Network Coding for Delay Challenged Environments

by

Daniel Enrique Lucani

B.S., E.E., (*Summa cum Laude*) Universidad Simón Bolívar (2005)
M.S., E.E., (Honors) Universidad Simón Bolívar (2006)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2010

Author .....
Department of Electrical Engineering and Computer Science
April 5, 2010

Certified by .....
Muriel Médard
Professor
Thesis Supervisor

Certified by .....
Milica Stojanovic
Associate Professor, Northeastern University
Thesis Supervisor

Accepted by .....
Terry P. Orlando
Chairman, Department Committee on Graduate Students

Network Coding for Delay Challenged Environments

by

Daniel Enrique Lucani

Submitted to the Department of Electrical Engineering and Computer Science
on April 5, 2010, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Electrical Engineering

Abstract

Delay is a fundamental problem of data communication and networks, a problem that is not usually addressed in classical coding, information or networking theory. We focus on the general problem of delay challenged networks. This delay challenge may be related to different reasons, for example, 1) large latency, which can affect the performance of the system in delay, throughput or energy efficiency, 2) half-duplex constraints on the nodes, which precludes a node to receive and transmit at the same time, and/or 3) application-level requirements for reliable, fast and efficient dissemination of information.

We consider three main problems of study and the role of network coding on solving these problems. The first is related to the problem of reliable communication in time-division duplexing channels, also known as half-duplex channels, in the presence of large latency. In large latency channels, feedback about received packets may lag considerably the transmission of the original packets, limiting the feedback's usefulness. Moreover, the time duplex constraints may entail that receiving feedback may be costly. In this work, we consider tailoring feedback and (network) coding jointly in such settings to reduce the mean delay for successful in order reception of packets. We find that, in certain applications, judicious choices provide results that are close to those that would be obtained with a full-duplex system.

The second part of this thesis studies the problem of data dissemination in arbitrary networks. In particular, we study the problem of minimizing the delay incurred in disseminating a finite number of data packets. We show that the optimal solution to the problem can be thought of as a scheduling problem, which is hard to solve. Thus, we consider the use of a greedy linear network coding algorithm that only takes into account the current state of the system to make a decision. The proposed algorithm tries to maximize the impact on the network at each slot, i.e., maximize the number of nodes that will benefit from the coded packet sent by each active transmitter. We show that our scheme is considerably better, in terms of the number of slots to complete transmission, than schemes that choose the node with more information as the transmitter

The third part of this work studies the case of underwater acoustic networks as an

example of delay challenged networks. We consider the use of network coding under two different lights. First, as a means to obtain a lower bound on the transmission power of multicast connections in underwater networks. Second, to develop practical schemes useful in such networks. Finally, we study upper bounds on the transport capacity of underwater acoustic networks under unicast connections. We show that the amount of information that can be exchanged by each source-destination pair in underwater acoustic networks goes to zero as the number of nodes n goes to infinity. This occurs at least at a rate $n^{-1/\alpha}e^{-W_0(O(n^{-1/\alpha}))}$, where W_0 represents the branch zero of the Lambert W function, and a path loss exponent of α. Note that typical values of the path loss exponent are $\alpha \in [1, 2]$ for underwater acoustic networks. This is significantly different to the $\alpha \geq 2$ of radio wireless applications.

Thesis Supervisor: Muriel Médard
Title: Professor

Thesis Supervisor: Milica Stojanovic
Title: Associate Professor, Northeastern University

Preface

Wolfgang Amadeus Mozart once said that "to talk well and eloquently is a very great art, but an equally great one is to know the right moment to stop." Although it is a wise prescription for good conversation, this phrase came to have new meaning to me during my doctoral studies. To know when to stop transmitting and start to listen is quite relevant in communications systems and networks, for example, when nodes cannot talk and listen at the same time. Without discussing the eloquence of random coding, the first part of this thesis takes steps into formalizing a means to knowing how much to talk before stopping. Of course, this is not the only focus of this work, but it holds a special place in my heart.

The problem of delay challenged networks is fascinating and it is one that warrants further study. It is my hope that this work has provided meaningful insights and contributions to this topic. Of course, I cannot claim sole credit for this work. Many others have been involved and have made it a challenging but, most of all, enjoyable experience.

I would like to thank my adviser, Professor Muriel Médard, for her support and for providing invaluable insight and experience to our discussions, not only on academic issues but also on professional and personal ones. I would like to thank my co-adviser, Professor Milica Stojanovic, the "great approximator", for her constant search and grasp of solutions that are meaningful in theory and practice. I would also like to thank for her excellent spirit and good humor. Finally, I thank Frank H. P. Fitzek and David Karger for our discussions.

On a personal note, I would like to thank my wife Bianca, for her patience, encouragement, trust, and love. Life has not been the same since we met. I would also like to thank my friends MinJi, Shirley, Fang, Ali, Jay-Kumar and Soheil for their good spirit and for heeding Miguel de Cervantes' advice to "not see life as it is, but as it should be." It has been fun to work and share with you. Finally, I would like to thank my family for their unconditional support on any and all my endeavours.

This work was supported in part by the National Science Foundation (NSF) under

grants No. 0520075, 0831728 and CNS-0627021, by ONR MURI Grant # N00014-07-1-0738, and subcontract # 060786 issued by BAE Systems National Security Solutions, Inc. and supported by the Defense Advanced Research Projects Agency (DARPA) and the Space and Naval Warfare System Center (SPAWARSYSCEN), San Diego under Contract # N66001-06-C-2020 (CBMANET), subcontract # 18870740-37362-C issued by Stanford University and supported by the DARPA.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Delay is a fundamental problem of data communication and networks, a problem that is not usually addressed in classical coding, information or networking theory. Typically, the design of codes and networks has been grounded on the search of improving throughput in order to come closer to the capacity of the system. However, an improvement in throughput does not necessarily translate into improving delay performance: one can always have good throughput by delivering one truck with a large memory storage once a year. In fact, the random codebook design used by Shannon to determine the achievability result for channel capacity assumes that arbitrarily large codewords. Since decoding the information is not possible until the entire codeword is received, the delay to recover the information is also arbitrarily large.

A change of paradigm is then warranted for some applications. For instance, if an application requires a prompt delivery of a message to some or all the nodes in the network, the design objective or criteria should be to reduce transmission or dissemination delay. If this happens in unreliable transmission channels, e.g., channels in which data packets can be lost, coding should be aimed to minimize delay rather than using codes designed for improving throughput and hoping for a good delay performance.

