more important than payload bits and have to be protected more. Another example is in a compressed video stream, such as an MPEG stream, where I-frames need more protection than P-frames and B-frames. In Chapter 4, we develop, for the first time, rateless codes that can provide unequal error protection. In addition to providing UEP, these codes can also be used in applications for which *unequal recovery time* (URT) is desirable, i.e., when more important parts of data must be recovered faster than less important parts. In our design and analysis, we consider both asymptotic and finite-length cases. We investigate the asymptotic behavior of UEP-rateless codes when using iterative decoding. We also study the performance of maximum-likelihood (ML) decoders for the proposed finite-length UEP-rateless

codes. Moreover, we analyze the ML decoding performance of traditional rateless codes.

Further, we investigate the problem of reliable and energy-efficient broadcasting in multihop wireless ad-hoc and sensor networks. The goal is to deliver a large number of data packets, in a multihop fashion, to all the nodes in a network, such that the energy consumption is minimized. Our approach to tackle this problem is based on utilizing rateless codes. In Chapter 5, we propose a two-phase protocol called *collaborative rateless broadcast* (CRBcast). This scheme utilizes an application-layer rateless coding in conjunction with a simple and scalable broadcast scheme called *probabilistic broadcast* (PBcast). In the first phase of CRBcast, rateless encoded packets are broadcast based on PBcast. The second recovery phase, which is based on simple collaborations of nodes, ensures that all nodes can recover original data. CRBcast is a localized scheme that does not need any knowledge of the network. We study both the theoretical and practical aspects of PBcast and CRBcast for large-scale networks. In order to do this, we deal with many properties of random graphs such as connectivity, coverage, and giant components.

In Chapter 6, we study the same problem for the cases in which nodes in the network do have some local knowledge of their neighboring nodes. This local information can be exploited to design more efficient data dissemination protocols for wireless sensor networks. We propose a scheme called the *fractional transmission scheme* (FTS). In FTS, different neighbors of a node share the data delivery, and each node sends only a fraction of the total encoded packets required by a receiving node. FTS is a distributed and efficient broadcast scheme that has also low-complexity and provides load balancing.

In Chapter 7, we consider the problem of efficient broadcasting for a general wireless network model, where packet transmissions are associated with costs based on parameters such as the distance between nodes, link losses, etc. We also consider

nodes with adjustable transmission power. We propose a distributed protocol for finding low-cost broadcast trees, which can be used for reliable, energy-efficient, and low-latency data broadcast in wireless networks. The proposed scheme, referred to as *broadcast decremental power* (BDP), evolves a given spanning tree of the network and forms other spanning trees with lower broadcast costs. In our scheme, the Bellman-Ford tree is considered as the initial spanning tree. We then propose a generalization of BDP for efficient *multicast*, where only a subset of nodes in the network requires the data. This scheme is referred to as *multicast decremental power* (MDP).

Finally, Chapter 8 summarizes the completed work and points out some of the possible future research directions.

CHAPTER II

BACKGROUND

In this chapter, we briefly present the necessary background on which the thesis is based. We begin by introducing the binary erasure channel model. Then, we review low-density parity-check and rateless codes and important developments in these areas. Next, we provide background on unequal error protection and related previous work. Finally, we review the problem of efficient data broadcasting in wireless sensor networks.

2.1 Binary Erasure Channel

The *binary erasure channel* (BEC) was introduced by Elias in 1955 [13]. For about 40 years, BEC was considered just a theoretical model rather than a realistic model. However, after the emergence of the Internet, the erasure channel model became a realistic one. Indeed, erasure channels can be used to model data networks, where data is transmitted in the form of packets, which either arrive correctly or are lost for many reasons such as buffer overflow or packet checksum mismatch. In binary erasure channels, bits are either received correctly or are lost. The BEC is represented by a parameter ε, which is called the *channel erasure probability*. In other words, a bit is either lost or received correctly with probability ε or $1-\varepsilon$, respectively. Figure 2.1 depicts the BEC model. The Shannon capacity of the BEC with parameter ε is $C_{BEC} = 1-\varepsilon$ bits per channel use. In this work, we mostly consider channels as erasure channels.

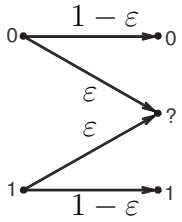


Figure 2.1: Binary erasure channel.

2.2 Low-Density Parity-Check Codes

A low-density parity-check (LDPC) code over $GF(2)$ is defined as a linear block code (n, k) with an $m \times n$ $(m = n - k)$ *sparse* parity-check matrix $H = [h_{ij}]$ (i.e., most of the elements of H are 0 and a few of them are 1). A codeword x is a valid codeword if and only if $Hx^T = 0$. Each column of H corresponds to one bit of a codeword x, and each row corresponds to a parity-check equation. An LDPC code can also be represented by a graph called a Tanner graph [71]. A Tanner graph is a bipartite graph with bipartition V and C, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of variable nodes and $C = \{c_1, c_2, \ldots, c_m\}$ is the set of check nodes. Variable nodes correspond to the bits of the codeword, and check nodes correspond to the set of parity-check constraints. Check node i (c_i) and variable node j (v_j) are adjacent (connected by an edge) if and only if $h_{ij} = 1$. We denote such a graph by $G(n, m)$. As an example, Figure 2.2 depicts the Tanner graph of the code defined by

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{1}$$

The degree of each node is defined as the number of edges connected to that node. An LDPC code is called regular if all variable nodes have the same degree and also all check nodes have the same degree. Otherwise, the code is called irregular. Let λ_i denote the fraction of the edges connected to variable nodes of degree i. Similarly,

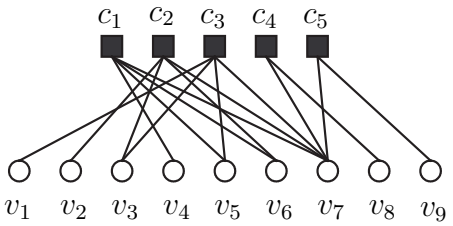


Figure 2.2: The Tanner graph of the H matrix given in (1). Circular and rectangular nodes correspond to variable nodes and check nodes, respectively.

let ρ_i denote the fraction of the edges connected to check nodes of degree i. A degree distribution for the Tanner graph is the pair (λ, ρ), where

$$\lambda = \lambda(x) = \sum_i \lambda_i x^{i-1} \text{ and } \rho = \rho(x) = \sum_i \rho_i x^{i-1}. \tag{2}$$

For a given length n and given degree distribution pair (λ, ρ), an *ensemble* $\mathcal{C}^n(\lambda, \rho)$ of LDPC codes is defined by choosing the edges randomly [64]. More precisely, for each variable node of degree i we assign i sockets. Similarly, for each check node of degree i we assign i sockets. The total number of sockets in each side of the graph (total number of edges) is $E = \frac{n}{\int_0^1 \lambda(x)dx}$. We enumerate the E variable sockets in some arbitrary order and proceed in the same way with the check sockets. Then, a code (a particular instance of this ensemble) can be specified with a permutation π on E letters. For each $i = 1, 2, \ldots, E$, we connect the variable node associated with the i^{th} variable socket to the check node associated with the $\pi(i)^{th}$ check socket.

2.2.1 Iterative Decoding on the Binary Erasure Channel

The decoding of LDPC codes is based on a message passing algorithm called *belief propagation* (BP). This algorithm is iterative and in each iteration messages are passed from variable nodes to check nodes, and from check nodes back to variable nodes. The messages that are passed between the nodes are in the form of a log-likelihood ratio (LLR) [64]. The message passed from a variable node v to a check node c contains the information about the probability that v has a certain value given the observed value of that variable node, and all the values communicated to v in the prior round

from check nodes adjacent to v other than c. On the other hand, the message passed from c to v contains the information about the probability that v has a certain value given all the messages passed to c in the previous round from message nodes other than v. In general, memoryless binary-input output-symmetric channel LLRs can take values in the range $[-\infty, +\infty]$. However, for the special case when the channel is BEC, LLRs can only take values in $\{-\infty, 0, +\infty\}$. LLR=0 means that the message is absolutely uncertain and |LLR|=$+\infty$ means that the message is absolutely certain. The sign of the LLR determines whether the value of the associated bit is 0 or 1. For the BEC, the BP algorithm is very simple and can be described much easier in the following [28, 67]:

- Initialize the value of all the check nodes to zero.
- For every variable node v in the graph that its value is known, add (modulo 2) the value of v to the values of all adjacent check nodes of v. Remove v and all edges incident to it from the Tanner graph.
- While there is a check node c of degree one, substitute the value of c into the value of its unique adjacent variable node v', add that value to the values of all check nodes adjacent to v' and remove v' and all of its incident edges from the graph.

The number of operations performed by this algorithm is proportional to the number of edges in the graph. Therefore, for LDPC codes, the algorithm runs in time linear in the block length of the code ($O(n)$).

2.2.2 Density Evolution on the Binary Erasure Channel

Density evolution (DE) is a technique for analyzing the performance of LDPC codes under BP decoding when code length (n) goes to infinity. As the name suggests, DE tracks the evolution of the distribution of messages that are passed from variable nodes to check nodes and vice versa. For the sake of analysis and without loss of

generality we can assume the all-zero codeword is transmitted [64]. Therefore, when the transmission channel is BEC, the LLR messages can only take two values (either $+\infty$ or 0). Hence, we only need to keep track of one parameter x_i, which is the probability that the messages passed from a variable node to a check node at round i of BP is 0. It was shown in [28, 30] that given the degree distribution pair (λ, ρ), the DE formula is given by

$$x_i = x_0\lambda(1 - \rho(1 - x_{i-1})) \quad \text{for} \quad i \geq 1, \tag{3}$$

where $x_0 = \varepsilon$ for a BEC with channel erasure probability ε. It was shown in [64] that the density evolution recursion given in (3) always converges to a fixed point. Moreover, there exits a threshold $\varepsilon^{th}(\lambda, \rho)$ for the fixed point of (3), defined as

$$\varepsilon^{th}(\lambda, \rho) = sup\{\varepsilon \in [0, 1] : x_i \overset{i\to\infty}{\longrightarrow} 0\}. \tag{4}$$

The threshold ε^{th} is the supremum value of ε such that the error rate can be made arbitrarily small.

2.3 Rateless Codes

Rateless (Fountain) codes are a new class of linear error-control codes. Examples of such codes are LT codes [31], Raptor codes [68], and Online codes [39]. The idea behind rateless codes is that every receiver continues collecting the encoded data until the decoding can be finished successfully. Unlike traditional codes, rateless codes on lossy channels do not assume any knowledge of the channel. Therefore, rateless codes are very suitable candidates for applications in which the channel erasure probability is unknown, nonuniform, or time varying. It has been shown that rateless codes have very simple encoding and decoding algorithms. Asymptotically good degree distributions for them have also been developed [31], [68], [39]. Raptor and Online codes are extensions of LT codes, in which an outer traditional pre-code is concatenated to an inner LT code to get practically better results.

Next, we review LT codes. Suppose we want to transmit a message comprised of n input symbols. Let $\Omega_1, \ldots, \Omega_n$ be a probability distribution on $\{1, \ldots, n\}$ so that Ω_i denotes the probability that the value i is chosen. We may also denote this distribution by its generator polynomial $\Omega(x) = \sum_{i=1}^{n} \Omega_i x^i$. An encoding (output) symbol is formed as follows:

- Randomly choose a degree d according to the distribution $\Omega_1, \ldots, \Omega_n$
- Choose uniformly at random d input symbols
- Perform bitwise XOR operations on the selected d input symbols to form the output symbol

The output symbol is then transmitted. We repeat this process until a sufficient number of output symbols is obtained at the receiver. In general, the number of output symbols required to give a high probability of decoding n input symbols can be expressed as γn for a fraction $\gamma \gtrsim 1$ (γ is called the rateless overhead). When transmitting information using a traditional code, both the sender and the receiver are in possession of a description of the coding method used. For rateless codes, this is not necessarily the case, since the code is being generated concurrently with the transmission. Therefore, in order to be able to recover the original data from the output symbols, it is necessary to transmit a description of the code together with the output symbols. In a setting where the symbols are data packets, we can include the information that describes the set of neighbors of an output symbol in the header of the packet [68]. There are also several other methods to accomplish this [31]. Without loss of generality and for simplicity, we may assume that the symbols are binary symbols. Following [39] and [30] we may view the input and output symbols as vertices of a bipartite graph G, where the input symbols correspond to the variable nodes and the output symbols correspond to the check nodes. The ensemble of LT codes can be specified by parameters $\Omega(x)$, n, and γ. The average check-node degree is given by $\mu = \sum_{i=1}^{n} i\Omega_i = \Omega'(1)$, where $\Omega'(x)$ is the derivative of $\Omega(x)$ with respect

to x. Moreover, it is straightforward to show that the degree of variable nodes has a binomial distribution. Specifically, the probability λ_d that a variable node has degree d is given by

$$\lambda_d = \binom{\mu\gamma n}{d} (\frac{1}{n})^d (1 - \frac{1}{n})^{\mu\gamma n - d}. \tag{5}$$

Asymptotically (as n goes to infinity), and assuming that $\mu\gamma$ is constant, we can approximate distribution (20) by a poisson distribution with mean $\mu\gamma$ expressed as follows:

$$\lambda_d = \frac{e^{-\mu\gamma}(\mu\gamma)^d}{d!}. \tag{6}$$

2.3.1 Iterative Decoding on the Binary Erasure Channel

The decoding of LT codes on the BEC is similar to the decoding of LDPC codes on the BEC. The BP decoder can be best described in terms of the graph associated with the decoder. It performs the following steps until either no output symbols of degree one are present in the graph or until all the input symbols have been recovered [68]. At each step of the algorithm, the decoder identifies an output symbol of degree one. If none exists, and if not all the input symbols have been recovered, the algorithm reports a decoding failure. Otherwise, the value of the output symbol of degree one recovers the value of its unique neighbor among the input symbols. Once this input symbol value is recovered, its value is added to the values of all the neighboring output symbols, and the input symbols and all edges emanating from it are removed from the graph.

Figure 2.3 depicts a small example of an LT code, where $n = 7$ and $\gamma = 8/7$. Circular nodes correspond to the input symbols, and the rectangular nodes correspond to the output symbols. The values of the output symbols are known at the receiver, and the goal is to find the values of the input symbols. The decoding starts by copying the value of c_3 to its unique neighbor v_2. Next, since c_2 has only one unknown neighbor, it recovers the value of v_1. The next output symbol with only one unknown

neighbor is c_4 and recovers v_6. The decoding continues until no output symbol with exactly one unknown neighbor exists. In this example, the decoding is successful since the values of all the input symbols are determined.

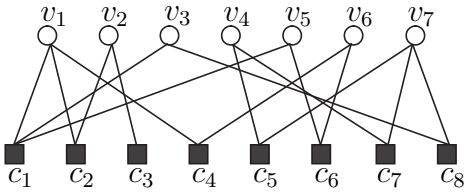


Figure 2.3: An example of an LT code, where $n = 7$ and $\gamma = 8/7$. The circular and rectangular nodes correspond to input and output symbols, respectively.

2.3.2 Density Evolution on the Binary Erasure Channel

Similar to LDPC codes, we can use the density evolution tool in the analysis and design of rateless codes under the iterative decoding. Consider an LT code with parameters $\Omega(x)$, information length n, and overhead γ. Define x_i to be the probability that the value of a variable node is unknown at round i of BP. It is clear that $x_0 = 1$ since at first the value of all input nodes are unknown. The DE can be formulated as follows [39]:

$$x_i = \delta(1 - \beta(1 - x_{i-1})), \quad \text{for} \quad i \geq 1, \tag{7}$$

in which $\delta(x) = e^{\mu\gamma(x-1)}$ and $\beta(x) = \frac{\Omega'(x)}{\Omega'(1)}$.

2.4 Unequal Error Protection

Most error-control coding schemes protect all data equally. In other words, they provide equal error protection (EEP). However, in several important applications, some parts of information may need a higher level of protection against error than other parts. For example, in a packet sent through a network, header bits are more important than payload bits and have to be protected more. Another very important application is in transmitting multimedia over error-prone wireless networks. In such

cases, we are left with three options. First, EEP codes with high protection for the entire data could be used. This is not efficient since EEP codes provide far more protection than is necessary by adding excessive redundancy. Second, different codes could be used for different parts of data. This approach is not prudent since the performance of codes degrades as the length of data decreases. Finally, a more interesting and challenging solution is the construction of a single code that induces a selective protection property known as unequal error protection (UEP).

The first UEP codes were proposed by Masnick and Wolf [38] in 1967. Later, many other UEP design methodologies were developed, e.g., [8, 9, 21, 36, 50, 73]. Because of the outstanding performance of LDPC and rateless codes, it is desirable to design these codes such that they provide UEP. In this work, we propose two schemes for designing UEP-LDPC codes. The proposed schemes are among the very first studies on UEP-LDPC codes. We also propose a novel scheme for having UEP-rateless codes, which to the best of our knowledge is devised for the first time.

2.5 Data Broadcast in Wireless Sensor Networks

Wireless sensor networks are becoming a part of our everyday lives [3]. They are finding applications in many areas such as health, disaster relief, environmental surveillance, and military sensing. In such networks, sensor nodes have limited energy, storage, and computational capability. Further, these networks are characterized by short communication range, low data rate, and dense deployment. The topology of a sensor network is subject to frequent changes because of node mobility or frequent node failure. In most of the applications, the deployment of nodes is such that recharging their batteries is infeasible; hence, low energy expenditure is of paramount importance. For this reason, protocols tailored to such networks must place the utmost emphasis on the conservation of energy, and the direct application of protocols used for general wireless networks is not efficient.

One of the common models for wireless ad-hoc and sensor networks with omni-directional antennas is the *random geometric graph* model [46]. A network with N nodes and transmission range r is modeled by a graph $G(N, r)$ such that there is an edge between any two nodes if and only if their distance is less than or equal to r. Gupta et al. studied the connectivity of such graphs when N is very large [17]. They showed that if N nodes are deployed uniformly at random in a field with area A and the transmission range of the nodes is such that

$$\frac{\pi r^2}{A} = \frac{\ln N + \omega(N)}{N}, \tag{8}$$

then the resulting network is asymptotically connected with probability one if and only if $\omega(N) \to \infty$.

By the nature of their applications, one-to-all (broadcast) and one-to-many (multicast) communication tasks are very common in wireless ad-hoc and sensor networks. One situation that calls for one-to-all broadcasting is the software update (needed for adding new functionalities, maintenance, and debugging) in all the sensor nodes after their deployment. Another example is route discovery in reactive routing protocols (where route query packets are forwarded to all nodes in the network). Some important factors that influence the efficiency of a broadcasting scheme can be reliability (defined as the percentage of nodes in the network that are able to retrieve the data), energy-efficiency, complexity, scalability, and latency. Based on the application, some factors might be more important than others. For example, for updating the software in all the nodes in the network, reliability is very important, while latency might have less importance. Broadcasting streaming media is a case where latency is of paramount importance. Energy is usually an important issue especially for battery-powered sensor networks.

In this study, we consider the case that a large number of packets have to be broadcast in a multihop wireless sensor network with our main concerns being *reliability* and *energy-efficiency*. We are also interested in broadcasting schemes that have

low complexity of implementation and are distributed and practical.

The most straightforward way to perform broadcasting is *flooding* [20]. In the flooding method, a node rebroadcasts a packet that it receives for the first time. In a connected and lossless network, flooding guarantees reliability. Although flooding is simple and scalable, it has the following disadvantage. Many redundant rebroadcasts occur especially in dense networks, which over-consume the precious network resources of energy and bandwidth. This problem is known as the broadcast storm problem [42].

The problem of *reliable* and *energy-efficient* broadcasting in wireless networks has different solutions in the two following models.

- In the first model, the nodes have only relaying capability. In this case, reliable and energy-efficient broadcasting in a wireless network is equivalent to the problem of finding a *minimum-connected dominating-set* (MCDS) for the corresponding network graph, if we model the network as a geometric graph. Unfortunately, determining an MCDS is an NP-complete problem [15] even if a centralized algorithm utilizing the full knowledge of the graph topology is applied. This forces the employment of some heuristic and suboptimal schemes. One of the important schemes is called *probabilistic broadcast* (PBcast) [18, 42]. This approach was originally introduced in [7] to reduce traffic for multicast wired networks and later was tailored for wireless applications. In PBcast, a source broadcasts all the packets with probability one. Any other node rebroadcasts every packet that it receives for the first time with some probability $p < 1$. Therefore, the number of unnecessary rebroadcasts is decreased. However, some nodes may not receive all the packets because of disconnectivity caused by the probabilistic relaying. A high value for p may be chosen to achieve reliability; however, if p is too high, energy efficiency will be lost. Some other heuristic algorithms for attacking this problem have been proposed,

e.g., [18, 26, 42, 45, 51, 70, 74]. Most of them assume considerable knowledge of network topology and are either impractical or suffer from lack of reliability. Note that in all the above schemes, to ensure reliability one, every single packet needs to be received by all the nodes in the network. This constraint may cause lots of retransmissions and may require in-sequence data delivery. Moreover, these schemes would be far less efficient for lossy networks (i.e., networks with unreliable communication links).

- In the second model, in addition to relaying, each node has the capability of doing local processing and coding. This model was first introduced in [2] and opened a new research path known as network coding.

We illustrate the difference between the optimal solutions for these two models in a small example. Assume that we want to broadcast two symbols x and y from a source S to all the nodes in the wireless network shown in Figure 2.4

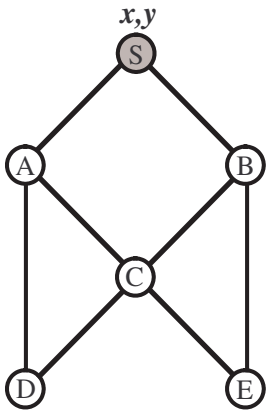


Figure 2.4: Two symbols x and y have to reach all the nodes in this wireless network with minimum number of transmissions.

The optimal solutions for the first and second model are shown in Figures 2.5(a) and 2.5(b), respectively. In the former model, nodes S, A, and B forward x and y. The total number of transmissions is six, and this is an optimal solution. In the second model, node S forwards x and y. Node A forwards only x, and node B forwards only y. Node C collects x and y and forwards bitwise XOR $x + y$. Clearly, node D can

recover x and y by receiving x and $x+y$. Similarly, node E can recover x and y by receiving y and $x+y$. Hence, $x+y$ is beneficial to both nodes D and E. In this case, the optimal broadcasting happens with only five transmissions. We conclude that local coding in the network can reduce the number of transmissions and offers in a better energy efficiency.

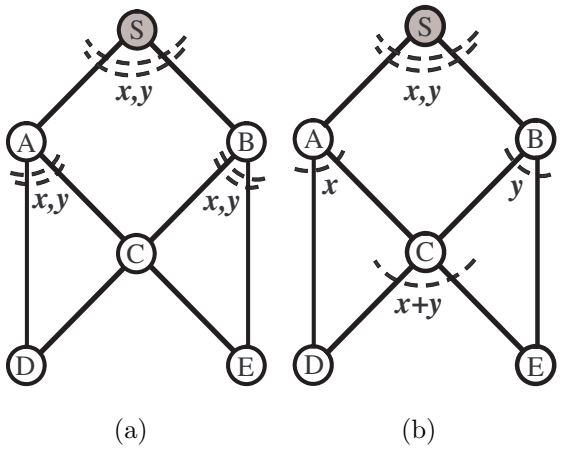


Figure 2.5: Optimal schemes for broadcasting x and y from source S to all the nodes in the network. (a) Nodes only have relaying capability. Minimum number of required transmissions is 6. (b) Nodes have relaying and coding abilities. Minimum number of required transmissions is 5.

Considerable work has been done in the area of coding within networks including [32, 34] and references therein. Next, we briefly review the approach that is well known as *Network Coding* in the community.

2.5.1 Network Coding

The problem of minimum-cost multicast/braodcast in wireless networks can be addressed by *Network Coding* (NC) approach [33]. In NC, a network is modeled with a directed hypergraph $\mathcal{H} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of hyperarcs. A hypergraph is a generalization of a graph, where we have hyperarcs instead of arcs (edges). A hyperarc is a pair (i, J), where $i \in \mathcal{N}$ is the start point and J is the set of end points and is a non-empty subset of $\mathcal{N}$ [33]. Each hyperarc (i, J)

represents a lossless broadcast link[1] from node i to nodes in the set J. The source node and the set of destinations nodes are denoted by s, and T, respectively. For the general multicast problem, T could be any subset of $\mathcal{N}$. For the special case of broadcasting, we have $T = \mathcal{N} \backslash s$. By z_{iJ} we denote the rate at which coded packets are injected and received on hyperarc (i, J).

NC can be divided into two decoupled problems. The first one is to determine the subgraph over which coding has to be performed and the flow rate on each link (z_{iJ}) such that multicast/broadcast cost is minimum. The other problem is to determine the code to use over that subgraph.

In the first problem, to achieve a minimum-cost multicast/broadcast of rate arbitrarily close to R, we need to solve the following optimization problem.

$$minimize\ f(z) \tag{9}$$

subject to

$$z_{iJ} \geq \sum_{j \in J} x_{iJj}^{(t)}, \qquad \forall (i, J) \in \mathcal{A}, t \in T,$$

$$\sum_{\{J|(i,J)\in\mathcal{A}\}} \sum_{j \in J} x_{iJj}^{(t)} - \sum_{\{j|(j,I)\in\mathcal{A}, i\in I\}} x_{jIi}^{(t)} = \sigma_i^{(t)}, \qquad \forall i \in \mathcal{N}, t \in T \tag{10}$$

$$x_{iJj}^{(t)} \geq 0, \qquad \forall (i, J) \in \mathcal{A}, j \in J, t \in T. \tag{11}$$

where,

$$\sigma_i^{(t)} = \begin{cases} R, & \text{if i=s,} \\ -R, & \text{if i=t,} \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

An example for the cost function $f(z)$ could be $\sum_{(i,J)\in\mathcal{A}} z_{iJ}$. In this case, the problem is a linear programming (LP) problem and can be solved in polynomial time.

The second problem (coding problem) is solved as follows. Whenever a node has to inject a packet, it sends a random linear combination of the packets it has in its

[1] A generalized model that considers lossy broadcast links is also studied in [33].

memory. The coefficients of the linear combinations are chosen independently and uniformly over all elements of a finite field $GF(q)$. We will refer to this type of coding as *random sum coding*. Clearly, information about the chosen coefficients must also be sent along with the sent packet for the decoding purpose.

We note the following shortcomings with NC.

- The assumption of a directed graph is a limiting one since wireless networks consisting of nodes with omnidirectional antennas are, by their nature, not directed. This means that for any two nodes i and j that are in the transmission range of each other, the transmission can happen in both directions from i to j or from j to i in different time slots. The issue of finding optimal directions for the edges is an intractable problem by itself considering the fact that the number of possible combinations of assignable directions grows exponentially with the number of edges in the network.
- The size of the finite field $GF(q)$ from which the coefficients of linear combinations are selected must be very large for optimality. This makes the computations costly.
- The overhead for network coding (due to the transmission of coefficients with each packet) is $n_p \log_2 q$ bits for each sent packet, where n_p is the number of original packets. This overhead might be prohibitive if $n_p \log_2 q$ is comparable to the size of the packets.
- The decoding is required to be Gaussian elimination with cubic complexity with respect to n_p.
- The computational complexity of NC for broadcasting over a lossy wireless network of N nodes and average node degree J is prohibitive for large networks since it involves an expensive optimization with over $N^2 2^J$ constraints and variables [33].

Instead of using random sum codes that have cubic complexity of decoding, other

coding techniques can also be employed at each node. One of the best options is *rateless (Fountain) erasure coding* [31, 39, 68]. Rateless codes have linear encoding and decoding complexity. The encoding is a low-weight packet-level addition of input packets over $GF(2)$, and the decoding is done by a simple iterative decoding as we explained in Section 2.3.1. Unlike traditional codes, rateless codes do not assume any knowledge of the channel and are adaptable to different channel conditions. In [10,11, 41, 68], the applicability of rateless codes for reliable multicast/broadcast in *single-hop* lossy networks was mentioned. The original packets are first encoded using a rateless code. The encoded packets are then broadcast. In this case, the redundancy is optimal for all clients independent of their packet loss rates. No prior knowledge of the channel status is needed. However, the performance of broadcasting encoded data over *multihop* wireless networks depends on the routing scheme as well. One option is to find the optimal sub-network as in the case of NC (using LP) and then use rateless coding over the sub-network. However, the problem of finding routes using LP is not very practical for large networks such as sensor networks.

CHAPTER III

UNEQUAL ERROR PROTECTION USING LOW-DENSITY PARITY-CHECK CODES

3.1 Introduction

Unequal Error Protection (UEP) property is very desirable for applications where different bits have different significance. The first UEP codes were proposed by Masnick and Wolf [38]. Later, other UEP design methodologies were developed, e.g., [8, 9, 21, 22, 36] based on different coding schemes such as cyclic codes and convolutional codes. After the rediscovery of LDPC codes and the invention of rateless codes, many researchers started studying them from different aspects. These codes show outstanding performance and low complexity of implementation. However, there was not much work on studying these codes for unequal error protection. In the next two chapters, we investigate this problem.

In this chapter, we propose two schemes to construct efficient LDPC codes that provide UEP [49, 52, 57, 58]. The first scheme is based on traditional bipartite Tanner graphs, and the second scheme is a novel approach based on combining two Tanner graphs resulting a 3-partite ensemble. We derive density evolution formulas for both the proposed unequal error protection LDPC ensembles over the binary erasure channel. Using the density evolution formulas, we can optimize the codes based on the requirements of our problem. We compare our schemes with some other LDPC codes, the time-sharing method, and a previous work on UEP using LDPC codes [73]. Simulation results confirm the superiority of the proposed design methodologies for UEP. We also compare the two proposed schemes. It is shown that by employing the scheme based on combined Tanner graphs we can achieve improved performance over

the first proposed scheme. Specifically, error floor decreases for more important parts of data in the second proposed scheme.

3.2 UEP Using LDPC Codes

Up until now different schemes for designing capacity achieving (CA) LDPC codes over the BEC have been devised, e.g., [43]. These schemes are based on designing codes of rate R with the threshold channel erasure probability (ε^{th}) as close as possible to $1-R$. When the channel erasure probability is less than ε^{th}, the average bit error rate (BER) (the probability that a bit is not recovered after the decoding stops) goes to zero when long enough code lengths and large enough number of decoding iterations are considered. Therefore, CA codes are superior to the UEP codes asymptotically as they provide small enough error rates for all data. However, short- to moderate-length codes are preferable in practice. For these lengths, UEP codes are desirable. In the proposed UEP designs, we neither optimize the codes based on ε^{th} nor use the average BER of all data in our analysis. Instead, we divide the codeword into different groups and investigate the average BER for each group. The codes are optimized such that some information bits have lower BER than the other bits.

Throughout this chapter, we are only concerned with the performance of *information bits*, thus UEP for information bits is considered. Therefore, we need to determine the positions of the information bits in the codeword. For an (n,k) LDPC code that is defined by a parity-check matrix H, not every arbitrary collection of k bits in the codeword (correspondingly k columns of H) can correspond to the information bits. The following should be satisfied by H.

Lemma 3.1. *Let $(n-k)\times n$ matrix $H=[h_1h_2\dots h_n]$ be the parity-check matrix corresponding to an (n,k) linear code. To have $[i_1,i_2,\dots,i_k]$ as the positions of the information bits in the codeword, matrix $H_P = H \setminus H_I$ must be full rank, where $H_I=[h_{i_1}h_{i_2}\dots h_{i_k}]$.*

Proof. Let us define $X = (x_1, x_2, \ldots, x_n)$, $X_I = (x_{i_1}, \ldots, x_{i_k})$, and $X_P = X \setminus X_I$. Then, X is a valid codeword if and only if $H_I X_I^T + H_P X_P^T = 0^T$. Using this equation, we can find parity bits X_P as a function of information bits X_I if and only if H_P is full rank. □

In this work, the information bits are divided into two groups with two levels of importance. One group consists of the *more important bits* (MIB) that need higher protection. The other group contains the *less important bits* (LIB). More specifically, the following problem is studied.

3.3 Problem Statement

Suppose we want to transmit k information bits with two levels of importance over an erasure channel with erasure probability ε. To do this, we want to design an (n, k) UEP code $\mathcal{C}$ having rate $R = k/n$. Let $k_M = \alpha k$ (where $0 < \alpha < 1$) be the number of MIB and $k_L = (1 - \alpha)k$ be the number of LIB as in Figure 3.1. Let $m = n - k$ be the number of parity bits (PB).

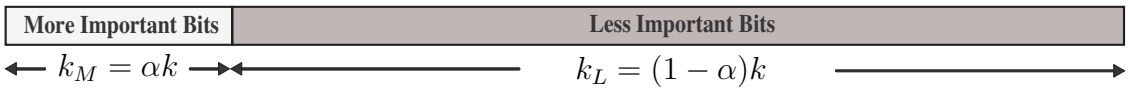


Figure 3.1: Information is divided into two parts. A fraction α of the data are more important bits, and a fraction $1 - \alpha$ of the data are less important bits.

It should be noted that although we consider that information bits have two levels of importance, the generalization of the proposed schemes for the cases with more levels of importance is straightforward.

In this chapter, we first study the design of UEP-LDPC codes based on traditional bipartite Tanner graphs [52, 58]. Then, we develop the second type of UEP-LDPC codes that are constructed based on combining two Tanner graphs [57].

3.4 UEP-LDPC Codes: Traditional Tanner Graphs

Before explaining our design criteria, it is good to provide some insight into how an LDPC code can have different error protection levels for different bits. It is known that it is best to have high degrees for variable nodes. This is because the more information a variable node receives from its adjacent check nodes, the more accurately it can judge about its correct value. In contrast, from the point of view of a check node, it is best to have a low degree, since the lower the degree of a check node, the more valuable the information it can transmit back to its neighbors [29]. Assuming a fixed number of edges in the graph, increasing degrees of some variable nodes results in decreasing degrees of some other variable nodes, and this provides UEP.

Next, a method for providing UEP is proposed. In this method, we consider the conventional bipartite Tanner graph with n variable nodes and m check nodes. For the simplicity of design, we assume to have partially regular ensembles. By partially regular, we mean that all the MIB, LIB, and PB have the same degrees d_M, d_L, and d_p, respectively. Further, all check nodes have the same degree d_c. Figure 3.2 shows the Tanner graph of this ensemble. Let $H = [H_M|H_L|H_p]$ denote the corresponding parity-check matrix of this graph, where H_M, H_L, and H_p are submatrices that correspond to the MIB, LIB, and PB, respectively. By Lemma 3.1, we conclude that the assumption of separating information bits and parity bits as specified above is valid if and only if H_p is full rank. Next, we derive density evolution formulas for the proposed partially regular ensemble.

3.4.1 UEP Density Evolution

Let us consider the standard iterative decoding algorithm for the BEC. To achieve UEP with a significant gap among the different protection levels, we modify the density evolution formulas introduced in [30]. In our formulation, three parameters M_i, L_i, and p_i are introduced. These parameters denote the expected fractions of

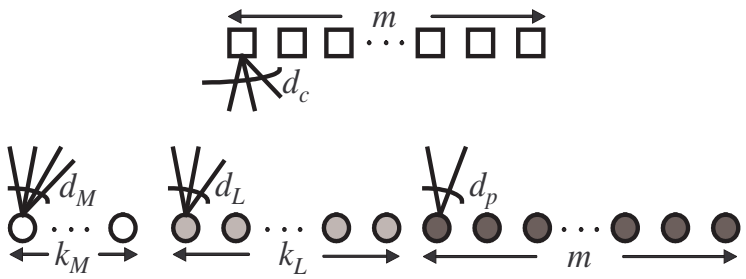


Figure 3.2: The Tanner graph of the proposed ensemble for providing UEP.

the erasure messages at the i^{th} iteration that are passed from the variable nodes that correspond to the MIB, LIB, and PB, respectively. Furthermore, let q_i denote the probability that an erasure message is passed from the check nodes to the variable nodes at the i^{th} iteration. Then, the unequal density evolution (UDE) formulas are given as

$$M_0 = L_0 = p_0 = \varepsilon, \tag{13}$$

$$M_{i+1} = M_0 q_i^{d_M - 1}, \quad L_{i+1} = L_0 q_i^{d_L - 1}, \quad p_{i+1} = p_0 q_i^{d_p - 1}, \tag{14}$$

$$q_i = 1 - (1 - \lambda_{d_M} M_i - \lambda_{d_L} L_i - \lambda_{d_p} p_i)^{d_c - 1}, \tag{15}$$

where λ_{d_M}, λ_{d_L}, and λ_{d_p} are the fractions of the edges that are connected to the MIB, LIB, and PB, respectively. These parameters can be obtained by

$$\lambda_{d_M} = \frac{\alpha R d_M}{\alpha R d_M + (1-\alpha) R d_L + (1-R) d_p},$$
$$\lambda_{d_L} = \frac{(1-\alpha) R d_L}{\alpha R d_M + (1-\alpha) R d_L + (1-R) d_p},$$
$$\lambda_{d_p} = \frac{(1-R) d_p}{\alpha R d_M + (1-\alpha) R d_L + (1-R) d_p}.$$

The following lemma points out the UEP property of the proposed ensemble.

Lemma 3.2. *Let ε be the erasure probability of a BEC and $\beta_{i,\varepsilon} \triangleq \frac{L_{i+1,\varepsilon}}{M_{i+1,\varepsilon}}$ be the UEP gain at the i^{th} decoding iteration. Then, $\beta_{i,\varepsilon}$ increases when the erasure probability of the channel, ε, decreases. Moreover, $\beta_{i,\varepsilon}$ is an increasing function of the number of iterations i.*

Proof. Using (14), we have $\beta_{i,\varepsilon} = (\frac{1}{q_{i,\varepsilon}})^{d_M - d_L}$. Since $d_M > d_L$, we need to show that $q_{i,\varepsilon}$ is an increasing function of ε. This can be proven by induction. Assume $\varepsilon_2 > \varepsilon_1$. From (15) we have $q_{0,\varepsilon} = 1 - (1-\varepsilon)^{d_c - 1}$. This implies that $q_{0,\varepsilon_2} > q_{0,\varepsilon_1}$. Now we assume that $q_{i-1,\varepsilon_2} > q_{i-1,\varepsilon_1}$. From (14) we have $M_{i,\varepsilon_2} > M_{i,\varepsilon_1}$, $L_{i,\varepsilon_2} > L_{i,\varepsilon_1}$, and $p_{i,\varepsilon_2} > p_{i,\varepsilon_1}$. Using (15) we conclude that $q_{i,\varepsilon_2} > q_{i,\varepsilon_1}$. This proves the first part of the lemma.

To prove the second part, we must show that $q_{i,\varepsilon}$ is a decreasing function of i. This can be done by induction on i. First, note that $q_{0,\varepsilon} < 1$. From (15), we have $q_{1,\varepsilon} < q_{0,\varepsilon}$. Now assume that $q_{i,\varepsilon} < q_{i-1,\varepsilon} < \ldots < q_{1,\varepsilon} < q_{0,\varepsilon} < 1$. Let

$$f(q_{i,\varepsilon}) \triangleq \varepsilon \lambda_{d_M} q_{i,\varepsilon}^{d_M - 1} + \varepsilon \lambda_{d_L} q_{i,\varepsilon}^{d_L - 1} + \varepsilon \lambda_{d_p} q_{i,\varepsilon}^{d_p - 1}. \tag{16}$$

We have $f(q_{i,\varepsilon}) < 1$ and

$$\begin{aligned} q_{i+1,\varepsilon} - q_{i,\varepsilon} &= (1 - f(q_{i-1,\varepsilon}))^{d_c - 1} - (1 - f(q_{i,\varepsilon}))^{d_c - 1} \\ &= (f(q_{i,\varepsilon}) - f(q_{i-1,\varepsilon})) \times K \end{aligned}$$

in which $K > 0$. Using (16), the value of $f(q_{i,\varepsilon}) - f(q_{i-1,\varepsilon})$ can be seen to be negative since $q_{i,\varepsilon} < q_{i-1,\varepsilon}$. Therefore, we have $q_{i+1,\varepsilon} < q_{i,\varepsilon}$. This completes the proof. □

Using the UDE formulas, the asymptotic behavior of a code for a given degree distribution can be estimated[1]. Moreover, we can optimize the degrees such that we have low error rates for MIB while keeping the overall performance comparable with other codes. For a given R and α, we need to find optimal values for d_M, d_L, d_p, and d_c. However, we have one equality constraint that is imposed by the edges as

$$\alpha R d_M + (1-\alpha) R d_L = (1-R)(d_c - d_p). \tag{17}$$

Therefore, we have three independent variables to optimize. We considered d_c as the dependent variable. By assuming a maximum value for the degrees (d_{max}) and

[1]An alternative way to obtain the performance of a code over the BEC is by determining the stopping sets characteristics. Such an approach is more complicated especially for the UEP case. However, the results of two approaches will be consistent asymptotically.

considering $d_M > d_L > d_p$, we can easily search through all the possible values for the degrees and select the ones that result in very low error rates for MIB. The cost function is considered as M_I (for some large integer I).

3.4.2 Simulation Results

Assume we want to design a UEP code with $\alpha = 0.1$ and rate 1/2. By setting $d_{max} = 25$, $\varepsilon = 0.45$[2], and $I = 25$, we minimized the cost function M_I. Table 3.1 shows the degrees, M_I, L_I, and p_I for two optimized codes.

Table 3.1: Degree distributions, M_{25}, L_{25}, and p_{25} of some optimized UEP-LDPC codes of rate 1/2 and $\alpha = 0.1$ found by the proposed method.

Code	d_M	d_L	d_p	d_c	M_{25}	L_{25}	p_{25}
1	23	3	2	7	2.18e-6	1.48e-1	2.58e-1
2	24	4	2	8	2.31e-12	1.52e-2	1.45e-1

As is shown in the table, asymptotically, the performance gaps between the BERs of MIB and the rest of the codeword bits are several orders of magnitude for 25 decoding iterations. Increasing the number of iterations results in even larger gaps.

To measure the performance of the proposed codes for the finite-length case, we found the BER versus ε for Code 1 ($\varepsilon^{th} = 0.455$) when the length of the code is $n = 4000$ [Figure 3.3(a)]. Two other codes were chosen for comparison with our code: the regular $(3, 6)$ ($\varepsilon^{th} = 0.429$) and a BEC-optimized irregular code, referred to as Code A, found from [1] by setting the maximum allowable degree to 25. The degree distribution of Code A[3]is given by $\lambda(x) = 0.249765x + 0.247164x^2 + 0.148003x^5 + 0.0033269x^6 + 0.351741x^{19}$ and $\rho(x) = x^7$ with $\varepsilon^{th} = 0.489$. To have a fair comparison, we showed the performance of $k_M = 200$ highest-degree nodes (as MIB) and the rest

[2]If we optimize a code for a large value of ε, asymptotically, the code will have a good performance for large $\varepsilon's$. On the other hand, if we optimize a code for a small value of ε, asymptotically, the code will have a good performance in the error floor region.