We are particularly interested in studying environments that are delay challenged. The term delay challenged is not meant only to characterize networks in which delay

is the main design criteria, for instance, when the application requires a relatively low delay. We also use it to characterize networks in which the transmission channel imposes an important propagation delay, which can affect the performance of the system in delay, throughput or energy efficiency.

This work focuses in the general problem of data transmission in networks that are delay challenged. In particular, we are interested in studying the role of network coding in these scenarios from both a practical perspective, e.g., to propose schemes that combine coding and feedback to provide reliable communications, and as a means to derive fundamental limits in special cases, e.g., using network coding techniques to derive a lower bound on the transmission power of underwater acoustic networks. Although our study of network coding in delay challenged environments is developed under assumptions that are valid for a wide variety of channels, ranging from satellite to terrestrial radio to underwater acoustic channels, we analize in greater detail the case of underwater acoustic networks as an important example of delay challenged networks.

We begin by presenting previous results on network coding and underwater acoustic networks to motivate our work and contributions to the field in Sections 1.1 and 1.2, respectively. Section 1.3 summarizes the different scenarios of delay challenged networks studied in this work and our contributions.

1.1 Previous Results in Network Coding

Network coding was introduced by Ahlswede *et al* [1]. Network coding considers the nodes to have a set of functions that operate upon received or generated data packets [24]. Today's networks would represent a subset of the coded packet networks, in which each node has two main functions: forwarding and replicating a packet. A classical network's task is to transport packets provided by the source nodes unmodified. In contrast, network coding considers information as an algebraic entity, on which one can operate.

Network coding research originally studied throughput performance without delay

considerations for the transmitted information. The seminal work by Ahlswede *et al* [1] considers a channel with no erasures and, therefore, no need for feedback. Work in [2] and [3] showed that linear codes over a network are sufficient to implement any feasible multicast connection, again considering a channel with no erasures. Also, [3] provides an algebraic framework for studying this subset of coded networks. In both of these cases, the nodes are considered to transmit a linear combination of the packets previously received. Work in [4] presents the idea of using linear codes generated randomly in a network.

For networks with packet erasures, two approaches have been used. The first approach relies on rateless codes, i.e., transmitting coded data packets until the receiver sends an acknowledgement stating that all data packets have been decoded successfully. Reference [5] studies random linear network coding in lossy networks showing that it can achieve packet-level capacity for both single unicast and single multicast connections in wireline and wireless networks. Reference [6] presents network codes that preserve the communication efficiency of a random linear code, while achieving better computational efficiency. Reference [7] presented a random linear coding scheme for packet streams considering nodes with a fixed, finite memory, establishing a trade-off between memory usage and achievable rate. In terms of practical issues and implementation, work in [54] presents MORE, a MAC-independent opportunistic protocol for wireless networks, and provides experimental results with some emphasis on the throughput gains provided by network coding.

The work in [10] and [11] has studied delay performance gains and their scaling laws for network coding with and without channel side information, respectively. They focus on transmission of large files in a rateless fashion. In [12] the delay performance of network coding for a tree-based multicast problem is studied and compared to various Automatic Repeat reQuest (ARQ) and Forward Error Correcting (FEC) techniques. For network coding, it assumes reliable and instantaneous feedback to acknowledge a correct decoding of all data packets. Note that the focus of these references has been on either throughput or delay performance, usually considering minimal feedback.

Finally, the work in [13] couples the benefit of network coding and ARQ by acknowledging degrees of freedom (dofs), defined as linearly independent combinations of the data packets, instead of original data packets to show that queue size in a node follows degrees of freedom.

The second approach uses block transmissions. Reference [14] studies the problem in wireless networks and shows that linear codes achieve capacity in the network. In [15] a queueing model for random linear coding is presented, which codes data packets in a block-by-block fashion using acknowledgements to indicate successful transmission of each block. Interestingly, the coding block size depends on the number of packets available in the queue, up to some maximum block size.

The general problem of data transmission using network coding under half-duplex constraints on the nodes has not been considered for network coding before our work. In particular, we are interested in the problem of minimizing the time to completely transmit a block of data packet to one or several nodes in the network. The key questions that we are interested in answering are 1) how much should a node transmit before it stops to listen for an acknowledgement or other node's transmission, and 2) which nodes should transmit in order to reduce the time to complete the transmission of a block of data.

1.2 Previous Results in Underwater Acoustic Networks

With recent advances in acoustic communication technology, the interest in study and experimental deployment of underwater networks has been growing [46]. References [51] and [52] present studies of the characteristics and design challenges of underwater acoustic networks.

Underwater acoustic channels impose many constraints that affect the design of wireless networks. They are characterized by a path loss that depends on both transmission distance and signal frequency, in a far more pronounced way than a terrestrial

radio system. Thus, not only the transmission power, but also the useful bandwidth depend strongly on transmission distance [47]. Specifically, signals transmitted over a distance l are subject to a power loss of $l^{-\alpha}a(f)^{-l}$. Although a terrestrial radio channel can be modeled similarly, the underwater acoustic channel has important differences. The spreading factor α, related to the geometry of propagation, has values in the range $1 \leq \alpha \leq 2$, where $\alpha = 1$ corresponds to cylindrical spreading and $\alpha = 2$ to spherical spreading. Also, the absorption coefficient $a(f)$ is a rapidly increasing function of frequency, e.g., it is three orders of magnitude greater at 100 kHz than at a few Hz [47]. Finally, the background noise is not white, but has a power spectral density that is highly dependent on frequency.

In general, studies on underwater acoustic networks have focused on practical schemes. Over the years, a series of routing schemes have been proposed for underwater networks. References [51], [52] and [53] present surveys of different routing schemes used in underwater networks. In [48] two distributed routing algorithms are introduced for delay-insensitive and delay-sensitive applications. Reference [49] presents a modification of the dynamic source routing protocol that adds location awareness and link quality metrics. In [50] a routing protocol based on local depth of the nodes is studied.

However, the study of fundamental limits in underwater networks has been limited. For example, there was no lower bound for the transmission power prior to our work that could be used as a comparison point for different practical network layer schemes.

Another interesting fundamental limit that had not been analyzed is the problem of capacity scaling laws for underwater networks. The seminal work by Gupta and Kumar [66] studied wireless networks, modeled as a set of n nodes that exchange information, with the aim of determining what amount of information the source nodes can send to the destination as the number n grows. The original results obtained for nodes deployed in a disk of unit area motivated the study of capacity scaling laws in different scenarios, ranging from achievability results in random deployments using percolation theory [67] or cooperation between nodes [68], to the impact of node mobility on the capacity of the network, e.g., [69]. Reference [70] provides a good

overview of the different assumptions and scaling laws for radio wireless networks.

However, existing capacity scaling laws for wireless radio networks correspond to scenarios for which $a(f) = 1$, or a constant greater than one, and $\alpha \geq 2$, e.g., [66], [67]. These results cannot be directly applied to underwater acoustic networks in which the attenuation varies over the system bandwidth and $\alpha \leq 2$.

1.3 Thesis Contributions

This thesis studies several cases of networks with delay challenges. However, we have focused on three main areas of study. Hereafter, we provide an overview of the challenges and results in these areas.

1.3.1 Network Coding for Large Latency Time Division Duplexing Channels

We have studied channels in which time division duplexing is necessary, i.e., when a node can only transmit or receive, but not both at the same time. This problem has not been considered in any of the previous network coding references or, to the best of our knowledge, for network coding before our work. This type of channel is usually called half-duplex in the literature, but we use the term time division duplexing (TDD) to emphasize that the transmitter and receiver do not use the channel half of the time each or in any pre-determined fashion. As mentioned before, important examples of time division duplexing channels are infrared devices (IrDA), which have motivated many TDD ARQ schemes [17] [18], and underwater acoustic communications [19]. Other important applications are found in channels with very high latency, e.g., in satellite [20][21], and deep space [22] communications.

Originally, we focused on the problem of transmitting M data packets through a link using random linear network coding (Section 2.1) and we have extended these results to more complex network scenarios, namely, 1) a node broadcasting information to several receivers (Section 2.2), and 2) an all-to-all broadcast network, where

every node has some information to be transmitted to all other nodes in the network, all nodes are within range of each other but possibly with different probabilities of packet loss (Section 2.3).

Let us explain the simple link case, which provides the insight that was used in more complex network scenarios. For a link, we considered that the sender can transmit random linear coded packets back-to-back before stopping to wait for an acknowledgement (ACK) packet. This ACK packet conveys the remaining number of degrees of freedon (dofs) required at the receiver to decode all M data packets. Each dofs is defined as an independent linear combination of the original packets. We consider that the number of coded packets N_i to be transmitted before waiting for a new ACK packet depends on the number of dofs i needed at the receiver, as indicated by the last ACK packet received successfully. If it is the first transmission, we consider that the required dofs is M. The system transmits N_i coded packets (CP), and waits to receive an ACK packet that updates the value of i to j, at which point it will transmit N_j coded packets. The system will keep transmitting and stopping to update i, until $i = 0$. When $i = 0$, the transmitter can start with M new data packets, or simply stop.

There is a natural trade off in the choice of the N_i's. Every time the system stops to wait for an ACK, it incurs in an additional delay, which can be large in high latency channels. In general, the system requires at least one stop to get confirmation of complete transmission. However, we want to minimize the number of stops required to complete transmission of the M packets. Note that if the N_i's are too small given the channel conditions, the system will have to transmit more ACK packets to complete transmission of the block of M data packets, which will cause a larger delay. On the other hand, if the N_i's are too large, the receiver will have decoded the M packets, for example, before the transmitter stops sending the first N_M coded packets. Since the block of M original packets is considered to be completely transmitted when the ACK requests no more dofs, the system causes unnecessary delay by transmitting too many coded packets by delaying transmission of the ACK by the receiver. In other words, the transmitter could have sent a smaller number of coded packets before stopping

and still transmit the M packets successfully.

We showed that there exists an optimal number of coded data packets to be transmitted back-to-back before stopping to wait for an ACK packet from the receiver, in terms of mean completion time, i.e., mean time to decode the M original data packets at the receiver and get an ACK at the transmitter. In fact, the optimal number of coded data packets N_i depends on the number of dofs i that the receiver requires to decode the information, and also on the packet erasure probability (Pe) and the latency, i.e., the number of packets in flight. Thus, we showed that there is an optimal time for stop transmitting coded packets and start listening to an ACK packet from the receiver.

Our objective was then to minimize the expected time to complete transmission of a block, i.e., the delay in block transmissions, using feedback. This delay to decode a block is different from the usual packet delay measure. Since coding is carried out on blocks of packets, the delay to decode a block successfully determines the delay of each of the packets in that block. We also showed that minimizing the expected time to complete transmission of a block of M packets with a fixed packet size also maximizes the throughput performance. However, we show that a correct choice of M and number of bits in the data packet can further improve throughput performance.

Although both standard ARQ techniques and our scheme achieve reliability by detecting errors in received packets or packet erasures, and recover the information using a retransmission scheme, there are some important differences. First, we rely on transmission of coded packets, i.e., there is no need to specify a particular data packet to retransmit as in ARQ, but only a random linear combination. The ACK packet of our scheme thus differs from common ARQ techniques [23] in that it does not give acknowledgement to particular data packets [23], but to degrees of freedom needed at the receiver to decode the M original packets. Second, the number of coded packets transmitted in our scheme is not fixed by design of the algorithm, but chosen given channel characteristics and information in the ACK packet. In fact, the information in the ACK packet of our algorithm can be used to update an estimate of the probability of packet error and improve the overall performance.

As mentioned before, we extend this idea to more complex scenarios, such as, broadcast from one node to all other nodes in a network, and all-to-all broadcast where every node has some information to transmit to all other nodes in the network. We also study the energy characteristics of our schemes, the effect of the choice of field size for the Galois Field operations used in the (network) coding, and queueing analysis and characteristics under a model of Poisson arrivals. We also study the use of systematic network coding, which consists in transmitting the original M packets first followed by random linear coded packets for the rest of the communication, as a means to reduce average computation requirements, use a small field size for operations (e.g., only use XORs), and maintain completion time performance.

1.3.2 Network Coding For Data Dissemination in Arbitrary Time Division Duplexing Networks

The problem of data dissemination has been widely studied for routing scenarios, focusing on theoretical analysis, e.g., [26], and protocol design, e.g., [32]. More recently, Reference [27] studied the effect of using network coding showing significant improvement over routing in terms of completion time. Reference [28] provides a wireless medium access control combined with network coding for multi-hop content distribution. The authors focus on a protocol that uses a content-directed medium access control (MAC), through which transmission priority is given to those nodes based on the rank of the coefficient matrix associated with the coded content the node holds, i.e., nodes with more information are given higher priority.