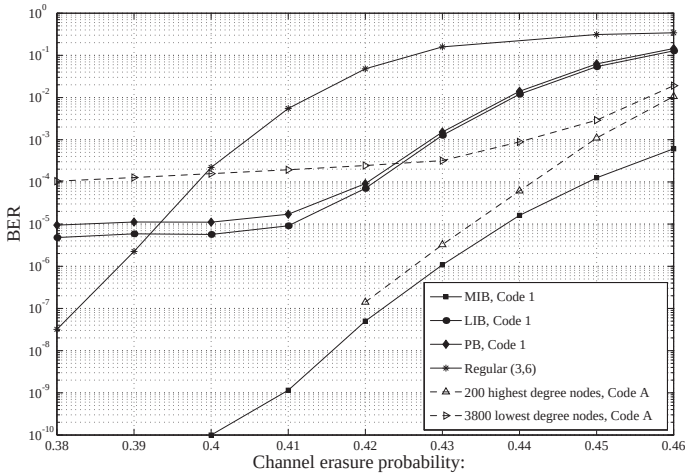
[3]We need to make a subtle change to the distribution of finite-length codes. For example, we used $\lambda(x) = 0.249625x + 0.2475x^2 + 0.148125x^5 + 0.0035x^6 + 0.35125x^{19}$ and $\lambda(x) = 0.249x + 0.2475x^2 + 0.15x^5 + 0.0035x^6 + 0.35x^{19}$, for $n = 4000$ and $n = 1000$, respectively. In both cases $\varepsilon^{th} = 0.489$.

of the nodes (LIB and PB) separately for Code A. As we can see in Figure 3.3(a), there is a large gap between the BERs of the MIB and LIB in the proposed code[4]. This gap is at least two orders of magnitude, and it increases when the channel erasure probability decreases as in Lemma 3.2 for the asymptotic case. Moreover, the performance of the MIB in the proposed code is always better than the performance of the MIB in the two other codes. In addition, the error floor in the LIB and PB are lower in the proposed code in comparison with Code A. We also note that the performance of the proposed code is far better than the performance of the regular $(3, 6)$ for $\varepsilon > 0.3921$. For smaller $\varepsilon's$, the performance of the regular $(3, 6)$ beats the performance of LIB in the proposed method. This is because of the well known result that the regular $(3, 6)$ does not show an error floor unlike the irregular codes. However, we note that the performance of MIB in the proposed code is superior to the performance of the regular $(3, 6)$.

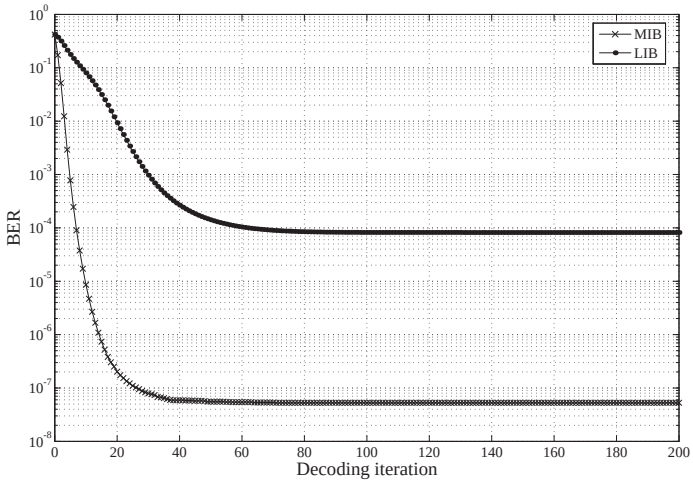
It is worth noting that not only will MIB be retrieved with much less error than LIB, but also MIB converges in fewer decoding iterations than LIB. This can be seen in Figure 3.3(b) for Code 1 at $\varepsilon = 0.42$. This is useful when fast decoding for MIB is needed.

We also illustrated the performance of Code 1 when $n = 1000$ (Figure 3.4). Again, we compared the proposed code with the regular $(3, 6)$ and Code A of lengths 1000. As we can see, the proposed code is superior to the regular $(3, 6)$ in the shown range. Moreover, although the performances of LIB and PB are close in Code 1 and Code A, the performance of MIB is far better than the performance of the 50 highest-degree nodes in Code A.

[4]The BER for MIB is found by averaging over the fraction of the bits in MIB that has not been recovered when decoding stops. Similarly, BERs of LIB and PB can be obtained.



(a)



(b)

Figure 3.3: (a) Comparison of the bit error rates of Code 1 with Code A and the regular $(3, 6)$. All codes are of length 4000 and rate 1/2. (b) Recovery convergence rate of MIB and LIB in Code 1 at $\varepsilon = 0.42$.

3.4.3 Efficient Encoding

As we can see in Table 3.1, the degree-distribution optimization has resulted in $d_p = 2$. We also observed the same result for most of our other UEP code designs. In fact, we exploit this property of the parity nodes to simplify the encoding of the proposed codes as follows. Since $d_p = 2$, all columns of H_p have weight two. However, given that H_p is $m \times m$ and full rank by Lemma 3.1, no more than $m-1$ columns of weight two are allowed. To overcome this problem we use the method proposed in [76]. One

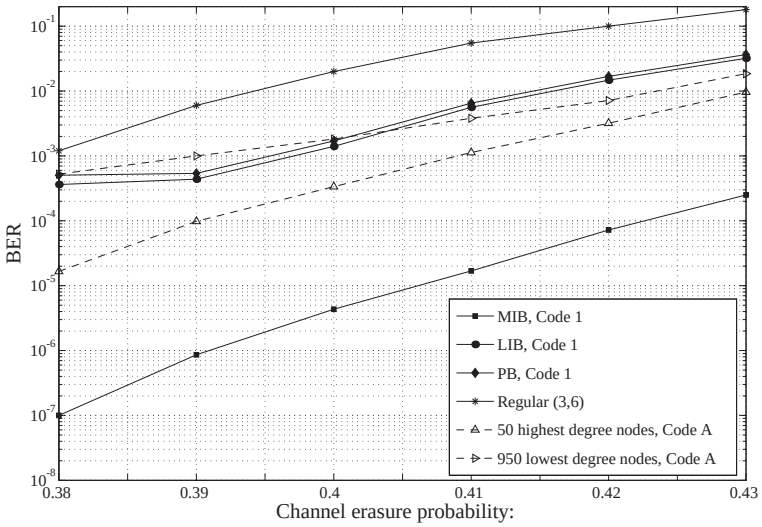


Figure 3.4: Comparison of the bit error rates of Code 1 with Code A and the regular (3,6). All codes are of length 1000 and rate 1/2.

of the weight-two columns is replaced with a weight-one column. This does not have an important effect on the performance of the code but ensures that H_p is full rank. It was shown in [76] that H_p can be either a dual-diagonal matrix Q or a column permutation of Q. In other words, $H_p = Q\Pi$, where Π is a random permutation matrix. An $m \times m$ dual-diagonal matrix Q is defined as :

$$Q = \begin{pmatrix} 1 & & & & & & \\ 1 & 1 & & & & & \\ & 1 & 1 & & & & \\ & & & \cdots & & & \\ & & & & 1 & 1 & \\ & & & & & 1 & 1 \end{pmatrix} \tag{18}$$

A systematic generator matrix for the parity-check matrix $H = [H_M|H_L|Q\Pi]$ is given by $G = [I|H_I^T Q^{-T}\Pi]$ in which $H_I = [H_M|H_L]$. It can be easily verified that

$GH^T = 0$. It can also be verified that Q^{-T} has the following form

$$Q^{-T} = \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & \cdots & 1 \\ & 1 & 1 & 1 & \cdots & \cdots & 1 \\ & & \ddots & & & & \vdots \\ & & & & 1 & 1 & 1 \\ & & & & & 1 & 1 \\ & & & & & & 1 \end{pmatrix} \tag{19}$$

which is precisely the transformation matrix corresponding to a differential encoder whose transfer function is $\frac{1}{1 \oplus D}$ [76]. The transfer function $\frac{1}{1 \oplus D}$ is equivalent to the feedback system depicted in Figure 3.5, in which D represents one unit delay.

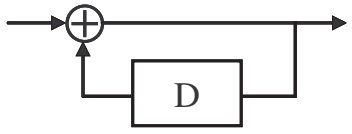


Figure 3.5: Linear feedback system equivalent to $\frac{1}{1 \oplus D}$.

The encoder for these codes is depicted in Figure 3.6. Thus, these codes are systematic and are a generalized form of the Repeat Accumulate (RA) codes (for which Π is equal to the identity matrix).

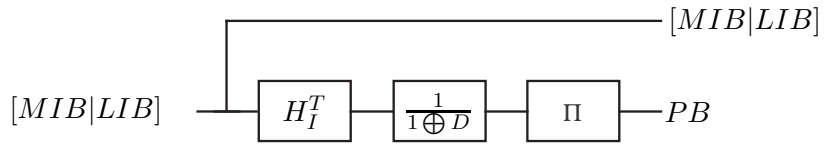
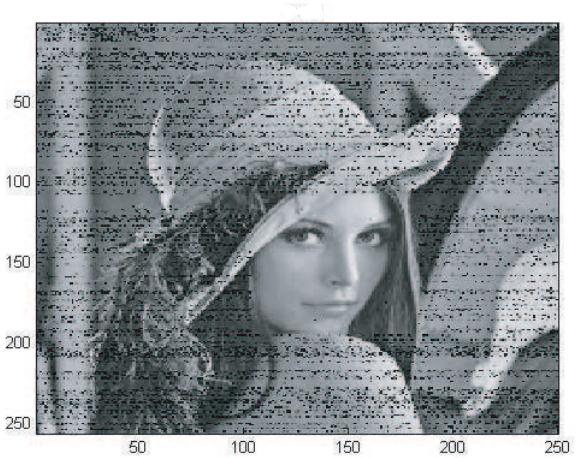


Figure 3.6: Efficient encoding for the proposed UEP codes when $d_p = 2$.

Pictorial Example: As an experiment, Code 1 with rate $R = 1/2$ was used to provide UEP for the Lena image. Figure 3.7(a) depicts the Lena image before transmission. We encode the image employing the encoding scheme explained above. The transmission channel is an erasure channel with $\varepsilon = 0.45$. The face in the image is considered as MIB, and the rest is considered as LIB. Figure 3.7(b) shows the

reconstructed image at the receiver. Clearly, the MIB have been protected much better than the LIB.

(a)



(b)

Figure 3.7: (a) Original Lena image. (b) Reconstructed Lena image. The transmission is over an erasure channel with $\varepsilon = 0.45$. Code 1 with rate $R = 1/2$ was used as the channel encoder to provide UEP.

3.4.4 Comparison with the Time-Sharing Method

One approach to provide UEP is the time-sharing method. In this method, several codes of different rates are used for different parts of the data. This method increases the complexity of the system. Additionally, since the MIB is usually very short, the

code length would be short. We expect that this causes performance degradation. The following simulation confirms that the time-sharing technique does not perform as well as the proposed method. Suppose we want two levels of protection for a message whose α fraction is MIB. In the first method, a UEP code of rate R is considered. Alternatively, we can design two codes TS_M and TS_L with rates R_M and R_L for MIB and LIB, respectively. By fixing the total number of the parity bits in both methods, we get

$$\frac{\alpha}{R_M} + \frac{1-\alpha}{R_L} = \frac{1}{R}.$$

For a given R and α, we can have different pairs of R_M and R_L, which choosing the best pair can be done by trial and error. Figure 3.8 compares the performance of UEP Code 1 of length 4000, $R = 0.5$, and $\alpha = 0.1$ with the time-sharing method having $R_L = 0.52$ and $R_M = 0.37$. The codes that are used in the time-sharing

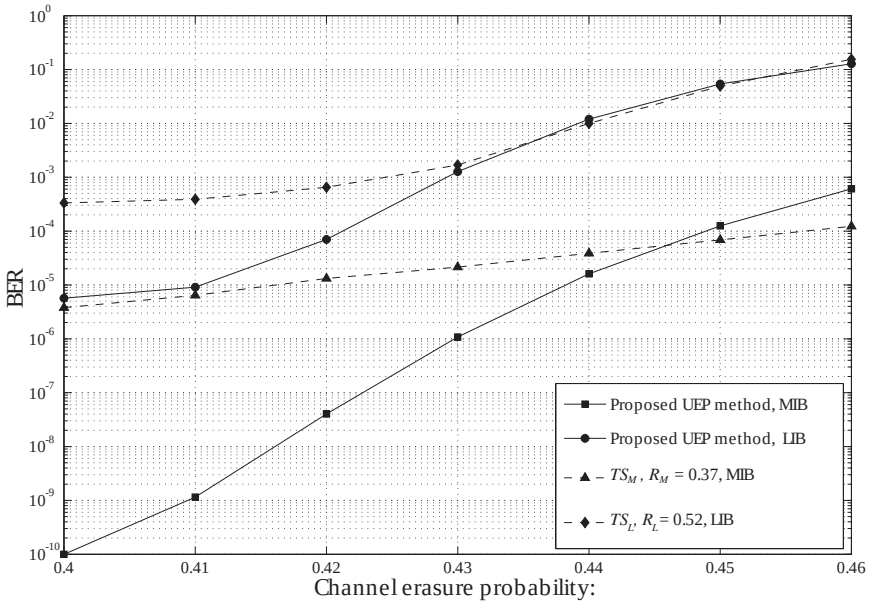


Figure 3.8: Comparison of the proposed UEP method with the time-sharing method.

method are the best codes that we could find from [1] for the given rates. They have

the following degree distributions:

$$\begin{aligned}
\lambda_{TS_L}(x) &= 0.299488x + 0.144426x^2 + 0.0159922x^3 + 0.165696x^4 \\
&\quad + 0.175288x^9 + 0.134321x^{24} + 0.0647887x^{25}, \\
\rho_{TS_L}(x) &= x^7, \\
\lambda_{TS_M}(x) &= 0.249839x + 0.128926x^2 + 0.113474x^4 + 0.0514865x^5 \\
&\quad + 0.0767592x^{10} + 0.0636197x^{11} + 0.142641x^{27} \\
&\quad + 0.174203x^{99}, \\
\rho_{TS_M}(x) &= 0.7x^6 + 0.3x^7.
\end{aligned}$$

with $\varepsilon_L^{th} = 0.474$ and $\varepsilon_M^{th} = 0.628$, respectively. To have better performance for MIB and LIB in the time-sharing method, we assume that MIB and LIB correspond to the higher degree variable nodes in TS_M and TS_L, respectively. Figure 3.8 indicates that the proposed UEP scheme outperforms the time-sharing scheme for $\varepsilon < 0.446$. For example, at $\varepsilon = 0.42$, the MIB (LIB) in Code 1 has more than two orders of magnitude (about one order of magnitude) less BER in comparison with the case that TS_M (TS_L) is used. Further, the superiority of the proposed method versus time-sharing increases when the channel erasure probability decreases.

3.4.5 Comparison with the Previous UEP-LDPC Codes based on CDFs

In [73], authors proposed UEP-LDPC codes constructed based on the orbits of cyclic difference families (CDFs). We note that the codes have very high protection for some codeword bits. This approach is desirable in the applications such as holographic memory systems where the noise has nonuniform pattern. Therefore, different protection levels for *codeword bits* are provided to achieve uniform BER after the decoding. In applications where UEP for *information bits* is needed, this approach may not be efficient. Specifically, it can be shown that the most highly protected codeword bits in [73] are not the information bits. This is because of the parity-check matrix structure that is used. As an example, a code of length $n = 553$ and $R \cong 0.57$

would have an H matrix of the form depicted in Figure 3.9, in which H_1 is a 316×79 submatrix, H_2 and H_3 are 158×79 and H_4, H_5, H_6, and H_7 are 79×79 submatrices. Note that all elements in the gray part of H are zeros. Moreover, the bits correspond

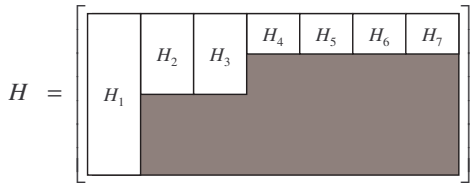


Figure 3.9: The structure of the parity-check matrix constructed using the CDF method.

to H_1 are the most protected bits. We claim that the codeword bits corresponding to H_1 are the parity bits. Otherwise, we must have $\text{rank}(H \backslash H_1) = n(1-R) = 237$, which is impossible. Therefore, the most highly protected bits are the parity bits. By a similar argument, it is shown that H_2 and H_3 cannot together correspond to the information bits. Therefore, a possible choice for information bits can correspond to H_2, H_4, and H_5. For comparison, we also give a code based on our proposed method having length $n = 555$, $R = 0.6$, $d_M = 23$, $d_L = 3$, $d_p = 2$, $d_c = 11$ and $\alpha \cong 0.15$. It should be mentioned that in this example the channel is AWGN. We used the same code that we designed using density evolution formulas over the BEC. Figure 3.10 shows the BER versus SNR for the information bits. Note that BER of the parity bits was not shown in this figure. The number over each graph represents the number of information bits in each part. It is concluded that although our proposed code has a slightly higher rate, it has much better performance than the code in [73].

3.5 UEP-LDPC Codes: Combined Tanner Graphs

Let us consider again the UEP problem mentioned in Section 3.3. Let $G(n, m)$ be defined as the Tanner graph corresponding to $\mathcal{C}$ with n variable nodes and m check nodes. Let H denote an $m \times n$ binary full-rank parity-check matrix corresponding to $G(n, m)$.

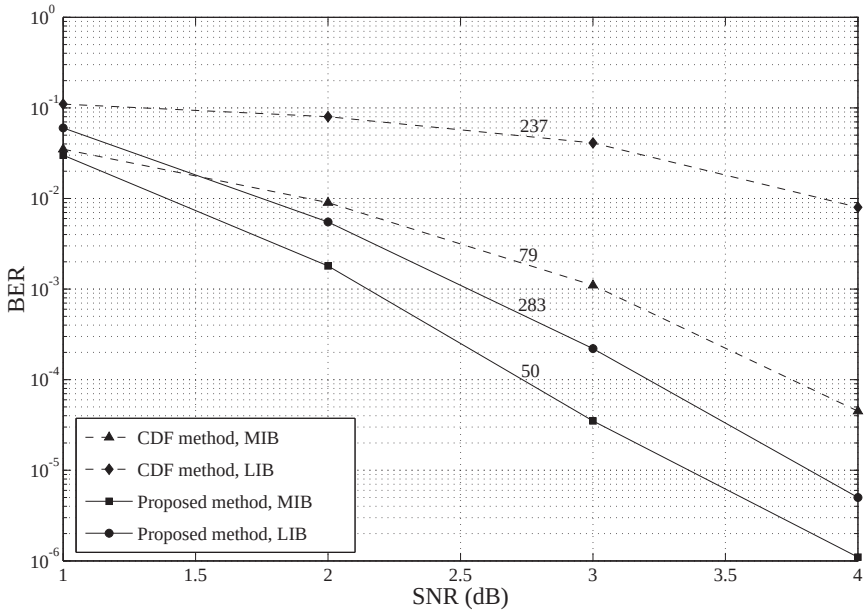


Figure 3.10: Comparison of the code designed by the CDF method and the proposed method.

The previous scheme on UEP-LDPC codes studied in Section 3.4 is based on having different degrees for MIB, LIB, and PB. To further reduce the error rates for the MIB, we propose another scheme. In this scheme, we combine two Tanner graphs. The first Tanner graph corresponds to a high-rate LDPC code that is for protecting MIB. The second graph is for protecting all the data. The first Tanner graph has the role of determining the values of those bits in MIB that the second graph failed to determine. Therefore, the error rate for MIB can be reduced. Let $G_1 = G(n_1, m_1)$ and $G_2 = G(n, m_2)$ denote the first and second graph, respectively. Here, $m_1 = \gamma m$ and $m_2 = (1-\gamma)m$ for some $0 < \gamma < 1$. The proposed ensemble is depicted in Figure 3.11. Let us call the proposed ensemble as G_c. The first n_1 variable nodes in G_c, are protected by both G_1 and G_2. It should be noted that not all of these n_1 bits can be taken as information bits. In the following lemma we prove that we have $n_1 - m_1$ information bits in this part of the codeword.

Lemma 3.3. *Consider two Tanner graphs $G_1 = G(n_1, m_1)$ and $G_2 = G(n, m_2)$ that are combined to form an ensemble as in Figure 3.11. Then, n_1 variable nodes that*

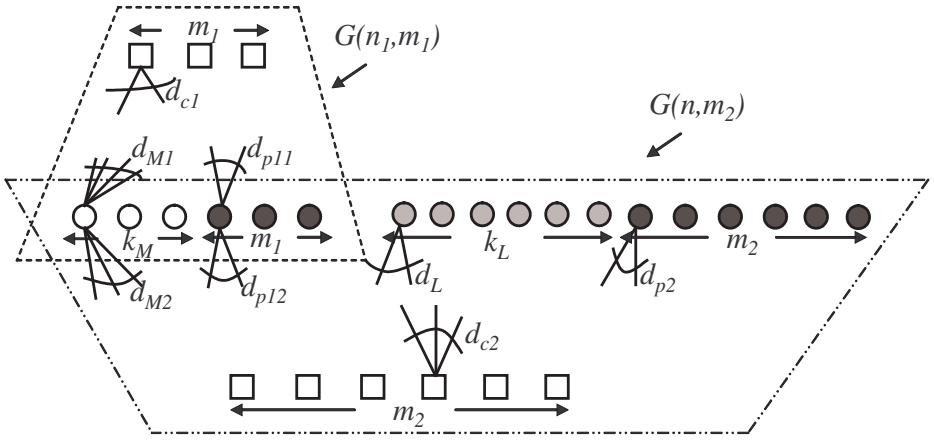


Figure 3.11: The Tanner graph of the proposed combined ensemble for providing UEP.

are common in both graphs contain $n_1 - m_1$ information bits.

Proof. Let H_1 and $H_2 = [H_{21}|H_{22}]$ denote $m_1 \times n_1$ and $m_2 \times n$ parity-check matrices corresponding to G_1 and G_2, respectively. It is easy to see that the parity-check matrix of the combined code is given by

$$H = \begin{pmatrix} H_1 & 0 \\ H_{21} & H_{22} \end{pmatrix}.$$

Since H is full rank, using algebra we can show that H_1 and H_{22} are also full rank. Since H_{22} is full rank, we conclude that all the first n_1 bits can potentially be information bits (their values can be set independently). However, since H_1 is also full rank with rank m_1, we conclude that only $n_1 - m_1$ bits of the first n_1 bits are information bits, and the values of the other m_1 bits are determined by the $n_1 - m_1$ information bits. This completes the proof. □

We consider all of these $n_1 - m_1$ bits as MIB, i.e., $k_M = n_1 - m_1$. To impose different protection levels for MIB and LIB, it is necessary to know the positions of MIB and LIB in G_c. The following lemma states the necessary and sufficient conditions for arranging MIB, LIB, and parity bits as in Figure 3.11. The corresponding codeword is in the form of $\underline{c} = [MIB|P_1|LIB|P_2]$, where the parity bits have been divided into

two parts P_1 and P_2.

Lemma 3.4. *Let $H_1 = [A|H_{p1}]$ and $H_2 = [B|C|E|H_{p2}]$ denote the parity-check matrices that correspond to G_1 and G_2, respectively. Here A, H_{p1}, B, C, E, and H_{p2} are matrices of size $m_1 \times k_M$, $m_1 \times m_1$, $m_2 \times k_M$, $m_2 \times m_1$,$m_2 \times k_L$, and $m_2 \times m_2$, respectively. The assumption of separating MIB, LIB, and PB as shown in Figure 3.11 is valid if and only if H_{p1} and H_{p2} are full rank.*

Proof. Let us define H_p as

$$H_p = \begin{pmatrix} H_{p_1} & 0 \\ C & H_{p_2} \end{pmatrix}.$$

The columns of H_p correspond to the PB if and only if H_p is full rank. This is possible if and only if H_{p_1} and H_{p_2} are full rank. □

Next, we derive density evolution formulas for the proposed ensemble.

3.5.1 UEP Density Evolution

Here, we derive the UDE formulas for the proposed ensemble. See Figure 3.11 for the definitions of d_{M1}, d_{M2}, d_{p11}, d_{p12}, d_L, d_{p2}, d_{c1}, and d_{c2}. Let $M_{1,i}$ and $p_{11,i}$ denote the expected fractions of erasure messages that are received by the check nodes in G_1 from the variable nodes that correspond to MIB and P_1, respectively. Let $M_{2,i}$, $p_{12,i}$, L_i, and $p_{2,i}$ denote the expected fractions of erasure messages that are received by the check nodes in G_2 from the variable nodes that correspond to MIB, P_1, LIB, and P_2, respectively. Let M_i and $p_{1,i}$ denote the expected fractions of erasure messages that are sent to an incident edge from the variable nodes that correspond to MIB and P_1, respectively. Let also q_i (r_i) denote the probability that an erasure message is passed from the check nodes to the variable nodes in G_1 (G_2). Note that subscript i is the iteration number. The UDE formulas for $i \geq 0$ are given by

$$M_{1,0} = M_{2,0} = L_0 = p_{11,0} = p_{12,0} = \epsilon,$$

$$M_{1,i+1} = \epsilon q_i^{d_{M1}-1} r_i^{d_{M2}}, \quad M_{2,i+1} = \epsilon r_i^{d_{M2}-1} q_i^{d_{M1}},$$

$$p_{11,i+1} = \epsilon q_i^{d_{p11}-1} r_i^{d_{p12}}, \quad p_{12,i+1} = \epsilon r_i^{d_{p12}-1} q_i^{d_{p11}},$$

$$L_{i+1} = \epsilon r_i^{d_L-1}, \qquad p_{2,i+1} = \epsilon r_i^{d_{p2}-1},$$

$$M_{i+1} = \frac{d_{M1} M_{1,i+1} + d_{M2} M_{2,i+1}}{d_{M1} + d_{M2}},$$

$$p_{1,i+1} = \frac{d_{p11} p_{11,i+1} + d_{p12} p_{12,i+1}}{d_{p11} + d_{p12}},$$

$$q_i = 1 - (1 - \lambda_{d_1} M_{1,i} - \lambda_{d_2} p_{11,i})^{d_{c1}-1},$$

$$r_i = 1 - (1 - \lambda_{d_3} M_{2,i} - \lambda_{d_4} p_{12,i} - \lambda_{d_5} L_i - \lambda_{d_6} p_{2,i})^{d_{c2}-1},$$

where λ_{d_1}, λ_{d_2} are the fractions of edges that are connected to the MIB and P_1 in G_1, respectively. Furthermore, λ_{d_3}, λ_{d_4}, λ_{d_5}, and λ_{d_6} are the fraction of edges that are connected to the MIB, P_1, LIB, and P_2 in G_2, respectively. These parameters are obtained by $\lambda_{d_1} = \frac{\alpha R d_{M1}}{T_1}$, $\lambda_{d_2} = \frac{\gamma(1-R) d_{p11}}{T_1}$, $\lambda_{d_3} = \frac{\alpha R d_{M2}}{T_2}$, $\lambda_{d_4} = \frac{\gamma(1-R) d_{p12}}{T_2}$, $\lambda_{d_5} = \frac{(1-\alpha) R d_L}{T_2}$, and $\lambda_{d_6} = \frac{(1-\gamma)(1-R) d_{p2}}{T_2}$, in which $T_1 = \alpha R d_{M1} + \gamma(1-R) d_{p11}$ and $T_2 = \alpha R d_{M2} + \gamma(1-R) d_{p12} + (1-\alpha) R d_L + (1-\gamma)(1-R) d_{p2}$.

Using the UDE formulas, the asymptotic behavior of a code with a given degree distribution can be estimated. Moreover, we can optimize the degrees so that we have low error rates for MIB while keeping the overall performance comparable to other codes. For a given R and α, optimal values for d_{M1}, d_{M2}, d_{p11}, d_{p12}, d_L, d_{p2}, d_{c1}, d_{c2}, and γ need to be found. However, we have two equality constraints imposed by edge constraints. These constraints are given by

$$\alpha R d_{M1} = \gamma(1-R)(d_{c1} - d_{p11}),$$

$$\alpha R d_{M2} + \gamma(1-R) d_{p12} + (1-\alpha) R d_L = (1-\gamma)(1-R)(d_{c2} - d_{p2}).$$

Therefore, we require to optimize seven independent variables. We considered d_{c2} and γ as dependent variables. By setting some upper bounds for the degrees, we can search through all the possible values for degrees and select the ones that result

in very low error rates for MIB. The cost function is considered as M_I (for some large integer I for which M_I is very close to its steady state value). It should be noted that the rate of the code corresponding to G_1 is given by $R_p = \frac{\alpha R}{\alpha R+\gamma(1-R)}$. For a fixed R and α, the larger is γ, the smaller are R_p and BER for MIB. On the other hand, we need to keep R_p large such that the performance of LIB remains acceptable. Therefore, we impose a lower bound on the rate R_p. Note that since the UDE formulas represent the asymptotic performance, every code obtained by the UDE formulas would not be necessarily optimal for finite-length codes. Therefore, we further refine the solutions for finite-length codes by choosing the one that has highest performance using iterative decoding.

3.5.2 Simulation Results

Consider the problem of designing a rate 1/2 UEP code with $\alpha = 0.1$. Let us assume the following search space: $d_{M1}, d_{M2}, d_L \leq 25$, $d_{p11}, d_{p12}, d_{p2} \leq 5$, $d_{c1}, d_{c2} \leq 15$, and $R_p \geq 0.8$. Using the UDE formulas, we optimize the codes. For example, we picked a code that results in $M_I = 0$ and $L_I = 7.9 \times 10^{-31}$ for $\epsilon = 0.45$ and $I = 1000$ iterations. This code also results in $M_I = 2.85 \times 10^{-26}$ and $L_I = 2.49 \times 10^{-12}$ for $\epsilon = 0.45225$. Table 3.2 summarizes the degrees for the optimized code. For the finite-length case,

Table 3.2: Degree distributions of the proposed rate 1/2 UEP-LDPC code.

d_{M1}	d_{M2}	d_{p11}	d_{p12}	d_L	d_{p2}	d_{c1}	d_{c2}	γ
1	22	2	2	3	2	9	7	0.0143

we found the BERs versus the channel erasure probability for this code when the code length is $n = 4000$ ($k_M = 200$, $k_L = 1800$, $m_1 = 28$, $m_2 = 1972$) and the maximum number of decoding iterations is 200. Figure 3.12 shows the performance of the proposed code compared to our previous code (Code 1) presented in Section 3.4.2 with $d_M = 23$, $d_L = 3$, $d_p = 2$, and $d_c = 7$. Figure 3.12 shows that the performance of MIB has improved by about one order of magnitude. On the other hand, the

performance of LIB has degraded slightly for large ϵ, although LIB does not show an error floor as opposed to Code 1. We also included the BERs for P_1 and P_2 in Figure 3.12. Although P_1 has a total degree that is much smaller than that of MIB, the BER performances of P_1 and MIB are close. This is because the only neighbors of the check nodes in G_1 are MIB and P_1. Hence, certain messages from MIB help P_1 to be determined.

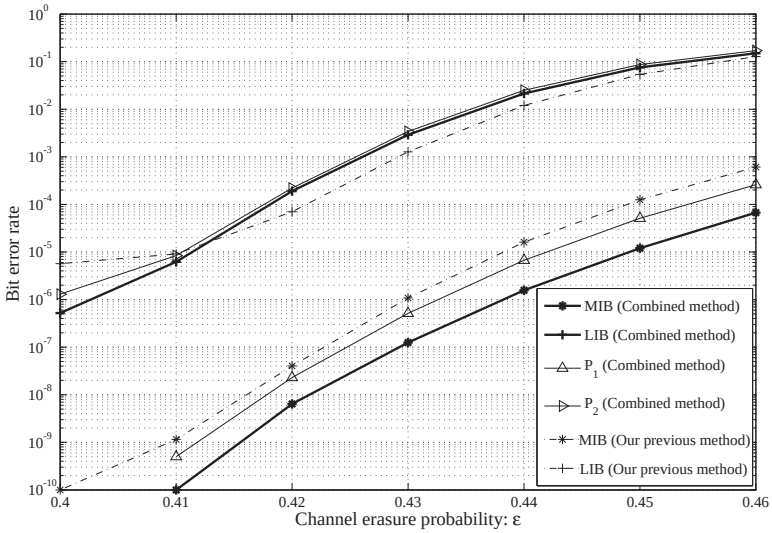


Figure 3.12: Comparison of the proposed method based on combined Tanner graphs and the method in Section 3.4. The codes are of length $n = 4000$, rate $1/2$, and $\alpha = 0.1$.

We also illustrated the performance of the proposed code when $n = 1000$ ($k_M = 50$, $k_L = 450$, $m_1 = 7$, $m_2 = 493$) in Figure 3.13. For comparison, we depicted the performance of Code 1 and a BEC-optimized irregular code, referred to as Code B, found from [1] by setting the maximum allowable degree to 25. The degree distribution for Code B is given by $\lambda(x) = 0.24976x + 0.24716x^2 + 0.148x^5 + 0.003326x^6 + 0.35174x^{19}$ and $\rho(x) = x^7$. We showed the performance of $k_M = 50$ highest degree nodes (as MIB) and rest of the nodes separately for Code B. We note that the performance of MIB in the proposed code is by far (three orders of magnitude for $\epsilon = 0.38$) better than the performance of MIB in Code B.

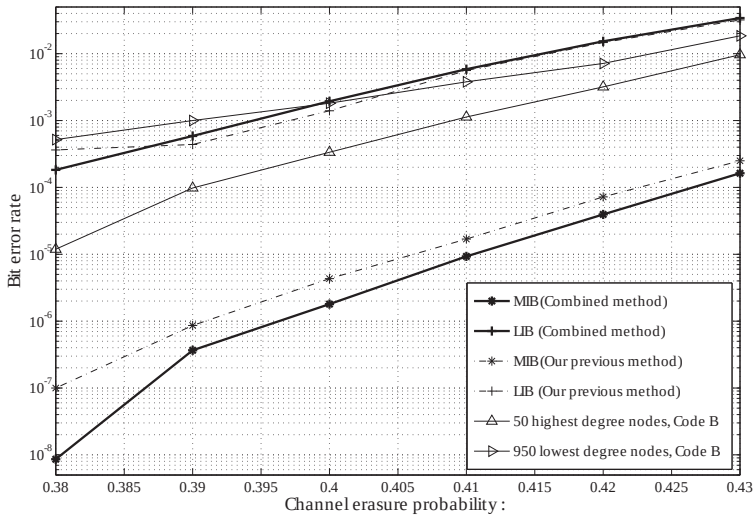


Figure 3.13: Comparison of the proposed method based on combined Tanner graphs, the method in Section 3.4, and Code B. All the codes are of length $n = 1000$ and rate $1/2$.

3.5.3 Efficient Encoding

Here, we present an efficient encoding scheme for the case that $d_{p11} = 2$ and $d_{p2} = 2$, which occurs in many optimized cases. We have H_{p1} and H_{p2} as full-rank matrices by Lemma 3.4. It can be seen easily that H_{p1} (H_{p2}) is either an $m_1 \times m_1$ ($m_2 \times m_2$) dual-diagonal matrix Q_1 (Q_2) or its column permutation. Let us say $H_{p1} = Q_1\Pi_1$ and $H_{p2} = Q_2\Pi_2$ for some random permutation matrices Π_1 and Π_2. A systematic generator matrix for the parity-check matrix

$$H = \begin{pmatrix} A & Q_1\Pi_1 & 0 & 0 \\ B & C & E & Q_2\Pi_2 \end{pmatrix},$$

is given by

$$G = \begin{pmatrix} I_{k_M \times k_M} & A^T Q_1^{-T}\Pi_1 & 0 & (B^T + A^T Q_1^{-T}\Pi_1 C^T)Q_2^{-T}\Pi_2 \\ 0 & 0 & I_{k_L \times k_L} & E^T Q_2^{-T}\Pi_2 \end{pmatrix}.$$

It can be easily verified that $GH^T = 0$. The matrix Q_1^{-T} (Q_2^{-T}) corresponds to a differential encoder whose transfer function is $\frac{1}{1 \oplus D}$ [76]. The encoder for these codes is depicted in Figure 3.14. We assumed that the information bits are $\underline{I} = [MIB|LIB]$,

and therefore the codeword bits are in the form of $\underline{c} = [MIB|P_1|LIB|P_2]$.

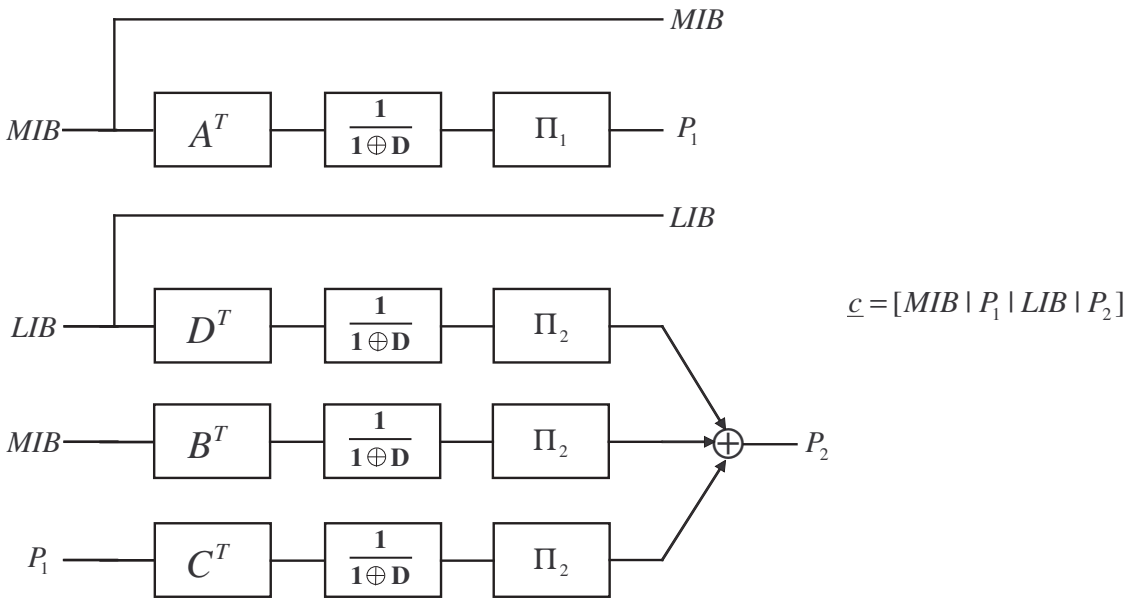


Figure 3.14: Efficient encoding for the proposed combined ensemble when $d_{p11} = d_{p2} = 2$.

3.6 Conclusion

In this chapter, we proposed two frameworks to design unequal error protection LDPC codes. We considered the cases in which information bits have two levels of importance, more important bits (MIB) and less important bits (LIB). We considered two different ensembles in our designs. The first ensemble is similar to the ensemble of traditional bipartite Tanner graphs. The second one is a combination of two bipartite graphs resulting in a 3-partite one. We assumed partially regular ensembles, which simplifies the optimization problems by reducing the number of variables (degrees of different variable nodes and check nodes). We derived density evolution formulas for the proposed schemes and optimized the codes based on them. The optimization problems can be solved easily even if we search through all feasible solutions, despite the design of conventional LDPC codes for which very complex degree-optimization algorithms are required.

We demonstrated the BER performance of different parts of the UEP-LDPC codes designed by our proposed schemes for several code lengths. Our results indicate that our proposed codes provide strong UEP with several orders of magnitude lower BER for MIB than LIB. We also compared our codes with other competitive codes, and simulation results confirmed the superiority of our schemes.

In our second proposed scheme, we have more degrees of freedom for designing UEP codes. This offers a trade off between the complexity of design and the performance. This scheme is very desirable, especially if very low error rates for MIB are required. It was shown in our examples that we can further decrease BER of MIB by using a UEP code designed by the combined Tanner graph scheme instead of a UEP code designed based on traditional Tanner graphs.

Finally, we proposed simple and efficient encoding schemes for special cases of the proposed UEP codes, i.e., for optimized codes having parity bits of degree two in each Tanner graph. Since these cases occur often for optimized codes, the proposed efficient encoding schemes would be desirable.

CHAPTER IV

UNEQUAL ERROR PROTECTION USING RATELESS CODES

4.1 Introduction

In Chapter 3, we proposed two design methodologies for providing unequal error protection of data based on LDPC codes. In this chapter, we develop UEP schemes that are based on *rateless (Fountain) codes*. Rateless codes are a new class of error-control coding schemes. LT codes [31], Raptor codes [68], and Online codes [39] are examples of such codes. It has been shown that these codes have very simple encoding and decoding algorithms. Asymptotically good degree distributions for them were also developed [68], [39]. Rateless codes on lossy channels do not assume any knowledge of the channel. Therefore, rateless codes are very suitable candidates for applications such as transmitting data on lossy multicast channels, nonuniform channels, and time-varying channels. In some of these applications, we may not have an estimate of the channel erasure probability at all times. In some others, different users may receive data that is passed through different channels. Traditional codes cannot be optimal for such cases because of the unknown or varying characteristics of the channels. In particular, rateless codes can fit very well for networking applications, such as wireless networks and the Internet. Because of many reasons such as buffer overflow at intermediate nodes, collision, and noisy channels, some packets may become lost or may be declared as lost if the internal checksum does not match. Therefore, these networks are a very good model of *erasure channels* with unknown and time-varying erasure probabilities. Although, the schemes based on *automatic repeat request* (ARQ) such as the *transmission control protocol* (TCP) ensure reliability by retransmission of the

lost packets, it is well known that such protocols behave poorly in many cases, such as one-to-many transmissions, or transmission of data over very noisy channels such as wireless links. Therefore, forward error-control schemes are more desirable and among them rateless erasure codes have the potential of replacing TCP [41].

In all previous studies on rateless codes, equal error protection (EEP) of all data was considered. The EEP property would be sufficient for applications such as multicasting bulk data (e.g., a software file) [10]. However, in several applications, a portion of data may need more protection than the rest of data. For example, in an MPEG stream [69], I-frames need more protection than P-frames. In some other applications, a portion of data may need to be recovered prior to the other parts. An example would be video-on-demand systems, in which the stream should be reconstructed in sequence [41], [75]. Such applications raise a need for having codes with *unequal error protection* (UEP) or *unequal recovery time* (URT).

For the applications similar to the ones we described above, designing rateless codes with unequal error protection property (UEP-rateless codes) is of great interest. In this work, we develop, for the first time, rateless codes that can provide UEP [54, 56, 59]. This implies that some portion of data would be protected more than the other parts. Theoretical and simulation results illustrate that a strong UEP can be achieved by the proposed rateless codes. These codes can also be employed in applications for which URT is desirable, i.e., the number of received symbols for recovering more important parts is less than that number for recovering less important parts. In our design and analysis, we consider both asymptotic and finite-length cases. We investigate the asymptotic behavior of UEP-rateless codes under the iterative decoding. We also study the performance of maximum-likelihood (ML) decoders for the proposed finite-length UEP-rateless codes. Moreover, we analyze the ML decoding performance of traditional rateless codes [53].

This chapter is organized as follows. In Section 4.2 design and *asymptotic* analysis

of UEP-rateless codes under the *iterative decoding* is studied. Section 4.3 investigates design and analysis of *finite-length* UEP-rateless codes when the *maximum-likelihood* (ML) decoding is considered. Finally, we conclude the chapter in Section 4.4.

Throughout this chapter, we assume the following terminologies. In a graph $G(V, E)$, where V is the set of vertices (nodes) and E is the set of edges, two vertices u and v are *adjacent* or *neighbor* if there is an edge $e = (u, v) \in E$ with ends u and v. Two edges e_1 and e_2 are *adjacent* if they share an end. A vertex v and an edge e are *incident* if v is an end of e. The *degree* of a vertex v is defined as the number of edges of G incident to v. We call $G'(V', E')$ is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$. Moreover, G' is a subgraph of G *induced by* V' if G' contains all the edges $(u, v) \in E$ with $u, v \in V'$.