In Chapter 3, we advocate for the combination of network coding and medium access strategies, similar to the idea in Reference [28]. However, we show that giving priority to the nodes with the most information in the network is not necessarily going to promote a faster dissemination of the data. We determine some key ideas and insights to help in the development of ad-hoc protocols that combine network coding and MAC considerations.

In particular, we focus on the problem of minimizing the completion time to dis-

seminate a finite number of data packets in arbitrary networks assuming a time slotted system, and that nodes cannot transmit and receive information simultaneously (Half-Duplex constraint). The dissemination process is completed when all terminals can decode the original data packets. This problem can be stated as a scheduling problem which is hard to solve in general. We use a greedy linear network coding scheme to solving the problem, in which the nodes with the greatest impact on the network at each time slot should transmit, instead of choosing the node with the most information.

We show that our scheme is considerably better, in terms of the number of slots to complete transmission, than schemes that choose the node with more information as the transmitter at every time slot. We show that an order of magnitude reduction in the completion time is possible in networks with a small fixed number of nodes and a large number of packets to be disseminated (e.g., 1000 data packets to be disseminated to 20 nodes), or large networks that require dissemination of a small number of data packets (e.g., 20 data packets to be disseminated to 1000 nodes). The improvement is more dramatic if we increase the number of nodes in the network and the number packets to be disseminated.

1.3.3 Underwater Acoustic Networks: Fundamental Limits and Practical Schemes

This area of work focuses on the case of underwater acoustic networks. In order to better understand the problem and be able to derive relevant fundamental limits for it, we start by studying the exact relationship among power, transmission band, distance and capacity for the underwater acoustic channel under a Gaussian noise assumption. Since this complete model is a complicated one, we provide a closed-form approximate model for 1) transmission power and 2) optimal frequency band to use, as functions of distance and capacity. The model is obtained through numerical evaluation of analytical results that take into account physical models of acoustic propagation loss and ambient noise.

We show that the complete model that relates transmission power, transmission band, distance, and link capacity is convex. However, since we are interested in tractable models that could be used in network optimization problems, the present work shows the operating conditions under which the approximate model is convex.

We then assess the minimum transmission power required for an underwater acoustic network. A lower bound for transmission power is obtained by neglecting interference between the nodes and using subgraph selection [24] to establish minimum-cost multicast connections with network coding. The convex cost function for the network optimization is given by the transmission power which depends on the distance and a desired data rate via the approximate model for each active link. We show that the no-interference assumption in an underwater scenario is justified for low multicast rates, and randomly placed nodes with inter-node distances less than 10 km.

Finally, the network coding based lower bound for transmission power is used to compare different routing and network coding schemes.

We also study upper bounds on the transport capacity of underwater acoustic networks for three cases of interest: an arbitrarily chosen narrow transmission band; the case of power limited nodes which transmit in disjoint narrow bands; and the case of nodes with high power capabilities that use of a wide transmission band. The choice of transmission band in the last two cases depends on the transmission distance and the physical characteristics of the channel, and is made in accordance with the waterfilling principle.

We show that the amount of information that can be exchanged by each source-destination pair in underwater acoustic networks goes to zero as the number of nodes n goes to infinity. This occurs at least at a rate $n^{-1/\alpha}e^{-W_0(O(n^{-1/\alpha}))}$, where W_0 represents the branch zero of the Lambert function. We illustrate that this throughput per source-destination pair has two different regions. For small n, the throughput decreases very slowly as n increases. For large n, it decreases almost as $n^{-1/\alpha}$. Thus for large enough n, the throughput decreases more rapidly in underwater networks than in typical radio channels, because of the difference in the path loss exponent α. In wireless radio channels, $\alpha \geq 2$ while in underwater acoustic networks $\alpha \in [1, 2]$

typically. We study this in more detail in Chapter 5. This rule is valid for the different scenarios in general, requiring only changes in the scaling constants.

Chapter 2

Network Coding for Large Latency Time Division Duplexing Channels

This chapter studies networks that operate in time division duplexing channels, i.e., when a node can only transmit or receive, but not both at the same time. This problem has not been considered in any of the previous network coding references or, to the best of our knowledge, for network coding before our work. This type of channel is usually called half-duplex in the literature, but we use the term time division duplexing (TDD) to emphasize that the transmitter and receiver do not use the channel half of the time each or in any pre-determined fashion. Important examples of time division duplexing channels are infrared devices (IrDA), which have motivated many TDD ARQ schemes [17] [18], and underwater acoustic communications [19]. Other important applications may be found in channels with very high latency, e.g., in satellite [20][21], and deep space [22] communications.

The key question to ask for the problem of achieving reliable communication TDD channels, and the one that initiated this work, is quite simple and natural: how much should a node talk before stopping to listen to others? Answering this question is particularly relevant in scenarios in which latency, i.e., the number of bits/packets in flight, is large because the penalty for not transmitting (talking) the correct amount of time before stopping is very high. We consider that packets can be lost/erased, i.e., they are not successfully received. The simplest case of this problem is that of a

link. Let us provide some intuition about this case before mathematically analyzing the problem and extending it to more complex scenarios.

The case of a link corresponds to the problem of a node transmitting M data packets through a link using random linear network coding. We consider that the sender can transmit random linear coded packets back-to-back before stopping to wait for an acknowledgement (ACK) packet. This ACK packet conveys the remaining dofs required at the receiver to decode all M data packets. This will be a key feature of the different cases discussed in this chapter. We consider that the number of coded packets N_i to be transmitted before waiting for a new ACK packet depends on the number of dofs i needed at the receiver, as indicated by the last ACK packet received successfully. This number of coded packets is directly mapped to the time of talking before stopping to listen.

There is a natural trade off in the choice of the N_i's. Every time the system stops to wait for an ACK, it incurs in an additional delay, which can be large in large latency channels. In general, the system requires at least one stop to get confirmation of complete transmission, in order to ensure reliable transmission of the data. However, we want to minimize the time to complete a transmission, which is related to reducing the number of stops required to complete transmission of the M packets. Note that if the N_i's are too small given the channel conditions, the system will have to transmit more ACK packets to complete transmission of the block of M data packets, which will cause a larger delay. On the other hand, if the N_i's are too large, the receiver will have decoded the M packets, for example, before the transmitter stops sending the first N_M coded packets. Since the block of M original packets is considered to be completely transmitted when the ACK requests no more dofs, the system causes unnecessary delay by transmitting too many coded packets by delaying transmission of the ACK by the receiver. In other words, the transmitter could have sent a smaller number of coded packets before stopping and still transmit the M packets successfully.