4.2 Design and Asymptotic Analysis of UEP-Rateless Codes

Let $\Omega(x) = \sum_{i=1}^{n} \Omega_i x^i$ be the generator polynomial corresponding to the probability distribution of the degrees of output symbols (check nodes) in an LT code. In our proposed scheme, the neighbors of a check node are selected nonuniformly at random. Let us partition the n input symbols (variable nodes) into r sets $s_1, s_2, \ldots, s_r$ of sizes $\alpha_1 n, \alpha_2 n, \ldots, \alpha_r n$ such that $\sum_{j=1}^{r} \alpha_j = 1$. Let $p_j(n)^2$ be the probability that an edge is connected to a particular variable node in s_j, for $j = 1, \ldots, r$ (see Figure 4.1).

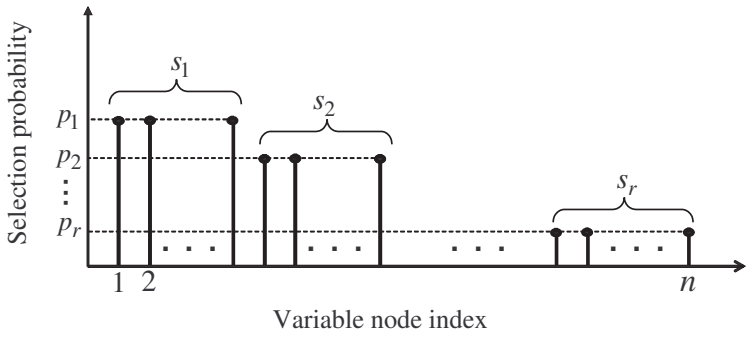


Figure 4.1: Nonuniform probability distribution function for selecting a variable node (input symbols) by an edge.

Clearly, we have $\sum_{i=1}^{r} p_i(n)\alpha_i n = 1$. The proposed ensemble at the receiver is specified by parameters $\Omega(x)$, n, γ, and $P(x,z)$, in which $P(x,z) = \sum_{i=1}^{r}(\alpha_i x^i + p_i z^i)$. The average check-node degree is given by $\mu = \sum_{i=1}^{n} i\Omega_i = \Omega'(1)$, where $\Omega'(x)$ is the derivative of $\Omega(x)$ with respect to x. Moreover, it is straightforward to show that the degree of variable nodes in s_j has a binomial distribution, for $j = 1, 2, \ldots, r$. Specifically, the probability $\lambda_{d,j}$ that a variable node in s_j has a degree d is given by

$$\lambda_{d,j} = \binom{\mu\gamma n}{d} p_j^d (1-p_j)^{\mu\gamma n - d}. \tag{20}$$

Asymptotically (as n goes to infinity), we can approximate distribution (20) by a Poisson distribution if the following two conditions are satisfied for $j = 1, \ldots, r$:

C_1 : $p_j(n) = o(1)$

C_2 : $\mu\gamma n p_j = \theta_j$ is a constant

Satisfying these conditions, $\lambda_{d,j}$ approaches to

$$\frac{e^{-\theta_j}(\theta_j)^d}{d!}, \tag{21}$$

which is a Poisson distribution with mean θ_j.

Throughout this chapter we assume conditions C_1 and C_2 are satisfied. For example, we can have $p_j(n) = \frac{k_j}{n}$, for some non-negative constant k_j that satisfy $\sum_{j=1}^{r} \alpha_j k_j = 1$. Accordingly, C_2 reduces to $\mu\gamma$ has to be a constant. This condition can be easily addressed if we consider both μ and γ as constants. Assuming that μ is a constant results in constant average variable-node and check-node degrees. This is desirable since the resulting graph will be a tree as $n \to \infty$ [39], and the encoding complexity will be linear in n.

To investigate the recovery probability of an input symbol in a generalized rateless code, we use a technique called And-Or tree analysis ([27] and [39]). Next, we describe this technique and will generalize it to fit our problem. Then, we will see how And-Or tree analysis and recovery probability of input nodes in rateless coding are related.

[2]The special case $p_1 = \ldots = p_r = \frac{1}{n}$, results in the previously studied EEP-rateless codes.

4.2.1 And-Or Tree Analysis Technique

An And-Or tree T_l is defined as following. Let T_l be a tree of depth $2l$. The root of the tree is at depth 0, its children are at depth 1, their children at depth 2, and so forth. Each node at depth $0, 2, 4, \ldots, 2l-2$ is called an *OR-node* (that evaluates logical OR operation on the value of its children), and each node at depth $1, 3, 5, \ldots, 2l-1$ is called an *AND-node* (that evaluates logical AND operation on the value of its children). Figure 4.2 depicts an T_l.

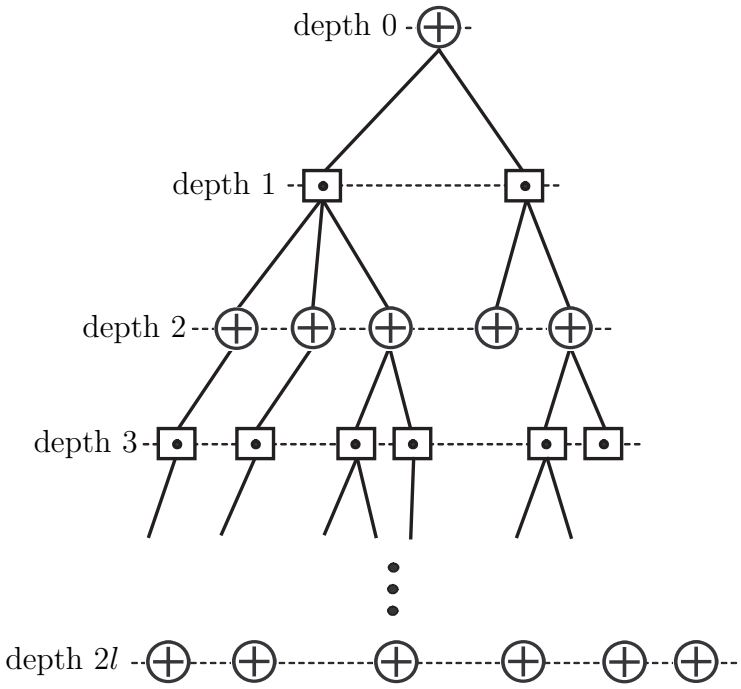


Figure 4.2: An And-Or tree of depth $2l$. Nodes represented by $\boxdot$ and $\oplus$ are AND-nodes and OR-nodes, respectively.

Suppose that each OR-node independently chooses to have i children with probability δ_i, where $\sum_i \delta_i = 1$. Similarly, each AND-node chooses to have i children with probability β_i, where $\sum_i \beta_i = 1$. Each node at depth $2l$ is assigned a value 0 or 1 independently, with y_0 being the probability that it is 0. Also, OR-nodes with no children are assumed to have a value 0, whereas AND-nodes with no children are assumed to have a value 1. We are interested in finding y_l, the probability that the

root node evaluates to 0, if we treat the tree as a boolean circuit.

The following lemma from [27], which is called the And-Or tree lemma, formulates y_l. The proof is straightforward, considering that the OR-nodes at depth 2 in T_l are the roots for independent And-Or trees T_{l-1}. Therefore, y_l can be computed as a function of y_{l-1}, the probability that the root of an And-Or tree T_{l-1} evaluates to 0.

Lemma 4.1. *The probability y_l that the root node of an And-Or tree T_l evaluates to 0 is $y_l = f(y_{l-1})$, where y_{l-1} is the probability that the root node of an And-Or tree T_{l-1} evaluates to 0, and*

$$\begin{aligned} f(x) &= \delta(1-\beta(1-x)), \\ \delta(x) = \sum_i \delta_i x^i, \quad &\text{and} \quad \beta(x) = \sum_i \beta_i x^i. \end{aligned} \tag{22}$$

Next, we generalize the And-Or tree construction to the case that OR-nodes are unlike each other. Specifically, suppose we have r different types of OR-nodes: Type 1,Type 2,..., Type r. Number of OR-nodes of each type is sufficiently large. Suppose the root of a generalized And-Or tree $GT_{l,j}$ is an OR-node of Type j, and the depth of the tree is $2l$. We construct $GT_{l,j}$ similar to T_l except that each OR-node of Type k chooses to have i children with probability $\delta_{i,k}$, for $k = 1, \ldots, r$. Each AND-node, as before, chooses to have i children with probability β_i. However, each child of an AND-node independently will be an OR-node of Type k with probability q_k. Each node of Type k at depth $2l$, is assigned a value 0 or 1 independently, with $y_{0,k}$ being the probability that it is 0. Also, OR-nodes with no children are assumed to have a value of 0, whereas AND-nodes with no children are assumed to have a value of 1. We are interested in finding $y_{l,j}$, the probability that the root node evaluates to 0, if we treat the tree as a boolean circuit. Lemma 4.2 formulates $y_{l,j}$.

Lemma 4.2. *Let $y_{l,j}$ be the probability that the root of an And-Or tree $GT_{l,j}$ evaluates to 0. Then*

$$y_{l,j} = \delta_j(1-\beta(1-\sum_{k=1}^{r} q_k y_{l-1,k})), \tag{23}$$

in which $\delta_j(x) = \sum_i \delta_{i,j} x^i$ *and* $\beta(x) = \sum_i \beta_i x^i$.

The proof is straightforward and is similar to the proof of Lemma 4.1. The relation between the above analysis and the error probabilities for the generalized rateless codes is given in the following subsection.

4.2.2 Analysis of the Generalized Rateless Codes

In this section, we examine the generalized rateless codes under iterative decoding. Let G denote the bipartite graph corresponding to the code at the receiver. Following [68] and [39], we can rephrase the belief propagation decoding algorithm for our analysis as following. At every iteration of the algorithm, messages (0 or 1) are sent along the edges from check nodes to variable nodes, and then from variable nodes to check nodes. A variable node sends 0 to an adjacent check node if and only if its value is not recovered yet. Similarly, a check node sends 0 to an adjacent variable node if and only if it is not able to recover the value of the variable node. In other words, a variable node sends 1 to a neighboring check node if only if it has received at least one message with value 1 from its other neighboring check nodes. Also, a check node sends 0 to a neighboring variable node if only if it has received at least one message with value 0 from its other neighboring variable nodes. Therefore, we see that variable nodes indeed do the logical OR operation and the check nodes do the logical AND operation. We can use the results of Lemma 4.2 on a subgraph G_l of G to find the probability that a variable node is not recovered after l decoding iterations (its value evaluates to zero). We choose G_l as following. Choose an edge (v, w) uniformly at random from all edges. Call the variable node v the root of G_l. Subgraph G_l is the graph induced by v and all neighbors of v within distance $2l$ after removing the edge (v, w). We can see G_l is a tree asymptotically [27]. We can map each check node to an AND-node and each variable node in s_j to an OR-node of Type j. We only need to compute the probabilities β_i, $\delta_{i,j}$, and q_k. We have β_i is the

probability that a randomly chosen edge is connected to a check node with i children. This is the probability that the edge is connected to a check node of degree $i+1$. We know the probability that a check node has degree $i+1$ is Ω_{i+1}. Therefore, we have

$$\beta_i = \frac{(i+1)\Omega_{i+1}}{\sum_i i\Omega_i} = \frac{(i+1)\Omega_{i+1}}{\Omega'(1)},$$

and consequently

$$\beta(x) = \sum_i \beta_i x^i = \frac{\Omega'(x)}{\Omega'(1)}.$$

Similarly, we have $\delta_{i,j}$ is the probability that the variable node connected to a randomly selected edge has degree $i+1$ given that the variable node belongs to s_j. Therefore,

$$\delta_{i,j} = \frac{(i+1)\lambda_{i+1,j}}{\sum_i \lambda_{i,j}}$$

Using (21), we conclude that

$$\begin{aligned}
\delta_{i,j} &= \frac{(i+1)\lambda_{i+1,j}}{p_j\mu\gamma n} \\
&= \frac{(i+1)e^{-\mu\gamma np_j}(\mu\gamma np_j)^{i+1}}{\mu\gamma np_j(i+1)!} \\
&= \frac{e^{-\mu\gamma np_j}(\mu\gamma np_j)^i}{i!}.
\end{aligned}$$

After substitution, we have

$$\begin{aligned}
\delta_j(x) &= \sum_i \delta_{i,j}x^i \\
&= \sum_i \frac{e^{-\mu\gamma np_j}(\mu\gamma np_j x)^i}{i!} \\
&= e^{np_j\mu\gamma(x-1)}.
\end{aligned}$$

Additionally, we have $q_k = p_k\alpha_k n$. We summarize our results in the following lemma.

Lemma 4.3. *Consider a generalized rateless code with parameters $\Omega(x)$, $P(x,z)$, n, and γ. Let $y_{l,j}$ be the probability that a variable node in s_j is not recovered after l*

decoding iterations. For $j = 1, \ldots, r$ *we have*

$$y_{0,j} = 1$$

and

$$y_{l,j} = \delta_j(1 - \beta(1 - \sum_{k=1}^{r} p_k \alpha_k n y_{l-1,k})), \quad l \geq 1 \tag{24}$$

in which

$$\beta(x) = \Omega'(x)/\Omega'(1)$$

and

$$\delta_j(x) = e^{np_j\mu\gamma(x-1)}$$

with $\mu = \Omega'(1)$.

Next, we prove a few lemmas that mostly represent the properties of the proposed codes.

Lemma 4.4. *$y_{l,j}$ is a decreasing function of the number of iterations l.*

Proof. We prove this lemma by induction. We have $y_{1,j} = e^{-np_j\gamma\Omega_1} < y_{0,j}$. Now suppose $y_{l,j} < y_{l-1,j}$ for $j = 1, \ldots, r$. We need to show $y_{l+1,j} < y_{l,j}$. This can be shown easily using the fact that $\beta(\cdot)$ and $\delta_j(\cdot)$ are both increasing functions of their argument. □

From Lemma 4.4, $\{y_{l,j}\}_l$ is a monotone decreasing sequence. Moreover, $\{y_{l,j}\}_l$ is a bounded sequence since we have $y_{l,j} \in [0, 1]$ for $l \geq 0$. From the monotone convergence theorem [4], we conclude that $\{y_{l,j}\}_l$ is a convergent sequence that converges to a fixed point in $[0, 1]$.

The following lemma can be proved similar to Lemma 4.4.

Lemma 4.5. *$y_{l,j}$ decreases when γ increases (more output symbols are collected).*

Definition 4.1. *Define* $G_{l,i,j} \triangleq \frac{y_{l,i}}{y_{l,j}}$. *This parameter compares the recovery probabilities of nodes in* s_i *and* s_j. *The larger the value of* $G_{l,i,j}$, *the higher the recovery probability of the nodes in* s_j *in comparison with the nodes in* s_i.

It can be shown that $G_{l,i,j} = e^{n(p_j - p_i)\mu\gamma\beta(1-\sum_{k=1}^{r} p_k\alpha_k n y_{l-1,k})}$. Therefore, we have:

Lemma 4.6. *For* $l \geq 1$, $G_{l,i,j} > 1$ *if and only if* $p_j > p_i$.

Lemma 4.7. *Consider two sets* s_i *and* s_j. *Suppose that* $p_j > p_i$. *Then,* $G_{l,i,j}$ *is an increasing function of the number of iterations* l *and the overhead* γ.

Proof. First we need to show that $G_{l+1,i,j} > G_{l,i,j}$. This can be shown easily using Lemma 4.4 and the fact that $\beta(\cdot)$ is an increasing function of its argument. The second part is concluded using Lemma 4.5. □

From Lemmas 4.6 and 4.7, we conclude the following. To increase the recovery probability of nodes in a set, we need to increase the selection probability of the nodes in that set. Moreover, if two nodes in different sets have different selection probabilities, the difference between their recovery probabilities increases by receiving more check nodes or by increasing the number of iterations in the iterative decoding algorithm.

4.2.3 A Special Case

In this section, a special case of the generalized rateless codes with parameters $\Omega(x)$, $P(x, z)$, n, γ, and $r = 2$ is investigated.

Assume we have two levels of importance on n information bits. Assume $n_1 = \alpha n$ $(0 < \alpha < 1)$ is the number of *more important bits* (MIB), which reside in the first part of the information, and $n_2 = (1-\alpha)n$ is the number of *less important bits* (LIB). To ensure lower average BERs for MIB than LIB, the probability of selecting MIB has to be more than that of LIB by Lemma 4.6. We set $p_1 = \frac{k_M}{n}$ and $p_2 = \frac{k_L}{n}$ for some $0 < k_L < 1$ and $k_M = \frac{1-(1-\alpha)k_L}{\alpha}$. Let $y_{l,M}$ and $y_{l,L}$ denote the error probabilities

of MIB and LIB at the l^{th} decoding iteration, respectively. From Lemma 4.3, we conclude that

$$y_{l,M} = e^{-k_M \mu\gamma\beta(1-(1-\alpha)k_L y_{l-1,L} - \alpha k_M y_{l-1,M})} \tag{25}$$

and

$$y_{l,L} = e^{-k_L \mu\gamma\beta(1-(1-\alpha)k_L y_{l-1,L} - \alpha k_M y_{l-1,M})} \tag{26}$$

with $\beta(x) = \Omega'(x)/\Omega'(1)$, $\mu = \Omega'(1)$, and $y_{0,L} = y_{0,M} = 1$.

The sequences $\{y_{l,M}\}_l$ and $\{y_{l,L}\}_l$ are convergent by Lemma 4.4. Let us call the corresponding fixed points as y_L and y_M, respectively. It can be shown that $\frac{\partial y_M}{\partial k_M}|_{k_M=1} = -\varphi$ and $\frac{\partial y_L}{\partial k_M}|_{k_M=1} = \varphi\frac{\alpha}{1-\alpha}$, where $\varphi = -y\ln y > 0$. Here, y is the bit error probability when uniform selection ($k_M = 1$) is done and satisfies $y = e^{-\gamma\Omega'(1-y)}$. These results express the variations of the bit error rates when k_M is slightly greater than one. We note that y_M decreases but y_L increases. However, for $0 < \alpha < \frac{1}{2}$, the decreasing slope of y_M is $\frac{1-\alpha}{\alpha}$ times greater than the increasing slope of y_L.

Example: In this example, we consider the degree distribution as in [68]:

$$\begin{aligned}\Omega_1(x) = {} & 0.007969x + 0.493570x^2 + 0.166220x^3 + 0.072646x^4 \\ & + 0.082558x^5 + 0.056058x^8 + 0.037229x^9 \\ & + 0.055590x^{19} + 0.025023x^{64} + 0.003135x^{66}.\end{aligned} \tag{27}$$

Figure 4.3 shows y_L and y_M versus k_M for $\alpha = 0.1$. We considered two overheads $\gamma = 1.03$ and $\gamma = 1.05$. As an example, we consider the case that $\gamma = 1.05$. Uniform selection ($k_M = 1$) results in the BER of 3.4×10^{-3} for all data whereas $y_M = 5 \times 10^{-5}$ and $y_L = 9 \times 10^{-3}$ when $k_M = 1.9$. This shows that the BER of MIB has improved substantially (about two orders of magnitude) at the cost of a slight performance loss on the LIB.

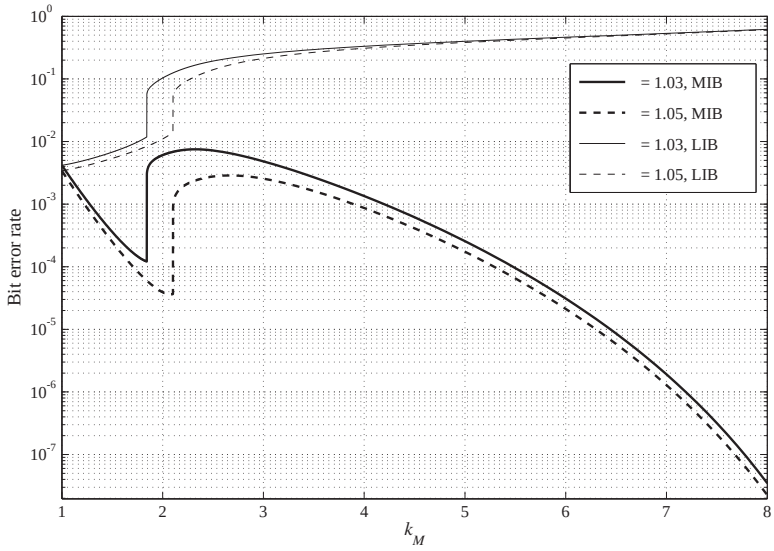


Figure 4.3: Asymptotic analysis of bit error rates versus k_M for the UEP-rateless code with parameters $\Omega_1(x)$, n, and $P(x,z) = 0.1x + \frac{k_M}{n}z + 0.9x^2 + \frac{k_L}{n}z^2$.

Figure 4.4 compares the average BER and the BER of MIB with the BER of the EEP-code for $\gamma = 1.05$. For example, for $k_M = 2$, the average performance of the UEP code is tripled. However, the performance of MIB is 87 times better than the case of EEP.

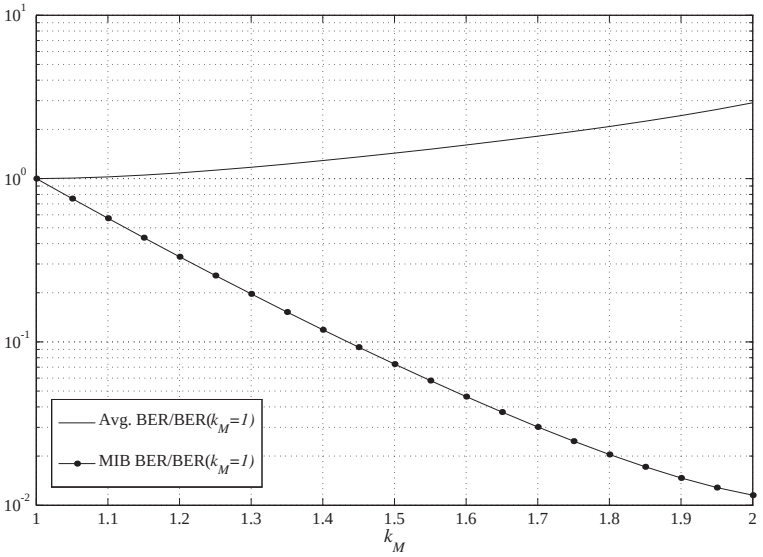


Figure 4.4: The ratios of the average BER and the MIB error rate to the BER of the EEP-code versus k_M. In this case $\gamma = 1.05$.

Figure 4.5 depicts the BERs of MIB and LIB versus the overhead γ for $k_M = 2$.

We have also included the BERs for the EEP code. Interestingly, nonuniform selection reduces BERs of both MIB and LIB for small overheads. For large overheads, the BER of MIB improves significantly while in return the performance of LIB slightly degrades.

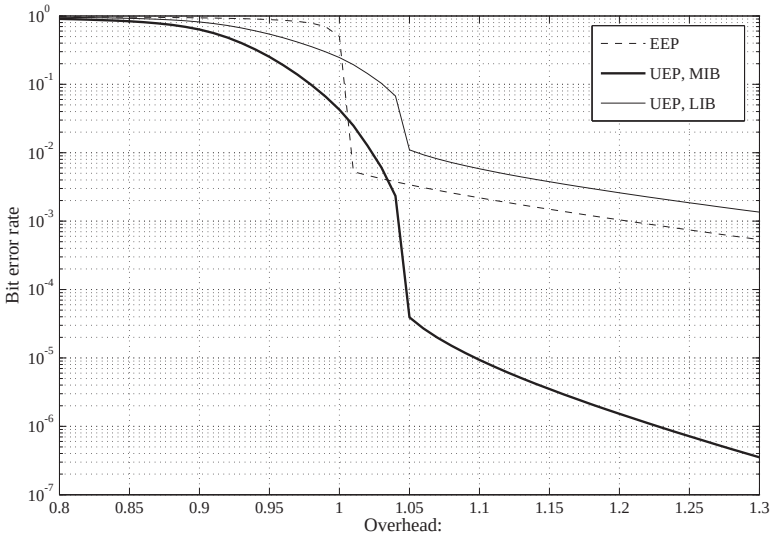


Figure 4.5: Asymptotic BERs of MIB and LIB versus overhead γ for $k_M = 2$, as well as the BERs of the EEP code ($k_M = 1$).

It should be mentioned that we can also interpret the UEP as the URT. This implies that given a target bit error rate, different parts of information bits can be decoded after receiving different numbers of encoded bits. In other words, the BER of MIB reaches a target BER sooner (smaller overhead) than the BER of LIB (see Figure 4.5).

4.2.4 Simulation Results on the Iterative Decoding of a Moderate-Length UEP-Rateless Code

Here, we give simulation results for the case that the number of information bits is $n = 2000$. We considered two cases, an EEP code and a UEP code with $k_M = 2$ and $\alpha = 0.1$. We considered $\Omega_1(x)$ in both cases. Figure 4.6 shows the bit error rates after performing LT decoding. We notice that the performance of MIB improves substantially in the UEP case. Even LIB has better performance than the case of

EEP for small overheads. We conclude that for small overheads, UEP is provided while the overall performance of the UEP code is better than that of the EEP code. Figure 4.6 also depicts a large gap between the BERs of MIB and LIB. For example, the BER of MIB is about two orders of magnitude better than that of LIB when $\gamma = 1.3$. This gap increases monotonically with the overhead.

Next, let us consider the URT problem. In URT, the BER of MIB reaches a target BER faster (smaller overhead) than the BER of LIB. For example in Figure 4.6, we need to collect $1.16n = 2320$ output symbols to have $BER = 10^{-3}$ for MIB. However, $1.33n = 2660$ output symbols need to be collected to achieve the same BER for LIB. This implies faster recovery for MIB than LIB.

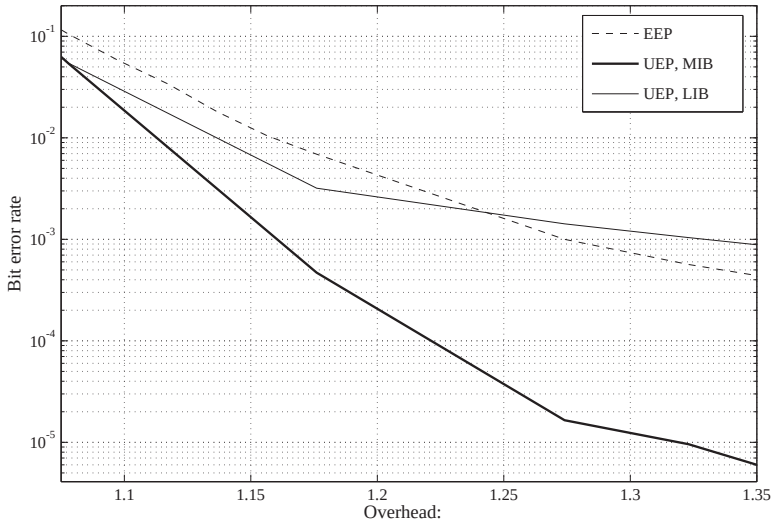


Figure 4.6: Iterative decoding performance of the UEP-rateless code with parameters $\Omega_1(x)$, $n = 2000$, $k_M = 2$, and $\alpha = 0.1$ in comparison with the EEP-rateless code.

4.3 Finite-Length Analysis of UEP-Rateless Codes

In this section, finite-length analysis of LT and Raptor codes over the BEC is investigated. First, we derive upper and lower bounds on the maximum-likelihood (ML) decoding error probabilities of LT and Raptor codes when they provide EEP. This not only provides a ground for comparison between the EEP- and UEP-rateless codes, but also offers a lower bound on the performance of EEP-rateless codes under the

iterative decoding. We then study this for UEP-LT and UEP-Raptor codes. ML decoding is computationally complex especially for long codes. However, the derivation of bounds on the ML decoding is of interest, as it provides an ultimate indication on the code performance.

4.3.1 Bounds on the Maximum-Likelihood Decoding Error Probabilities of Finite-Length LT and Raptor Codes over the BEC

We investigate the performance of finite-length LT and Raptor codes under the ML decoding. In our analysis, we consider the non-replacement selection of the input nodes of LT codes. This means that given a check-node degree is d, a sequence of d different input nodes is selected uniformly at random from the n input nodes. Thus, a particular sequence is selected with a probability $\frac{1}{\binom{n}{d}}$.

4.3.1.1 ML Decoding of LT Codes over the BEC

The ML decoding of LT codes over the BEC is the problem of recovering n information bits from $n\gamma$ received check bits. This is equivalent to solving the linear equation

$$Hx^T = b, \tag{28}$$

in which $H = [h_{ij}]$ is an $n\gamma \times n$ adjacency matrix corresponding to the graph that is formed by the input nodes and the received check nodes. Here, $h_{ij} = 1$ if the i^{th} received check node and the j^{th} input node are adjacent, otherwise $h_{ij} = 0$. Moreover, b is an n dimensional column vector in which b_i is the value of the i^{th} received check node. Equation (28) has at least one solution. It has multiple solutions if and only if H is not full rank. Moreover, the i^{th} bit does not have a unique solution if and only if H_i (the i^{th} column of H) is in the column space spanned by $H \backslash H_i$. In the following lemma, we derive an upper bound on the ML decoding bit error probability of LT codes under the ML decoding.

Lemma 4.8. *Given an LT code with parameters $\Omega(x)$, n, and overhead γ_L, an upper bound on the bit error probability of the LT code under the ML decoding is*

$$p_b^{ML} \leq \min\left\{1, \sum_{w=1}^{n} \binom{n-1}{w-1} \left(\sum_d \Omega_d \frac{\sum_{s=0,2,\ldots,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s}\binom{n-w}{d-s}}{\binom{n}{d}} \right)^{n\gamma_L} \right\}. \tag{29}$$

Proof. Let p_b^{ML} be the probability that the i^{th} bit cannot be determined by the ML decoder, for an arbitrary $i \in \{1, 2, \ldots, n\}$. We have

$$\begin{aligned} p_b^{ML} &= Pr\{\exists x \in GF(2)^n, x(i) = 1 : Hx^T = 0^T\} \\ &\leq \sum_{x \in GF(2)^n, x(i)=1} Pr\{Hx^T = 0^T\}. \end{aligned}$$

Let $x \in GF(2)^n, x(i) = 1$, and $I = \{i_1, i_2, \ldots, i_w\}$ be the set of indices such that $j \in I$ if and only if $x(j) = 1$. The rows of H, when viewed as random binary vectors, are generated from *independent* trials of a random variable R, such that for any vector $\nu \in GF(2)^n$, $Pr(R = \nu) = \frac{\Omega_d}{\binom{n}{d}}$, where d is the weight of ν. Therefore,

$$Pr\{Hx^T = 0^T\} = \left(Pr\{Rx^T = 0\}\right)^{n\gamma_L}.$$

Suppose that weight$(R) = d$. Moreover, let $R(I)$ be defined as a sub-vector of R containing components of R that are specified by the elements of I, i.e., $R(I) = \{R(i_1), R(i_2), \ldots, R(i_w)\}$. We have

$$\begin{aligned} Pr\{Rx^T = 0\} &= Pr\{R(I) \text{ contains even number of 1's}\} \\ &= \frac{\sum_{s=0,2,\ldots,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s}\binom{n-w}{d-s}}{\binom{n}{d}}. \end{aligned}$$

Since each row of H has weight d with probability Ω_d, and there are $\binom{n-1}{w-1}$ choices of x with weight w, we conclude the assertion. □

A lower bound on the bit error probability of LT codes under ML decoding can be found by computing the probability that a variable node is not adjacent to any of the check nodes. This lower bound is given by [68]

$$p_b^{ML} \geq \left(1 - \frac{\mu}{n}\right)^{n\gamma_L}, \tag{30}$$

in which $\mu = \sum_d d\Omega_d$ is the average check-node degree.

Figure 4.7 shows the upper and lower bounds on ML decoding error probabilities versus overhead γ_L for an LT code with distribution $\Omega_1(x)$ and length 500. The results imply that the bound is almost tight for $\gamma > 1.3$.

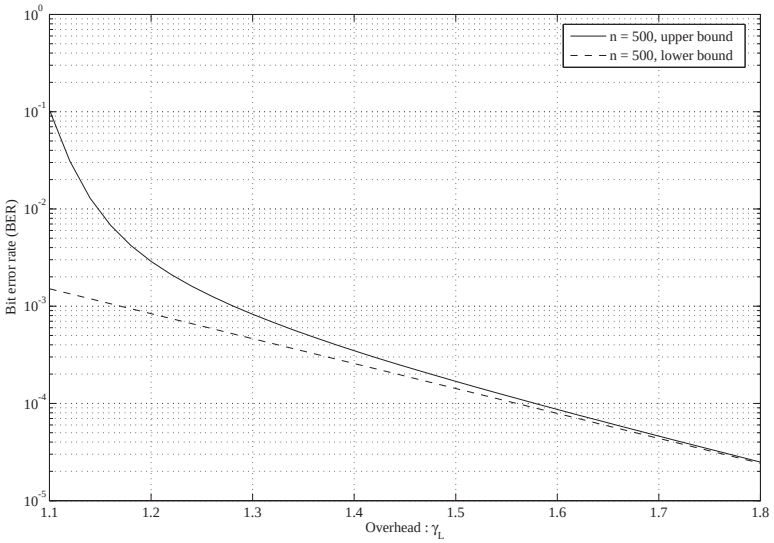


Figure 4.7: Upper and lower bounds on the ML decoding BERs versus overhead γ_L for an LT code with distribution $\Omega_1(x)$ and length $n = 500$.

4.3.1.2 ML Decoding of Raptor Codes over the BEC

Raptor codes introduced by Shokrollahi [68] are an extension of LT codes, in which an outer high-rate traditional pre-code is concatenated to an inner LT code to get practically better results than the LT code. Let $\mathcal{C}$ be a linear code of length n, rate $R = 1-\frac{m}{n}$, and dimension $k = n-m$. A Raptor code with parameters $(k, \mathcal{C}, \Omega(x))$ is an LT code with distribution $\Omega(x)$ on n bits that are the codeword bits of the pre-code $\mathcal{C}$. If γ_L denotes the overhead of the LT code, the overhead of the Raptor code is $\gamma = \frac{\gamma_L}{R}$. In this study, we assume the pre-code is an (n, k) LDPC code with a parity-check matrix $H' = [h'_{ij}]$ whose entries are independent and identically distributed (i.i.d) Bernoulli random variables with parameter ρ. We denote such a code by (n, k, ρ)

LDPC code. The following lemmas, whose proofs are given in Appendix A, develop upper and lower bounds on the ML decoding error probability of Raptor codes.

Lemma 4.9. *Let $\mathcal{C}$ be an (n, k, ρ) LDPC code. Given a $(k, \mathcal{C}, \Omega(x))$ Raptor code with overhead γ, an upper bound on the ML decoding bit error probability is obtained as*

$$p_b^{ML} \leq \sum_{e=0}^{n} \binom{n}{e} \epsilon_U^e (1-\epsilon_U)^{n-e} \frac{e}{n} \min\left\{1, \sum_{w=1}^{e} \binom{e-1}{w-1} \Big(A\big(w,\rho\big)\Big)^m\right\},$$

where

$$A(w,\rho) := \frac{1+(1-2\rho)^w}{2}. \tag{31}$$

Here, $m = n - k$. Also, ϵ_U is the upper bound on the ML decoding bit error rate of the LT code with parameters $\Omega(x)$, n, and overhead $\gamma_L = \frac{k}{n}\gamma$ that was found by Lemma 4.8.

Lemma 4.10. *Let $\mathcal{C}$ be an (n, k, ρ) LDPC code. Given a $(k, \mathcal{C}, \Omega(x))$ Raptor code with overhead γ, a lower bound on the ML decoding bit error probability is given by*

$$\begin{aligned} p_b^{ML} \geq \max\Bigg\{ & 0, \sum_{e=0}^{n} \binom{n}{e} \epsilon_L^e (1-\epsilon_L)^{n-e} \frac{e}{n} \min\Big\{1, \sum_{w=1}^{e} \binom{e-1}{w-1} A^m(w,\rho)\Big\} \\ & - \frac{1}{2} \sum_{e=0}^{n} \binom{n}{e} \epsilon_U^e (1-\epsilon_U)^{n-e} \frac{e}{n} \min\Big\{1, \sum_{w_0=1}^{e-1} \sum_{w_1=0}^{e-w_0} \sum_{w_2=0}^{e-w_0-w_1} \mathbf{1}(w_1+w_2) \\ & \cdot \binom{e-1}{w_0-1} \binom{e-w_0}{w_1} \binom{e-w_0-w_1}{w_2} D^m(w_0, w_1, w_2, \rho, \rho, \rho)\Big\}\Bigg\}, \end{aligned}$$

where

$$\begin{aligned} D(w_0, w_1, w_2, \rho, \rho, \rho) := & A(w_0,\rho)A(w_1,\rho)A(w_2,\rho) \\ & + \overline{A}(w_0,\rho)\overline{A}(w_1,\rho)\overline{A}(w_2,\rho), \end{aligned}$$

$$\mathbf{1}(x) := \begin{cases} 0 & \textit{if } x = 0 \\ 1 & \textit{otherwise} \end{cases}$$

$\overline{A(\cdot)} = 1 - A(\cdot)$, *and* $m = n - k$. *Also,* $A(\cdot)$ *is defined as (31). Moreover,* ϵ_L (ϵ_U) *is the lower bound (upper bound) on the ML decoding bit error rate of the LT code with parameters* $\Omega(x)$, n, *and overhead* $\gamma_L = \frac{k}{n}\gamma$ *found by (30) and (29).*

Figure 4.8 depicts the upper and lower bounds on the ML decoding bit error probabilities versus overhead γ for the fixed degree distribution $\Omega_1(x)$. We considered an LT code with $n = 500$ and a Raptor code with $k = 500$ and a pre-code $\mathcal{C}$ as an $(510, 500, 0.4)$ LDPC code with $R \approx 0.98$. Note that in each case we assumed the decoder starts the decoding after receiving 500γ check bits. As we can see, the bounds are tight for small error rates. Moreover, as expected and was shown in [68], Raptor codes can achieve lower error rates than LT codes.

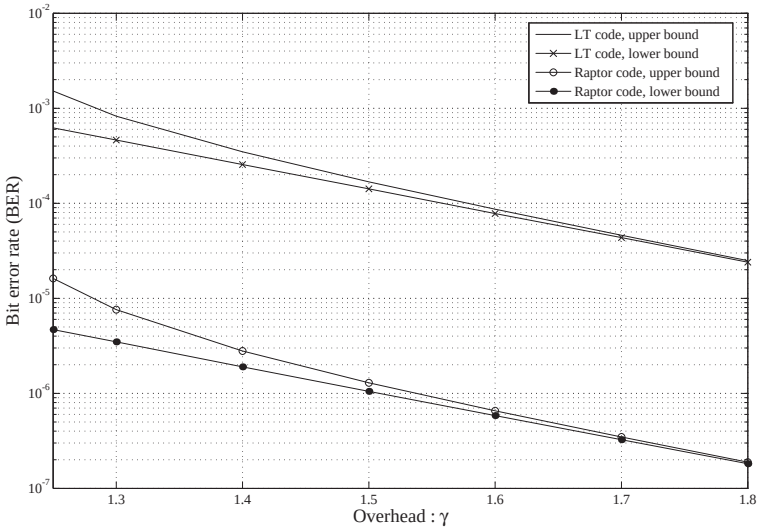


Figure 4.8: Upper and lower bounds on the ML decoding BERs of LT and Raptor codes versus overhead γ for transmitting 500 information bits over an erasure channel.

4.3.2 Bounds on the Maximum-Likelihood Decoding Error Probabilities of Finite-Length UEP-LT and UEP-Raptor Codes over the BEC

In this section, we consider the problem of finite-length UEP-rateless codes. Suppose we want to transmit n bits with two different levels of importance over a BEC. Assume $n_1 = \alpha n$ $(0 < \alpha < 1)$ is the number of MIB and $n_2 = (1 - \alpha)n$ is the number of LIB. A UEP-LT code is constructed similar to a traditional LT code except that

the check nodes select their adjacent variable nodes *nonuniformly* at random. This means that a check node with degree d, selects $d_1 = \min([\alpha d k_M], n_1)$ ($[x]$ means the nearest integer to x) variable nodes from MIB (for some $k_M > 1$) and $d_2 = d - d_1$ variable nodes from LIB as shown in Figure 4.9. Note that here the non-replacement selection is considered. This means that any sequence of d_1 (d_2) different variable nodes in MIB (LIB) is selected uniformly with probability $\frac{1}{\binom{n_1}{d_1}}$ ($\frac{1}{\binom{n_2}{d_2}}$). By cascading

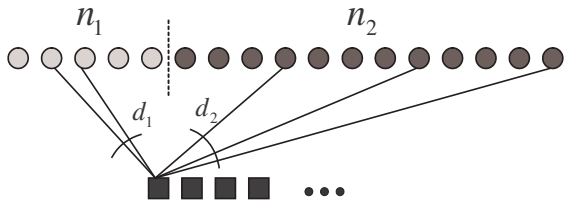


Figure 4.9: Nonuniform selection of variable nodes (input symbols) in UEP-LT codes.

a UEP-LT code and a traditional pre-code $\mathcal{C}$, we can form a UEP-Raptor code[2]. This implies that the codeword bits of $\mathcal{C}$ are the input bits of the UEP-LT code. Let $\mathcal{C}$ be a linear code of length n, rate $R = 1 - \frac{m}{n}$ and dimension $k = n - m$. Let also H' be the parity-check matrix that corresponds to $\mathcal{C}$. Here the number of information bits is k. We may design the pre-code $\mathcal{C}$ such that all the first n_1 bits of the codeword bits correspond to the more important information bits. This is possible if and only if the submatrix of H' containing the last n_2 columns has full rank. In this case, the ratio of the number of more important information bits to the total number of information bits is $\alpha_R = \frac{\alpha}{R}$. As before, let us assume the pre-code $\mathcal{C}$ is an (n, k, ρ) LDPC code. Next, we derive upper and lower bounds on the ML decoding error probabilities of the UEP-LT and UEP-Raptor codes.

[2]An alternative way to form a UEP-Raptor code is by cascading a traditional LT code and a UEP pre-code. Although we do not consider this case in this study, the analysis will be similar.

4.3.2.1 ML Decoding of UEP-LT Codes

In this section, we examine the performance of UEP-LT codes under the ML decoding. In the following lemma, upper bounds on the ML decoding error probabilities of the proposed ensemble are derived.

Lemma 4.11. *Consider a UEP-LT code with parameters $\Omega(x)$, n, α, k_M, and overhead γ_L. The upper bounds on the bit error probabilities of MIB and LIB under the ML decoding are*

$$p_{b,MIB}^{ML} \leq \min\Bigg\{1, \sum_{w=1}^{n}\sum_{w_1=1}^{w}\binom{n_1-1}{w_1-1}\binom{n_2}{w_2} \cdot\left(\sum_d \Omega_d \frac{\sum_{t=0}^{1}\Big(\prod_{r=1}^{2}\Big(\sum_{s=t,2+t,\ldots,2\lfloor\frac{d_r}{2}\rfloor-t}\binom{w_r}{s}\binom{n_r-w_r}{d_r-s}\Big)\Big)}{\binom{n_1}{d_1}\binom{n_2}{d_2}}\right)^{n\gamma_L}\Bigg\}$$

and

$$p_{b,LIB}^{ML} \leq \min\Bigg\{1, \sum_{w=1}^{n}\sum_{w_1=0}^{w-1}\binom{n_1}{w_1}\binom{n_2-1}{w_2-1} \cdot\left(\sum_d \Omega_d \frac{\sum_{t=0}^{1}\Big(\prod_{r=1}^{2}\Big(\sum_{s=t,2+t,\ldots,2\lfloor\frac{d_r}{2}\rfloor-t}\binom{w_r}{s}\binom{n_r-w_r}{d_r-s}\Big)\Big)}{\binom{n_1}{d_1}\binom{n_2}{d_2}}\right)^{n\gamma_L}\Bigg\},$$

respectively. Here, $w_2 = w - w_1$, $n_1 = \alpha n$, $n_2 = (1-\alpha)n$, $d_1 = \min([\alpha d k_M], n_1)$, and $d_2 = d - d_1$.

Proof. Let $H = [h_{cv}]$ be the adjacency matrix corresponding to the graph that is formed by the input nodes and the received check nodes. This means that $h_{cv} = 1$ if and only if the c^{th} received check node is adjacent to v^{th} variable node. Let $p_{b,i}^{ML}$ be the bit error probability of the i^{th} bit under ML decoding. For an arbitrary $i \in \{1, 2, \ldots, n\}$ we have

$$\begin{aligned} p_{b,i}^{ML} &= Pr\{\exists x \in GF(2)^n, x(i) = 1 : Hx^T = 0^T\} \\ &\leq \sum_{x \in GF(2)^n, x(i)=1} Pr\{Hx^T = 0^T\}. \end{aligned} \tag{32}$$

Let $x \in GF(2)^n, x(i) = 1$, and $I = \{i_1, i_2, \ldots, i_{w_1}\}$ be the set of indices such that $j \in I$ if and only if $x(j) = 1$ and $j \in \{1, \ldots, n_1\}$. Similarly, $J = \{j_1, j_2, \ldots, j_{w_2}\}$ is the set of indices such that $j \in J$ if and only if $x(j) = 1$ and $j \in \{n_1 + 1, \ldots, n\}$. As in the proof of Lemma 4.8 we have

$$Pr\{Hx^T = 0^T\} = \big(Pr\{Rx^T = 0\}\big)^{n\gamma_L},$$

where R is any row of H. Suppose that weight$(R) = d$.

$$\begin{aligned} Pr\{Rx^T = 0\} &= Pr\{R(I) \quad \text{contains even number of 1's}\} \\ &\quad \cdot Pr\{R(J) \quad \text{contains even number of 1's}\} \\ &\quad + Pr\{R(I) \quad \text{contains odd number of 1's}\} \\ &\quad \cdot Pr\{R(J) \quad \text{contains odd number of 1's}\} \\ &= \frac{\displaystyle\sum_{t=0}^{1}\Big(\prod_{r=1}^{2}\Big(\sum_{s=t,2+t,\ldots,2\lfloor\frac{d_r}{2}\rfloor-t}\binom{w_r}{s}\binom{n_r-w_r}{d_r-s}\Big)\Big)}{\binom{n_1}{d_1}\binom{n_2}{d_2}} \end{aligned}$$

For $i \in$ MIB, there are $\binom{n_1-1}{w_1-1}\binom{n_2}{w_2}$ possible different x's, and for $i \in$ LIB, this value is $\binom{n_1}{w_1}\binom{n_2-1}{w_2-1}$. This completes the proof. □

Lower bounds on the bit error probabilities of MIB and LIB under the ML decoding are given by

$$p_{b,MIB}^{ML} \geq \left(1 - \sum_d \Omega_d \frac{d_1}{n_1}\right)^{n\gamma_L} \tag{33}$$

and

$$p_{b,LIB}^{ML} \geq \left(1 - \sum_d \Omega_d \frac{d_2}{n_2}\right)^{n\gamma_L}, \tag{34}$$

respectively. These are the probabilities that a node in MIB or LIB is not a neighbor of any of the check nodes.

Figure 4.10 shows the upper bound (UB) and lower bound (LB) on the ML decoding BERs of MIB and LIB versus overhead γ_L for a UEP-LT code with parameters

$n = 500$, $\Omega_1(x)$, $k_M = 2$, and $\alpha = 0.1$. We also included the bounds on the ML decoding performance of an EEP-LT code with $n = 500$ and $\Omega_1(x)$. As an example, for $\gamma = 1.8$ where the bounds are tight, we note that BER of LIB degrades less than one order of magnitude in comparison with the EEP code. However, BER of MIB improves by about four orders of magnitude.

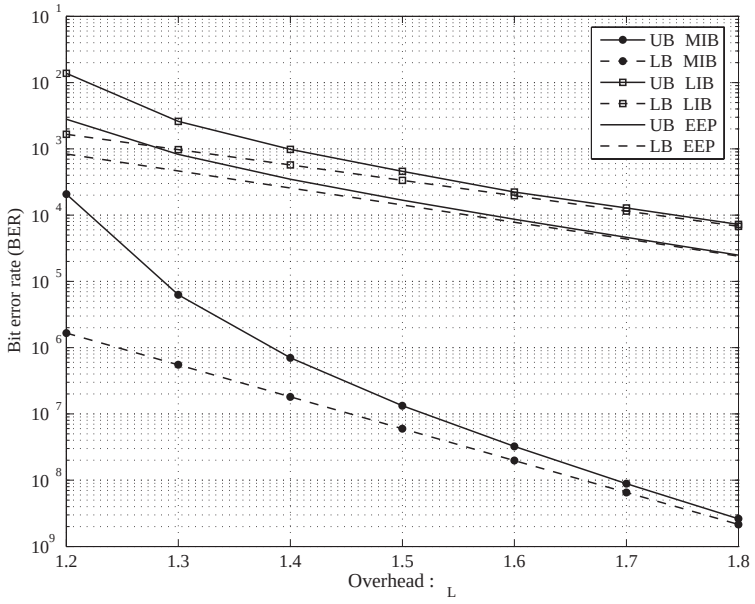


Figure 4.10: Upper and lower bounds on the ML decoding BERs of MIB and LIB versus overhead γ_L for a UEP-LT code with parameters $n = 500$, $\Omega_1(x)$, $k_M = 2$, and $\alpha = 0.1$. The bounds on the decoding performance of the EEP-LT code are also depicted.

4.3.2.2 ML Decoding of UEP-Raptor Codes

Let us consider the case that we cascade a UEP-LT code by a pre-code $\mathcal{C}$ to form a UEP-Raptor code. Similar to Lemma 4.9, we can show the following.

Lemma 4.12. *Let $\mathcal{C}$ be an (n, k, ρ) LDPC code. Consider a UEP-Raptor code that has a UEP-LT code with parameters $\Omega(x)$, n, γ_L, α, and k_M together with the pre-code $\mathcal{C}$. Upper bounds on the bit error probabilities of MIB and LIB under the ML*

decoding are given by

$$p_{b,MIB}^{ML} \leq \sum_{e=1}^{n} \sum_{e_1=\max(1,e-n_2)}^{\min(n_1,e)} \binom{n_1-1}{e_1-1}\binom{n_2}{e-e_1}\epsilon_{U1}^{e_1}(1-\epsilon_{U1})^{n_1-e_1} \cdot \epsilon_{U2}^{e-e_1}(1-\epsilon_{U2})^{n_2-e+e_1} \min\Big\{1, \sum_{w=1}^{e}\binom{e-1}{w-1}A^m(w,\rho)\Big\}$$

and

$$p_{b,LIB}^{ML} \leq \sum_{e=1}^{n} \sum_{e_1=\max(0,e-n_2)}^{\min(n_1,e-1)} \binom{n_1}{e_1}\binom{n_2-1}{e-e_1-1}\epsilon_{U1}^{e_1}(1-\epsilon_{U1})^{n_1-e_1} \cdot \epsilon_{U2}^{e-e_1}(1-\epsilon_{U2})^{n_2-e+e_1} \min\Big\{1, \sum_{w=1}^{e}\binom{e-1}{w-1}A^m(w,\rho)\Big\},$$

respectively. Here, ϵ_{U1} and ϵ_{U2} are the upper bounds on the ML decoding BERs of MIB and LIB in the UEP-LT code, respectively, $m = n-k$, and $A(.)$ is defined as in Lemma 4.9.

Likewise, similar to Lemma 4.10, we can show the following.

Lemma 4.13. *Let $\mathcal{C}$ be an (n,k,ρ) LDPC code. Consider a UEP-Raptor code that has a UEP-LT code with parameters $\Omega(x)$, n, γ_L, α, and k_M together with the pre-code $\mathcal{C}$. Lower bounds on the bit error probabilities of MIB and LIB under the ML decoding are given by*

$$\begin{aligned} p_{b,MIB}^{ML} \geq \max\Bigg\{0, &\sum_{e=1}^{n} \sum_{e_1=\max(1,e-n_2)}^{\min(n_1,e)} \binom{n_1-1}{e_1-1}\binom{n_2}{e-e_1}\epsilon_{L1}^{e_1}(1-\epsilon_{L1})^{n_1-e_1} \\ &\cdot \epsilon_{L2}^{e-e_1}(1-\epsilon_{L2})^{n_2-e+e_1} \min\Big\{1, \sum_{w=1}^{e}\binom{e-1}{w-1}A^m(w,\rho)\Big\} \\ &- \frac{1}{2}\sum_{e=1}^{n} \sum_{e_1=\max(1,e-n_2)}^{\min(n_1,e)} \binom{n_1-1}{e_1-1}\binom{n_2}{e-e_1}\epsilon_{U1}^{e_1}(1-\epsilon_{U1})^{n_1-e_1} \\ &\cdot \epsilon_{U2}^{e-e_1}(1-\epsilon_{U2})^{n_2-e+e_1} \min\Big\{1, \sum_{w_0=1}^{e-1}\sum_{w_1=0}^{e-w_0}\sum_{w_2=0}^{e-w_0-w_1} \mathbf{1}(w_1+w_2) \\ &\cdot \binom{e-1}{w_0-1}\binom{e-w_0}{w_1}\binom{e-w_0-w_1}{w_2} D^m(w_0,w_1,w_2,\rho,\rho,\rho)\Big\}\Bigg\} \end{aligned}$$

and

$$p_{b,LIB}^{ML} \geq \max\Bigg\{0, \sum_{e=1}^{n} \sum_{e_1=\max(0,e-n_2)}^{\min(n_1,e-1)} \binom{n_1}{e_1}\binom{n_2-1}{e-e_1-1} \epsilon_{L1}^{e_1}(1-\epsilon_{L1})^{n_1-e_1}$$
$$\cdot \epsilon_{L2}^{e-e_1}(1-\epsilon_{L2})^{n_2-e+e_1} \min\Big\{1, \sum_{w=1}^{e}\binom{e-1}{w-1} A^m(w,\rho)\Big\}$$
$$- \frac{1}{2}\sum_{e=1}^{n} \sum_{e_1=\max(0,e-n_2)}^{\min(n_1,e-1)} \binom{n_1}{e_1}\binom{n_2-1}{e-e_1-1} \epsilon_{U1}^{e_1}(1-\epsilon_{U1})^{n_1-e_1}$$
$$\cdot \epsilon_{U2}^{e-e_1}(1-\epsilon_{U2})^{n_2-e+e_1} \min\Big\{1, \sum_{w_0=1}^{e-1}\sum_{w_1=0}^{e-w_0}\sum_{w_2=0}^{e-w_0-w_1} \mathbf{1}(w_1+w_2)$$
$$\cdot \binom{e-1}{w_0-1}\binom{e-w_0}{w_1}\binom{e-w_0-w_1}{w_2} D^m(w_0,w_1,w_2,\rho,\rho,\rho)\Big\}\Bigg\},$$

respectively. Here, ϵ_{L1} (ϵ_{U1}) and ϵ_{L2} (ϵ_{U2}) are the lower bounds (upper bounds) on the ML decoding BERs of MIB and LIB in the UEP-LT code, respectively and $m = n-k$. Moreover, $A(\cdot)$, $\overline{A(\cdot)}$, and $D(\cdot)$ are defined as in Lemmas 4.9 and 4.10.

Figure 4.11 shows the upper and lower bounds on the ML decoding BERs of MIB and LIB versus overhead γ for a UEP-Raptor code with parameters $k = 500$, $\Omega_1(x)$, $k_M = 2$, $\alpha = 0.1$, and a pre-code $\mathcal{C}$ as an $(510, 500, 0.4)$ LDPC code with $R \approx 0.98$. We also included the bounds on the ML decoding performance of an EEP-Raptor code with $k = 500$, $\Omega_1(x)$, and the same pre-code. As an example, for $\gamma = 1.8$ where the bounds are tight, the BER of LIB is increased less than one order of magnitude but the BER of MIB is decreased by about four orders of magnitude. This shows a large gap between the BERs of MIB and LIB and very low error rates for the MIB.

4.4 Conclusion

In this chapter, we proposed a modification in the structure of rateless codes to provide unequal error protection (UEP) and unequal recovery time (URT) properties. We analyzed the performance of the iterative decoding algorithm for the proposed

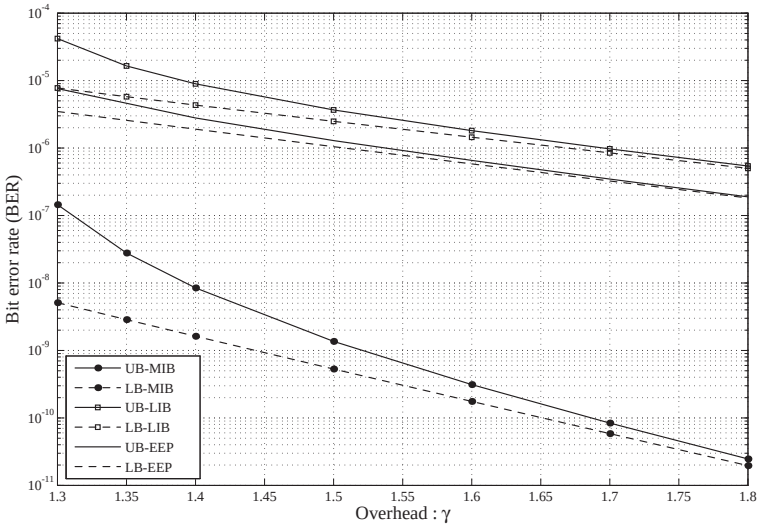


Figure 4.11: Upper and lower bounds on the ML decoding BERs of MIB and LIB for the UEP-Raptor code with parameters $k = 500$, $\Omega_1(x)$, $k_M = 2$, $\alpha = 0.1$, and a $(510, 500, 0.4)$ LDPC code as the pre-code. The bounds on the decoding performance of the EEP-Raptor code are also depicted.

codes asymptotically (when the length of the code goes to infinity). It was shown that UEP-rateless codes can provide very low error rates for more important bits with only a subtle loss in the performance of less important bits. Next, we focused on finite-length rateless codes and derived upper and lower bounds on the maximum-likelihood decoding bit error rates of EEP- and UEP-rateless codes. The results show not only that the bounds are tight for small error rates, but also that the bit error rates of more important bits are significantly improved with respect to the bit error rates of less important bits. We also discussed that the UEP problem can be viewed as a URT problem for a fixed bit error rate. That is to say, different parts of information can be retrieved if the receiver receives different numbers of encoded packets, which corresponds to different recovery times.

CHAPTER V

EFFICIENT BROADCASTING IN WIRELESS AD-HOC AND SENSOR NETWORKS WITH NO TOPOLOGY KNOWLEDGE

5.1 Introduction

An important issue in multihop wireless networks that has attracted a lot of attention is efficient network-wide broadcasting. The type of broadcast data could be bulk data, e.g., software files or short data like route discovery packets. Some important factors that influence the efficiency of a broadcasting scheme can be reliability (defined as the percentage of nodes in the network that are able to retrieve the data), energy efficiency, complexity, scalability, and latency. Depending on the application, some factors might be more important than others. For example, for updating the software in all the nodes in the network, reliability is very important, while latency might be less important. Energy is usually an important issue, especially for battery-powered sensor networks.

In this chapter, we consider a case in which a large number of packets have to be broadcast in a multihop wireless sensor network with no availability of information about the network topology, and our main concerns are *reliability* and *energy efficiency*. We propose an efficient two-phase broadcast scheme, which we refer to as *collaborative rateless broadcast* (CRBcast) [55, 60]. CRBcast is based on *probabilistic broadcast* (PBcast) and an application layer rateless coding. In the first phase, the rateless-encoded packets are broadcast based on PBcast, in which each node probabilistically relays every new received packet. The second recovery phase, which is based on simple collaborations of the nodes, ensures that all the nodes can recover

the original data.

Since the original packets are first encoded by a rateless code in CRBcast, the number of the original packets should be large enough to minimize the overhead incurred by the coding scheme. Therefore, CRBcast is suitable for applications such as data dissemination in which we are required to broadcast moderate to large number of packets.

Since the characteristics of PBcast influence CRBcast, we first investigate PBcast analytically and by simulations. PBcast has been studied before by simulations in some studies, e.g., [18,24,65]. We elaborate the problem here and provide asymptotic analysis for finding the optimal forwarding probability. Then, we investigate the effectiveness of CRBcast. CRBcast not only is a reliable and energy efficient scheme, but is also a scalable scheme that requires no knowledge of the network topology. This property makes CRBcast to be a very desirable scheme in some applications.

For our problem, we consider the following setup. We assume a wireless network of N static nodes with omnidirectional antennas and transmission range r deployed uniformly at random in a field with area A. We model the network by a random geometric graph $G(N, r)$. We also assume that r is large enough so that $G(N, r)$ is connected. Furthermore, we consider lossless networks by assuming lossless channels and the existence of a medium access control (MAC) layer, which prevents collision of packets. That is to say, every packet sent by a node correctly reaches all its neighbors.

We also apply a more specific MAC scheme for our simulations in Section 5.4.4. In this MAC scheme, when a node is transmitting, all of its neighbors up to two hops will be silent and will not transmit. This avoids interference and the hidden terminal problem [72]. We refer to this MAC as *two-hop blocking MAC*. A similar MAC was considered in [51].

In our model, we consider only the energy spent for RF transmissions as in [74].

Therefore, the energy consumption is proportional to the number of packet transmissions in the network.

5.2 Asymptotic Analysis of Probabilistic Broadcast

PBcast is a scalable and simple scheme for broadcasting in multi-hop wireless networks. In PBcast, every node relays a packet that it receives for the first time with some probability p. Let us assume that at any time slot, in which a packet travels in the network, we color each node as black or white with probability p and $1-p$, respectively. Therefore, black nodes forward a new received packet while white nodes do not forward it. Suppose B and W are the sets of the black and white nodes, respectively. Let $G_B(p,r) = G(N,r)\backslash W$ represents the subgraph of $G(N,r)$ induced by B. The following remark can be concluded.

Remark: The problem of energy-efficient and reliable broadcasting in wireless networks using PBcast can be rephrased as finding the lowest p such that $G_B(p,r)$ is connected and every white node is in the single-hop neighborhood of at least one black node.

For clarification, consider Figure 5.1 in which a random deployment of 16 nodes is depicted. The source node is in the center. The forwarding probability is $p = 1/2$. There is an edge between two nodes if they are in the transmission range of each other. All the black nodes are connected, and every white node has at least one black node as its neighbor. Therefore, if the source node broadcasts a packet, all the nodes in $G(N,r)$ receive it. This implies that reliability of PBcast is one in this case (although the number of transmissions is not minimum). Next, we will show that there exists a threshold p^{th} such that the reliability of PBcast is equal to one asymptotically almost surely if and only if $p > p^{th}$.

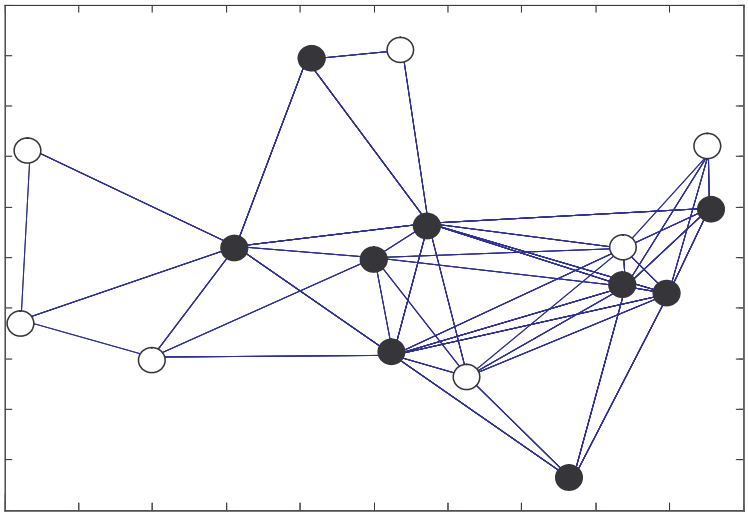


Figure 5.1: A random deployment of 16 nodes in a field. At any instant, each node is colored black with probability p and is colored white with probability $1-p$. In this figure $p = 1/2$.

5.2.1 Connectivity of Black Nodes

Our goal is to find the condition under which $G_B(p, r)$ is connected. Connectivity of geometric graphs is a well studied subject. Gupta et al. [17] derived the condition for asymptotic connectivity of a random geometric graph. Later, [47] generalized the case and considered the connectivity of a random geometric graph with unreliable nodes in which each node fails with probability $1-p$. Interestingly, the connectivity of random graphs with unreliable nodes can be used to find the connectivity condition in $G_B(p(N), r(N))$ by mapping the failed nodes into white nodes and the active nodes into black nodes. The following theorem states the necessary and sufficient condition for connectivity of $G_B(p(N), r(N))$ for large values of N. For proof we refer readers to [47].

Theorem 5.1. *Consider a random graph $G_B(p(N), r(N))$. Let A denote the area of a square field in which we deployed the N nodes at random. Assume $p(N) \times N \to \infty$ as $N \to \infty$ and let $\omega(N)$ be any slowly growing function such that $\omega(N) \to \infty$ as*

$N \to \infty$. *Suppose we have*

$$\lim_{N\to\infty} \left(\frac{p(N)N\pi r^2(N)}{(\ln(p(N)N) + \omega(N))\, A} \right) = \alpha. \tag{35}$$

If $\alpha > 1$, *then* $G_B(p(N), r(N))$ *is connected asymptotically almost surely. On the other hand, if* $\alpha < 1$, *then* $G_B(p(N), r(N))$ *is not connected asymptotically almost surely.*

Using Theorem 5.1, we conclude that p^{th} for the connectivity of $G_B(p(N), r(N))$ is given by:

$$\frac{p^{th}\pi r^2(N)}{A} = \frac{\ln(p^{th}N) + \omega(N)}{N} \quad \text{as } N \to \infty. \tag{36}$$

5.2.2 Sufficient Condition for the Coverage of White Nodes

In this section, we examine the sufficient condition under which every white node has at least one black node as its neighbor. We prove that (36) is a sufficient condition.

Theorem 5.2. $p > p^{th}$, *where* p^{th} *satisfies (36), provides a sufficient condition for the coverage of all white nodes.*

Proof. Let us define C_i as the event that the i^{th} node has at least one black node neighbor (within a single hop) provided that the i^{th} node is white. We show that asymptotically $Pr(\bigcap_{i=1}^{N} C_i)$ goes to one if (36) is satisfied. By union bound we have:

$$Pr(\bigcap_{i=1}^{N} C_i) \geq 1 - NPr(\overline{C_i}). \tag{37}$$

Moreover,

$$\begin{aligned} Pr(\overline{C_i}) &= \left(p(1 - \frac{\pi r^2}{A}) + (1-p) \right)^{N-1} \\ &= \left(1 - p\frac{\pi r^2}{A} \right)^{N-1}. \end{aligned} \tag{38}$$

Using inequality $1 - x < e^{-x}$ and (36) we have:

$$\begin{aligned} \left(1 - p\frac{\pi r^2}{A} \right)^{N-1} &< e^{-p\frac{\pi r^2}{A}(N-1)} \\ &< e^{-\ln(p^{th}N) - \omega(N)}. \end{aligned} \tag{39}$$

Therefore,

$$Pr(\bigcap_{i=1}^{N} C_i) \geq 1 - \frac{N}{p^{th}N} e^{-\omega(N)} \rightarrow 1 \quad \text{as } N \rightarrow \infty. \tag{40}$$

This concludes the assertion. □

From Theorems 5.1 and 5.2, we conclude the following corollary.

Corollary 5.1. *Broadcasting a single packet to a network by PBcast achieves reliability one asymptotically almost surely if and only if $p > p^{th}$, where p^{th} satisfies (36).*

So far, we have considered the case of broadcasting *a single packet*. We now study the case of broadcasting n_p packets in a multi-hop wireless network. Let R_1 denote the fraction of nodes that receive a particular packet in PBcast. It is clear that in a uniform packet dissemination, R_1 also denotes the probability that a node receives the packet. Since the transmissions of packets are independent, the probability R_{n_p} that a node receives all n_p packets is equal to $R_1^{n_p}$ for a uniform packet dissemination. However, packet transmissions are not uniform in general. For instance nodes at the neighborhood of the source always receive the packets while the border nodes receive less packets. Therefore, R_{n_p} is not equal to $R_1^{n_p}$ in general. In the next lemma, we derive bounds on R_{n_p}.[1]

Lemma 5.1. *Consider the PBcast protocol for broadcasting n_p packets in a large wireless network with N nodes. Let R_1 and R_{n_p} denote the probabilities that a node receives a particular packet and n_p packets, respectively. Then,*

$$R_1^{n_p} \leq R_{n_p} \leq R_1. \tag{41}$$

[1]Alternatively, broadcasting n_p packets can be performed such that a node keeps its forwarding status during the broadcasting session. This implies that a node is a forwarding node (with probability p) for the whole broadcasting session. In this case $R_{n_p} = R_1$. However, this unevenly distributes the energy consumption in the network, i.e., the fixed forwarding nodes consume much more energy than non-forwarding nodes. This is not desirable especially for sensor networks.

Proof. The right-hand side inequality is obvious. We prove the left-hand side inequality. Let us partition the nodes in the network into groups $1, 2, \ldots, j$ such that the packet dissemination in each group is uniform. Let the fraction of the nodes in the i^{th} group be α_i ($0 < \alpha_i \leq 1$ and $\sum_{i=1}^{j} \alpha_i = 1$). Also, let $R_{1,i}$ be the probability that a node in the i^{th} group receives a packet. A circular partitioning may be suitable since the nodes with similar distances from the source are expected to have the same probability of receiving the packets. Assuming that the number of nodes in each partition is large enough, it is clear that we have

$$R_1 = \sum_{i=1}^{j} \alpha_i R_{1,i}. \tag{42}$$

Since the packet transmissions are independent in PBcast and assuming uniform reception of data in each group of nodes, we have

$$R_{np} = \sum_{i=1}^{j} \alpha_i R_{1,i}^{n_p}. \tag{43}$$

Using Jensen's inequality, we conclude that $R_{n_p} \geq R_1^{n_p}$. □

Since energy consumption is proportional to the number of packet transmissions, it is also desirable to obtain the total number of required transmissions per original packet (N_{tx}/n_p). Since not all the black nodes receive a packet to transmit, N_{tx}/n_p in PBcast is upper bounded by the total number of black nodes plus one (for the source node), which is equal to $pN + 1$ on the average. Moreover, in the area spanned by the nodes that receive a particular packet, on the average, fraction p of them are transmitting nodes. Therefore, we have

$$N_{tx}/n_p = pNR_1. \tag{44}$$

5.2.3 Simulation Results for Probabilistic Broadcast

In this section, we demonstrate PBcast properties by simulation. First, we consider random deployment of N nodes uniformly in a field with area $A = 2000m \times 2000m$, for

$N = 10^4, 10^5$, and 10^6. For each N, we chose transmission range r using $\frac{\pi r^2}{A} > \frac{\ln N}{N}$ [17], so that the network $G(N, r)$ is connected. We computed p^{th}; the threshold for the reliable broadcasting of a single packet from (36). Table 5.1 gives the analytical results. We note that as N increases the required p^{th} decreases.

Table 5.1: The values of p^{th} for reliable broadcasting of a single packet in the geometric graph $G(N, r)$ deployed randomly in an area $A = 2000m \times 2000m$.

N	$r(m)$	p^{th}
10^4	50	0.43
10^5	20	0.34
10^6	8	0.25

Next, we consider the following network topology. We assume $N = 10^4$ nodes with transmission range $r = 50m$ are deployed uniformly at random in an area $A = 2000m \times 2000m$. We call this topology as $\mathcal{T}$ for our future references. We developed event driven softwares in C++ for our simulations.

We first verify our theoretical analysis for p^{th}. We also confirm that for reliable broadcasting of $n_p > 1$ packets, the required relaying probability is much higher than p^{th}. In Figure 5.2, the fraction of nodes that successfully receive a particular packet (denoted by R_1) and $n_p = 2000$ packets (denoted by R_{n_p}) are shown. Each point in the figure is the result of averaging over 300 different random graphs with the topology $\mathcal{T}$. We also depicted $R_1^{n_p}$, which is a lower bound for R_{n_p} by Lemma 5.1. We confirm that R_1 is very close to one for $p > 0.43$, which is the analytical threshold value given by Table 5.1. However, for $n_p > 1$, the reliability decreases and a larger forwarding probability p is needed. We note that for $R_{n_p} \approx 1$ in PBcast, p has to be very close to one. For example, forwarding probability of at least $p = 0.7$ is required for $R_{n_p} \geq 0.99$.

Next, we give the simulation results for the required number of transmissions per packet (N_{tx}/n_p) versus the forwarding probability p, since it is the criterion for energy

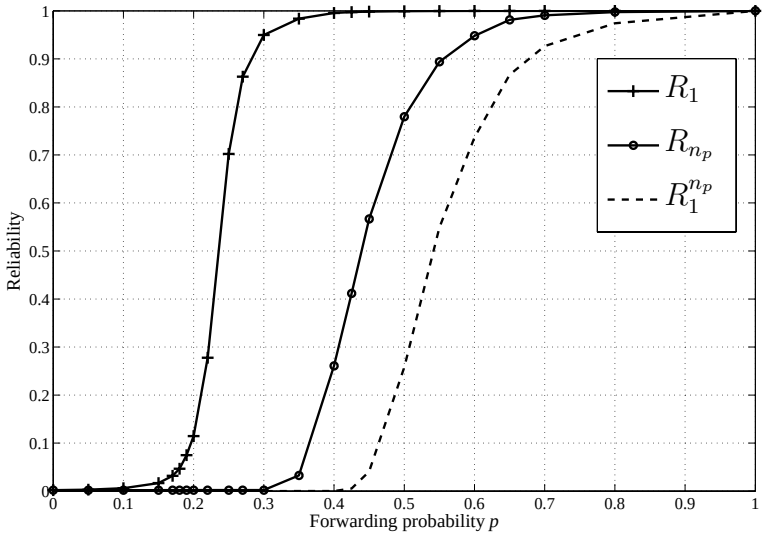


Figure 5.2: R_1, R_{n_p}, and $R_1^{n_p}$ versus forwarding probability p for a wireless sensor network with the topology $\mathcal{T}$ (n_p is equal to 2000).

consumption. Figure 5.3 shows N_{tx}/n_p versus p for the topology $\mathcal{T}$ when $n_p = 2000$.

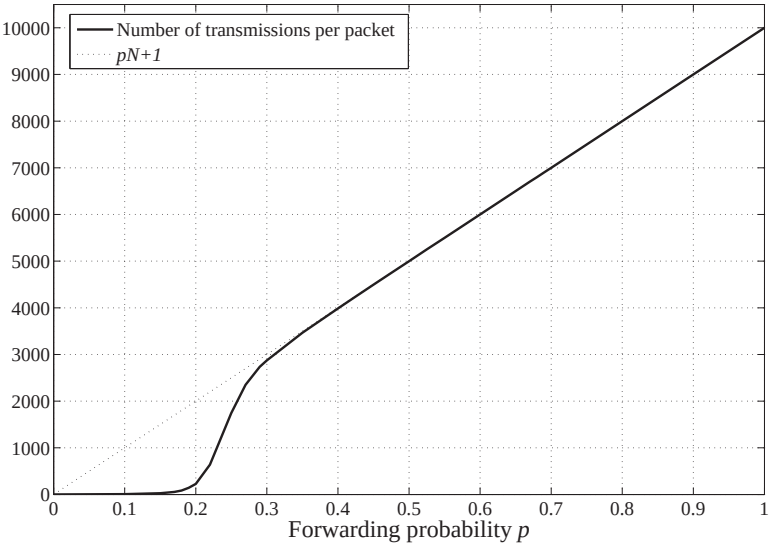


Figure 5.3: N_{tx}/n_p versus forwarding probability p for the network topology $\mathcal{T}$.

We note that when the number of required transmissions per packet is plotted using Equation (44), we get the same result as the simulation provided in Figure 5.3. As we can see N_{tx}/n_p is an increasing function of p. The greatest rate of increase in N_{tx}/n_p happens around $p \approx 0.24$. This point is the threshold for occurrence of a *giant component* in $G_B(p, r)$. Let p^G denote this threshold. As given in [47], the

asymptotic value of p^G can be calculated by

$$p^G N r^2/A = \lambda_c \approx 1.44. \tag{45}$$

We refer to the giant component again in Section 5.4, where we observe that the optimal value of p in our proposed protocol (p^*) is close to p^G. Therefore, (45) can be used for the approximation of p^*.

Figures 5.4, 5.5, 5.6, 5.7, 5.8, 5.9 depict screen shots of broadcasting *a single packet* using PBcast with different forwarding probabilities p. The network topology $\mathcal{T}$ is considered, and the source node is located in the center of the area. We see three types of nodes. Black, white, and red[1]. The black nodes are those that forwarded the packet after receiving it. The white nodes are those that only received the packet (did not forward it). The red nodes are those that did not receive the packet. As can be seen, when p is small only a very small fraction of nodes received the packet. For example, for $p = 0.2$ only 12% of the nodes received the packet. However, for $p = 0.25$, 74% of nodes received the packet. This is the point that a giant component has happened. More nodes received the packet when p is increased.

[1]Nodes in red may appear as light gray in a non-color print version of this document.

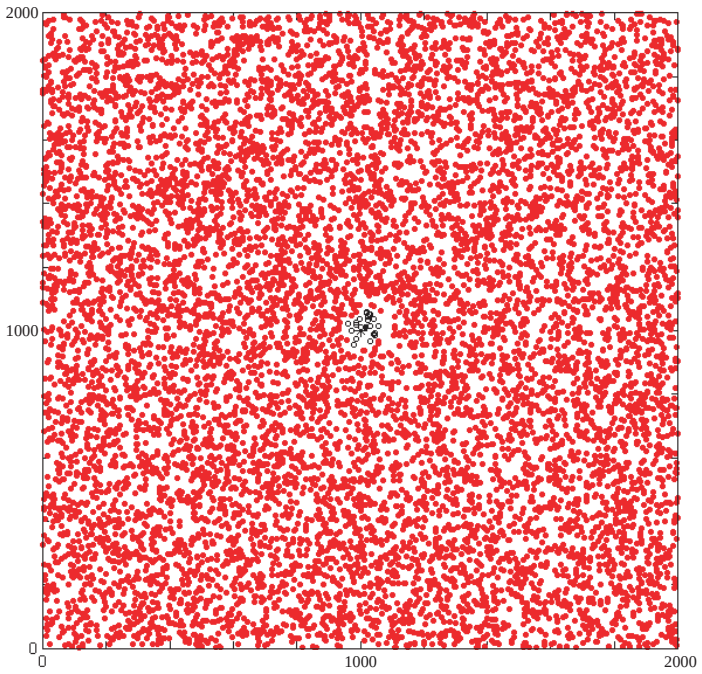


Figure 5.4: A screen shot of broadcasting a single packet over a network with topology $\mathcal{T}$ using PBcast with forwarding probability $p = 0.1$

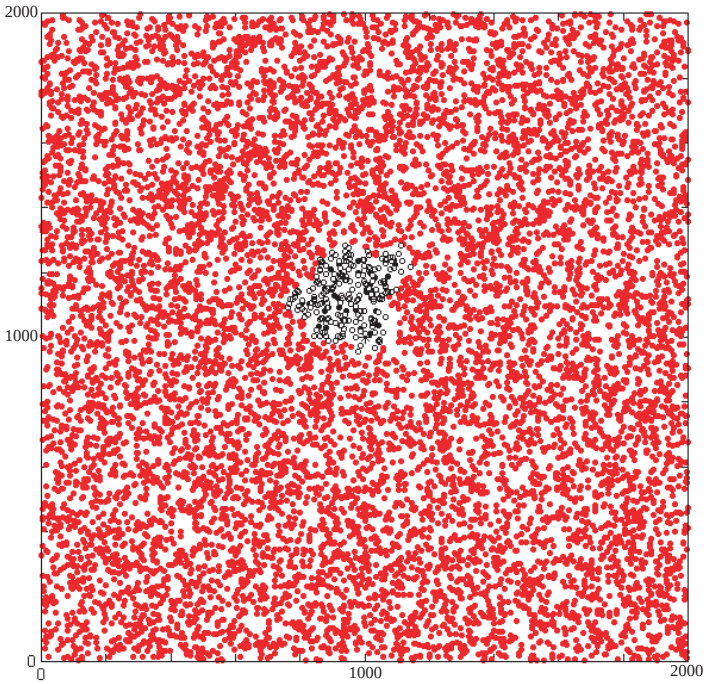


Figure 5.5: A screen shot of broadcasting a single packet over a network with topology $\mathcal{T}$ using PBcast with forwarding probability $p = 0.15$

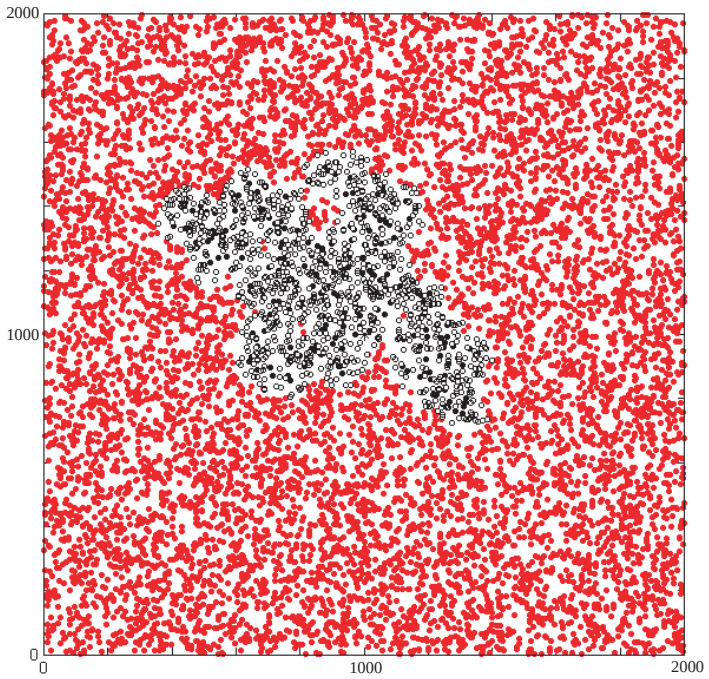


Figure 5.6: A screen shot of broadcasting a single packet over a network with topology $\mathcal{T}$ using PBcast with forwarding probability $p = 0.2$

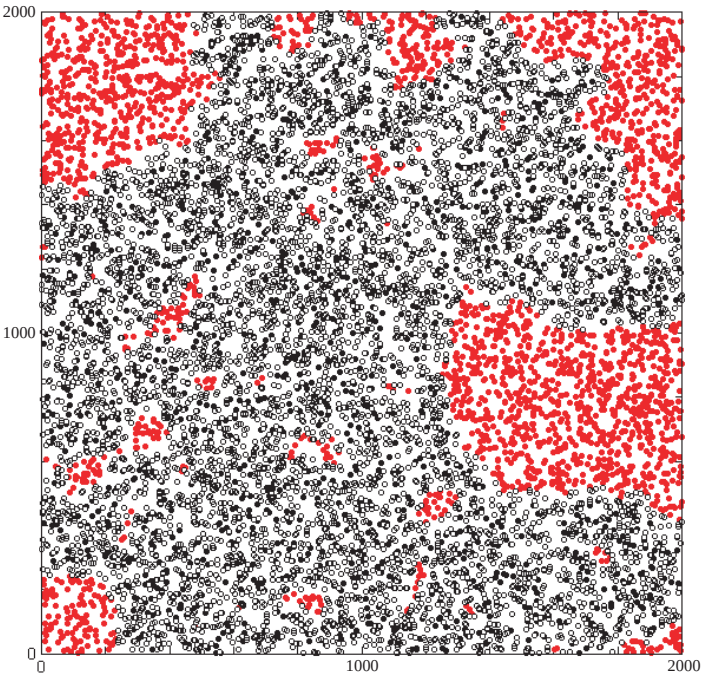


Figure 5.7: A screen shot of broadcasting a single packet over a network with topology $\mathcal{T}$ using PBcast with forwarding probability $p = 0.25$

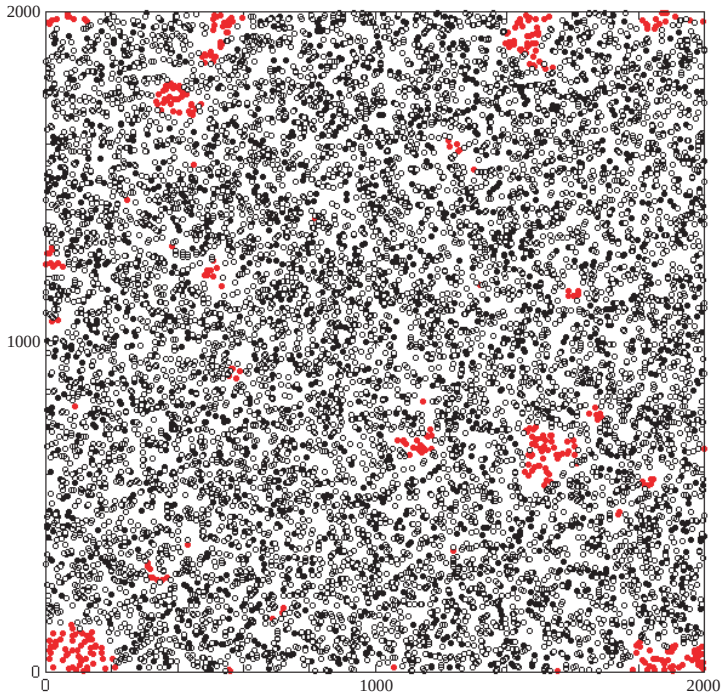


Figure 5.8: A screen shot of broadcasting a single packet over a network with topology $\mathcal{T}$ using PBcast with forwarding probability $p = 0.3$

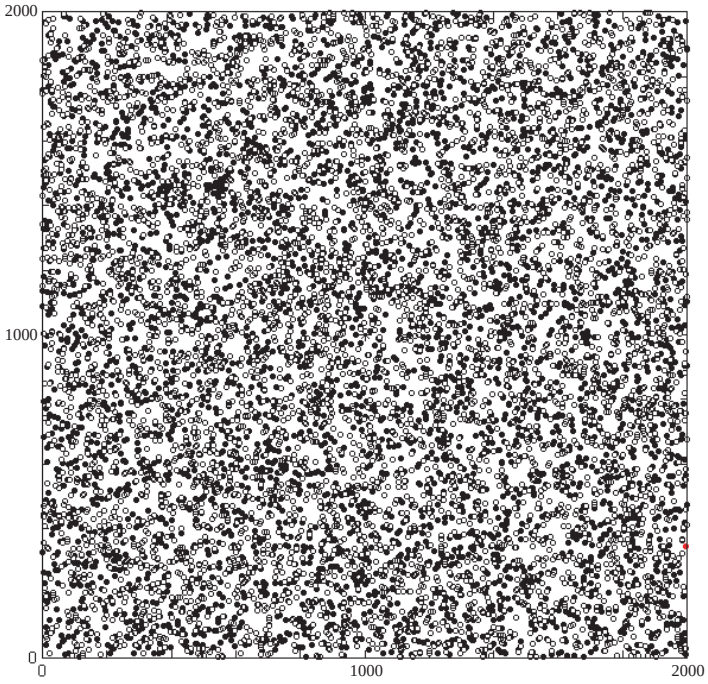


Figure 5.9: A screen shot of broadcasting a single packet over a network with topology $\mathcal{T}$ using PBcast with forwarding probability $p = 0.5$

5.3 RBcast: Rateless Probabilistic Broadcast

We noted in Section 5.2.3 that as the number of packets for broadcasting (n_p) increases, the forwarding probability p also has to increase so that PBcast provides high reliability. The reason is that every single packet must be received by all the nodes in the network; however, as the number of packets increases the probability that some nodes miss some packets increases as well. Therefore, as n_p increases, the performance of a reliable PBcast becomes close to the performance of flooding ($p = 1$), and PBcast will not be energy efficient.