We show that there exists an optimal number of coded data packets to be transmitted back-to-back before stopping to wait for an ACK packet from the receiver, in terms of mean completion time, i.e., mean time to decode the M original data packets

at the receiver and get an ACK at the transmitter. In fact, the optimal number of coded data packets N_i depends on the number of dofs i that the receiver requires to decode the information, and also on the packet erasure probability and the latency. Thus, we show that there is an optimal time for stop transmitting coded packets and start listening to an ACK packet from the receiver.

Thus, our objective is to find the N_i's that optimize a specific metric. In particular, we are interested in minimizing the mean time to complete transmission of a block of data packets, i.e., the delay in block transmissions, using feedback. This delay to decode a block is different from the usual packet delay measure. Since coding is carried out on blocks of packets, the delay to decode a block successfully determines the delay of each of the packets in that block. We also show that minimizing the expected time to complete transmission of a block of M packets with a fixed packet size also maximizes the throughput performance. However, we show that a correct choice of M and number of bits in the data packet can further improve throughput performance.

Although both standard ARQ techniques and our schemes achieve reliability by detecting errors in received packets or packet erasures, and recover the information using a retransmission scheme, there are some important differences. First, we rely on transmission of coded packets, i.e., there is no need to specify a particular data packet to retransmit as in ARQ, but only a random linear combination. The ACK packet of our scheme thus differs from common ARQ techniques [23] in that it does not give acknowledgement to particular data packets [23], but to degrees of freedom needed at the receiver to decode the M original packets. Second, the number of coded packets transmitted in our scheme is not fixed by design of the algorithm, but chosen given channel characteristics and information in the ACK packet. In fact, the information in the ACK packet of our algorithm can be used to update an estimate of the probability of packet error and improve the overall performance.

Note that this scheme is rateless in nature. However, the feedback mechanism is different to that of typical rateless schemes, which assume an independent feedback channel through which the transmitter can receive an ACK indicating full decoding

of the data. Our schemes assume that there is only one channel for both data and ACK. Other rateless codes, e.g., LT codes [40], Raptor Codes [41], will face a similar challenge under the TDD channel, namely how many coded packets to send before stopping to listen for an ACK. The advantage of random linear network coding is that we can extend these ideas to general networks with no modifications from the coding perspective. That is, our results can be extended to the case of a network, in which each node performs a random linear combination of packets received from different nodes. In this extension, each node transmitting through a link, or, more generally, a hyperarc (using the terminology in [24]) will have an optimal number of coded packets to transmit back-to-back before stopping to listen. Fountain codes, e.g., [40] [41], do not share this trait: intermediate nodes have to either relay packets with no modifications or completely decode the information before transmitting to other nodes in order to preserve the structure of the code.

We begin by studying the case of a link in Section 2.1 analyzing different aspects of this problem. We then proceed to extend our result to the case of one node broadcasting information to several receivers (one-to-all broadcast) in Section 2.2. Finally, Section 2.3 studies the case of a network in which every node has information to transmit to all other nodes (all-to-all broadcast).

2.1 Random Linear Network Coding for TDD channels: Link Case

Let us consider the simplest case of transmission of information: a node that transmits information to another node in a single hop.

2.1.1 Model

We consider a sender in a link wants to transmit M data packets at a given link data rate R. The channel is modeled as a packet erasure channel. Nodes can only transmit or receive, but not both at the same time. The sender uses random linear network

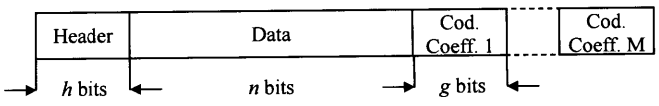

Figure 2-1: Structure of coded data packet

coding [4] to generate coded data packets. Each coded data packet contains a linear combination of the M data packets of n bits each, as well as the random encoding coefficients used in the linear combination. Each coefficient is represented by g bits. For encoding over a field size q, we have that $g = \log_2 q$ bits. Also consider an information header of size h. Thus, the total number of bits per packet is $h+n+gM$. Figure 2-1 shows the structure of each coded packet consider in our scheme.

The sender can transmit coded packets back-to-back before stopping to wait for the ACK packet. The ACK packet feeds back the number of degrees of freedom, that are still required to decode successfully the M data packets. Since random linear coding is used, there is some probability of choosing encoding vectors that are all zero for one coded packet or encoding vectors that are linearly dependent on vectors of previously received packets. Thus, using arguments similar to [10], the expected number of successfully received packets before having M linearly independent combinations, is

$$\sum_{k=1}^{M} \frac{1}{(1-(1/q)^k)} \leq M\frac{q}{q-1}. \tag{2.1}$$

In the following analysis, we assume that the field size q is large enough so that the expected number of successfully received packets at the receiver, in order to decode the original data packets, is approximately M. This is not a necessary assumption for our analysis. In fact, in the Practical Considerations chapter we show how to include the effect of the field size into our models. We prove that the performance degradation is very small, specially if M is moderately large. However, making the large field size assumption simplifies the expressions and provides a good approximation for large enough q. Also, all the techniques discussed in this chapter are directly applicable, with minor or no modifications, if we take into account the effect of field size.