To overcome this problem, we investigate the following potential solution. The source node encodes the data using a channel code before broadcasting it. The encoded packets then are broadcast using PBcast. By doing this, nodes in the network do not require to receive all the broadcast packets. They only need to receive enough packets to be able to decode the data. We use rateless codes, because these codes do not require any information about the channel and also for their simple encoding and decoding. A node is able to decode and retrieve the original n_p packets if it receives at least $n_p\gamma$ encoded packets, where $\gamma \geq 1$ is the overhead of rateless codes. We refer to this scheme as *rateless probabilistic broadcast* (RBcast). Figure 5.10 illustrates the schematic of RBcast. Next, we consider broadcasting $n_p = 2000$ packets over a

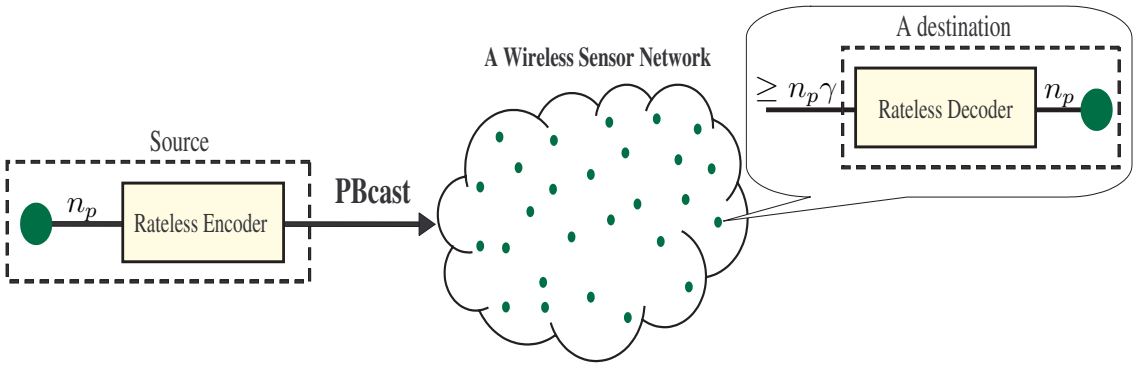


Figure 5.10: The schematic of RBcast.

random graph with the topology $\mathcal{T}$ using RBcast. In our simulations, we use the

rateless code in [39] that results in a decoding failure probability less than 10^{-8} for $n_p = 2000$ and $\gamma = 1.03$. The source node generates encoded packets, and these packets are broadcast in the network based on PBcast. The broadcast session ends if all the nodes in the network receive at least $n_p\gamma = 2060$ packets. In this way, it is guaranteed that all the nodes can retrieve the original data with probability almost one. Figure 5.11 depicts the number of transmissions per packet versus p for RBcast.

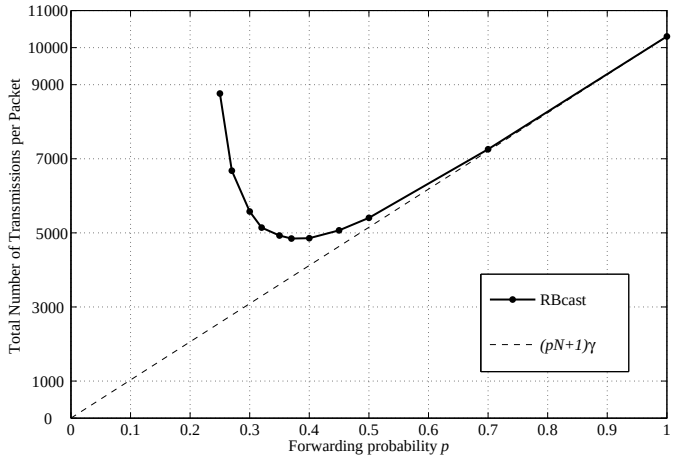


Figure 5.11: The number of transmissions per packet for providing reliability of at least $1 - 10^{-8}$ in RBcast. We considered $n_p = 2000$ and $\gamma = 1.03$ for a random network with topology $\mathcal{T}$.

As is shown, when p is very small, a large number of transmissions is required. The minimum number of transmissions per packet is about 4850 and happens at $p = 0.37 \sim 0.4$. To compare RBcast with PBcast, we recall that $N_{tx}/n_p \approx 7000$ transmissions are necessary for gaining reliability of 99% in PBcast. Therefore, RBcast results in 30% less transmissions in comparison with PBcast. This result is promising; however, we next propose another broadcasting scheme which is similar to RBcast and improves the efficiency even more.

5.4 CRBcast: Collaborative Rateless Broadcast

In this section, we propose a scheme for reliable and energy-efficient broadcasting in multihop wireless networks. In the proposed scheme, we provide reliability by employing rateless coding. In rateless coding, potentially unlimited number of encoded packets can be generated by simple XOR operations on the original packets. A receiver is able to retrieve the original packets when it receives a sufficient subset of the encoded packets. No information about the channel is needed, and there is no need for in-sequence data delivery. In other words, it is not important which encoded packets a node receives. The only important thing is the number of the received encoded packets. We propose to use rateless codes in conjunction with a light-weight PBcast algorithm, which is a simple and scalable broadcasting scheme. In light-weight PBcast, we choose a small value for p. Light-weight PBcast reduces the probability of multiple reception of the same packet. This prevents many redundant transmissions. Therefore, the total number of transmissions decreases as does energy consumption. Since not every node in the network can recover the original packets, we need a second recovery phase to guarantee reliability. Next, we explain our proposed scheme, which we call *collaborative rateless broadcast* (CRBcast). CRBcast consists of the following two phases.

5.4.1 The CRBcast Protocol

CRBcast consists of the following two phases.

5.4.1.1 CRBcast-Phase I

In Phase I, the original n_p packets at source are first encoded to $n_p\gamma$ encoded packets. Then, the encoded packets are broadcast using PBcast.

At the end of Phase I, some nodes, referred to as *complete nodes*, receive all $n_p\gamma$ different packets and can reconstruct the original packets. We refer to the rest of the nodes that did not receive $n_p\gamma$ different packets as *incomplete nodes*. The

number of complete nodes after Phase I and the number of transmissions per packet can be approximated by $NR_{(n_p\gamma)}$ and $N_{tx}/n_p = pNR_1\gamma$, respectively, as discussed in Section 5.2.

The parameter $\gamma \geq 1$ is the overhead imposed by the rateless coding and is selected such that the probability of successful decoding ($P_R(n_p, \gamma)$) is almost one. For large values of n_p, the probability P_R for rateless codes described in [39,68] approaches one when γ is slightly greater than one, while the complexity of encoding and decoding is linear.

5.4.1.2 CRBcast-Phase II

Phase II is based on a simple collaboration among complete and incomplete nodes such that each complete node sends only once the required number of packets to its neighbors to complete them. The new complete nodes repeat this and the process continues until no new complete nodes remain. Therefore, we need two types of very short handshake messages between complete and incomplete nodes: advertisement messages (ADV) and request messages (REQ). Whenever a node becomes complete, it advertises its completeness to its neighbors once using an ADV message (which includes the ID of the complete node and some flag bits that indicate the message is an ADV message). Any incomplete neighbor that receives the ADV message responds by a REQ message including the required number of new packets for its completion, the ID of the complete node, and some flag bits associated with the REQ message. For example, suppose a node has already received n_1 packets when it receives an ADV message. This node would use an REQ message asking for $n_p\gamma - n_1$ new packets. It should be noted that the complete node generates new encoded packets by performing rateless-encoding on the retrieved original packets. In this way, it can be guaranteed that *new* encoded packets are sent to the incomplete node.

Here are more details about the collaboration between nodes. If an incomplete

node receives multiple ADV messages from different neighbors, it will respond only to the one with the lowest ID number. Moreover, after sending an ADV message, each complete node waits long enough to receive the REQ messages from all of its incomplete neighbors. When a complete node receives different REQ messages from different incomplete neighbors, it sends the maximum number of the required packets. Each complete node advertises once and each incomplete node requests once. In a connected network with lossless channels, we prove in Lemma 5.2 that all nodes are eventually completed. The total number of ADV messages (n_{adv}) plus REQ messages (n_{req}) is less than $2N$ packets for the whole broadcasting session (equivalently, $2N/n_p$ transmissions per packet). These handshake messages result in negligible overhead due to their relatively short packet sizes and the fact that n_p is large.

Lemma 5.2. *Let $G(N, r)$ be a lossless, stationary, and connected wireless network. Then, CRBcast provides reliability one independent of the value p.*

Proof. Let S denote the set of complete nodes after the completion of Phase II. We note that S is a nonempty set since the neighbors of the source are definitely complete. If $|S| = N$ then we are done. Otherwise, connectivity of $G(N, r)$ implies that there exists at least an edge (i, j) such that $i \in S$ and $j \notin S$. However, this contradicts the protocol in Phase II. Since, once node i becomes complete, it completes all its incomplete neighbors based on the ADV and REQ mechanism in CRBcast. Therefore, j is also complete, i.e., $j \in S$, which contradicts our assumption. □

5.4.2 Extensions of CRBcast

In this section, we propose two extensions of CRBcast. These extensions can further reduce the number of transmissions.

Probabilistic Advertising in Phase II: In the original CRBcast protocol, every complete node advertises its status to its neighbors once. However, to decrease the number of transmissions, we can modify Phase II such that every complete node

advertises once with probability p_{adv}. Similar to Corollary 5.1, asymptotically, the condition $p_{adv} > p^{th}$ is sufficient for reliable broadcasting. However, it should be noted that although the total number of advertisements reduces by a factor of $1 - p_{adv}$, the total number of transmissions does not reduce by the same factor. The rational is that even if a complete node advertises, it does not send any encoded packet if all of its neighbors are complete.

Multi-Stage Recovery in Phase II: In our original CRBcast protocol, every complete node sends the maximum number of packets that is required by its neighbors to become complete. However, in many cases, a smaller number of transmissions suffices due to the redundancy that is inherent in the wireless medium. For example, consider the following scenario shown in Figure 5.12. Suppose nodes A and E are complete. However, nodes B, C, D, and F require 50, 100, 500, and 250 packets, respectively. In the original CRBcast protocol, nodes A and E must send 500 and 250 packets, respectively. However, since node D also receives the transmitted packets from node E, a more efficient scheme would only require that nodes A and E send 250 packets each. In general, finding the optimal number of transmissions for each

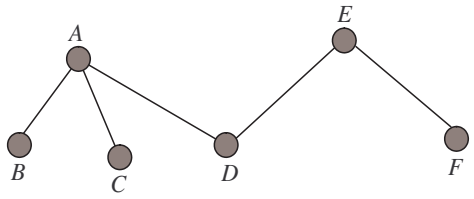


Figure 5.12: A scenario in which multi-stage recovery reduces the number of transmissions.

complete node needs a global knowledge of the network status, which is not practical. Here, we propose a simple modification to CRBcast for reducing the number of transmissions.

We put an upper bound n_{max} on the number of packets that each complete node sends after receiving the requests from its neighbors. Therefore, a complete node may not complete all its neighbors at the first try. Instead, it would hope that

its neighbors would receive more packets from other complete nodes. However, to ensure that all the neighbors become complete ultimately, a complete node advertises periodically with sufficient time interval between advertisements until all its neighbors become complete (no REQ message is received). If we consider $n_{max} = \rho n_p \gamma$ for some $0 < \rho \leq 1$, we have $n_{adv}, n_{req} \leq \lceil 1/\rho \rceil N$. Therefore, the total overhead because of the handshaking in Phase II is upper bounded by $2\lceil 1/\rho \rceil N$. Decreasing n_{max} reduces the number of data packet transmissions. However, it increases the overhead of handshake messages.

5.4.3 Simulation Results for CRBcast: Time-Relaxed Implementation

Here, we provide the simulation results for the number of transmissions per packet (N_{tx}/n_p) that is necessary for all nodes to receive at least $n_p\gamma$ distinct encoded packets as a function of p. This quantity is the indication of the energy consumption in the network. In our simulations, we used the rateless code in [39] that results in a decoding failure probability less than 10^{-8} for $n_p = 2000$ and $\gamma = 1.03$. Therefore, the recovery probability (P_R) is almost one. We considered the network topology $\mathcal{T}$ described in Section 5.2. Figure 5.13 depicts the result. We also included the number

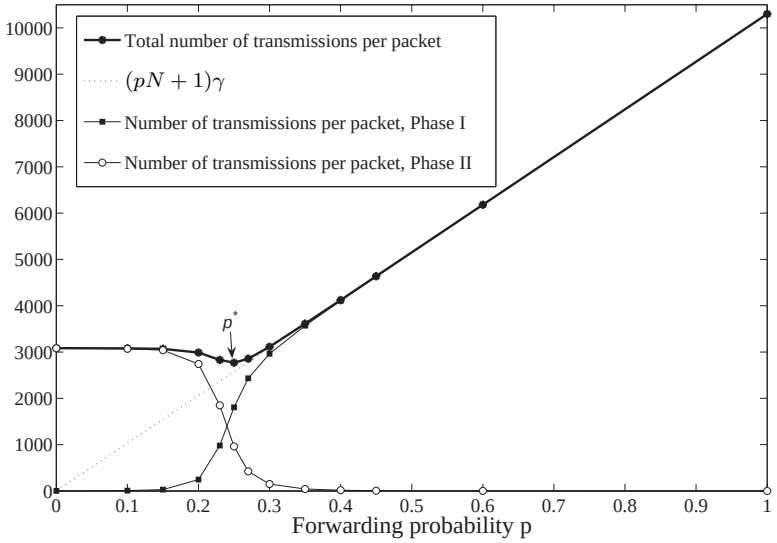


Figure 5.13: The number of transmissions per packet for providing reliability at least $1 - 10^{-8}$ in CRBcast. We considered $n_p = 2000$ and $\gamma = 1.03$ for the network topology $\mathcal{T}$.

of transmissions per packet in Phase I and II. As expected, the total number of transmissions in Phase I increases with p, while the total number of transmissions in Phase II decreases with p. The minimum total number of transmissions occurs at $p^* = 0.25$, for which N_{tx}/n_p is equal to 2769 transmissions per packet. This results in saving of more than 72% packet transmissions in comparison with flooding. To compare CRBcast with PBcast, we recall that $N_{tx}/n_p \approx 7000$ transmissions are necessary for gaining reliability of 99% in PBcast. Therefore, CRBcast results in 60% less transmissions in comparison with PBcast, while CRBcast's reliability is almost one. To compare CRBcast with RBcast, we recall that RBcast can achieve the minimum of $N_{tx}/n_p \approx 4850$. Therefore, CRBcast outperforms RBcast in terms of number of transmissions per packet by about 43%. This shows the effectiveness of introducing the second recovery phase in CRBcast.

It is worth noting that using (45), the giant component for $G_B(p, r)$ happens theoretically at $p^G \approx 0.24$. We observe that p^* is close to p^G. We can explain this by recalling that at p^G a large fraction of nodes receive each packet (in Phase I). This balances the recoveries in the two phases. Therefore, (45) can be used as an approximation for p^*. We should also mention that the number of handshaking transmissions is at most $2N = 20000$ packets, which is about 0.3% of the total number of transmissions. Considering that handshake packets are much shorter than data packets, we see that this overhead is wholly negligible.

Next, we implement the two modifications of CRBcast for the above example when $p = 0.25$. First, we consider probabilistic advertisements with $p_{adv} = 0.45$. The number of transmissions per packet is reduced to 2587 while the reliability is 0.9999. This is 6.5% improvement in energy consumption in comparison with the original CRBcast. We also consider the multi-stage recovery in Phase II for CRBcast with $p = 0.25$. We consider $n_{max} = 50$. The number of transmissions per packet including the number of handshake messages is reduced to 2584. Thus, almost the

same improvement as probabilistic advertisement is achieved. Table 5.3 summarizes N_{tx}/n_p and reliability in different methods.

Table 5.2: Comparison of energy consumption and broadcast reliability in different broadcast schemes. The value of p is chosen as 0.7, 0.37, and 0.25 for PBcast, RBcast and CRBcast-based methods, respectively.

Broadcasting Scheme	N_{tx}/n_p	Reliability
Flooding	10001	1
PBcast	6999	0.99
RBcast	4850	1
CRBcast	2769	1
CRBcast with $p_{adv} = 0.45$	2587	0.9999
Multi-stage CRBcast with $n_{max} = 50$	2584	1

Finally, we implement the approximation algorithm for finding an MCDS proposed in [37], which is a centralized algorithm and needs full knowledge about the topology of the network. The algorithm results in 1956 transmissions per packet, which outperforms CRBcast. However, the algorithm in [37] is neither scalable nor practical for large networks. Moreover, it has the uneven-load-balancing problem (i.e., some nodes run out of battery power much faster than the others). More importantly, broadcast schemes that are based on finding an MCDS cannot be adapted easily for mobile or lossy networks. In contrast, generalization of CRBcast for lossy and mobile networks is straightforward. Phase I remains the same (however, a higher p would be required). In Phase II, each complete node including the source node sends ADV. Lost ADV messages can be compensated by periodical re-advertisements. Lost REQ and data packets can be compensated by requesting the number of necessary packets that do not arrive within a fixed time period.

5.4.4 Simulation Results for CRBcast: Time-Constrained Implementation

In our simulations in Section 5.4.3, we made some assumptions that may not be necessary or very practical. The first assumption disregarded the waiting time to access

the channel. The second assumption in our simulations of CRBcast was that Phase I and Phase II are executed sequentially. In other words, we assumed that Phase II starts when Phase I is over, i.e., all the nodes have already forwarded their packets in Phase I. Using these assumptions, we were able to carry out theoretical work and find the optimal forwarding probability. We can argue that these assumptions could be realized if we do not have any constraint on the latency of broadcasting, i.e., if we give broadcasting sufficient time for completion. We refer to this implementation of CRBcast as *time-relaxed* implementation.

In this section, we repeat the experiment assuming the nodes that have some packets to send will *contend* for a channel to avoid any collision. For this purpose, we consider the *two-hop blocking MAC* scheme. We assume that the complete transmission of one packet from a node to its one-hop neighbors takes one time unit. By considering such a MAC layer, we will also be able to compare the latency of the broadcasting algorithms.

Bringing the MAC into the picture, slightly changes the implementation of CRBcast in that we no longer can assume the separation of Phase I and Phase II. This means while some of the nodes in the network may be in the probabilistic forwarding phase, other nodes are in Phase II. A node is in Phase I at the beginning, and it forwards the packets that it receives for the first time with a probability p, whenever it gets the channel. If the channel is not free, it will put the packets in its queue and send one packet at a time, whenever the channel becomes available. A node is considered to be in Phase II if it is either complete or is a neighbor of a complete node (if it hears an ADV message). In these cases, the node will not continue to send those packets that are waiting in its queue. Instead, if it is a complete node, it will re-encode and send new packets to its neighbors based on the requests that it receives. If it is a neighbor of a complete node, it will request the number of packets it requires to be complete. Upon reception of the required packets, it becomes a complete node

and, when it has the channel, sends ADV message. This process continues until all the nodes become complete.

We compare CRBcast, PBcast, and another scheme called *Multipoint Relaying* (MPR) [51]. In MPR, each node i is aware of its neighborhood up to two hops and selects a subset $\mathcal{M}_i$ of its one-hop neighbors as forwarding nodes such that for any node j that is two hops away from i, there exists a node in $\mathcal{M}_i$ that is connected to j. Therefore, if i transmits a packet and only nodes in $\mathcal{M}_i$ forward it, all the nodes in the two-hop neighborhood of i will receive the packet. Yet, when broadcasting is performed, a node k forwards a message received from node i if and only if $k \in \mathcal{M}_i$.

Figure 5.14 compares N_{tx}/n_p with respect to forwarding probability p when $n_p = 2000$ packets are broadcast over the network topology $\mathcal{T}$ using CRBcast for both time-relaxed and time-constrained cases. We see that there is discrepancy between the two graphs. This is because of different simulation setups. For example, we see that for large values of p, time-constrained CRBcast has smaller N_{tx}/n_p. This is because there are some nodes in the network that start their Phase II before completing their Phase I. As can be seen, similar to the time-relaxed case, there is an optimal forwarding probability for the time-constrained case, and the corresponding optimal forwarding probability values are close. The minimum N_{tx}/n_p, which is equal to 3053 transmissions per packet, occurs at $p = 0.27$.

Figure 5.15 depicts the latency of PBcast and CRBcast versus forwarding probability p for broadcasting $n_p = 2000$ packets over the network topology $\mathcal{T}$. The two-hop blocking MAC was considered for both schemes. Clearly, latency is an increasing function of p for PBcast. The case is different for CRBcast. When p is very small, each packet is forwarded by very few nodes in Phase I. Therefore, most of the load will be in Phase II, which is more time consuming. This causes a high latency. By increasing p, a better balance between Phase I and Phase II is in place. Hence,

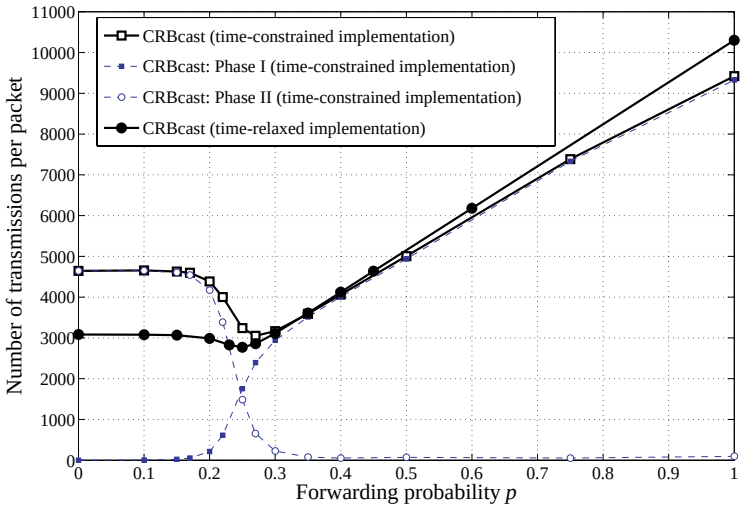


Figure 5.14: N_{tx}/n_p versus forwarding probability p for CRBcast over the network topology $\mathcal{T}$ for two different implementations.

we expect that the latency would decrease since more nodes can forward the packets simultaneously. However, we should note that increasing p beyond a threshold causes many unnecessary transmissions, which increases the latency. This explains the variation of CRBcast's latency in Figure 5.15.

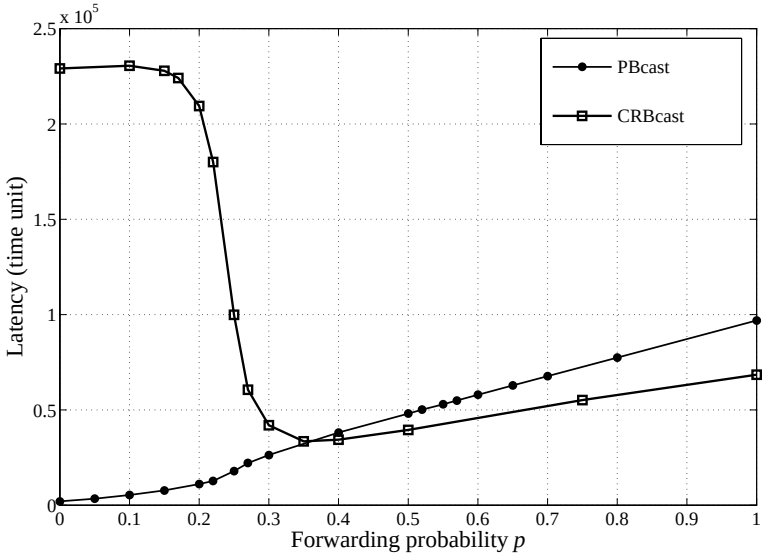


Figure 5.15: Latency of PBcast and CRBcast versus forwarding probability p for broadcasting $n_p = 2000$ packets over the network topology $\mathcal{T}$. The two-hop blocking MAC was considered for both schemes.

Table 5.3 summarizes the results for the minimum number of transmissions per

packet and the corresponding latency for different broadcast schemes when the two-hop MAC is considered. As can be seen, CRBcast has 69.5%, 56.4%, and 63.3% less energy consumption in comparison with flooding, PBcast ($p = 0.7$), and MPR, respectively. In terms of latency, PBcast and CRBcast are quite close, though CRBcast has slightly lower latency.

Table 5.3: Comparison of energy consumption and latency in different broadcast schemes. The value of p is chosen as 0.7 and 0.27 for PBcast and CRBcast, respectively.

Broadcasting Scheme	N_{tx}/n_p	Latency (time unit)
Flooding	10001	9.7e4
PBcast	6999	6.7e4
CRBcast	3053	6.1e4
MPR	8311	8.0e4

5.5 Conclusion

In this chapter, we studied the problem of reliable and energy-efficient broadcasting in wireless sensor networks when no knowledge of the topology of the network is available. We first studied the probabilistic broadcast (PBcast) scheme as a simple and scalable method. We derived the formula for the optimal forwarding probability p^{th} when only a single packet is broadcast over a network with a large number of nodes. This p^{th} is the minimum forwarding probability that guarantees almost surely all the nodes in the network receive the packet. We further showed that to reliably broadcast a large number of packets, a large forwarding probability (p) has to be used for PBcast. This makes PBcast energy inefficient.

Next, we studied the integration of rateless codes into PBcast. To do this, the original data is first encoded at the source using a rateless (Fountain) code. The encoded packets are then broadcast by PBcast. We called this scheme rateless probabilistic broadcast (RBcast). We showed that this scheme could be efficient and will outperform PBcast if p is not too small. We then modified RBcast and proposed our

main, even more efficient, two-phase broadcast scheme, referred to as collaborative rateless broadcast (CRBcast). The first phase of CRBcast is similar to RBcast. In this phase, however, it is not required that all nodes be able to decode the data. The second recovery phase, which is based on simple collaborations of the nodes, ensures that all the nodes can recover the original data.

We studied two different implementations for CRBcast, namely time-relaxed and time-constrained. In the former, the latency of broadcasting is not an issue, and an ideal MAC and relaxed timing are considered. These assumptions made the analysis more tractable. In the time-constrained implementation, a more realistic MAC referred to as two-hop blocking MAC is in place. We showed by simulation that CRBcast substantially reduces energy consumption for reliable broadcasting in comparison with other schemes such as flooding, PBcast, RBcast, and MPR. A very important property of CRBcast is that it is not only a reliable and energy-efficient scheme, but also a scalable and practical one that does not require any information about the network topology. Hence, CRBcast can be easily generalized for mobile and lossy networks.

CHAPTER VI

EFFICIENT BROADCASTING IN WIRELESS AD-HOC AND SENSOR NETWORKS WITH KNOWLEDGE OF LOCAL TOPOLOGY

6.1 Introduction

In Chapter 5, we proposed the CRBcast protocol, which is a viable approach for reliable and energy-efficient broadcasting in multihop wireless sensor networks that also addresses scalability and load balancing, while requiring no knowledge of the network topology. However, for the cases in which some local information about the network is available, we can exploit the extra information and improve the energy efficiency of broadcasting. In this chapter, we investigate reliable and energy-efficient broadcasting in multihop wireless sensor networks assuming the *availability of local information* about the neighborhood around each node. By the availability of local information, we mean that each node knows its hop-distance from the source and those of its neighbors. We propose a scheme, referred to as *Fractional Transmission Scheme* (FTS), which is scalable and has low complexity[1] [62,63]. FTS utilizes rateless coding and the broadcasting nature of wireless channels to reduce energy consumption while ensuring the reliable delivery of packets to all the nodes in a network, even if the links in the network are lossy. For that, FTS even does not require any knowledge of the values of link losses.

In FTS, based on its local knowledge of the network topology, each node converges on a fraction of encoded data that it must send after decoding and re-encoding. In

[1]This study was done collaboratively with my colleague Badri N. Vellambi.

other words, different neighbors of a node share the data delivery, and each node sends only a fraction of the total encoded packets required by a receiving node. If each node knows its geographical location and those of its neighbors, energy consumption can be even further reduced. In this case, each node can adapt its transmission power based on its farthest neighbor. This scheme is referred to as FTS_{adapt}.

We analyze FTS for grid networks and compare the results with the corresponding optimal case. Further, extensive simulation results are provided considering grid and random geometric networks for both lossless and lossy networks. These results suggest better or comparable energy consumption in comparison with some other competitive techniques such as CRBcast, Network Coding (NC), Broadcast Incremental Power (BIP) [74], and Multipoint Relaying (MPR) [51]. Due to their low complexity and requirement of only local information, unlike NC and BIP, FTS and FTS_{adapt} can easily be applied to large networks.

6.2 Network Models and Terminologies

For simplicity and tractability of analysis, we consider the following setup. A network of N static nodes with omnidirectional antennas and a transmission range r is assumed. Such a wireless network can be modeled as a geometric undirected graph $G(V, E)$, in which V is the set of nodes and E is the set of edges. There is an edge between two nodes u and v if and only if the Euclidean distance between u and v is less than or equal to r. We also assume that r is large enough so that $G(V, E)$ is connected. The neighborhood $\mathcal{N}_r(u)$ of a node u is the set of all nodes that are within a distance of r units away from u. The wireless nature of the network is modeled by the assumption that each node u transmits the same message to all the nodes in its neighborhood. For a graph $G(V, E)$ with a unique source node s, $\mathbf{H} : V \longrightarrow \mathbb{N} \cup \{0\}$ denotes the *hop-distance* function that maps each node v to the length of the shortest path connecting s to v. Finally, we assume that the network is equipped with a

one-to-one node identifier function $id : V \longrightarrow \mathbb{N}$ such that each node v knows $id(v)$. For simplicity, we assume that for any $u, v \in V$, if u is closer to the source than v then $id(u) < id(v)$.

We consider only the energy expended on RF transmissions (as in [74]) as the cost criteria for energy-efficiency. The cost for sending a packet from a node with transmission range r is taken to be r^2. We also denote the *number of transmissions per packet per node* by $\mathcal{N}_{/p/n}$ and the corresponding *energy consumption per packet per node* by $\mathcal{C}_{/p/n}$.

In our simulations for lossy networks, we consider that the transmissions in networks are subject to distance attenuation and Rayleigh fading. Therefore, when a node u with a nominal transmission range r transmits, the signal-to-noise (SNR) of the signal received at a node v with distance d_{uv} from the node u is $\lambda r^2/d_{uv}^{\alpha}$, where λ is an exponentially-distributed random variable with unit mean, and α is an attenuation parameter called *path loss*. The value of α is usually between 2 and 4 depending on the characteristics of the channel; however, in this work, we consider $\alpha = 2$. We assume that a packet transmitted by a node u is successfully received by a node $v \in \mathcal{N}_r(u)$ if and only if the received SNR exceeds a threshold β, i.e., $\lambda r^2/d_{uv}^2 > \beta$. We took $\beta = 1/2$.

We also consider the *two-hop blocking* MAC in our simulations. In this MAC scheme, when a node is transmitting, all of its neighbors up to two hops will be silent and will not transmit. This avoids interference and the hidden terminal problem [72]. A similar MAC was considered in [51]. By considering the MAC layer, we will also be able to compare the latency performance of the different broadcasting algorithms.

Some standard networks that are considered in this study are grid networks and random deployment models. In grid networks, we consider two cases. First, the source is considered to be one of the four corners of the grid, and second the source is in the center of the grid. When the source is in the corner, the grid is assumed to

be a square of l rows and l columns with neighbors spaced equally apart from each other. When the source is placed in the center, the network is assumed to be a square of $2l-1$ rows and $2l-1$ columns. In both scenarios, the transmission range r is chosen to be equal to the distance between any two neighboring nodes, constraining the maximum degree of a node in the network to be four. In random deployment networks, the nodes are placed independently and uniformly at random in a field of $A \times A$ distance unit^2.

6.3 FTS: Fractional Transmission Scheme

In this section, we propose a broadcasting scheme referred to as *Fractional Transmission Scheme* (FTS). FTS is based on the idea that multiple neighbors of a node u can *share* the work of packet transmission to u. It suffices that each neighbor of a node just sends a fraction of the data such that the total sum of all fractions received by the node is enough for successful data recovery. To be efficient, packets from different neighbors should be innovative. To ensure this, we employ rateless coding. First, the source encodes the data and forwards it. When other nodes receive enough encoded data packets, they perform *decoding and re-encoding* to generate new (innovative) packets.

It is assumed that each node is aware of its hop-distance from the source. Each node can also be aware of the hop-distances of its immediate neighbors from the source by simple HELLO message exchanges. In FTS, over each link between two nodes, the one with the smaller hop-distance (from the source) transmits to the one with the higher hop-distance. In the case that both the nodes have the same hop-distance, the one with lower *id* transmits to the other. Therefore, a node v will expect to receive data from the nodes in its *parent set* $\mathcal{P}_r(v)$ defined as:

$$\mathcal{P}_r(v) = \{w \in \mathcal{N}_r(v) : \mathbf{H}(w) < \mathbf{H}(v)\} \cup \{w \in \mathcal{N}_r(v) : (\mathbf{H}(w) = \mathbf{H}(v)) \wedge (id(w) < id(v))\}.$$

On the other hand, each node w is responsible for providing a fraction of data to a set of

nodes that are called its *children* defined as $\mathcal{C}_r(w) = \mathcal{N}_r(w) \backslash \mathcal{P}_r(w)$. The fraction of data that each node w must send, α_w, has to be determined. Next, we provide details on how FTS determines these fractions and how it operates.

6.3.1 Description of FTS

FTS includes three phases: *Initial Fraction Exchange Phase*, *Fraction Reduction Phase*, and *Data Transmission Phase*. In the first two phases, each node determines the fraction of data that it must send. The last phase is the actual data transmission phase. In *Initial Fraction Exchange Phase* (Algorithm 1), each node v determines the number of neighbors κ_v that will send data to v. In other words, $k_v = |\mathcal{P}_r(v)|$ represents the number of neighbors of v that have either a smaller hop-distance from the source or the same hop-distance but a smaller *id*. Therefore, v expects a fraction of $1/\kappa_v$ of the required data from each of them. Once v determines the fraction it expects, it declares this fraction to nodes in $\mathcal{P}_r(v)$. Each node w collects all the requested fractions that it must send to its children $\mathcal{C}_r(w)$. It considers the maximum of these fractions as the sufficient fraction of data that it must send. For example, if $\mathcal{C}_r(w) = \{u, v\}$ with $\kappa_u = 3$ and $\kappa_v = 4$, this means that u and v expect fractions of $\frac{1}{3}$ and $\frac{1}{4}$ of data from w, respectively. Then, $\alpha_w = \frac{1}{3}$. After this phase, the sum of the fraction of data that each node receives might be larger than one. Therefore, we can further reduce the fraction by another phase called *Fraction Reduction Phase* (Algorithm 2). In this phase, each node v asks its neighbors in $\mathcal{P}_r(v)$ to reduce their fractions equally by f_v such that the total fraction that v receives adds up to one. Each node then reduces its fraction by the minimum of requested reduced fractions. The new fraction that node w will send is denoted by α'_w. Figure 6.1 shows a small part of a network as an example. Suppose $\mathcal{P}_r(u_1) = \{w_1\}$, $\mathcal{P}_r(u_2) = \{w_1, w_2\}$, and $\mathcal{P}_r(u_3) = \{w_2, w_3, w_4\}$. Moreover, $\mathcal{C}_r(w_1) = \{u_1, u_2\}$, $\mathcal{C}_r(w_2) = \{u_2, u_3\}$, $\mathcal{C}_r(w_3) = \{u_3\}$, and $\mathcal{C}_r(w_4) = \{u_3\}$. We have $\kappa_{u_1} = 1$, $\kappa_{u_2} = 2$, and $\kappa_{u_3} = 3$. Therefore, $\alpha_{w_1} = 1$, $\alpha_{w_2} = 1/2$, and $\alpha_{w_3} = \alpha_{w_4} = 1/3$. By the fraction reduction phase, we have $f_{u_1} = 0$, $f_{u_2} = 1/4$, $f_{u_3} = 1/18$. The final fractions will be $\alpha'_{w_1} = 1$, $\alpha'_{w_2} = 1/2 - 1/18$, and $\alpha'_{w_3} = \alpha'_{w_4} = 1/3 - 1/18$.

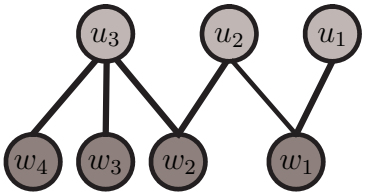


Figure 6.1: A small example.

Algorithm 1 *Initial Fraction Exchange Phase*

Require: Graph $G(V, E)$ with source $s \in V$ where each node $v \in V$ knows its minimum hop-distance $\mathbf{H}(v)$ from source.

for $v \in V \setminus \{s\}$ **do**
  $\alpha_v = 0$.
end for
$\alpha_s = 1$.
for $v \in V$ **do**
  transmit: $(\mathbf{H}(v), id(v))$
end for
for $v \in V$ **do**
  identify $\kappa_v = |\mathcal{P}_r(v)|$
  transmit: $(v, \mathbf{H}(v), \kappa_v)$
end for
for $v \in V$ **do**
  for $w \in \mathcal{N}_r(v) \backslash \mathcal{P}_r(v)$ **do**
    $\alpha_v = \max(\alpha_v, \frac{1}{\kappa_w})$
  end for
end for
for $v \in V$ **do**
  transmit: α_v
end for

Algorithm 2 *Fraction Reduction Phase*

for $v \in V$ **do**
  transmit: $f_v = \max\left(\left(\sum_{w \in \mathcal{P}_r(v)} \alpha_w - 1\right)/\kappa_v, 0\right)$
end for
for $v \in V$ **do**
  $\alpha'_v = \max(\alpha_v - \min_{w \in \mathcal{C}_r(v)} f_w, 0)$
end for

Once the first two phases are complete, we start the *Data Transmission Phase* (Algorithm 3). In this phase, once a node v receives fraction $\max(\alpha_w - f_v, 0)$ of packets from a neighbor w it sends acknowledgement of partial completeness ($P - ACK(v \rightarrow w)$), and

when it becomes complete (receives enough encoded packets to decode), it sends acknowledgement of completeness ($C - ACK(v)$) to its neighbors. It then decodes and re-encodes data using a rateless code and sends new encoded packets until all the children of v are either complete or do not need any more packets from v.

Algorithm 3 *Data Transmission Phase*

$send(s) = 1$
transmit: $n_p\gamma$ encoded packets
for $v \in V \setminus \{s\}$ **do**
  $send(v) = 0, done(v) = 0$
end for
while $\exists\, v \in V$ with $send(v) = 0, done(v) = 0$ **do**
  $A = \{u \in V : send(u) = 0, done(u) = 0\}$
  for $w \in A$ **do**
    if w has received $n_p\gamma$ encoded packets **then**
      decode n_p data packets
      transmit: $C - ACK(w)$
      set $send(w) = 1$
    else
      if w has received $\max(\alpha_z - f_w, 0)n_p\gamma$ packets from neighbor z **then**
        transmit: $P - ACK(w \to z)$
      end if
    end if
  end for
  $B = \{u \in V : send(u) = 1\}$
  for $t \in B$ **do**
    if all neighbors have either sent a $C - ACK$ or a $P - ACK$ directed at t **then**
      set $send(t) = 0$, $done(t) = 1$
    else
      generate an encoded packet P.
      transmit: P
    end if
  end for
end while

6.3.2 Discussion of Various Overheads

The proposed FTS scheme has three overhead messages/signals whose effects on latency and transmission cost are described below. Since the network is assumed to be stationary, the setup overheads correspond to the *initial fraction exchange phase* and the *fraction reduction*

phase, which have to be performed just once. In the former phase, the determination of the hop counts and the initial fractions can be initiated by the source by flooding a single packet. This can be completed in $\Theta(N)$ time units and by spending $\Theta(1)$ units of energy per node. In the latter, each node has to declare the amount by which its neighbors can deduct the fraction that it needs to send. This can again be performed in $\Theta(Nr^2(N))$ time units by spending $\Theta(1)$ energy units per node.

The second overhead is that of coding. In general, the number of output packets required to give a high probability of retrieving n_p input packets is expressible as γn_p for a fraction $\gamma \geq 1$ (γ is called the coding overhead). The coding overhead corresponds to the number of additional packets that are required over and above the bare minimum number of packets required for decoding all the transmitted packets. If one performed maximum likelihood (ML) decoding, one would have a smaller coding overhead. However, the complexity would make this option restrictive. On the other hand, using iterative decoding will enable lower decoding complexity but will have a higher coding overhead. Throughout this work, we set the rateless coding overhead γ to be 1.03 for $n_p = 2000$ packets as described in [39].

The third overhead is the bits that need to be appended to the header of an encoded packet to identify the index of the packets that were added to form the current packet. In rateless codes, at most n_p bits must be added to each output packet. The overhead imposed by these additional bits is considerably smaller than the corresponding overhead in random sum coding, which is $n_p \log_2 q$ bits[2].

The last overhead, which is also the most important one, is that of acknowledgement of the completion or acknowledgement of partial receipt (from a particular neighbor). The acknowledgement of completion needs to be transmitted only once. It must be noted that the acknowledgement of partial receipt is necessary only if we want a node to stop transmitting once all its children have received the required number of packets prior to sending all its determined share of packets. In partial acknowledgment, we can achieve benefits in conserving energy while transmitting encoded data packets by allowing each node to

[2]In random sum coding, each packet is multiplied by a coefficient chosen randomly from $GF(q)$. These coefficients must also be appended to the header of packets.

transmit control packets, which are very short in length, of order $\Theta(Nr^2(N))$.

6.3.3 Analysis of FTS on Grid Networks

FTS lends itself to easy analysis over grid networks and yields the following results on the average energy consumed.

6.3.3.1 Lossless Grid Networks

For simplicity, we first consider the case that all the links are lossless, and we then extend it to the case in which we have signal attenuation and Rayleigh fading channels.