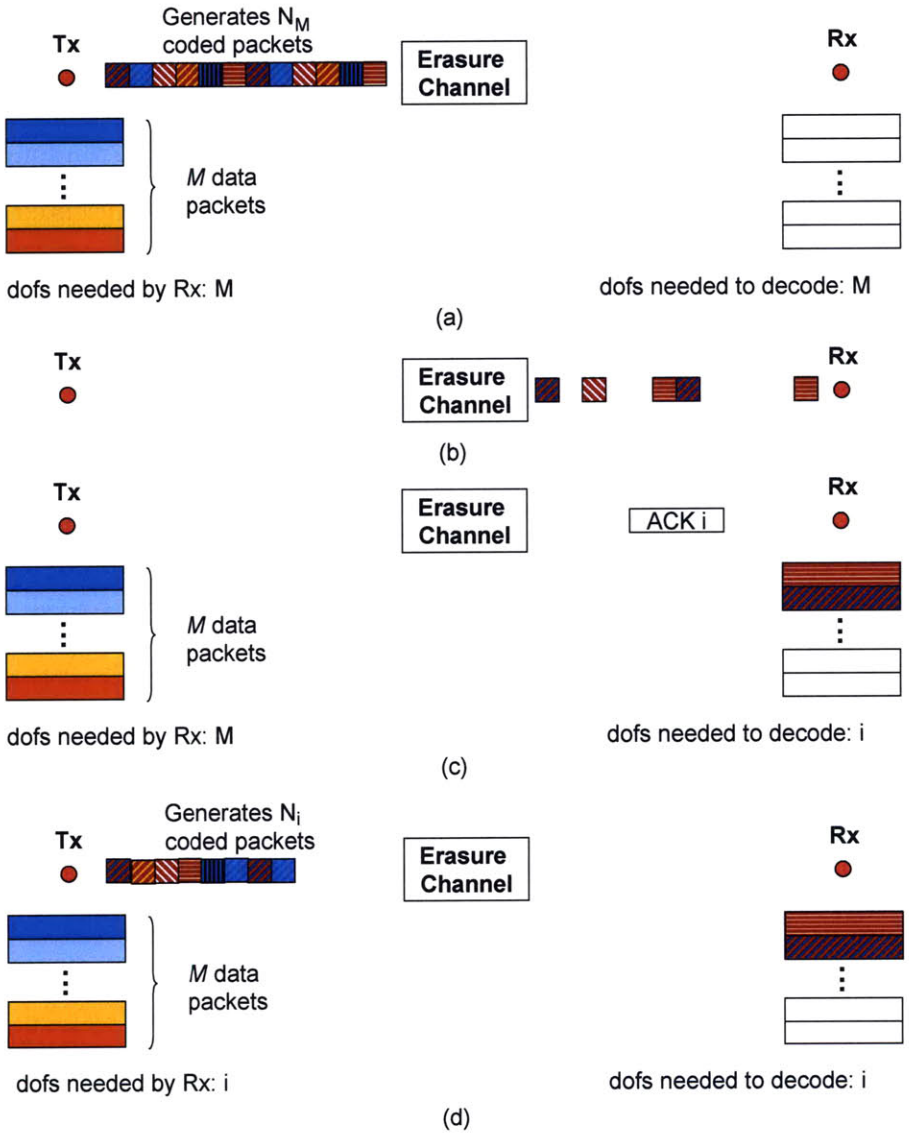

Figure 2-2: Network coding TDD scheme in a link. (a) The transmitter generates N_M coded packets and sends them to the receiver. (b) Some of the packets are lost when going through the channel. (c) The receiver sends an ACK packet indicating that i dofs are needed to decode. (d) Upon reception of the ACK packet, the transmitter generates N_i coded packets and sends them to the receiver if $i > 0$.

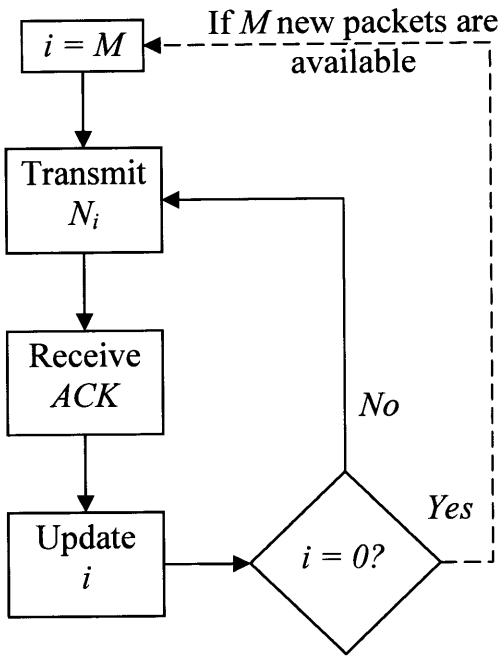

Figure 2-3: Algorithm of network coding for time division duplexing channels. i represents the remaining number of degrees of freedom to decode the packets, and N_i the corresponding number of coded packets transmitted before stopping to listen for a new ACK. The ACK packet has the information to update i

2.1.2 Description of Scheme

We are interested in determining the optimal number of coded packets that should be sent back-to-back before waiting for an ACK packet from the receiver in order to minimize the time for successfully transmitting the M data packets over the link. Note that if M packets are in the queue, at least M degrees of freedom have to be sent in the initial transmission, i.e., $N_M \geq M$ coded packets. We are interested not only in the number of dofs that are required at the first transmission, but also at subsequent stages.

Transmission begins with M information packets, which are encoded into N_M random linear coded packets and transmitted (Figure 2-2 (a)). As shown in Figure 2-2 (b), some of the coded packets are lost (suffer an erasure) when going through the erasure channel. After waiting for the N_M-th packet to be received, the receiver sends an ACK packet to the transmitter. At this point, the receiver has accumulated a certain number of dofs. The ACK informs the transmitter how many dofs are missing at

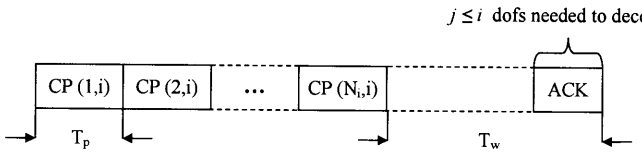

Figure 2-4: View in time of a round of transmission followed by an ACK packet in our network coding TDD scheme in a link, where $CP(k, d)$ represents the k-th coded packet transmitted when we start transmission with d dofs needed at the receiver to decode the information.

the receiver, say i, in order to decode the information (Figure 2-2 (c)). After the ACK packet is received, the transmitter updates its knowledge of the requirements of the receiver. If all M packets are decoded successfully ($i = 0$), the process is completed. Otherwise, the transmitter sends N_i coded packets (Figure 2-2 (d)) and stops to listen for an ACK, and so on, until all M packets have been decoded successfully. We are interested in the optimal number N_i of coded packets to be transmitted back-to-back in the next transmission to complete the remaining i dofs. Figure 2-3 shows the communication process as a system transmits N_M coded packets initially and awaits reception of an ACK packet that updates the value of i, at which point it will transmit N_i coded packets. The system will keep transmitting and stopping to update i, until $i = 0$. When $i = 0$, the transmitter can start with M new data packets or simply stop. In Figure 2-4 we show the process in time of a round of communication, starting by transmission of N_i coded packets and ending with the reception of an ACK packet. In Figure 2-4, $CP(k, d)$ represents the k-th coded packet transmitted when we start transmission with d dofs needed at the receiver to decode the information.

The process can be modelled as a Markov Chain (Figure 2-5). The states are defined as the number of dofs required at the receiver to decode successfully the M packets. Thus, these states range from M to 0. This is a Markov Chain with M transient states and one recurrent state (state 0). Let us define N_i as the number of coded packets that are sent when i dofs are required at the receiver in order to decode the information. Note that the time spent in each state depends on the state itself, because $N_i \neq N_j, \forall i \neq j$ in general.