Lemma 6.1. *For an $l \times l$ grid network with the source node in the corner of the grid, the average number of transmissions per packet per node under FTS when a rateless code of overhead γ is used is given by*

$$\mathcal{N}_{/p/n} = \frac{(l^2 + 2l - 4)\gamma}{2l^2}.$$

Proof. Without loss of generality, we assume that the source node is in the bottom left corner of the grid. It is clear that the source node has to inject a fraction of one; otherwise, no node in the network can perform decoding. Nodes at the bottom and left edges that are not corner nodes, which are $2(l-2)$ in number, have to send a fraction of one as their neighbors have $\kappa = 1$. The node at the top right corner transmits a fraction of zero. All other nodes, which are $(l-1)^2 + 1$ nodes in number, transmit a fraction of $\frac{1}{2}$, since each of their neighbors have $\kappa = 2$. Therefore, the total number of transmissions per packet can be easily calculated as $\frac{(l^2+2l-4)\gamma}{2}$, where γ is coding overhead. One can easily check that for such a grid network, the Fraction Reduction Phase does not change the fraction of data for all but one of the nodes. This node, which we call v, is in the position (1,1) assuming the source is at (0,0). The unique property of node v is that the two children of v have other parents that will send a fraction of one of data (all data). Therefore, $\alpha'_v = 1/2 - 1/4 = 1/4$. This change is negligible. □

Figure 6.2 shows the fractions of data that each node sends in FTS in a 4×4 grid, when source is in the bottom left corner. We have $\mathcal{N}_{/p/n} = 10/16\gamma$. Similarly, we can prove the

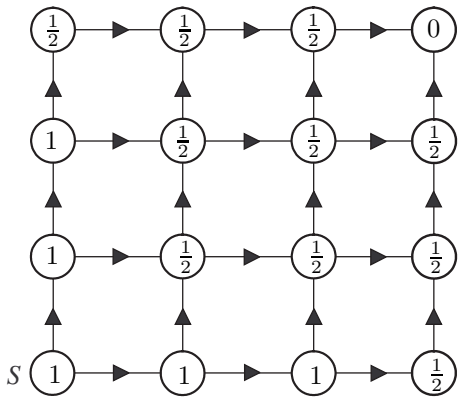


Figure 6.2: The fraction of data that each node sends to its neighbors after decoding and re-encoding in FTS. Source is in the bottom left corner. We have $\mathcal{N}_{/p/n} = 10/16\gamma$.

following lemma for the case in which the source is located in the center of a grid.

Lemma 6.2. *For a $(2l-1)\times(2l-1)$ grid network with the source in the center of the grid, $\mathcal{N}_{/p/n} = \frac{(2l^2-5)\gamma}{(2l-1)^2}$, under FTS when a rateless code of overhead γ is used.*

Assuming that each node has transmission range r, the average cost of transmissions per packet per node is given by $\mathcal{C}_{/p/n} = \mathcal{N}_{/p/n} \times r^2$. From Lemmas 6.1 and 6.2 we note $\mathcal{C}_{/p/n} = \frac{\gamma r^2}{2}$ for FTS; asymptotically (when l is very large).

Next, we would like to compare the cost of broadcasting with FTS and NC. If we fix the direction of the edges from a node with lower hop count to its neighbor with higher hop count, as for example in Figure 6.2, we will prove next that asymptotically the broadcasting costs of FTS and NC are very close.

Lemma 6.3. *For an $l\times l$ grid network with the source node in the corner of the grid, under NC (with the default direction that the node closer to the source always transmits to its neighbors that are farther away) we have $\mathcal{N}_{/p/n} = \frac{(l^2+l-2)}{2l^2}$.*

Proof. (By Badri N. Vellambi) With the default direction, it can be seen that each node can potentially get data from at most two of its neighbors and transmit to potentially at most two neighbors. It can be seen that for the set of nodes in any diagonal below the main diagonal of that of the grid, the fractions that are transmitted by the nodes satisfy the conditions of Lemma B.1 in Appendix B. For the set of nodes in any other diagonal,

the fractions transmitted satisfy the hypotheses of Lemma B.2 in Appendix B. Thus, for an $l \times l$ grid,

$$\mathcal{N}_{/p/n} = \frac{1 + T_2 + \ldots + T_{l-1} + T_l^{'} + T_{l-1}^{'} + \ldots + T_2^{'}}{l^2} = \frac{(l^2 + l - 2)}{2l^2}.$$

□

Figure 6.3 shows the fractions of data that each node sends in NC in a 4×4 grid. Source is in the bottom left corner. We have $\mathcal{N}_{/p/n} = 9/16$. Recall that FTS results in $\mathcal{N}_{/p/n} = 10/16\gamma$.

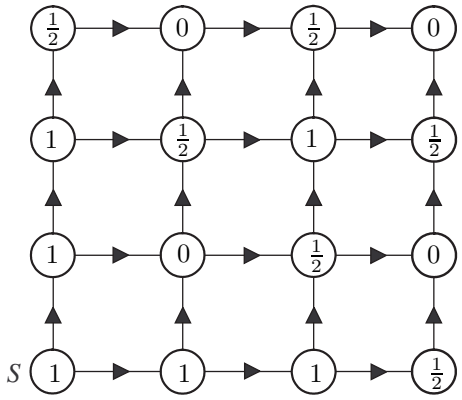


Figure 6.3: The fraction of data that each node sends to its neighbors after decoding and re-encoding in NC. Source is in the bottom left corner. We have $\mathcal{N}_{/p/n} = 9/16$.

We can easily extend this directed grid analysis to the $(2l-1) \times (2l-1)$ grid when the source is in the center and directions on each edge are chosen to be directed from a node closer to the source to one farther. In this case, $\mathcal{N}_{/p/n} = \frac{2l^2-2l+1}{(2l-1)^2}$.

Asymptotically, NC makes $\frac{1}{2}$ transmissions per packet per node for broadcasting in the assumed directed graph. It should be mentioned that the considered direction for the edges are not the best directions, and we can reduce the number of transmissions by choosing another set of directions.

A lower bound for the $\mathcal{N}_{/p/n}$ for an arbitrary (connected) directed grid network is $\frac{1}{3}$. This follows from the fact that each node in a connected directed grid can have, at most, three outgoing edges and hence each transmission can benefit, at most, three nodes. This

minimum $\mathcal{N}_{/p/n}$ of $\frac{1}{3}$ can be asymptotically ($l \to \infty$) achieved for an $l \times l$ grid under the following scheme, referred to as asymptotically-optimal scheme:

- $3|l$: Nodes in the bottommost row (except the rightmost one) and those in columns with index of the form $3j+2$ ($j = 0, 1, \ldots, \frac{l}{3} - 1$) transmit the whole data set. Other nodes do not transmit. Figure 6.4(a) illustrates the case for a 6×6 grid.
- $3|(l-1)$: Nodes in the bottommost row and in columns with index of the form $3j+1$ ($j = 0, 1, \ldots, \frac{l-1}{3}$) transmit a fraction of 1. Other nodes do not transmit. Figure 6.4(b) illustrates the case for a 7×7 grid.
- $3|(l-2)$: Nodes on the bottommost row except the last node and also nodes in columns with index of the form $3j+1$ ($j = 0, 1, \ldots, \frac{l-2}{3}$) transmit the whole data set. Other nodes do not transmit. Figure 6.4(c) illustrates the case for a 8×8 grid.

It can be shown easily that

$$\mathcal{N}_{/p/n} = \begin{cases} \frac{1}{3} + \frac{2}{3l} - \frac{1}{l^2} & 3|l \\ \frac{1}{3} + \frac{4}{3l} - \frac{2}{3l^2} & 3|(l-1) \\ \frac{1}{3} + \frac{1}{l} - \frac{4}{3l^2} & 3|(l-2) \end{cases} \tag{46}$$

These results can be extended to the case that source is in the center, keeping directions of edges symmetric with respect to the center. We have:

$$\mathcal{N}_{/p/n} = \begin{cases} \frac{4l^2+2l-9}{3(2l-1)^2} & 3|l \\ \frac{4l^2+4l-5}{3(2l-1)^2} & 3|(l-1) \\ \frac{4l^2-7}{3(2l-1)^2} & 3|(l-2) \end{cases} \tag{47}$$

6.3.3.2 Lossy Grid Networks

Here, we assume a more realistic case that the transmissions are subject to distance attenuation and Rayleigh fading as we described in Section 6.2. Assuming that the nodes have transmission range r, and two neighboring nodes in a grid are separated by distance r, we have

$$Pr\{\text{A packet is obtained by its neighbor}\}$$
$$= Pr\{\lambda > \beta\} = e^{-\beta} = e^{-1/2}$$

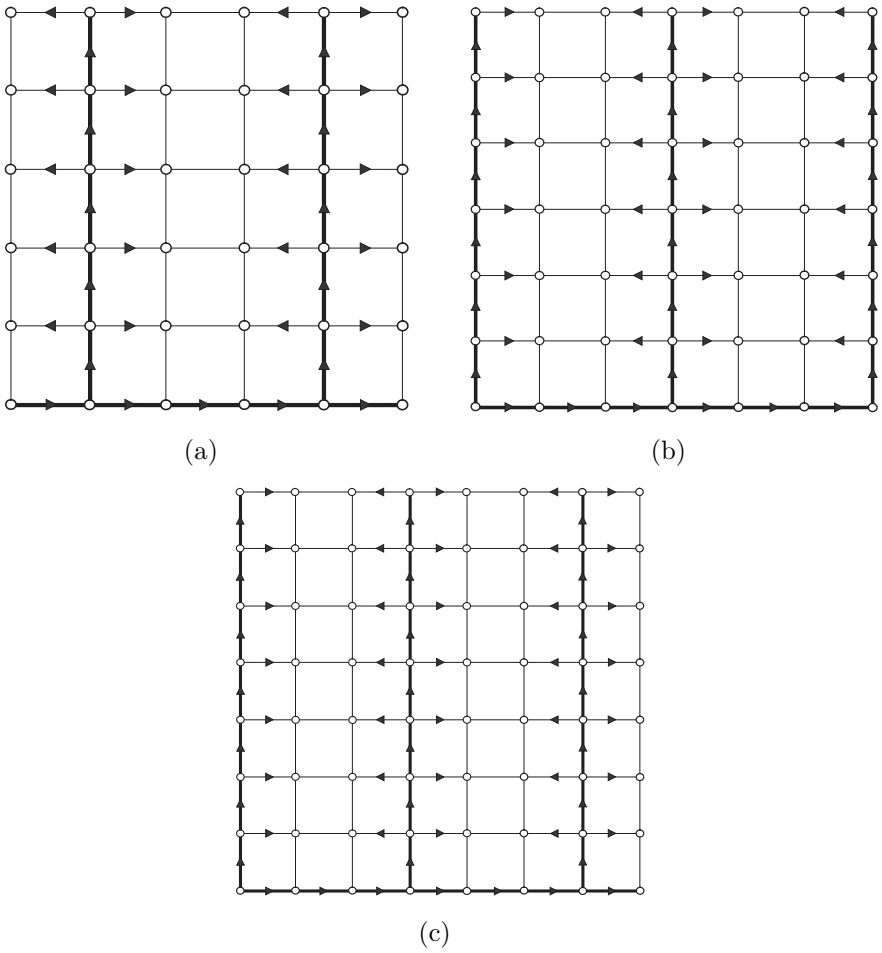


Figure 6.4: Optimal directions for an $l \times l$ grid. The nodes in the bold edges send all of the data and the other nodes are just receivers. (a) $l = 6$. (b) $l = 7$. (c) $l = 8$.

Therefore, each node v on the average needs to send fraction $\frac{\alpha'_v}{e^{-1/2}}$ instead of α'_v. We conclude that $\mathcal{N}_{/p/n}$ and $\mathcal{C}_{/p/n}$ that we calculated for FTS in the lossless case, need to be scaled by a factor of $\frac{1}{e^{-1/2}}$ for this case.

6.4 Simulation Results

To compare the energy-efficiency of FTS with present broadcasting algorithms such as BIP, NC, MPR, and CRBcast, we simulated these algorithms on square grids and randomly deployed networks. For the latter case, we considered N nodes with transmission range $r(m)$ are randomly deployed in an area $100m \times 100m$, for different N's and r's. The transmission range r in each case was selected such that the resulting graph is connected. We considered $\gamma = 1.03$ for rateless codes since this value is sufficient for the transmission

of $n_p \geq 2000$ packets with a decoding failure probability of less than 10^{-8} [39]. The size of the packets was assumed to be the same in all the schemes. Therefore, it is worth noting that the amount of information payload per packet would be higher for FTS in comparison with NC as described in Section 6.3.2. We disregarded this in our simulations.

We present our results in the two following subsections. The former presents the simulation results for lossless networks, and the latter presents the results for lossy networks.

6.4.1 Simulations for Lossless Networks

6.4.1.1 Lossless Grid Networks

Figure 6.5 shows the corresponding $\mathcal{N}_{/p/n}$ for different schemes for $l \times l$ grid networks with the source node in the corner. The results for FTS, NC, BIP, CRBcast, and the asymptotically-optimal scheme are presented. We implemented NC on a grid with the default directions (flow from the node closer to the source to the farther one). For large

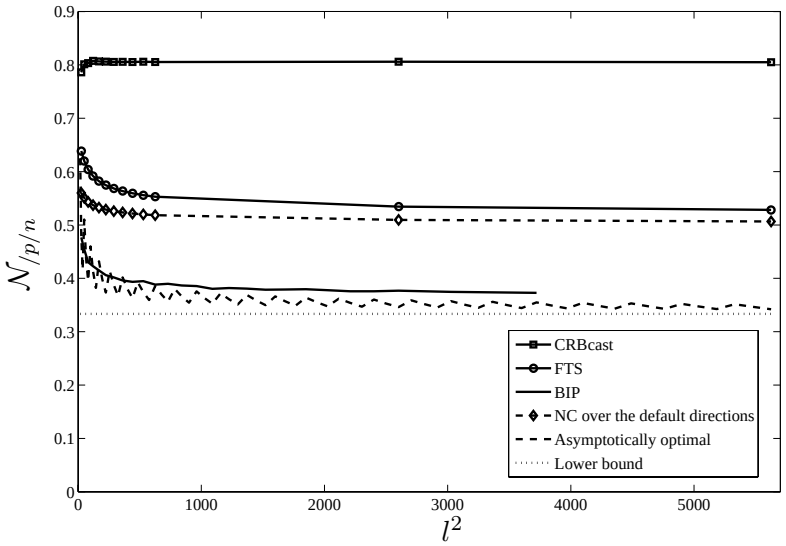


Figure 6.5: Comparing $\mathcal{N}_{/p/n}$ versus the size of the network in different broadcasting schemes for an $l \times l$ grid with the source node in the corner.

values of l, it can be noted that for the optimal case $\mathcal{N}_{/p/n}$ is about $\frac{1}{3}$. BIP performs slightly worse. NC and FTS perform very close with an asymptotic $\mathcal{N}_{/p/n}$ of 0.5 and 0.5γ, respectively. For large grid networks, the only difference in the cost of broadcasting between FTS and NC is the factor γ, the overhead of rateless coding. For NC, we assumed that the random sum coding over a large field $GF(q)$ results in no coding overhead (for the price of

a higher complexity of decoding). Increasing the number of packets n_p, γ becomes closer to one and FTS and NC result in the same energy consumption. Although NC and BIP have better energy-efficiency than FTS, it must be understood that they are considerably more sophisticated algorithms. BIP assumes global knowledge of the network for successful implementation, and NC requires LP optimization. On the other hand, FTS does not rely upon the presence of global knowledge but just on local knowledge of the topology of the network and does not require complex optimizations. Comparing to CRBcast, which does not assume any knowledge about the networks, we find that we are able to achieve more than 25% improvement by exploiting the local information.

Figure 6.6 shows the corresponding $\mathcal{N}_{/p/n}$ for different schemes for $(2l-1) \times (2l-1)$ grid networks with the source node in the center. Similar results are observed.

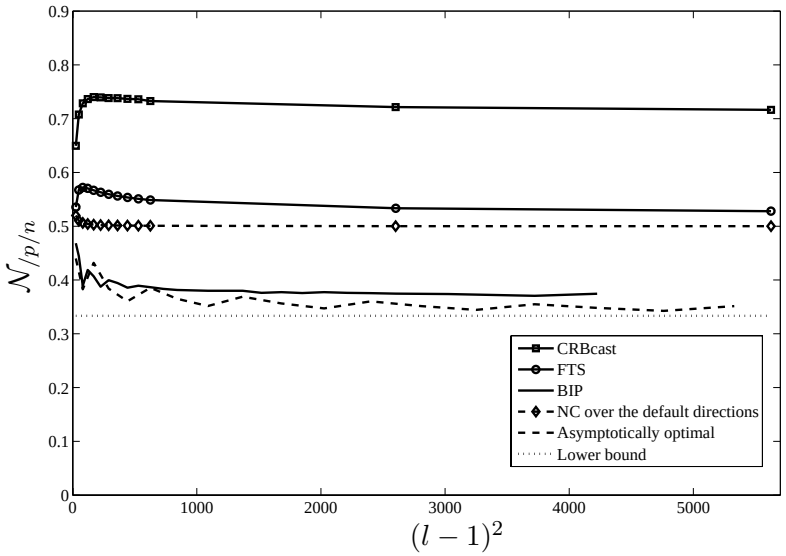


Figure 6.6: Comparing $\mathcal{N}_{/p/n}$ versus the size of the network in different broadcasting schemes for a $(2l-1) \times (2l-1)$ grid with the source node in the center.

6.4.1.2 Lossless Random Networks

First, we compare FTS and NC. We only give results for small number of nodes (i.e., 20 nodes) since we could not run the complex optimization for NC for large number of nodes. For FTS, we can easily assign direction for edges based on their hop distances from the source, as explained in Section 6.3. We also use these directions in our simulations for NC. The average $\mathcal{C}_{/p/n}$ for FTS and NC are tabulated in Table 6.1. Clearly, given the same

graph and directions, NC has a lower cost. As we see, the excess in energy consumption of FTS is less than 4% in comparison with NC. However, FTS is a very simple and easily implementable scheme and does not require any complex optimizations. This makes FTS to be more flexible and practical, with energy consumption relatively close to NC.

Next, we consider networks of up to 500 nodes. We compare FTS with CRBcast, MPR,

Table 6.1: Average broadcast cost per packet per node in randomly deployed wireless networks consisting of N nodes with transmission range r deployed in an area of $100m \times 100m$.

N	$r(m)$	Approach	$\mathcal{C}_{/p/n}$
15	23	NC	273.3
		FTS	283.3
20	22	NC	243.6
		FTS	253.7
25	21	NC	225.8
		FTS	231.6

and BIP.

In the BIP scheme, different nodes can have different transmission ranges to decrease the cost of broadcasting. Given the option that nodes can adjust their transmission range, we can extend FTS to FTS$_{adapt}$. FTS$_{adapt}$ is similar to FTS except that every node v in the network has the option of reducing its transmission range from r to r_v, where r_v equals the distance between v and the farthest neighbor of v that is going to listen to v (i.e., the farthest child of v) in the data transmission phase (Algorithm 3).

Figure 6.7 compares $\mathcal{C}_{/p/n}$ for different schemes FTS, FTS$_{adapt}$, MPR, CRBcast, and BIP. As we can see, BIP has the best possible performance. FTS$_{adapt}$ and FTS are next. We should note that BIP is a centralized scheme that needs the global knowledge of the network, and its complexity is very high. In contrast, FTS and FTS$_{adapt}$ are distributed schemes with low complexity. Comparing the performance of FTS$_{adapt}$ and FTS with MPR, which also assumes local information, we see that FTS and FTS$_{adapt}$ have much better performance. Comparing FTS and FTS$_{adapt}$ with CRBcast, we conclude FTS and FTS$_{adapt}$ could improve the energy efficiency by exploiting the local information.

We also simulate FTS for the same network topology that we used to simulate CRBcast

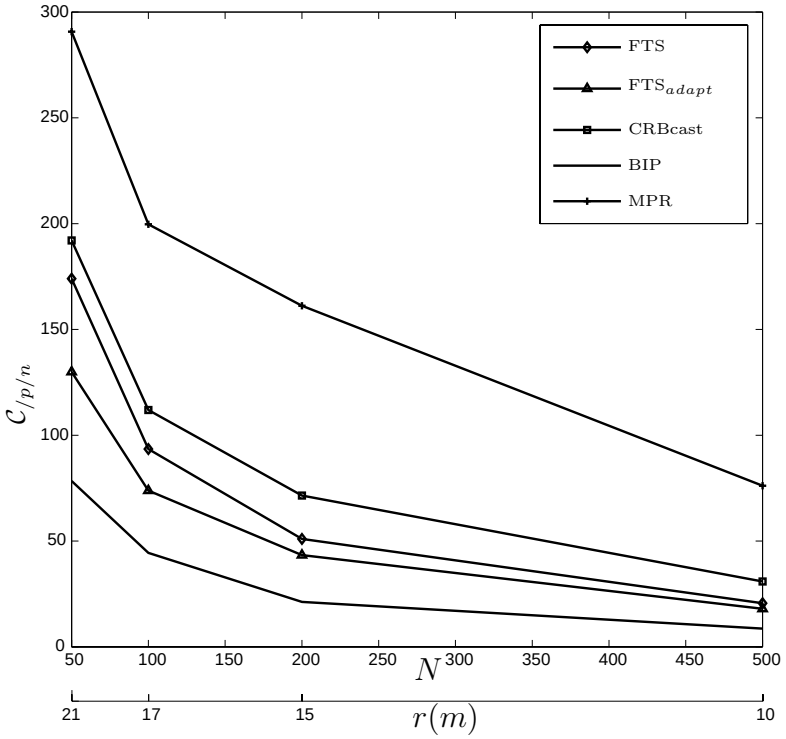


Figure 6.7: Comparing $\mathcal{C}_{/p/n}$ of different broadcasting schemes for random deployment of N nodes with transmission range r in an area $100m \times 100m$.

in Section 5.4.3. The network consists of $N = 10^4$ nodes with transmission range $r = 50m$, deployed uniformly at random in an area $A = 2000m \times 2000m$. The links are assumed to be lossless. Table 6.2 presents the results.

Table 6.2: Comparison of $\mathcal{N}_{/p/n}$ in FTS and CRBcast. The value of p is chosen as 0.25 for CRBcast.

Broadcasting Scheme	$\mathcal{N}_{/p/n}$
CRBcast	0.2769
FTS (without the fraction reduction phase)	0.2073
FTS (with the fraction reduction phase)	0.1600

As can be seen, FTS without and with the fraction reduction phase (Algorithm 2) improves the energy efficiency in comparison with CRBcast by 25% and 42%, respectively. This improved efficiency is the result of the fact that unlike CRBcast, FTS exploits local information about the network topology.

6.4.2 Simulations for Lossy Networks

In this section, we consider the channel model and MAC scheme as explained in Section 6.2 and compare FTS with some competitive broadcasting schemes such as BIP, NC, and MPR.

Schemes such as BIP and MPR are originally tailored for lossless networks, and they will not guarantee reliability in lossy networks. However, they can be extended to lossy networks using either multiple retransmissions or forward error correction at each link. In this way, if a channel has a corresponding loss probability of ϵ, the number of transmissions will be scaled, on the average, by a factor of $\frac{1}{1-\epsilon}$.

6.4.2.1 Lossy Grid Networks

Figure 6.8 depicts the transmission cost $\mathcal{C}_{/p/n}$ for different reliable broadcasting schemes for $l \times l$ grid networks with the source node in the corner. It is assumed that any two adjacent nodes have a distance of one unit and also a transmission radius of $r = 1$. The transmissions are subject to distance attenuation and Rayleigh fading with the probability of successful delivery of a packet represented as $e^{-1/2}$. For FTS, we depicted both the simulation and analytical results as derived in Section 6.3.3. As can be seen, the simulation and analytical results are almost the same. We also depicted the cost of broadcasting with NC. The asymptotically-optimal result corresponds to the scheme whose $\mathcal{N}_{/p/n}$ was derived in (46) and has been scaled by factor $\frac{r^2}{e^{-1/2}}$. The lower bound is also $\frac{r^2}{3e^{-1/2}}$. It should be noted again that the reason that NC does not reach optimality here is that the assumed direction of edges for them is not optimal.

We also depicted the latency of different schemes in Figure 6.9 for reliable delivery of 2000 packets. The two-hop blocking MAC was considered in our simulations. We did not include the latency of NC, since it is not very clear how the MAC layer and interference can be integrated in the NC. This does not affect energy consumption, but it does affect the latency. However, we expect NC to have a lower latency than FTS, since in FTS each node has to first wait to receive sufficient number of packets to be able to decode, before it can generate and send new encoded packets.

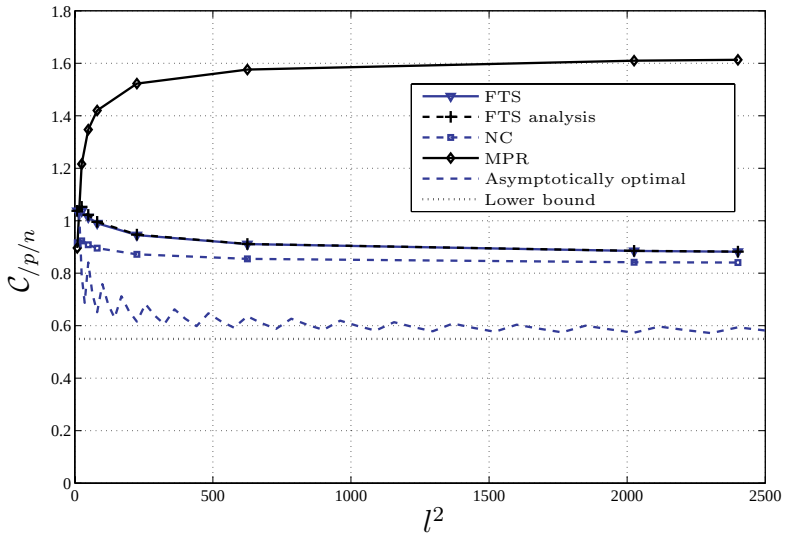


Figure 6.8: Comparing $\mathcal{C}_{/p/n}$ versus the size of the network in different broadcasting schemes for an $l \times l$ grid with the source node in the corner.

6.4.2.2 Lossy Random Networks

Here, we consider N nodes with transmission range r are randomly deployed in an area $100m \times 100m$, for different values of N and r. The source is assumed to be in the center of the area. The transmission range r in each case is selected such that the resulting graph is connected. Transmissions are subject to distance attenuation and Rayleigh fading with SNR threshold $\beta = 1/2$. First, we compare FTS and NC. For FTS, we can easily assign direction for edges based on their hop distances from the source, as explained in Section 6.3. We also use these directions in our simulations for NC. Table 6.3 compares the average $\mathcal{C}_{/p/n}$ for FTS and NC. Clearly, given the same graph and directions, NC has a lower cost. As we see, the excess in energy consumption of FTS is less than 8% in comparison with NC. However, FTS is a very simple and easily implementable scheme and does not require any complex optimizations.

Next, we consider networks of up to 500 nodes. Due to the complexity of optimization for large networks, we were not able to run NC for these experiments. Therefore, we only compare FTS with MPR and BIP for large networks.

Given the option that nodes can adjust their transmission range (as in BIP), we can implement FTS_{adapt}, in which every node v in the network has the option of reducing its

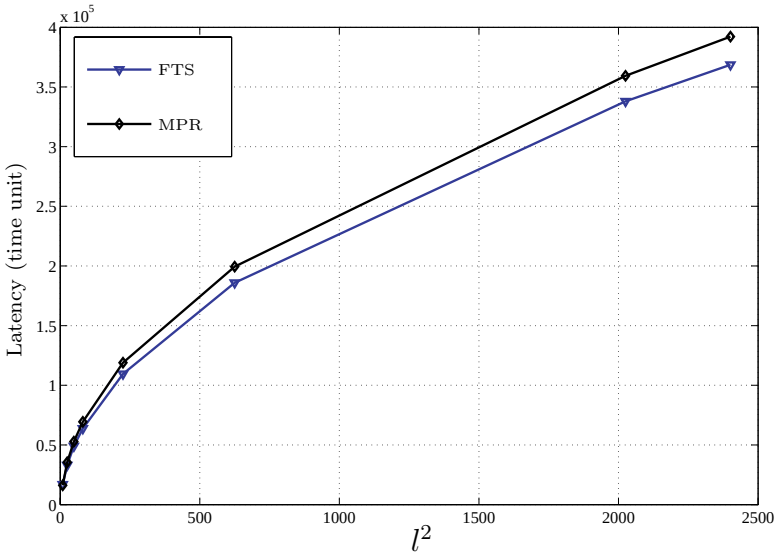


Figure 6.9: Comparing latency of different schemes.

Table 6.3: Average broadcast cost per packet per node in randomly deployed wireless networks consisting of N nodes with transmission range r deployed in an area of $100m \times 100m$.

N	$r(m)$	Approach	$\mathcal{C}_{/p/n}$
15	23	NC	331.10
		FTS	357.64
20	22	NC	326.04
		FTS	336.61
25	21	NC	300.29
		FTS	323.05

transmission range from r to r_v, where r_v equals the distance between v and the farthest neighbor of v that is going to listen to v (i.e., the farthest child of v) in the data transmission phase (Algorithm 3). Thus, the probability that a child w of v receives a packet from v decreases, since we have

$$Pr\{w \text{ receives a packet from } v\}$$
$$= Pr\{\lambda > \beta d_{vw}^2/r_v^2\} = e^{-\beta d_{vw}^2/r_v^2} < e^{-\beta d_{vw}^2/r^2}.$$

Therefore, by FTS$_{adapt}$ a greater number of transmissions is needed; however, each transmission has less cost. Overall, we would expect that FTS$_{adapt}$ would be more energy-efficient than FTS.

Figure 6.10 compares $\mathcal{C}_{/p/n}$ for different schemes FTS, FTS$_{adapt}$, MPR, and BIP. As

we can see, BIP has the best possible performance. FTS$_{adapt}$ and FTS are next. Again, it is important to keep in mind that BIP is a centralized scheme that requires the global network topology knowledge. In contrast, FTS and FTS$_{adapt}$ are distributed schemes with low complexity. Comparing the performance of FTS$_{adapt}$ and FTS with MPR, which is a distributed scheme, we see that FTS and FTS$_{adapt}$ have much better performance.

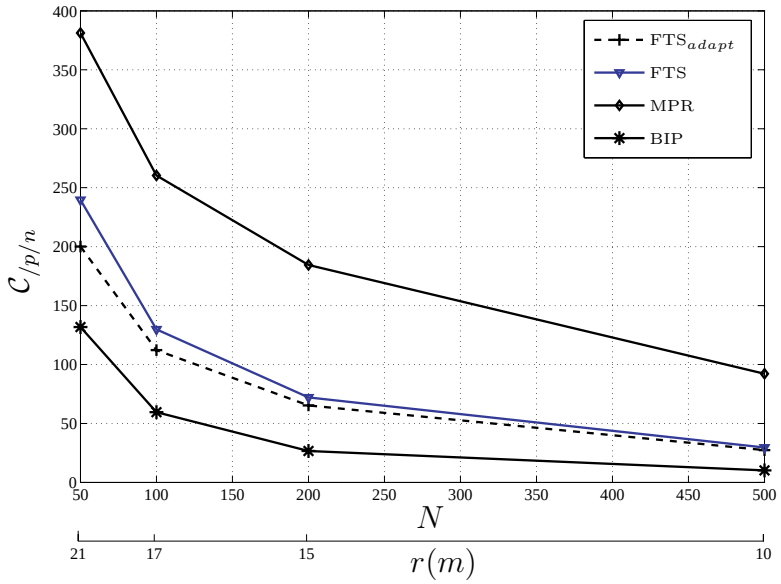


Figure 6.10: Comparing $\mathcal{C}_{/p/n}$ of different broadcasting schemes for random deployment of N nodes with transmission range r in an area $100m \times 100m$.

Figure 6.11 depicts the latency of different broadcasting schemes. The two-hop blocking MAC was considered in our simulations. FTS has the lowest latency. We note that FTS$_{adapt}$ has slightly larger latency than FTS, which is due to the fact that the number of transmissions increases when FTS$_{adapt}$ is implemented.

6.5 Conclusion

In this chapter, we studied the problem of reliable and energy-efficient broadcasting in wireless sensor networks when some local knowledge of the topology of the network is available. We proposed two low-complexity, distributed, and reliable broadcasting schemes. Both utilize rateless coding at each node of the network to reduce the redundancy of transmitted information, which is a key in reducing the energy consumption. The proposed protocols, referred to as FTS and FTS$_{adapt}$, assume only local knowledge of the wireless network in

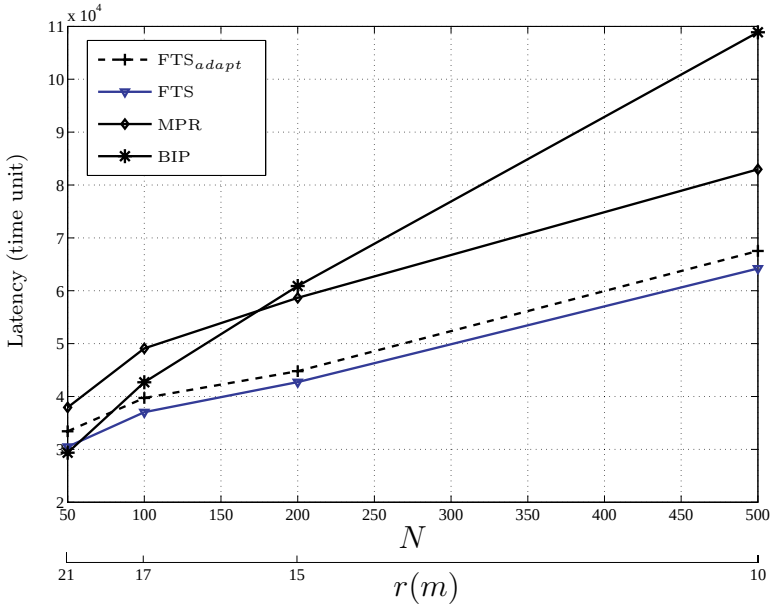


Figure 6.11: Comparing latency of different broadcasting schemes for sending 2000 packets over a random deployment of N nodes with transmission range r in an area $100m \times 100m$.

comparison to some current sophisticated algorithms that assume global knowledge, e.g., BIP. We compared our schemes with other broadcasting algorithms such as CRBcast, MPR, BIP and network coding (NC) in both lossy and lossless networks with grid and random deployment. Regarding energy efficiency, the results of our simulations show that, as expected, our algorithms perform better than CRBcast since CRBcast does not assume any knowledge of the network. The results also suggest FTS and FTS$_{adapt}$ perform much better than those schemes with similar assumptions and complexity, and that our schemes achieve energy efficiency close to that of more complex and efficient schemes such as BIP and NC. We conclude that because of their ease of implementation, low complexity, and competitive performance, FTS and FTS$_{adapt}$ are viable approaches for broadcasting in large-scale wireless ad-hoc and sensor networks, given the availability of limited and local information about the network topology.

CHAPTER VII

LOW-COST BROADCAST AND MULTICAST TREES IN WIRELESS SENSOR NETWORKS

7.1 Introduction

In Chapter 6, we introduced FTS as a reliable and energy-efficient scheme for broadcasting in wireless ad-hoc and sensor networks. As we mentioned in that chapter, FTS is applicable for lossy networks, with no required knowledge of link-loss probabilities. However, for the cases in which link losses are different, or in general the costs of packet delivery over different links are unlike and known to their incident neighbors, FTS may not be a very good solution. The reason is that FTS does not discriminate between good links (delivering a packet over them is less costly) and bad links (delivering a packet over them is very costly).

FTS also incurs a high latency because each node has to wait to receive sufficient number of packets for successful decoding before being able to re-encode and send the new encoded packets. This limits the applicability of FTS for delay-sensitive broadcast scenarios. A very interesting and challenging problem therefore would be to design a reliable broadcasting scheme for networks with different and locally-known link costs that reduces both energy consumption and latency.

Our main objective in this chapter is to develop broadcast and multicast schemes that reduce both the total broadcast/multicast energy consumption and latency with emphasis on low-complexity distributed protocols. Our approach is based on first finding a low-cost spanning tree of the network and then allowing only the non-leaf nodes to send data [61]. If the links on the tree are lossless, to ensure reliability, no error-control coding is required. Non-leaf nodes in the tree only relay the packets that they receive. However, if the links are lossy, to ensure reliability, each non-leaf node does coding. Though any capacity-achieving erasure code could be employed from energy-consumption point of view, not all of them are

efficient in terms of latency. Therefore, we employ special types of random codes that do not require decoding before re-encoding [44] to avoid high latency[1].

The benefits of having a spanning tree are as follows. First, we reduce the number of forwarding nodes and therefore the number of redundant packet transmissions is reduced. Second, we guarantee that every node is included in the tree. One choice of a spanning tree is the Bellman-Ford tree, which is formed in a distributed fashion and optimizes the unicast cost of transmitting data from source to every node in the network. However, the Bellman-Ford tree (BFT) does not consider the wireless channel medium property, which enables the reception of a single transmission potentially by all neighboring nodes.

In this chapter, we propose and evaluate two distributed protocols for finding *low-cost broadcast and multicast trees in wireless networks*. The constructed trees can then be used for reliable, energy-efficient, and low-latency data broadcast and multicast in wireless networks. The proposed schemes, referred to as *broadcast decremental power* (BDP) and *multicast decremental power* (MDP), evolve a given spanning tree of a network and form other spanning trees with lower broadcast/multicast costs. In our schemes, the Bellman-Ford tree is considered as the initial spanning tree.

Links in a network are assumed to have some cost based on parameters such as the distance between nodes, link losses, etc. We consider two different network scenarios. In the first one, the nodes in the network have adjustable transmission power, and in the second one, the transmission power is fixed. Links in networks are assumed to have some cost based on parameters such as the distance between nodes, link losses, etc. Exhaustive simulation results are provided for the two different communication power scenarios and different network topologies to evaluate the proposed schemes. We show that broadcast/multicast cost is substantially improved comparing to BF and well-known centralized schemes such as *Broadcast Incremental Power* (BIP) and *Multicast Incremental Power* (MIP), which can be implemented for the adjustable radius model. For the fixed power model, we compare

[1]The new broadcast scheme, in return, would lack some of the desirable properties of FTS such as load balancing, and simplicity (i.e., local fraction exchange without any need for distributed optimization).

our scheme with BF, NC, CRBcast, and FTS. Substantial improvement compared to BF and Network Coding (NC) is observed in most of our simulations. The proposed schemes are shown to have, at most, quadratic complexity, and, unlike NC, their application can be extended to large networks.

7.2 Network Model

Throughout this chapter, we model a wireless network by a weighted undirected graph $G(V, E, \mathbf{C})$, where V denotes the set of nodes in the network with $|V| = N$, E is the set of edges, and $\mathbf{C} : V \times V \rightarrow \mathbb{R}^+ \cup \{0, \infty\}$ represents the function denoting the cost of transmitting a packet between any two nodes. Specifically, for any $i, j \in V$, $c_{ij} := \mathbf{C}(i, j)$ is the cost of delivering one packet from node i to node j. These costs can be assigned based on different metrics such as the distance between nodes, link losses, etc. Links are assumed bidirectional and symmetric, i.e., $c_{ij} = c_{ji}$. We assume that each node knows the cost of reaching its adjacent neighbors.

To integrate the wireless channel property in our model, we assume the following, which we call the *wireless cost property*. If a node i transmits a packet with a cost C, then a node j will also receive the packet if and only if $c_{ij} \leq C$.

7.3 Related Work

The problem of minimum-cost broadcasting in wireless networks can be approached differently in the two models that follow.

1. In the first model, the nodes have only relaying capability. In this case, the problem of reliable and minimum-cost broadcasting in a wireless network is NP-complete. Generally, this problem is equivalent to finding a *minimum-cost broadcast tree* if the graph is weighted, and it is equivalent to finding a *minimum-connected dominating-set* (MCDS) for the corresponding network graph for the special case in which the costs can take only two values $\{c, \infty\}$[1]. Both of these problems have been shown to

[1]This happens, for example, when the transmission radius is fixed and nodes within the transmission range of each other can communicate with cost c; otherwise, they cannot communicate.

be NP-complete [16, 25] even if a centralized algorithm utilizing the full knowledge of the graph topology is employed.

Since broadcast is a special case of multicast, the minimum-cost multicast problem is at least as hard as the minimum-cost broadcast problem[2]. Some heuristic algorithms for tackling the broadcast problem have been proposed, e.g., [18, 23, 42, 45, 51, 70] for cases in which the transmission range of the nodes in the network is fixed and $c_{ij} \in \{c, \infty\}$ for $i, j \in V$ and $i \neq j$. In [74], the authors considered another transmission model, in which each node can adaptively select its communication radius and accordingly c_{ij} can take any non-negative value. For this model, they proposed novel heuristic broadcast and multicast schemes, which they called *broadcast incremental power* (BIP) and *multicast incremental power* (MIP), respectively. BIP and MIP are centralized schemes in which a central node adds nodes to the tree one at a time, based on its knowledge of the whole network topology (geographical positions of the nodes) and knowing the nodes have already been included in the tree. BIP and MIP demonstrate good performance, though the centralized nature of the schemes is limiting. Since the introduction of BIP and MIP, not much improvement has been achieved because of the complexity of the problems.

2. In the second model, in addition to relaying, each node has the capability to do local processing and coding. This model was first exploited in [2] and opened a new research area known as *network coding* (NC). Considerable work has been done in this area including [32, 34] and the references therein. In this model, the problem is solvable by a polynomial-time algorithm assuming that the network is *directed*. We discussed NC in detail in Section 2.5.1 and mentioned its advantages and shortcomings.

The difference between the NC approach and our proposed schemes can be stated in the following way. NC assumes a directed graph and tries to find a low-cost subgraph. The

[2]Even in wireline networks, the minimum-cost multicast problem, which is equivalent to solving the *Minimum Steiner Tree* problem, is an NP-complete problem, while the minimum-cost broadcast problem is equivalent to finding the minimum-spanning tree that runs in polynomial time.

resulting subgraph may not be a tree. Therefore, a node might have different incoming flows, in which case NC uses a random sum coding scheme to combine the incoming data from different neighbors. In contrast, our schemes work on an undirected graph. We try to find a low-cost subgraph which is a *tree*. In such a subgraph, each node has exactly one incoming flow and therefore there is no need to combine the data. If, however, the links are lossy, local rateless coding is implemented on each link for loss recovery.

7.4 Terminologies and Definitions

Throughout this chapter, we assume the following terminologies. We denote the set of neighbors of a node i (any node that is connected to i by an edge) by $\mathcal{N}(i)$. After directions are assigned to some edges in a network, if an edge (i, j) is directed from i to j, then i is called the parent of node j, denoted as $\mathcal{P}(j)$, and j is called a child of node i. On the other hand, if an edge (i, j) does not have any direction, we say i is a *neutral neighbor* of j, and vice versa. We denote the set of children of node i and the set of neutral neighbors of i by $\mathcal{C}(i)$ and $\mathcal{T}(i)$, respectively. For example, for node i in Figure 7.1, we have $\mathcal{P}(i) = \{A\}$, $\mathcal{C}(i) = \{B, C, D\}$, and $\mathcal{T}(i) = \{E, F\}$.

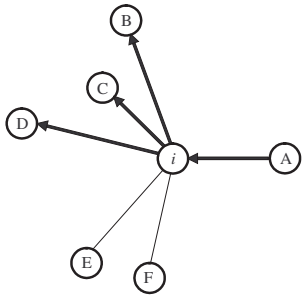


Figure 7.1: The subgraph induced by node i.

The cost of broadcast for each node i in the network is defined as

$$P_i = \max_{j \in \mathcal{C}(i)} c_{ij}, \tag{48}$$

and the total broadcast cost is

$$P = \sum_{i \in V} P_i. \tag{49}$$

For any node $j \in \mathcal{N}(i)$, the *additional cost* of having node j as a child of node i, is denoted by

$$\Delta c_{ij} = c_{ij} - \max_{k \in \mathcal{C}(i) \setminus \{j\}} \{c_{ik}\} \tag{50}$$

7.5 Cost Models

In our proposed schemes, c_{ij}'s can be selected arbitrarily. However, a reasonable cost model can be based on the required energy to reach a node j from a node i. In our models, we consider only the energy spent for RF transmissions as in [74]. We consider the following two cost models in this chapter.