The transition probabilities from state i to state j ($P_{i \to j}$) have the following ex-

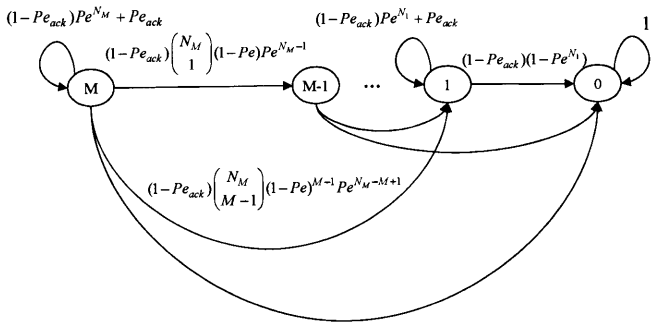

Figure 2-5: Markov chain representation of the scheme. State i represents that the receiver requires i more successfully received coded packets to decode the information

pression for $0 < j < i$ and $N_i \geq i$:

$$P_{i\to j} = (1 - Pe_{ack})\binom{N_i}{i-j}(1 - Pe)^{i-j}Pe^{N_i-i+j} \tag{2.2}$$

where Pe and Pe_{ack} represents the erasure probability of a coded packet and of an ACK packet, respectively.

More generally, the transition probability can be defined for any value of $N_i \geq 1$ as follows:

$$P_{i\to j} = (1 - Pe_{ack})f(i,j)(1 - Pe)^{i-j}Pe^{N_i-i+j} \tag{2.3}$$

where

$$f(i,j) = \begin{cases} \binom{N_i}{i-j} & \text{if } N_i \geq i, \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

For $j = i > 0$ the expression for the transition probability reduces to:

$$P_{i\to i} = (1 - Pe_{ack})Pe^{N_i} + Pe_{ack}. \tag{2.5}$$

Note that for $P(0 \to 0) = 1$. Finally, for $j = 0$ we have that

$$P(i \to 0) = 1 - \sum_{j'=1}^{i} P(i \to j'). \tag{2.6}$$

2.1.3 Moment Generating Function of Completion Time and Completion Energy

In this subsection we provide a full characterization to the problem of completion time and completion energy via a moment generating function. This result is general for finite state absorbing Markov chains, where the states are represented by positive integers ranging from 0 to M, and 0 being the absorbing state.

Let us define the moment generating function of the completion time when the Markov chain starts at state $n \in \{0, 1, ..., M\}$ as

$$M_{T,n}(s) = \sum_{t} \exp(st) P_T(T = t) \tag{2.7}$$

where $P_T(T = t)$ is the probability of the completion time T being t. Note that $M_{T,n}(s)$ is the moment generating function of the completion time when n data packets are taken by the source to be transmitted reliably to the receiver.

The following lemma states the expression for the moment generating function of the completion time.

Lemma 1. *The moment generating function $M_{T,n}(s)$ of the completion time when n linearly independent coded packets are needed to decode is given by*

$$M_{T,n}(s) = \frac{\exp(sT^n)}{1 - P_{n \to n} \exp(sT^n)} \sum_{i=0}^{n-1} P_{n \to i} M_{T,i}(s) \tag{2.8}$$

with $M_{T,0}(s) = 1$.

Proof. Using the Markov Chain structure of the problem, it can be shown that

$M_{T,n}(s)$ can be re-stated as

$$M_{T,n}(s) = \sum_{m_n \geq 1} \sum_{m_{n-1} \geq 0} \cdots \sum_{m_1 \geq 0} \exp\left(s \sum_{i=1}^{n} m_i T^i\right) C_n A_n \tag{2.9}$$

where T^i is the deterministic time required to send N_i coded packets and wait for an ACK when the Markov chain is in state i, e.g., $T^i = N_i T_p + T_w$, where T_p is the transmission time of a coded packet, and T_w is the waiting time to receive an ACK packet, as shown in Figure 2-4. The constant C_n captures the effect of returning to the same state repeatedly, while A_n captures the different paths that can be traversed without repetition of a state.

The expression for C_n is

$$C_n = \prod_{j=1}^{n} P_{j \to j}^{m_j - 1}.$$

The coefficient for A_n can be shown to obey a recursive expression of the form

$$A_n = \mathbf{1}_{\{m_n > 0\}} \left[\sum_{j=0}^{n-1} P_{n \to j} \left(\prod_{i=j+1}^{n-1} P_{i \to i} \mathbf{1}_{\{m_i = 0\}} \right) A_j \right]$$

with $A_1 = P_{1 \to 0} \mathbf{1}_{\{m_1 > 0\}}$. The indicator function $\mathbf{1}_{\{s \in S\}}$ is 1 when $s \in S$ and zero otherwise.

Substituting expression (2.10) into (2.9) we obtain the recursive equation for the moment generating function. □

Finally, note that the same structure is valid for computing the energy needed to complete transmission. To do so, one would substitute T^i by E^i, and $M_{T,n}(s)$ by $M_{E,n}(s)$, which leads to

$$M_{E,n}(s) = \frac{\exp(sE^n)}{1 - P_{n \to n} \exp(sE^n)} \sum_{i=0}^{n-1} P_{n \to i} M_{E,i}(s) \tag{2.10}$$

with $M_{E,0}(s) = 1$.

2.1.4 Mean Completion Time and Energy

The mean time for completing the transmission of the M data packets constitutes the mean time of absorption, i.e., the time to reach state 0 for the first time, given that the initial state is M. This can be expressed in terms of the expected time for completing the transmission given that the Markov chain is in state is i, T_i , $\forall i = 0, 1, ..M-1$. Let us denote the transmission time of a coded packet as T_p, and the waiting time to receive an ACK packet as T_w. For our scheme, $T_p = \frac{h+n+gM}{R}$ and $T_w = T_{rt} + T_{ack}$, where $T_{ack} = n_{ack}/R$, n_{ack} is the number of bits in the ACK packet, R is the link data rate, and T_{rt} is the round trip time. Note that $T_0 = 0$. Then, for $i > 1$:

$$T_i = \frac{N_i T_p + T_w}{(1 - Pe_{ack})(1 - Pe^{N_i})} + \frac{(1-Pe)^i Pe^{N_i - i} \sum_{j=1}^{i-1} f(i,j) \left(\frac{Pe}{1-Pe}\right)^j T_j}{1 - Pe^{N_i}}. \quad (2.11)$$

For example, for $i = 1$ we have that:

$$T_1 = \frac{(N_1 T_p + T_w)}{(1 - Pe_{ack})(1 - Pe^{N_1})}. \quad (2.12)$$

As it can be seen, the mean time for each state i depends on all the expected times for the previous states. Because of the Markov property, we can optimize the values of all N_i's in a recursive fashion, i.e., starting by N_1, then N_2 and so on, until N_M, in order to minimize the expected transmission time. We do so in the following subsection.