A. In this model, also studied in [74], we assume that the transmission range of a node can be adjusted a within a given range. Therefore, we assume that the cost (energy) needed for sending a packet from node i to node j, which are separated by distance r_{ij} is[2]

$$c_{ij} = r_{ij}^{\alpha}. \tag{51}$$

Parameter α is called the *path loss*, which is usually between 2 and 4, depending on the characteristics of the channel. The links in this model are assumed to be lossless. Therefore, nodes that are selected to be relay nodes (non-leaf nodes in the tree) will relay packets that they receive with a transmission power that is determined by the protocol. In this case, the number of packets (n_p) could be small or large. We refer to this model as Model A.

B. In the second model, referred to as Model B, we assume the transmission radius r is fixed for all nodes in the graph. Hence, the graph can be modeled by a random geometric graph, in which there is a link between two nodes i and j, which are separated by distance r_{ij} if and only if $r_{ij} \leq r$. In this case, we also assume the links are *lossy*, i.e., a packet sent through a link between i and j, will be

[2]In fact, we have $c_{ij} \propto r_{ij}^{\alpha}$. However, since we are interested in a comparison of different schemes, and not the exact values of the energy, without loss of generality, we can consider the constant factor as one.

lost with a probability of ϵ_{ij}. Therefore, the cost (energy) needed for successfully delivering a packet from node i to node j is given by

$$c_{ij} = \begin{cases} \frac{r^{\alpha}\gamma}{1-\epsilon_{ij}} & \text{if } r_{ij} \leq r \\ \infty & \text{otherwise} \end{cases} \tag{52}$$

The factor $\frac{1}{1-\epsilon_{ij}}$ reflects the fact that the channel capacity between two nodes i and j is $1-\epsilon_{ij}$ packets per channel use, and hence the least number of required packet transmissions to successfully deliver a packet is $\frac{1}{1-\epsilon_{ij}}$ packets. This bound can be achieved by using a capacity-achieving error-control code when n_p is large. In the cases for which n_p is not very large, more packets may need to be sent, and we consider this case by scaling with a parameter $\gamma(n_p) \geq 1$, which is called the coding overhead and is a decreasing function of n_p. To minimize the overhead γ imposed by the coding scheme, n_p should be large.

In [44], several coding schemes were proposed for coding over line networks. The proposed coding schemes all reach the capacity of each link, but they are different in terms of delay and complexity of processing at each node. One of the suggested coding schemes is to decode and re-encode at each node. The other coding scheme is called greedy random coding. In greedy random coding, at each time slot, each non-leaf node transmits random linear combinations (over $GF(2)$) of all the packets it has received so far. The main advantage of this greedy random scheme, aside from its adaptability to channel losses, is optimality in terms of delay. Its drawback is high decoding complexity, which is $O(n_p^2 \log n_p)$ XOR operations on packets [44].

It should be noted that for the cost models that we have assumed, the wireless cost property mentioned in Section 7.2 is well-justified. For the first case, when a node i sends a packet with a cost C, this packet reaches all the nodes within the radius $\sqrt[\alpha]{C}$. Any node j with $c_{ij} < C$, equivalently $r_{ij} < \sqrt[\alpha]{C}$, receives the packet. For the second model, since the transmission power for all the nodes is the same, the costs of a node i reaching its different neighbors is proportional to the *number* of packet transmissions required to

successfully deliver a packet to each of them. Clearly, communication with a higher cost C, which corresponds to a greater number of transmissions, will be more than sufficient for any neighbor j with $c_{ij} < C$ to be able to retrieve the packet.

7.6 The Proposed Protocol for Finding Low-Cost Broadcast Trees in Wireless Networks

We are interested in developing a simple and distributed protocol for finding low-cost broadcast trees in wireless networks. Before we explain our scheme, we give an example to show how we can take advantage of the wireless medium to reduce the broadcast cost. Figure 7.2 depicts a small network in which node S is the source, and the cost of communication on each link has been given on the top of it. We would like to establish the minimum-cost broadcast tree. Figure 7.3 shows the tree found by the Bellman-Ford scheme. The total cost for establishing this broadcast session is $P_S + P_A + P_D = 6$. It is not hard to see that the broadcast tree in Figure 7.4 has the minimum wireless cost, which is 4.5. We took advantage of the fact that node D is the only neighbor of node E and has to send data to it with a cost of two. Now, node B has two options for receiving data. It could get data from node S, with an additional cost of two, or get data from node D with an additional cost of 0.5. Clearly, the latter is more cost efficient.

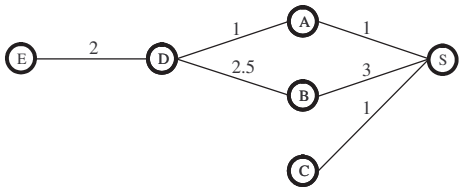


Figure 7.2: A network of 6 nodes, in which S is the source node.

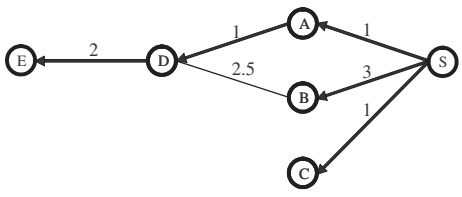


Figure 7.3: The broadcast tree found by the BF algorithm. Here, $P = 6$.

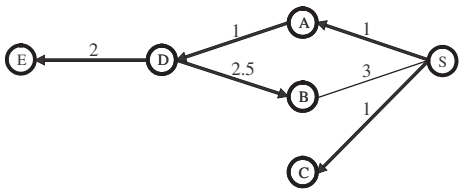


Figure 7.4: The minimum-cost broadcast tree. Here, $P = 4.5$.

Our proposed protocol works based on the following simple and local rule. Given a broadcast tree, each node j other than the source tries to select the best parent. By the best parent, we mean a neighbor that has to pay the smallest additional cost to be the parent of j. In other words, each node j chooses its parent as follows:

$$\mathcal{P}(j) = \underset{i \in \mathcal{P}(j) \cup \mathcal{T}(j)}{\arg\min} \{\Delta c_{ij}\} \tag{53}$$

Clearly, to find $\mathcal{P}(j)$ from (53), node j has to know how much additional cost will incur if j becomes a child to its non-child neighbors. This can be done by simple "HELLO" message exchanges between the neighbors. This means that node j asks its non-child neighbors $i \in \mathcal{P}(j) \cup \mathcal{T}(j)$ about its additional cost Δc_{ij}, by sending REQ-Δc_{ij} message. These neighbors reply back with the additional cost by sending an ACK-Δc_{ij} message, and j decides about its parent. When the decision is made, j informs its neighbors about the decision, and the neighbors update their lists accordingly. Specifically, j sends ACK-*join* to the new parent it chose, ACK-*release* to its previous parent in the tree, and ACK-*unchanged* to its other non-child neighbors. Clearly, the status of children of j will not change at this step. As we mentioned, a child decides which node will be its parent and not vice versa. To avoid any conflict, when a node i receives request for additional costs from different neighbors, it will handle them one by one. Specifically, i responds to the first requesting node and waits for its reply. After that, based on the new configuration in the tree, i replies to the next requesting node and so on.

To implement this protocol in a network, we again consider the example in Figure 7.3. We have $\Delta c_{SB} = 2$ and $\Delta c_{DB} = 0.5$. Therefore, node B chooses node D as its parent, instead of node S. This results in the broadcast tree in Figure 7.4.

Since each decision is made locally, it might not lead to the best possible one. Moreover,

since all the nodes are making new decisions, their new decisions may result in a different decision for their neighbors later. Therefore, the algorithm needs to be run several times to converge.

We call this scheme *Broadcast Decremental Power* (BDP) since each node tries to decrease the total broadcast power.

7.6.1 Cycle Occurrence

BDP, as explained above, may cause disconnectivity and cycles in a graph. This can be best explained by the following example. Figure 7.5 depicts a graph and the broadcast tree found by the Bellman-Ford scheme.

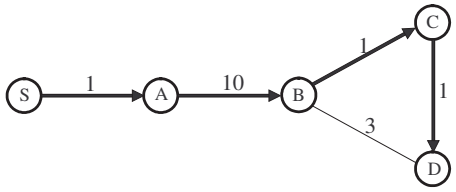


Figure 7.5: The broadcast tree found by the BF algorithm.

If we apply BDP, we have $\Delta c_{AB} = 10$, and $\Delta c_{DB} = 3$. Therefore, B chooses to have D instead of A as its parent. The resulting broadcast tree is depicted in Figure 7.6. As is

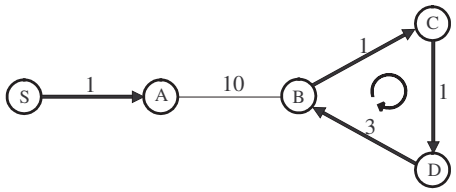


Figure 7.6: Nodes B, C, and D form a cycle, which has to be avoided.

shown, nodes B, C, and D form a cycle, and they are isolated from the source node. One way to overcome this problem is to have each node be aware of the list of all nodes included in the path from the source node to itself. In this way, a node i will consider itself as a potential parent for a neighboring node j if and only if j is not included in the path from the source node to i. In the previous example (Figure 7.5), the path from S to D includes

node B; therefore, node D cannot be a parent for B. Clearly, when the tree changes, all the nodes have to update their path list. However, this scheme is a centralized one, and we are interested in a decentralized scheme. Next, we propose a simple and decentralized scheme for cycle avoidance.

7.6.2 The Proposed Cycle-Avoidance Scheme

As mentioned, BDP starts with a spanning tree. In this study, we consider the BFT, due to the fact that it can easily be formed in a distributed fashion [6]. When the BFT is formed, not only does each node know which of its neighbors is a parent, child, or neutral, but each node also knows the total cost of the path from the source node to itself. We call this cost the *unicast cost*, and denote this as uc_i, for node i. For example, in Figure 7.5, $uc_B = 11$ and $uc_D = 13$. Our proposed cycle-avoidance scheme uses these unicast costs in such a way that a node i can be a potential parent for its neighboring node j only if $uc_i < uc_j$. It can be easily shown that this prevents the formation of cycles. Following this rule, in Figure 7.5, node D cannot be a parent for node B. Therefore, BDP and BF result in the same tree, which is the optimal one in this example. In contrast, in Figure 7.4, $uc_B = 3$ and $uc_D = 2$; hence, node D can potentially be a parent for node B. For a node j, we denote the set of its potential parents based on this cycle-avoidance scheme as $\mathcal{P}_{CA}(j)$.

7.6.3 BDP: Broadcast Decremental Power

We now summarize the BDP protocol for a given network graph $G(V, E, C)$ in Algorithm 4.

Algorithm 4 *Broadcast Decremental Power*

Require: A spanning tree of Graph G, $L_{\max} \in \mathbb{N}$. Each node has to know its unicast cost (uc) to the source S.

for $j \in V$ **do**
  transmit: uc_j
end for
for $j \in V$ **do**
  $\mathcal{P}_{CA}(j) = \{i | i \in \mathcal{N}(j) \wedge uc_i < uc_j\}$
end for
set $iteration = 1$
while $iteration < L_{\max}$ **do**
  for $j \in V \setminus S$ **do**
    transmit: REQ-Δc_{ij} to $\{i | i \in (\mathcal{P}(j) \cup \mathcal{T}(j)) \cap \mathcal{P}_{CA}(j)\}$
    if j received ACK-Δc_{ij} from all $i \in (\mathcal{P}(j) \cup \mathcal{T}(j)) \cap \mathcal{P}_{CA}(j)$ **then**
      $\mathcal{P}^{new}(j) = \arg\min_{i \in (\mathcal{P}(j) \cup \mathcal{T}(j)) \cap \mathcal{P}_{CA}(j)} \{\Delta c_{ij}\}$
      transmit: ACK_{join} to $\mathcal{P}^{new}(j)$
      transmit: $ACK_{release}$ to $\mathcal{P}(j)$
      transmit: $ACK_{unchanged}$ to $\big((\mathcal{P}(j) \cup \mathcal{T}(j)) \cap \mathcal{P}_{CA}(j)\big) \setminus \{\mathcal{P}^{new}(j), \mathcal{P}(j)\}$
      $\mathcal{P}(j) \longleftarrow \mathcal{P}^{new}(j)$
    end if
    if i received REQ-Δc_{ij} **then**
      $\Delta c_{ij} = c_{ij} - \max_{k \in C(i) \setminus \{j\}} \{c_{ik}\}$
      transmit: ACK-Δc_{ij} to j
      waits to receive ACK_{join}, $ACK_{release}$, or $ACK_{unchanged}$ from j
      if i received ACK_{join} from j **then**
        $\mathcal{C}(i) = \mathcal{C}(i) \cup \{j\}$
      end if
      if i received $ACK_{release}$ from j **then**
        $\mathcal{C}(i) = \mathcal{C}(i) \setminus \{j\}$
      end if
    end if
  end for
end while

Next, we prove that BDP protocol converges.

Lemma 7.1. *Let $\mathcal{C}^{(l)}(i)$ denote the set of children of a node i after the l^{th} iteration, and for each node i, $P_i^{(l)} = \max_{j \in \mathcal{C}(i)} c_{ij}$. Moreover, let us define the total broadcast cost after the l^{th} iteration of BDP as*

$$P_{BDP}^{(l)} = \sum_{i \in V} P_i^{(l)}. \tag{54}$$

Then, $P_{BDP}^{(l)}$ is a decreasing function of l, and it converges.

Proof. At any iteration l, each node in the BDP protocol selects its parent based on whether or not the new parent reduces the total broadcast cost ($P_{BDP}^{(l)}$). Therefore, the total cost is a decreasing sequence of l. Moreover, $P_{BDP}^{(l)}$ is bounded from below, since $P_{BDP}^{(l)} \geq 0$ for all l. From the monotone convergence theorem [4], we conclude the $P_{BDP}^{(l)}$ is a convergent

sequence that converges to a fixed point. □

7.7 The Proposed Protocol for Finding Low-Cost Multicast Trees in Wireless Networks

Even more complex than the broadcast problem is the problem of minimum-cost multicast in wireless networks. In this section, we propose a simple extension of BDP that is suitable for multicast.

7.7.1 MDP: Multicast Decremental Power

To obtain a low-cost multicast tree, we first form a broadcast tree using BDP. Next, the broadcast tree is pruned by eliminating all transmissions that are not needed to reach the destinations. In other words, nodes with no downstream destinations will not transmit. Moreover, for the case with adjustable transmission ranges, each node adjusts its transmission range based on its farthest child in the multicast tree, with some more-distant children perhaps already having been pruned. To form the multicast tree in a decentralized fashion, we propose the following scheme. After the broadcast tree is formed, each destination node that is a leaf in the multicast tree, sends a "*join token*" back to its parent. Parents that receive a *join token* then send the tokens back to their parent and so on, until all the tokens reach the source. Therefore, only the nodes that receive a *join token* know that they are included in the multicast tree. Moreover, if a node receives a *join token* from its child, it adds it to its children's list for the multicast tree.

7.8 Computational Complexity of BDP and MDP

In this section, we estimate the computational complexity of the proposed algorithms. From the setup (Algorithm 4), it can be seen that in the adjustable communication radius model, $\mathcal{N}(j)$ is potentially of the order of $\Theta(N)$ for almost all $j \in V$. Thus, $\mathcal{P}_{CA}(j)$ has a potential size of $\Theta(N)$. Thus, to implement steps 1 through 6 in Algorithm 4, one potentially does $\Theta(N^2)$ computations. It can also be seen that implementing step 10 takes $\Theta(N)$ amount of work, where as implementing 11 through 28 takes a constant amount of work. Thus, it can be seen that the sum total of the work to implement steps 8 to 30 is $\Theta(L_{\max}N^2)$, which

is also the same as the total order of complexity of the algorithm since steps 1 through 7 have a complexity of $\Theta(N^2)$. While the order of complexity of BF algorithm is also $\Theta(N^2)$, BIP has a complexity of $\Theta(N^3)$ [74]. Similarly, in the fixed communication radius scenario, the expected size of $\mathcal{N}(j)$ and $\mathcal{P}_{CA}(j)$ are both $\Theta(\log N)$ with a high probability for any $j \in V$. This will ensure that the BDP algorithm in the finite communication radius case has a complexity of $\Theta(L_{\max} N \log N)$. The complexity of MDP in the two models assumed here is the same as that of BDP for the respective models since MDP uses BDP, and the additional work is of the order of $\Theta(N)$, which is a small overhead compared to the work already done to complete BDP.

7.9 Performance Evaluation: Adjustable Communication Radius Model

In this section, we evaluate the performance of BDP and MDP for the case of the adjustable communication radius model. We compare the cost of BDP with that of BF and BIP. We also compare the cost of MDP with that of MIP and BF (as pruned for multicast).

7.9.1 Broadcast

First, we define P_{BDP}, P_{BF}, and P_{BIP} to be the total power of a broadcast tree generated by BDP, BF, and BIP, respectively. We can normalize these values by dividing them to the power of the best scheme, i.e.,

$$\eta_{BDP} = \frac{P_{BDP}}{\min(P_{BDP}, P_{BF}, P_{BIP})}, \tag{55}$$

$$\eta_{BF} = \frac{P_{BF}}{\min(P_{BDP}, P_{BF}, P_{BIP})}, \tag{56}$$

$$\eta_{BIP} = \frac{P_{BIP}}{\min(P_{BDP}, P_{BF}, P_{BIP})}. \tag{57}$$

We consider the example of a small network given in [74]. Figures 7.7, 7.8, and 7.9 depict the broadcast tree rooted at node S formed by the BIP, BF, and BDP schemes, respectively. We have $P_{BIP} = 10.9$, $P_{BF} = 12.7$, and $P_{BDP} = 6.3$ for $\alpha = 2$. Interestingly, BDP results in the optimal tree, which was found by exhaustive search in [74]. This example, though small, shows the potential that BDP has for reducing the broadcast cost.

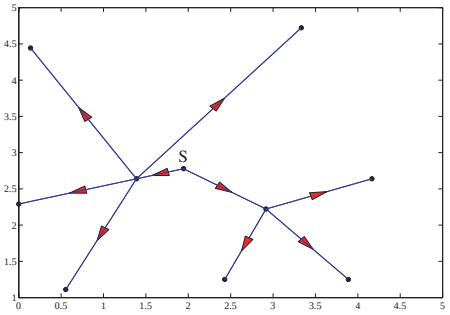


Figure 7.7: The broadcast tree formed by BIP. The broadcast cost is 10.9.

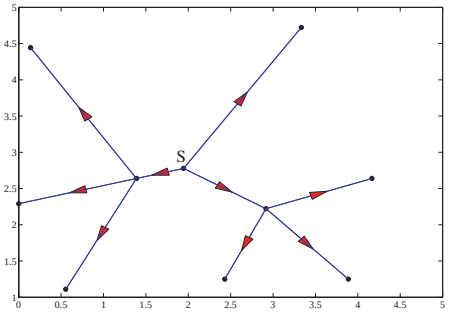


Figure 7.8: The broadcast tree formed by BF. The broadcast cost is 12.17.

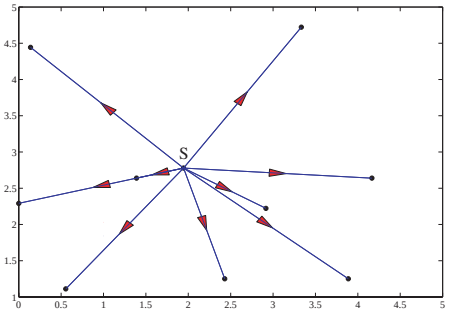


Figure 7.9: The broadcast tree formed by BDP. The broadcast cost is 6.3.

Next, we consider 100 instances of networks generated as follows. We have $N = 50$ nodes are randomly deployed in an area $50m \times 50m$ with the source node in the center of the area. Figure 7.10 shows the normalized broadcast tree power η_{BPD}, η_{BIP}, and η_{BF}, for different network instances. As we can see, for most of the network instances, BDP has a better performance than BIP. The average improvement in the power (cost) of the trees generated by BDP in comparison with that of BIP and BF is 7.2% and 25.1%, respectively.

Figure 7.11 depicts the empirical probability distribution function for the number of iterations required for the convergence of BDP. We note that in 96% of the network instances,

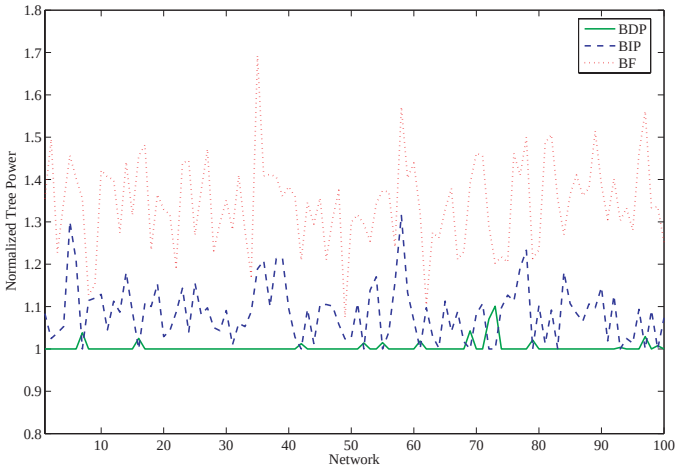


Figure 7.10: Normalized broadcast tree cost (power) for BDP, BIP, and BF, for 100 network instances; 50-node networks, deployed in an area $50m \times 50m$.

three iterations or less is sufficient for convergence. Here, we also note that the number of iterations is always less than or equal to five.

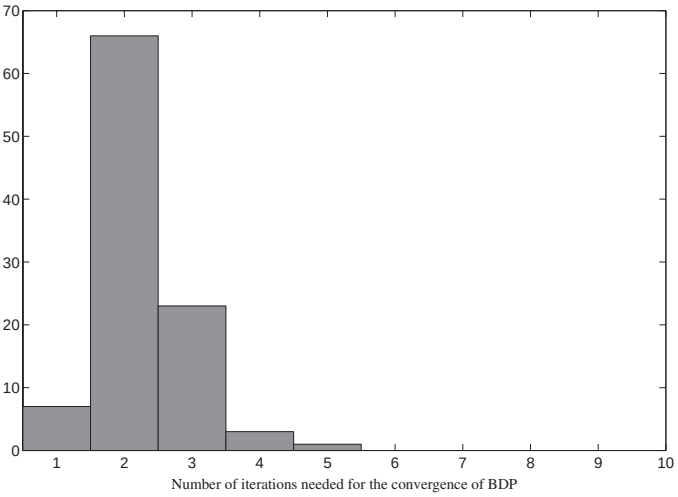


Figure 7.11: Distribution of the number of iterations required for the convergence of BDP; 50-node networks, deployed in an area $50m \times 50m$.

Next, we consider one instance of the aforementioned networks. Figures 7.12, 7.13, and 7.14 depict the broadcast trees formed by the BIP, BF, and BDP schemes, respectively. BDP converges in 3 iterations in this example. Assuming that $\alpha = 2$, we have $P_{BIP} = 1268$, $P_{BF} = 1516$, and $P_{BDP} = 964$. We see that BDP reduces the broadcast cost by 24% and 36% compared to BIP and BF, respectively.

Comparing Figures 7.12 and 7.14, we observe many interesting differences that show

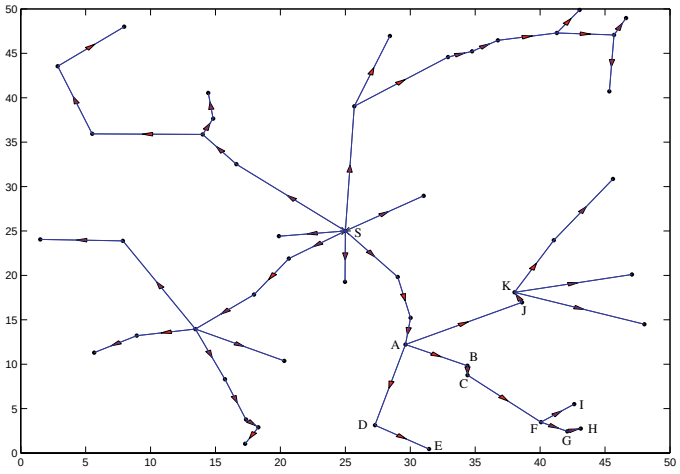


Figure 7.12: The broadcast tree formed by BIP. The broadcast cost is 1268.

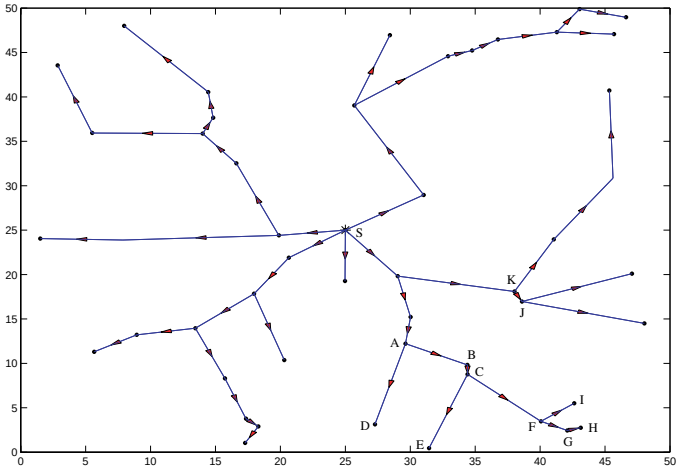


Figure 7.13: The broadcast tree formed by BF. The broadcast cost is 1516.

the effectiveness of BDP in comparison with BIP. For example, in BIP, node A is a parent for nodes B, D, and J. Moreover, node C has only F as its child, and D has only E as its child. Having so few children reduces transmission efficiency. In BDP, however, the power of A is just large enough to reach B. B is then a parent for C by a very small power, and C is a parent of many nodes, i.e., D, E, F, G, I, and K. The fact that each node may have more children means that the wireless channel property is better exploited in BDP.

Again, it should also be noted that BIP is a centralized scheme, while BDP is a distributed scheme. Therefore, not only does BDP result in a lower-cost tree, but it is also scalable and more practical.

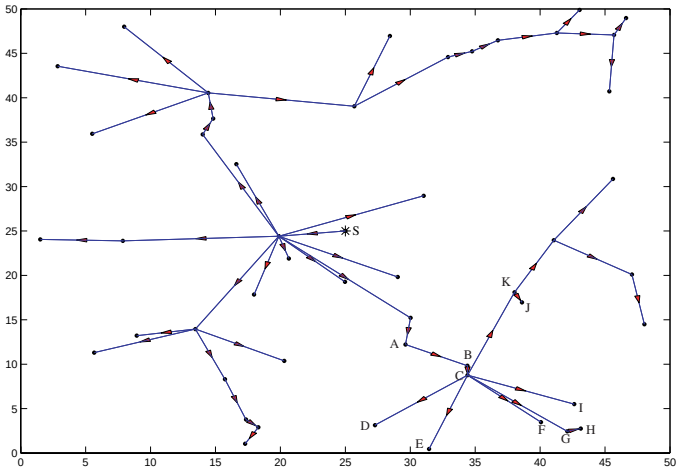


Figure 7.14: The broadcast tree formed by BDP. The broadcast cost is 964.

Next, we consider another example. Figure 7.15 shows the normalized broadcast tree cost (power) η_{BPD}, η_{BIP}, and η_{BF}, for 100 network instances. Networks include 20 nodes that have been randomly deployed in an area $10m \times 10m$, with the source node in the center of the area. As we can see, for most of the network instances, BDP has a better performance than BIP. The average improvement in the cost of the trees generated by BDP in comparison with that of BIP and BF is 15.8% and 26%, respectively.

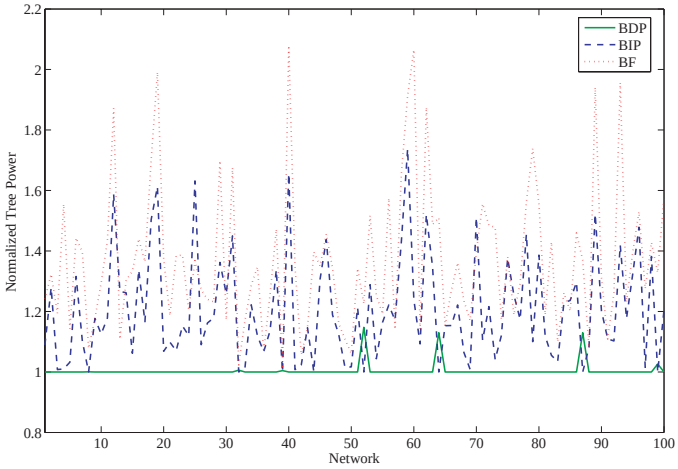


Figure 7.15: Normalized broadcast tree cost for BDP, BIP, and BF, for 100 network instances; 20-node networks, deployed in an area $10m \times 10m$.

Figure 7.16 depicts the empirical probability distribution function for the number of iterations required for the convergence of BDP. We note that in 93% of the network instances, only one or two iterations are enough for convergence. We also note that the number of

iterations is always less than or equal to four. Comparing Figures 7.16 and 7.11, we note that when the number of nodes in the network decreases, the average number of iterations for convergence of BDP also decreases.

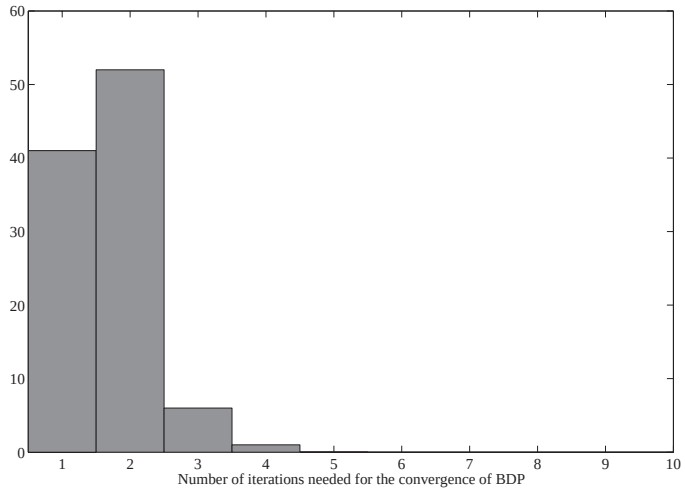


Figure 7.16: Distribution of the number of iterations required for the convergence of BDP; 20-node networks, deployed in an area $10m \times 10m$.

7.9.2 Multicast

We compare the cost of MDP with that of MIP, and BF (as pruned for multicast). For all three cases, we only consider the cost for reaching the destination nodes (which are a subset of the total nodes in the network). Table 7.1 compares the gain of using MDP versus MIP and BF, averaged over 100 network instances, for different number of destination nodes. The networks include 50 nodes that have been randomly deployed in an area $10m \times 10m$, with the source node in the center of the area. As we can see, when we only have one or two destinations, BF performs better than MDP and MIP. The reason is that BF provides the minimum-cost path to each destination. When the number of destinations is very small, the wireless medium cannot be well exploited. We also note that when we have just one destination (i.e., unicast), MIP has a better performance than MDP, since MDP takes advantage of the wireless medium more than MIP does, and wireless medium cannot be exploited in unicast. However, as the number of destination nodes increases, BF degrades and MDP outperforms both BF and MIP. Next, we consider one sample of these networks. Figures 7.17, 7.18, and 7.19 depict the multicast trees formed by MIP, BF, and MDP,

Table 7.1: Average gain of using MDP versus MIP and BF for different number of destinations. Average is taken over 100 network instances; 50-node networks, deployed in an area $10m \times 10m$.

# of destination nodes	Gain: MDP vs. MIP	Gain: MDP vs. BF
1	−15.6%	−38.5%
5	1.31%	−2.59%
15	5.07%	14.03%
25	5.88%	19.02%
35	6.7%	22.8%
50	7.2%	25.15%

respectively. The number of destination nodes is 15, all randomly chosen from the nodes in the network. The destination nodes are indicated by lager circles. MDP reduces the cost of multicast by 18.4% and 24.4% compared to MIP and BF, respectively.

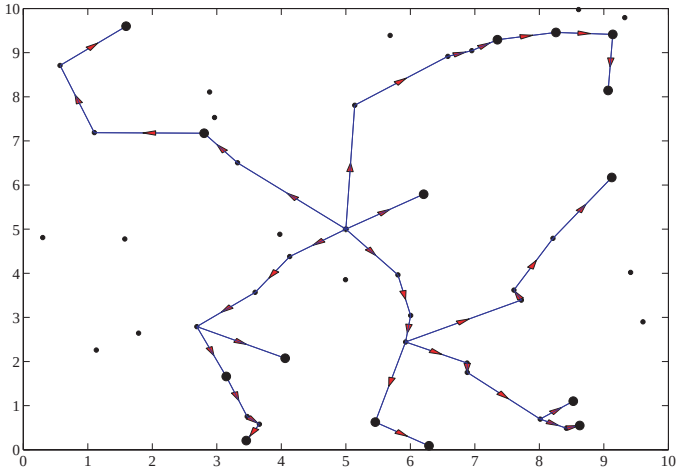


Figure 7.17: The multicast tree formed by MIP, for a network of 50 nodes and 15 randomly chosen destination nodes. Destination nodes are indicated by larger circles.

7.10 Performance Evaluation: Fixed Communication Radius Model

In this section, we consider broadcast and multicast problems for graphs $G(V, E, C)$, in which the nodes have a fixed transmission radius r, and, as before, each edge has a cost as was defined in (52). The proposed schemes remain the same even in this model, and we compare them with the unpruned and pruned BF schemes. However, we can no longer compare the results here with BIP and MIP, which work only when the nodes have an

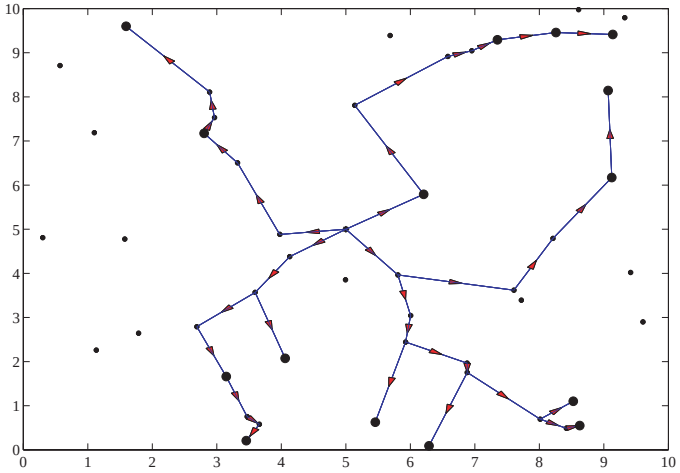


Figure 7.18: The multicast tree formed by BF, for a network of 50 nodes and 15 randomly chosen destination nodes. Destination nodes are indicated by larger circles.

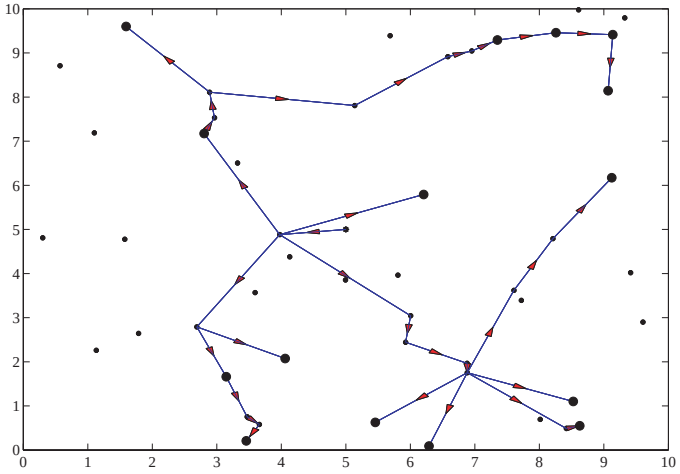


Figure 7.19: The multicast tree formed by MDP, for a network of 50 nodes and 15 randomly chosen destination nodes. Destination nodes are indicated by larger circles.

adjustable communication radius. Instead, in graphs with a small number of nodes, we compare our schemes with NC. We could not simulate NC for large networks because of the huge complexity of the optimization. NC is defined on a directed graph, and it becomes unclear as to what directions links must be assigned to reach the optimal performance. In this study, we have used directions based upon hop-distance from the source. On each edge, directions are chosen so that the node having a smaller hop-distance can potentially transmit packets to the one having the larger hop-distance. In cases where both ends of an edge have the same hop-distance, the one closer to the source in the sense of Euclidean distance can transmit to the one that is farther. We also present some simulations that

compare BDP with CRBcast and FTS, proposed in Chapters 5 and 6, respectively.

7.10.1 Broadcast

Figure 7.20 depicts the normalized tree cost (power) $\frac{P_{BDP}}{\min(P_{BDP},P_{BF})}$ and $\frac{P_{BDP}}{\min(P_{BDP},P_{BF})}$ formed by BDP and BF, respectively when $N = 200$ nodes with transmission radius $r = 20m$ are randomly deployed in an area $100m \times 100m$, for 500 network instances. The links are assumed to have packet erasure probabilities of either 0.45 or 0.55 with equal probability, and link costs are assumed to be proportional to $\frac{1}{1-\epsilon_{ij}}$, which are 1.8182 and 2.2222, respectively.

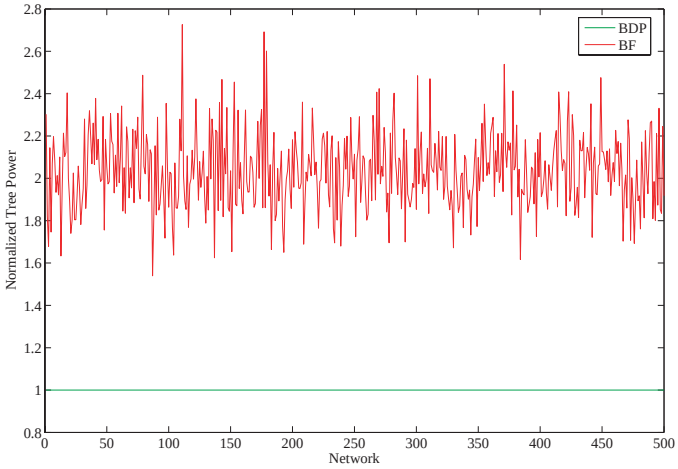


Figure 7.20: Normalized broadcast tree cost (power) for BDP and BF, for 500 network instances. $N = 200$ nodes with transmission radius $r = 20m$ are randomly deployed in an area $100m \times 100m$. The links are assumed to have packet erasure probabilities that are equiprobably either 0.45 or 0.55.

Figure 7.21 depicts average value $\frac{P_{BDP}^{(l)}}{P_{BF}}$ versus the number of iterations (l) of BDP. BDP starts with the BF tree (iteration zero) and improves the broadcast cost about 41% in just the first iteration. Improvement is about 49% in the second iteration, and 52% when the BDP converges. These substantial improvements gained in just one or two iterations are very desirable.

We also consider various scenarios with different link erasure values. Table 7.2 summarizes the average gains of employing BDP compared to BF. We observe the following interesting results. The highest gain among different link-cost scenarios is achieved when

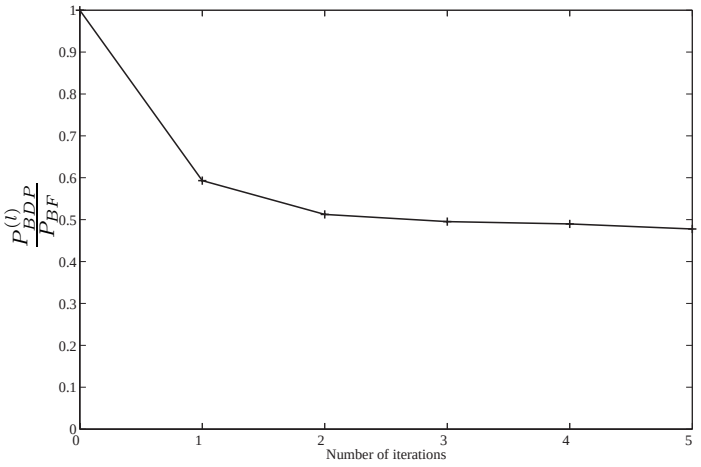


Figure 7.21: Average value $\frac{P_{BDP}^{(l)}}{P_{BF}}$ versus the number of iterations (l) of BDP. Substantial power reduction occurs in just the first two iterations.

link costs are about the same (i.e., when $\epsilon = 0.45$ or $\epsilon = 0.55$ equiprobably), and the lowest improvement happens when link costs are very different (i.e., when $\epsilon = 0.1$ or $\epsilon = 0.9$ equiprobably). We can explain this as follows. When the link costs are very different, BF avoids a large number of very costly links. Hence, by applying BDP, those remaining costly links are further avoided. Thus, the performance of BF and that of BDP are close. However, when the links have similar costs, the BDP algorithm can benefit a lot by changing the links in the tree and making it suitable for wireless cases. Comparing this case with the case that all links have the same cost (i.e., when $\epsilon = 0.5$), we see the former has a better gain.

Table 7.2: Average reduction in broadcast energy consumption of BDP compared to BF, for different link erasure probabilities (different link costs). Networks include 200 nodes with transmission radius $r = 20m$, which are randomly deployed in an area $100m \times 100m$.

Link erasure	Gain: BDP vs. BF
0.5	30%
0.45 or 0.55 equiprobably	52%
0.4 or 0.6 equiprobably	47%
0.35 or 0.65 equiprobably	43%
0.1 or 0.9 equiprobably	22%
Uniform in $[0, 1]$	42%

In Table 7.3, we compare the average gain of employing BDP versus NC for three

different network topologies, and also for three link erasure scenarios. As can be seen, in most cases, BDP outperforms NC because in BDP the directions of the edges are not pre-fixed and are selected based on the costs, whereas NC assumes a fixed directed graph. The maximum gain is 67.91%. There are few cases that NC outperforms BDP by less than 2%.

Table 7.3: Average reduction in broadcast energy consumption of BDP compared to NC for different network topologies and different link erasure scenarios. Networks consist of N nodes with transmission range r meter deployed uniformly at random in an area $100m \times 100m$.

Link erasure	(N, r)	Gain: BDP vs. NC
0.1 or 0.9 equiprobably	$(15, 23)$	11.30%
,,	$(20, 22)$	8.17%
,,	$(25, 21)$	3.25%
0.45 or 0.55 equiprobably	$(15, 23)$	5.27%
,,	$(20, 22)$	1.27%
,,	$(25, 21)$	-1.37%
Uniform in $[0, 1]$	$(15, 23)$	-1.95%
,,	$(20, 22)$	21.95%
,,	$(25, 21)$	67.91%

We also simulate BDP for the same network topology that we used to simulate CRBcast (in Section 5.4.3) and FTS (in Section 6.4.1.2). The network consists of $N = 10^4$ nodes with transmission range $r = 50m$, deployed uniformly at random in an area $A = 2000m \times 2000m$. The links are assumed to be lossless, i.e., $\epsilon = 0$. Table 7.4 presents the results. As can be seen, in comparison with CRBcast, BDP improves the energy efficiency by 13.5%. This is not a substantial improvement and as we see, FTS results in a better improvement than BDP. The reason is that in this case, the links are lossless and the transmission range is fixed for all the nodes. Therefore, all the links have the same cost. However, BDP improves the broadcast cost significantly when the link costs are very different, by avoiding the more costly links.

Our next results are based on simulating FTS, BF, and BDP in lossy networks, with different scenarios for link losses. We also considered two network topologies, $N = 200$ and 500 nodes with transmission ranges $r = 15m$ and $10m$, respectively, deployed randomly in

Table 7.4: Comparison of $\mathcal{N}_{/p/n}$ in BDP, BF, FTS, and CRBcast. The value of p is chosen as 0.25 for CRBcast.

Broadcasting Scheme	$\mathcal{N}_{/p/n}$
CRBcast	0.2769
FTS (without the fraction reduction phase)	0.2073
FTS (with the fraction reduction phase)	0.1600
BDP	0.2395

an area $100m \times 100m$. The simulation results are given in Table 7.5. As is shown, when link erasure probabilities or equivalently link costs are the same or very close, FTS has better or similar performance compared to BDP. However, when the link erasure probabilities are very different, BDP outperforms FTS since BDP takes into account the link costs and decreases the broadcast cost accordingly.

Table 7.5: Comparison of $\mathcal{N}_{/p/n}$ in FTS, BF, and BDP in lossy networks, with different scenarios for link losses. Networks consist of N nodes with transmission range r meter deployed uniformly at random in an area $100m \times 100m$.

Link erasure	(N, r)	FTS	BF	BDP
0.5	$(200, 15)$	0.442	0.734	0.541
,,	$(500, 10)$	0.401	0.76	0.523
$0.45, 0.55$ equiprobably	$(200, 15)$	0.487	0.83	0.473
,,	$(500, 10)$	0.442	0.868	0.45
$0.4, 0.6$ equiprobably	$(200, 15)$	0.55	0.83	0.50
,,	$(500, 10)$	0.49	0.86	0.47
$0.35, 0.65$ equiprobably	$(200, 15)$	0.62	0.831	0.51
,,	$(500, 10)$	0.567	0.84	0.49
$0.2, 0.8$ equiprobably	$(200, 15)$	1.07	0.62	0.51
,,	$(500, 10)$	0.978	0.607	0.482
$0.1, 0.9$ equiprobably	$(200, 15)$	2.12	0.57	0.48
,,	$(500, 10)$	1.95	0.55	0.442

7.10.2 Multicast

For multicast, we consider a network consisting of $N = 1000$ nodes with a transmission radius $r = 8m$, which are deployed uniformly at random in a square field with area $100m \times 100m$. The source node is placed in the center of the field. The links are assumed to have

packet erasure probabilities of either 0.45 or 0.55 with equal probability. We consider cases for which the number of destination nodes is 1, 10 ,25, 50, 100, 250, 500, and 1000. Table 7.6 summarizes the average reduction in multicast energy consumption of MDP compared to BF. The average has been taken over 50 network instances. As we can see, for the case that has only one destination node (unicast) BF provides a much better result than MDP. However, the gain of applying MDP increases as the number of destination nodes increases. For the special case of broadcast, MDP reduces the energy consumption by 55.46%.

Table 7.6: Average reduction in multicast energy consumption of MDP compared to BF for different number of destinations. The average is taken over 50 network instances; 1000-node networks, deployed in an area $100m \times 100m$. Transmission radius $r = 8m$, and the links' erasure probabilities are equiprobably either 0.45 or 0.55.

# of destination nodes	Gain: MDP vs. BF
1	-23.63%
10	-9.19%
25	2.81%
50	16.29%
100	26.8%
250	40.21%
500	48.46%
1000	55.46%

Next, we consider one instance of the above network topology with 100 destination nodes. Figures 7.22 and 7.23 depict the multicast trees formed by BF and MDP, respectively. The destination nodes are indicated by rings. The links' erasure probabilities are equiprobably either 0.45 or 0.55, and the links with larger losses are indicated by thinner (and lighter-color) lines. We note that the number of relaying nodes in the BF tree is 198 while this number is 128 in the MDP tree. In MDP, on the average, one transmission benefits more nodes than it does in BF. We have $P_{BF} = 2.42 \times 10^4$ and $P_{MDP} = 1.61 \times 10^4$. This means 33% reduction in multicast energy consumption.

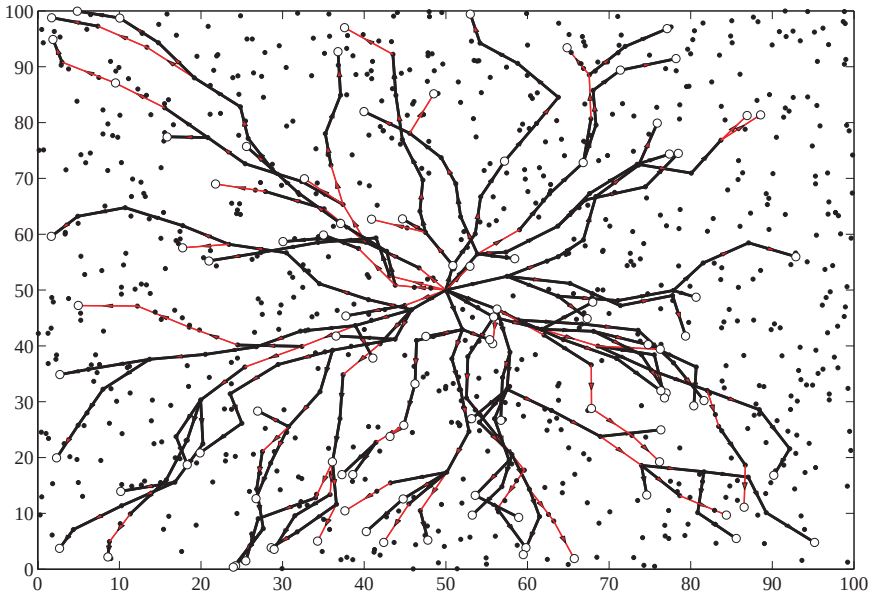


Figure 7.22: The multicast tree formed by BF, for a network of 1000 nodes with transmission radius $r = 8m$ and 100 randomly chosen destination nodes. The source node is in the center of the area. Destination nodes are indicated by rings. The links' erasure probabilities are equiprobably either 0.45 or 0.55, and links with larger losses are indicated by thinner (and lighter-color) lines.

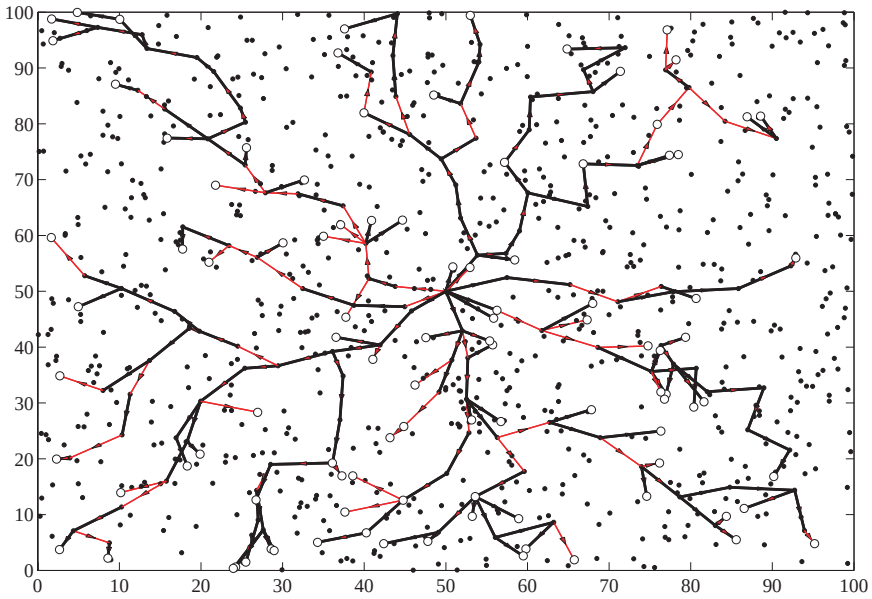


Figure 7.23: The multicast tree formed by MDP, for a network of 1000 nodes with transmission radius $r = 8m$ and 100 randomly chosen destination nodes. The source node is in the center of the area. Destination nodes are indicated by rings. The links' erasure probabilities are equiprobably either 0.45 or 0.55, and links with larger losses are indicated by thinner (and lighter-color) lines.

In Table 7.7, we compare the average gain achieved by employing MDP versus employing NC for three different network topologies when the links' erasure probabilities are equiprobably either 0.1 or 0.9, and when 10 nodes are randomly chosen as destination nodes. As we can see, MDP outperforms NC due to its flexibility to choose the directions of the links in networks.

Table 7.7: The average gain of MDP versus NC for different network topologies. Networks consist of N nodes with transmission range r meter deployed uniformly at random in a square area $100m \times 100m$. The links' erasure probabilities are equiprobably either 0.9 or 0.1. Number of destinations, which are randomly selected, is 10.

(N, r)	Gain: MDP vs. NC
$(15, 23)$	17.7%
$(20, 22)$	14.14%
$(25, 21)$	30.16%

7.11 Conclusion

In this chapter, we considered the important and challenging problem of finding minimum-cost broadcast and multicast trees in wireless networks. The problem is an NP-complete problem. We proposed two simple and heuristic suboptimal protocols, referred to as *broadcast decremental power* (BDP) and *multicast decremental power* (MDP), that exploit the wireless medium property to reduce broadcast/multicast costs. The proposed schemes also have the advantage of being distributed and scalable. They are based on evolving a given spanning tree (e.g., Bellman-Ford tree) such that the cost of the resulting tree is reduced by considering the wireless broadcast medium. We showed that the proposed schemes converge, and that the complexity of their implementation is at most quadratic in number of nodes in a network. We considered two communication radius models, adjustable and fixed. In the former, we compared our schemes with BF, BIP (for the broadcast case), a pruned BF and MIP (for the multicast case). For the latter, we compared the proposed scheme with BF, CRBcast, FTS, and NC (which we could implement only for small networks due to the complexity of optimization). For both models, simulation results demonstrated the superiority of the proposed schemes in terms of reducing broadcast/multicast costs in the

majority of the scenarios. Our simulations confirm that BDP substantially improves the broadcast cost in comparison with BF, BIP, and NC. MDP also improves the multicast cost considerably in comparison with BF, MIP, and NC, when the destination nodes are not very few.

CHAPTER VIII

CONCLUSION

8.1 Contributions

In this dissertation, we explored and investigated new theoretical and practical challenges in the application of modern error-control coding schemes, namely low-density parity-check (LDPC) codes and rateless codes, in error-prone wireless ad-hoc and sensor networks. The application of coding in such networks is necessary not only because of the noisy nature of wireless channels, but also because of the nature of communications in wireless ad-hoc and sensor networks.

First, we studied the problem of unequal error protection and designed several coding schemes for practical applications such as multimedia communication in wireless ad-hoc and sensor networks. Next, we examined the problem of reliable and energy-efficient broadcast and multicast in such networks. The communication in wireless ad-hoc and sensor networks is usually done in a multihop fashion, and each single transmission can potentially be received by all the neighboring nodes. These properties provide new opportunities and also challenges. One challenge is to determine the optimal set of nodes in the network that forward packets. A second challenge is to determine the content of the forwarded packets. In other words, should forwarding nodes only relay the incoming packets or also process the packets (i.e., do mathematical operations on the packets and encode them) before sending them? These challenges fit into the context of coding within a network, introduced for the first time by Ahlswede et al. in 2000 [2]. A part of our work was to tackle these problems using different approaches that touch on the four very large fields of wireless networking, modern coding theory, graph theory, and percolation theory.

Next, we summarize the contributions of this dissertation:

8.1.1 Design and Analysis of Unequal Error Protection Low-Density Parity-Check Codes

We introduced two frameworks for the design and analysis of LDPC codes that provide unequal error protection (UEP) [52, 57, 58], with a potential application in multimedia communication in wireless networks. The two proposed schemes [49, 52, 57, 58] can provide a higher protection for more important parts of data and less protection for less important parts. The first scheme is based on traditional bipartite Tanner graphs and the second scheme is based on combining two Tanner graphs resulting a 3-partite ensemble. For both schemes, we derived density evolution formulas on the binary erasure channel and used these formulas to optimize the degree distribution of the codes. Using these techniques, we found codes with overall good performance as well as significant UEP property. We also showed that the proposed codes have linear encoding complexity, which is very desirable for practical applications. Using the density evolution formulas, one can optimize codes based on the requirements of a given problem.

8.1.2 Design and Analysis of Unequal Error Protection Rateless Codes

We developed rateless codes that provide UEP [54, 56, 59]. All other studies thus far were based on designing rateless codes for equal error protection, and this study is the first one to propose UEP-rateless codes. To analyze the proposed codes, we first investigated the performance of these codes under the iterative decoding when the number of information symbols (n) goes to infinity. Then, we examined the performance of the proposed codes under the ML decoding for finite-length n. We derived upper and lower bounds on ML decoding bit error probabilities for the proposed codes and showed that UEP-rateless codes are able to provide very low error rates for more important bits with only a small performance loss for the less important bits. Moreover, we showed that given a target bit error rate, different parts of the information symbols can be decoded after receiving different numbers of encoded symbols. This implies that the information can be recovered in a progressive manner. This property is of interest in many practical applications such as media-on-demand systems.

8.1.3 Analysis of Rateless Codes under the Maximum-Likelihood Decoding

We derived upper and lower bounds on the maximum-likelihood (ML) decoding of finite-length rateless codes [53]. The bounds are shown to be tight for small error rates. These bounds on ML decoding are of interest, as they provide an ultimate limit on the code performance. Specifically, they can be used to optimize the degree distribution of rateless codes when the decoding scheme performs similar or close to the ML decoding. Examples of such decoding schemes are the Maxwell [40] and the guessing-based [48] decoders.

8.1.4 Rateless (Fountain) Coding for Reliable and Energy-Efficient Broadcast in Wireless Ad-Hoc and Sensor Networks

We studied the application of rateless codes in efficient data broadcasting in wireless sensor networks. We proposed two main protocols, each of which assumes a different level of information about the network topology and offers a solution accordingly. The proposed schemes are energy efficient and provide reliable data delivery to all the nodes in the network. They are also distributed and scalable and have low implementation complexity.

8.1.4.1 Collaborative Rateless Broadcast

We proposed the collaborative rateless broadcast (CRBcast) protocol [55], which is a two-phase protocol that utilizes rateless coding in conjunction with a simple and scalable broadcast scheme called probabilistic broadcast (PBcast). Since the characteristics of PBcast influence CRBcast, we investigated PBcast analytically and with simulations. Then, we investigated the effectiveness of CRBcast. We extensively studied the properties of CRBcast by analysis and simulations. We concluded that CRBcast is not only a reliable and energy efficient scheme but also a scalable scheme that requires no knowledge of the network topology.

8.1.4.2 Fractional Transmission Scheme

We proposed a fractional transmission scheme (FTS) for the cases in which each node knows its hop distance from the source and those of its immediate neighbors [62, 63]. This extra

information can be exploited so that different neighbors of a node share the data delivery, and each node sends only a fraction of the total encoded packets required by a receiving node. By utilizing rateless encoding and decoding at each node, FTS provides reliability and energy efficiency. We compared FTS with several competitive schemes such as broadcast incremental power (BIP) and network coding (NC). The results suggest that although BIP is a centralized scheme and NC utilizes a computationally expensive optimization, the energy efficiency of FTS is comparable to the energy efficiency of BIP and NC. Because of its ease of implementation, low complexity, and competitive performance, FTS is a viable approach for broadcasting in large-scale wireless ad-hoc and sensor networks.

8.1.5 Efficient Broadcast and Multicast Trees for Wireless Networks

We proposed broadcast decremental power (BDP) for general wireless networks, in which each link has been assigned a cost according to the distance between its incident nodes or its erasure probability, and each node in the network is aware of the value of the costs of its incident links [61]. BDP finds a low-cost (low-energy) broadcast tree for a wireless network. We also extended BDP for multicasting in wireless networks, denoted as multicast decremental power (MDP). Our simulations confirm that BDP substantially improves the broadcast cost in comparison with the Bellman-Ford (BF) scheme, BIP, and NC. MDP also improves the multicast cost considerably in comparison with BF, MIP, and NC, when the number of destination nodes is not too small. Both BDP and MDP also have the advantage that they can be implemented in such a way that they provide low-latency broadcast and multicast.

8.2 Suggestions for Future Research

This dissertation opened up many interesting theoretical and practical research possibilities in error-control coding, wireless ad-hoc and sensor networks, and other related areas in communications and signal processing. In the following, some of the interesting and potentially rich open directions for future research are listed.

- Applying the proposed unequal error protection schemes to practical scenarios such

as video streaming and media-on-demand systems

- Joint image processing and UEP coding
- Designing short- to moderate-length rateless codes, whose iterative decoding and ML decoding performance are close, using the ML decoding bounds derived in this thesis
- Multimedia broadcasting/multicasting in wireless sensor and actor networks utilizing the proposed UEP coding schemes
- Extending CRBcast and FTS to efficient multicasting in wireless ad-hoc and sensor networks
- Extending the application of rateless-coding based broadcasting to mobile ad-hoc networks

APPENDIX A

SUPPLEMENTARY FOR CHAPTER 4

Proof of Lemma 4.9. Let us assume that H' is the parity-check matrix corresponding to the pre-code $\mathcal{C}$. Moreover, let H'_e be an $m \times e$ matrix composed of the columns of H' that correspond to the variable nodes that have not been recovered after the LT-decoding process. Note that elements of H'_e are independently one with probability ρ. We want to obtain the probability that the i^{th} bit cannot be determined either by the LT decoder or by the pre-code decoder, for an arbitrary $i \in \{1, 2, \ldots, n\}$. Let the j^{th} column in H'_e correspond to the i^{th} input bit of the LT code. We have

$$\begin{aligned} &Pr\{\text{The pre-code fails to determine the } i^{th} \text{ bit}\} \\ &= Pr\{\exists x \in GF(2)^e, x(j) = 1 : H'_e x^T = 0^T\} \\ &\leq \sum_{x \in GF(2)^e,\, x(j)=1} Pr\{H'_e x^T = 0^T\} \end{aligned}$$

Let $x \in GF(2)^e$, $x(j) = 1$, and weight of x is w. Let R_l denote the l^{th} row of H'_e. We have

$$Pr\{H'_e x^T = 0^T\} = \prod_{l=1}^{m} Pr\{R_l x^T = 0\}.$$

Note that the events $R_l x^T = 0$ for $l = 1, \ldots, m$ are equiprobable and independent. Let $A(w, \rho)$ be the probability that even number of 1's occurs in a stream of independent 0's and 1's of length w when probability of 1 is ρ. We have

$$Pr\{R_l x^T = 0\} = A(w, \rho) \tag{58}$$

$$= \frac{1 + (1 - 2\rho)^w}{2}, \tag{59}$$

where (59) is concluded from Lemma A.2 in Appendix A. Assuming that ϵ is the bit error probability of the LT code, we conclude

$$p_b^{ML} \leq \sum_{e=0}^{n} \binom{n}{e} \epsilon^e (1 - \epsilon)^{n-e} \frac{e}{n} \min\left\{1, \sum_{w=1}^{e} \binom{e-1}{w-1} A^m(w, \rho)\right\}. \tag{60}$$

However, instead of the exact value of ϵ, we have bounds on it. Assuming ϵ_U is an upper bound on ϵ and using Lemma A.1 in Appendix A we can easily conclude the assertion. □

Proof of Lemma 4.10. Consider H'_e as was defined before. Let the j^{th} column in H'_e correspond to the i^{th} bit. We have:

$$\begin{aligned}
&Pr\{\text{The pre-code fails to determine the } i^{th} \text{ bit}\} \\
&= Pr\{\exists x \in GF(2)^e, x(j) = 1 : H'_e x^T = 0^T\} \\
&\geq \sum_{x \in GF(2)^e, x(j)=1} Pr\{H'_e x^T = 0^T\} \\
&\quad - \frac{1}{2} \sum_{x,y \in GF(2)^e,\, x(j)=1,\, y(j)=1,\, x \neq y} Pr\{H'_e x^T = 0^T, H'_e y^T = 0^T\}
\end{aligned}$$

in which the inequality results in from the Bonferroni inequality [12]. The first term can be calculated using Lemma 4.9. Let $x, y \in GF(2)^e$ such that $x(j) = 1$, $y(j) = 1$, and $x \neq y$. We define three binary vectors z_0, z_1, and $z_2 \in GF(2)^e$ such that for $t = 1, \ldots, e$, $z_0(t) = 1$ if and only if $x(t) = 1$ and $y(t) = 1$, $z_1(t) = 1$ if and only if $x(t) = 1$ and $y(t) = 0$, and $z_2(t) = 1$ if and only if $x(t) = 0$ and $y(t) = 1$. Let w_0, w_1, and w_2 be the weights of vectors z_0, z_1, and z_2, respectively. We have

$$Pr\{H'_e x^T = 0^T, H'_e y^T = 0^T\} \tag{61}$$

$$= \prod_{l=1}^{m} Pr\{R_l z_0^T = R_l z_1^T = R_l z_2^T\} \tag{62}$$

$$= \left(A(w_0, \rho) A(w_1, \rho) A(w_2, \rho) + \overline{A}(w_0, \rho) \overline{A}(w_1, \rho) \overline{A}(w_2, \rho) \right)^m \tag{63}$$

in which R_l denotes the l^{th} row of H'_e, (62) is resulted from the independency of the elements of H'_e, and (63) is obtained easily by the definition of $A(.)$ as (31) and $\overline{A(\cdot)} := 1 - A(\cdot)$. Summing over all possible values for e, w_0, w_1, and w_2 and noting that w_1 and w_2 cannot be zero simultaneously (since $x \neq y$), we conclude the assertion. □

Lemma A.1. *Let us define $g(\epsilon) = \sum_{e=0}^{n} \binom{n}{e} \epsilon^e (1-\epsilon)^{n-e} f(e)$. Then, $g(\epsilon)$ is a non-decreasing function of ϵ if $f(e)$ is a non-decreasing function of e.*

Proof. (By Badri N. Vellambi) Let us define $h_{\epsilon_1}(e) = \binom{n}{e} \epsilon_1^e (1-\epsilon_1)^{n-e}$ and $h_{\epsilon_2}(e) = \binom{n}{e} \epsilon_2^e (1-\epsilon_2)^{n-e}$. We need to show that if $\epsilon_2 > \epsilon_1$ then $\sum_{e=0}^{n} h_{\epsilon_2}(e) f(e) \geq \sum_{e=0}^{n} h_{\epsilon_1}(e) f(e)$. Since $\epsilon_2 > \epsilon_1$ and due to the nature of functions $h_{\epsilon_1}(e)$ and $h_{\epsilon_2}(e)$ it can be easily shown that

there exists an integer e_0 such that $h_{\epsilon_2}(e) > h_{\epsilon_1}(e)$ if and only if $e \geq e_0$. Therefore,

$$\begin{aligned}
&\sum_{e=0}^{n} \big(h_{\epsilon_2}(e) - h_{\epsilon_1}(e)\big) f(e) \\
&\quad = \sum_{e: h_{\epsilon_2}(e) > h_{\epsilon_1}(e)} \big(h_{\epsilon_2}(e) - h_{\epsilon_1}(e)\big) f(e) \\
&\qquad - \sum_{e: h_{\epsilon_2}(e) \leq h_{\epsilon_1}(e)} \big(h_{\epsilon_1}(e) - h_{\epsilon_2}(e)\big) f(e) \qquad (64) \\
&\quad \geq f(e_0) \sum_{e=e_0}^{n} \big(h_{\epsilon_2}(e) - h_{\epsilon_1}(e)\big) - f(e_0 - 1) \sum_{e=0}^{e_0-1} \big(h_{\epsilon_1}(e) - h_{\epsilon_2}(e)\big) \\
&\quad = \big(f(e_0) - f(e_0 - 1)\big) \sum_{e=e_0}^{n} \big(h_{\epsilon_2}(e) - h_{\epsilon_1}(e)\big) \geq 0,
\end{aligned}$$

where in (64), we use the fact that

$$\sum_{e=0}^{n} h_{\epsilon_1}(e) = \sum_{e=0}^{n} h_{\epsilon_2}(e) = 1.$$

Therefore,

$$\sum_{e=e_0}^{n} \big(h_{\epsilon_2}(e) - h_{\epsilon_1}(e)\big) = \sum_{e=0}^{e_0-1} \big(h_{\epsilon_1}(e) - h_{\epsilon_2}(e)\big).$$

□

Lemma A.2. *Let $\underline{x}$ be a random binary vector of length ω. Each element of $\underline{x}$ is independently 1 with probability ρ. Then, $A(\omega, \rho)$ defined as the probability that $\underline{x}$ has even number of 1 is given by*

$$A(\omega, \rho) = \frac{1 + (1 - 2\rho)^{\omega}}{2}. \qquad (65)$$

Proof. Since elements of $\underline{x}$ are chosen independently, we have:

$$A(\omega, \rho) = \sum_{i=0,2,\ldots,2\lfloor \frac{\omega}{2} \rfloor} \binom{\omega}{i} \rho^i (1 - \rho)^{\omega - i}. \qquad (66)$$

From binomial series, we have:

$$(\rho + (1 - \rho))^{\omega} = \sum_{i=0}^{\omega} \binom{\omega}{i} \rho^i (1 - \rho)^{\omega - i}, \qquad (67)$$

and

$$(-\rho + (1 - \rho))^{\omega} = \sum_{i=0}^{\omega} \binom{\omega}{i} (-\rho)^i (1 - \rho)^{\omega - i}. \qquad (68)$$

Adding two sides of (67) and (68), we conclude

$$1 + (1-2\rho)^{\omega} = \sum_{i=0}^{\omega} \binom{\omega}{i} [(-\rho)^i + \rho^i](1-\rho)^{\omega-i} \tag{69}$$

$$= \sum_{i=0,2,\ldots,2\lfloor \frac{\omega}{2} \rfloor} \binom{\omega}{i} 2\rho^i (1-\rho)^{\omega-i} \tag{70}$$

$$= 2A(\omega, \rho). \tag{71}$$

Therefore,

$$A(\omega, \rho) = \frac{1 + (1-2\rho)^{\omega}}{2}. \tag{72}$$

□

APPENDIX B

SUPPLEMENTARY FOR CHAPTER 6

Lemma B.1. *Let* $r \in \mathbb{N}$, $r \geq 2$, $M = \{\underline{\alpha} \in [0,1]^r : \alpha_i + \alpha_{i+1} \geq 1, i \in 1, \ldots, r-1, \alpha_1 = 1, \alpha_r = 1\}$, *then*

$$T_r \triangleq \min_{\underline{\alpha} \in M} \sum_i \alpha_i = \lceil \frac{r+1}{2} \rceil \tag{73}$$

Proof. (By Badri N. Vellambi) Suppose that r is even. Then,

$$T_r = \min_{\underline{\alpha} \in M} \Big(\alpha_1 + (\alpha_2 + \alpha_3) + \ldots + (\alpha_{r-2} + \alpha_{r-1}) + \alpha_r \Big) \geq \frac{r+2}{2}\,. \tag{74}$$

It can be seen that the equality is attained at $\underline{\alpha}^* = (\alpha_1^*, \ldots, \alpha_r^*)$, where $\alpha_i^* = \begin{cases} 1 & \text{if } i = 1, r \\ 0.5 & \text{otherwise} \end{cases}$

In the case that r is odd, we see that

$$T_r = 2 + \min_{\underline{\alpha} \in M} ((\alpha_2 + \alpha_3) + \ldots + (\alpha_{r-3} + \alpha_{r-2}) + \alpha_{r-1}) \geq \frac{r+1}{2}\,. \tag{75}$$

Clearly, the equality is attained at $\underline{\alpha}^* = (\alpha_1^*, \ldots, \alpha_r^*)$, where $\alpha_i^* = \begin{cases} 1 & \text{if } i \text{ is odd} \\ 0 & \text{otherwise} \end{cases}$ □

Lemma B.2. *Let* $r \in \mathbb{N}$, $r > 1$. *Let* $M = \{\underline{\alpha} \in [0,1]^r : \alpha_i + \alpha_{i+1} \geq 1, \forall\, i = 1, ..., r-1\}$. *Then*

$$T_r^{'} \triangleq \min_{\underline{\alpha} \in M} \sum_{i=1}^{r} \alpha_i = \lfloor \frac{r}{2} \rfloor \tag{76}$$

Proof. (By Badri N. Vellambi) Let r be even. Then

$$T_r^{'} = \min_{\underline{\alpha} \in M} (\alpha_1 + ... + \alpha_r) = \min_{\underline{\alpha} \in M} \Big((\alpha_1 + \alpha_2) + ...(\alpha_{r-1} + \alpha_r) \Big) \geq \frac{r}{2}\,. \tag{77}$$

It can be verified that this minimum is attained at $\underline{\alpha} = \underline{\alpha}^* = (\alpha_1^*, \ldots, \alpha_r^*)$, where $\alpha_i^* = \frac{1}{2}$, $i = 1, \ldots, r$. In the case that r is odd,

$$\begin{aligned} T_r^{'} &= \min_{\underline{\alpha} \in M} \Big((\alpha_1 + \alpha_2) + ...(\alpha_{r-2} + \alpha_{r-1}) + \alpha_r \Big) && (78) \\ &\geq \min_{\underline{\alpha} \in M} \Big((\alpha_1 + \alpha_2) + ...(\alpha_{r-2} + \alpha_{r-1}) \Big) + \min_{\underline{\alpha} \in M} \alpha_r \geq \frac{r-1}{2}\,. && (79) \end{aligned}$$

Moreover, it can be seen that the bound is reached at $\underline{\alpha}^* = (\alpha_1^*, \ldots, \alpha_r^*)$, where

$$\alpha_i^* = \begin{cases} 0 & \text{if } i \text{ is odd} \\ 1 & \text{otherwise} \end{cases} .$$

□

REFERENCES

[1] http://lthcwww.epfl.ch/research/ldpcopt/(Accessed 2004).

[2] Ahlswede, R., Cai, N., Li, S. Y. R., and Yeung, R. W., "Network information flow," *IEEE Trans. on Inf. Theory*, vol. 46, pp. 1204–1216, July 2000.

[3] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E., "Wireless sensor networks: A survey," *Computer Networks (Amsterdam, Netherlands: 1999)*, vol. 38, pp. 393–422, Mar. 2002.

[4] Bartle, R. G., *The elements of real analysis*. Wiley, second ed., 1976.

[5] Berrou, C., Glavieux, A., and Thitimajshima, P., "Near optimum error-correcting coding and decoding: Turbo-codes," *in Proc. IEEE Int. Conf. on Communications*, pp. 1064–1070, May 1993.

[6] Bertsekas, D. and Gallager, R., *Data Networks*. Englewood Cliffs, NJ: Prentice Hall, 1987.

[7] Birman, K. P., Hayden, M., Ozkasap, O., Xiao, Z., Budiu, M., and Minsky, Y., "Bimodal multicast," *ACM Transactions on Computer Systems*, vol. 17, pp. 41–88, May 1999.

[8] Boyarinov, I. and Katsman, G., "Linear unequal error protection codes," *IEEE Trans. on Information Theory*, vol. IT-27, pp. 168–175, Mar. 1981.

[9] Brissett, R. and Sodha, J., "Unequal error protection using convolutional codes," *IEEE SOUTHEASTCON '96*, pp. 242–245, Apr. 1996.

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

[11] Byers, J. W., Luby, M., and Mitzenmacher, M., "A digital fountain approach to asynchronous reliable multicast," *IEEE Journal on Selected Areas in Comm.*, vol. 20, pp. 1528–1540, Oct. 2002.

[12] Comtet, L., *Advanced Combinatorics*. 1974.

[13] Elias, P., "Coding for two noisy channels," *Information Theory, Third London Symposium*, pp. 61–76, 1955.

[14] Gallager, R. G., *Low-Density Parity-Check Codes*. PhD thesis, MIT, 1963.

[15] Garcy, M. R. and Johnson, D. S., *Computers and Tractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Company, New York, 1979.

[16] Guha, S. and Khuller, S., "Approximation algorithms for connected dominating sets," *Algorithmica*, vol. 20, no. 4, pp. 374–387, 1998.

[17] Gupta, P. and Kumar, P. R., "Critical power for asymptotic connectivity in wireless networks," *Stochastic Analysis, Control, Optimization and Applications: A volume in Honor of W. H. Fleming, W. M. McEneaney, G. Yin and Q. Zhang (Eds.)*, 1998.

[18] Haas, Z. J., Halpern, J. Y., and Li, L., "Gossip-based ad hoc routing," *in Proc. IEEE INFOCOM*, pp. 1707–1716, June 2002.

[19] Hamming, R., "Error detecting and error correcting codes," *Bell Systems Technical Journal*, vol. 29, pp. 147–160, Apr. 1950.

[20] Ho, C., Obraczka, K., Tsudik, G., and Viswanath, K., "Flooding for reliable multicast in multi-hop ad hoc networks," *in Proc. Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communication*, pp. 64–71, 1999.

[21] Kilgus, C. C. and Gore, W. C., "Cyclic codes with unequal error protection," *IEEE Trans. on Information Theory*, vol. 17, pp. 214–215, Mar. 1971.

[22] Kilgus, C. C. and Gore, W. C., "A class of cyclic unequal-error-protection codes," *IEEE Trans. on Information Theory*, vol. 18, pp. 687–690, Sept. 1972.

[23] Kim, J.-S., Zhang, Q., and Agrawal, D. P., "Probabilistic broadcasting based on coverage area and neighbor confirmation in mobile ad hoc networks," *in Proc. IEEE Globecom Workshops*, pp. 96–101, Nov-Dec 2004.

[24] Krishnamachari, B., Wicker, S. B., and Bejar, R., "Phase transition phenomena in wireless ad-hoc networks," *in Proc. IEEE GLOBECOM, San Antonio, TX*, Nov. 2001.

[25] Liang, W., "Constructing minimum-energy broadcast trees in wireless ad hoc networks," *in Proc. Third ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 112–122, 2002.

[26] Lim, H. and Kim, C., "Multicast tree construction and flooding in wireless ad hoc networks," *in Proc. the 3rd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 61–68, Aug. 2000.

[27] Luby, M., Mitzenmacher, and Shokrallahi, A., "Analysis of random processes via And-Or tree evaluation," *in Proc. 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 364–373, 1998.

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

[29] Luby, M. G., Mitzenmacher, M., Shokrollahi, M. A., and Spielman, D., "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. on Information Theory*, vol. 47, pp. 585–598, Feb. 2001.

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

[31] Luby, M., "LT codes," *in Proc. 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002.

[32] Lun, D. S., Medard, M., and Effros, M., "On coding for reliable communication over packet networks," *in Proc. Allerton conference on communication, control, and computing*, 2004.

[33] Lun, D. S., Medard, M., and Koetter, R., "Efficient operation of wireless packets networks using network coding," *International Workshop on Convergent Technology*, 2005.

[34] Lun, D. S., Ratnakar, N., Medard, M., Koetter, R., Karger, D. R., Ho, T., and Ahmed, E., "Minimum-cost multicast over coded packet networks," *submitted to IEEE Trans. on Information Theory*, 2006.

[35] Mackay, D. J. C., "Good error-correcting codes based on very sparse matrices," *IEEE Trans. on Information Theory*, vol. 45, pp. 399–431, Mar. 1999.

[36] Mandelbaum, D., "Unequal-error-protection codes derived from difference sets," *IEEE Trans. on Information Theory*, vol. IT-18, pp. 686–687, Sept. 1972.

[37] Marathe, M. V., Breu, H., Hunt-III, H. B., Ravi, S. S., and Rosenkrantz, D. J., "Simple heuristics for unit disk graphs," *Networks*, pp. 59–68, 1995.

[38] Masnick, B. and Wolf, J., "On linear unequal error protection codes," *IEEE Trans. on Information Theory*, vol. IT-3, pp. 600–607, Oct. 1967.

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

[40] Measson, C., Montanari, A., and Urbanke, R., "Maxwell's construction: the hidden bridge between maximum-likelihood and iterative decoding," *in Proc. IEEE International Symposium on Information Theory*, p. 225, June-July 2004.

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

[42] Ni, S.-Y., Tseng, Y.-C., Chen, Y.-S., and Sheu, J.-P., "The broadcast storm problem in a mobile ad hoc network," *in Proc. 5th Annual ACM/IEEE Int. Conf. on Mobile Computing and Networking*, pp. 151–162, 1999.

[43] Oswald, P. and Shokrollahi, A., "Capacity-achieving sequences for the erasure channel," *IEEE Trans. on Information Theory*, vol. 48, pp. 3017–3028, Dec. 2002.

[44] Pakzad, P., Fragouli, C., and Shokrollahi, A., "Coding schemes for line networks," *in Proc. IEEE International Symposium on Information Theory*, Sept. 2005.

[45] Park, S.-J., Vedantham, R., Sivakumar, R., and Akyildiz, I. F., "A scalable approach for reliable downstream data delivery in wireless sensor networks," *in Proc. 5th ACM international symposium on Mobile ad hoc networking and computing*, May 2004.

[46] Penrose, M., *Random Geometric Graphs*. Oxford University Press, 2003.

[47] Pishro-Nik, H., Chan, K., and Fekri, F., "On connectivity properties of large-scale wireless sensor networks," *First Annual IEEE International Conference on Sensor and Ad Hoc Communications and Networks*, pp. 498–507, Oct. 2004.

[48] Pishro-Nik, H. and Fekri, F., "On decoding of low-density parity-check codes over the binary erasure channel," *IEEE Tran. on Information Theory*, vol. 50, pp. 439–454, Mar. 2004.

[49] Pishro-Nik, H., Rahnavard, N., and Fekri, F., "Nonuniform error correction using low-density parity-check codes," *IEEE Trans. on Information Theory*, vol. 51, pp. 2702–2714, July 2005.

[50] Poulliat, C., Declercq, D., and Fijalkow, I., "Optimization of LDPC codes for UEP property," *IEEE International Symposium on Information Theory*, p. 450, June-July 2004.

[51] Qayyum, A., Viennot, L., and Laouiti, A., "Multipoint relaying for flooding broadcast message in mobile wireless networks," *in Proc. 35th Annual Hawaii Int. Conf. on System Science*, 2002.

[52] Rahnavard, N. and Fekri, F., "Unequal error protection using low-density parity-check codes," *in Proc. IEEE International Symposium on Information Theory, Chicago, IL*, p. 449, June-July 2004.

[53] Rahnavard, N. and Fekri, F., "Bounds on maximum-likelihood decoding of finite-length rateless codes," *in Proc. 39th Annual Conference on Information Sciences and Systems, Baltimore, MD*, Mar. 2005.

[54] Rahnavard, N. and Fekri, F., "Finite-length unequal error protection rateless codes: Design and analysis," *in Proc. IEEE GLOBECOM, St. Louis, MO*, pp. 1353–1357, Nov.-Dec. 2005.

[55] Rahnavard, N. and Fekri, F., "CRBcast: A collaborative rateless scheme for reliable and energy-efficient broadcasting in wireless sensor networks," *in Proc. 5th ACM/IEEE International Conference on Information Processing in Sensor Networks, Nashville, TN*, pp. 276–283, Apr. 2006.

[56] Rahnavard, N. and Fekri, F., "Generalization of rateless codes for unequal error protection and recovery time: Asymptotic analysis," *in Proc. IEEE International Symposium on Information Theory, Seattle, WA*, pp. 523–527, July 2006.

[57] Rahnavard, N. and Fekri, F., "New results on unequal error protection using LDPC codes," *IEEE Communications Letters*, vol. 10, pp. 43–45, Jan. 2006.

[58] Rahnavard, N., Pishro-Nik, H., and Fekri, F., "Unequal error protection using partially regular LDPC codes," *IEEE Trans. on Communications*, vol. 55, pp. 387–391, Mar. 2007.

[59] Rahnavard, N., Vellambi, B. N., and Fekri, F., "Rateless codes with unequal error protection property," *IEEE Trans. on Information Theory*, vol. 53, pp. 1521–1532, Apr. 2007.

[60] Rahnavard, N., Vellambi, B. N., and Fekri, F., "Collaborative rateless broadcast (CRBcast) in wireless sensor networks," *to be submitted to IEEE Trans. on Wireless Communications*, 2007.

[61] Rahnavard, N., Vellambi, B. N., and Fekri, F., "Distributed protocols for finding low-cost broadcast and multicast trees in wireless networks," *to be submitted*, 2007.

[62] Rahnavard, N., Vellambi, B. N., and Fekri, F., "Efficient broadcasting via rateless coding in multihop wireless networks with local information," *ACM International Wireless Communications and Mobile Computing Conference*, Aug. 2007.

[63] Rahnavard, N., Vellambi, B. N., and Fekri, F., "FTS: A fractional transmission scheme for efficient broadcasting via rateless coding in multihop wireless networks," *MILCOM 2007*, Oct. 2007.

[64] Richardson, T., Shokrollahi, A., and Urbanke, R., "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. on Information Theory*, vol. 47, pp. 619–637, Feb. 2001.

[65] Sasson, Y., Cavin, D., and Schiper, A., "Probabilistic broadcast for flooding in wireless mobile ad hoc networks," *Technical Report, Swiss Federal Institute of Technology*, 2002.

[66] Shannon, C., "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, July, October 1948.

[67] Shokrollahi, A., "LDPC codes: An introduction," 2002. `http://algo.epfl.ch/contents/output/pubs/ldpc-intro.pdf`.

[68] Shokrollahi, A., "Raptor codes," *IEEE Trans. on Information Theory*, vol. 52, pp. 2551–2567, June 2006.

[69] Sikora, T., "MPEG digital video coding standards," *IEEE Signal Processeing Mag.*, vol. 14, pp. 82–100, Sept. 1997.

[70] Stojmenovic, I., Seddigh, M., and Zunic, J., "Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks," *IEEE Trans. on Parallel and distributed systems*, vol. 13, pp. 14–25, Jan. 2002.

[71] Tanner, R. M., "A recursive approach to low complexity codes," *IEEE Trans. on Information Theory*, vol. 27, pp. 533–547, Sept. 1981.

[72] Tobagi, F. A. and Kleinrock, L., "Packet switching in radio channels: Part ii - the hidden terminal problem in carrier sense multiple-access modes and the busy-tone solution," *IEEE Trans. on Communication*, vol. 23, no. 12, pp. 1417–1433, 1975.

[73] Vasic, B., Cvetkovic, A., Sankaranarayanan, S., and Marcellin, M., "Adaptive error protection low-density parity-check codes for joint source-channel coding schemes," *IEEE International Symposium on Information Theory*, p. 267, Jun-July 2003.

[74] Wieselthier, J. E., Nguyen, G. D., and Ephremides, A., "Energy-efficient broadcast and multicast trees in wireless networks," *Mobile Networks and Applications*, vol. 7, pp. 481–492, 2002.

[75] Xu, L., "Resource-efficient delivery of on-demand streaming data using UEP codes," *IEEE Trans. on Communications*, vol. 51, pp. 63–71, Jan. 2003.

[76] Yang, M., Li, Y., and Ryan, W., "Design of efficiently encodable moderate-length high-rate irregular LDPC codes," *IEEE Transactions on Communications*, vol. 52, pp. 564–571, April 2004.

VITA

Nazanin Rahnavard received her B.S. and M.S. degrees in electrical engineering from Sharif University of Technology, Tehran, Iran, in 1999 and 2001, respectively. She then joined the Georgia Institute of Technology, Atlanta, GA, in 2002, where she received her Ph.D. degree in the School of Electrical and Computer Engineering in 2007. She will join the School of Electrical and Computer Engineering at Oklahoma State University as an assistant professor in Fall 2007. Her research interests lie in the area of telecommunications with a focus on error-control coding and wireless ad-hoc and sensor networks. Nazanin received the outstanding research award from the Center for Signal and Image Processing at the Georgia Institute of Technology in Spring 2007.