Now, our objective is to minimize the value of the mean completion time when M dofs are required in order to decode M data packets, T_M. Under the assumption that $N_i \geq i$, we have:

$$\min_{N_M,..,N_1} T_M = \min_{N_M,..,N_1} \frac{N_M T_p + T_w}{(1-Pe_{ack})(1-Pe^{N_M})} + \frac{(1-Pe)^M Pe^{N_M - M} \sum_{j=1}^{M-1} \binom{N_M}{M-j} \left(\frac{Pe}{1-Pe}\right)^j T_j}{1-Pe^{N_M}}$$

$$= \min_{N_M} \frac{N_M T_p + T_w}{(1-Pe_{ack})(1-Pe^{N_M})} + \frac{(1-Pe)^M Pe^{N_M - M} \sum_{j=1}^{M-1} \binom{N_M}{M-j} \left(\frac{Pe}{1-Pe}\right)^j \min_{N_j,..,N_1} T_j}{1-Pe^{N_M}}.$$

Without this assumption, we have that

$$\min_{N_M,..,N_1} T_M = \min_{N_M} \frac{N_M T_p + T_w}{(1-Pe_{ack})(1-Pe^{N_M})} + \frac{(1-Pe)^M Pe^{N_M - M} \sum_{j=1}^{M-1} f(M,j) \left(\frac{Pe}{1-Pe}\right)^j \min_{N_j,..,N_1} T_j}{1-Pe^{N_M}}.$$

Hence, regardless of the assumption on N_i, the problem of minimizing T_M in terms of the variables $N_M, .., N_1$ can be solved iteratively. First, we compute $\min_{N_1} T_1$, then use this results in the computation of $\min_{N_2,N_1} T_2$, and so on.

One approach to computing the optimal values of N_i is to ignore the constraint to integer values and take the derivative of T_i with respect to N_i and look for the value that sets it equal to zero. For our particular problem, this approach leads to solutions without a closed form, i.e., expressed as an implicit function. For $M = 1$, the optimal value of N_1 can be expressed using a known implicit function (Lambert W function), and it is given by

$$N_1^* = \frac{1 + W\left(-\exp\left(-1 + \frac{\ln(Pe)T_w}{T_p}\right)\right)}{\ln Pe} - \frac{T_w}{T_p} \tag{2.13}$$

where $W(\cdot)$ is the Lambert W function [25], defined as $W(z)e^{W(z)} = z$. The positive values are found for the branch W_{-1}, as denoted in reference [25].

The case of $M = 1$ can be thought as an optimized version of the uncoded Stop-and-Wait ARQ, which is similar to the idea presented in [20]. Instead of transmitting one packet and waiting for the ACK, our analysis suggests that there is an optimal number of back-to-back repetitions of the same data packet that should be transmitted before stopping to listen for an ACK packet.

Instead of using the previous approach, we perform a search for the optimal values $N_i, \forall i \in \{1, ...M\}$, using integer values. Thus, the optimal N_i's can be computed numerically for given Pe, Pe_{ack}, T_w and T_p. In particular, the search method for the optimal value can be made much simpler by exploiting the recursive characteristic of the problem, i.e., instead of making a M-dimensional search, we can perform M one-dimensional searches. Finally, these N_i's do not need to be computed in real time. They can be pre-computed for different channel conditions (e.g., Pe, T_{rt}) or

system settings (e.g., n, M, g, data rate), and stored in the receiver as look-up tables. This procedure makes the computational load on the nodes to be negligible at the time of determining the optimal number of coded packets in terms of the completion time, especially for dynamic environments.

2.1.5 Variance of Completion Time and Energy

Another figure of importance is the variance of the completion time and energy. We can use the moment generating function for our problem knowing that

$$Var_{T,n} = \frac{\partial^2 M_{T,n}(s)}{\partial s^2}\Big|_{s=0} - \left(\frac{\partial M_{T,n}(s)}{\partial s}\Big|_{s=0}\right)^2 \tag{2.14}$$

where $Var_{T,n}$ is the variance of T when $M = n$.

By taking derivatives, it is possible to prove that

$$\begin{aligned}\frac{\partial^2 M_{T,n}(s)}{\partial s^2}\Big|_{s=0} &= \frac{2T^n}{1-P_{n\to n}}\frac{\partial M_{T,n}(s)}{\partial s}\Big|_{s=0} - \frac{(T^n)^2}{1-P_{n\to n}} \\ &+ \frac{1}{1-P_{n\to n}}\sum_{i=1}^{n-1} P_{n\to i}\frac{\partial^2 M_{T,i}(s)}{\partial s^2}\Big|_{s=0}.\end{aligned} \tag{2.15}$$

Again, we can substitute the values of $T^i, \forall i$, and the values of the transition probabilities in order to compute the variance. As before, the same results apply for the case of energy making the appropriate substitutions of T^i by E^i, and $M_{T,i}$ by $M_{E,i}$.

2.1.6 Throughput

The mean throughput for a block scheme is strictly defined as

$$\text{Mean Throughput} = E[\frac{Mn}{T}] \tag{2.16}$$

where T is the time to complete transmission of M packets.

If we assume M and n to be constants, which is valid in our scheme, we have that

$$\text{Mean Throughput} = MnE[\frac{1}{T}]. \tag{2.17}$$

For the case of $M = 1$, i.e., the extended version of the Stop-and-Wait ARQ scheme we mentioned before, we can provide a simple expression for the mean throughput in terms of the transition probabilities $P_{1\to1}$ and $P_{1\to0}$,

$$E[\tfrac{1}{T}] = \tfrac{P_{1\to0}}{P_{1\to1}} \textstyle\sum_{k=1}^{\infty} \frac{{P_{1\to1}}^k}{kT^1} \tag{2.18}$$

$$= \tfrac{P_{1\to0}}{P_{1\to1}(T^1)} \textstyle\sum_{k=1}^{\infty} \frac{(1-P_{1\to0})^k}{k} \tag{2.19}$$

$$= -\tfrac{P_{1\to0}}{P_{1\to1}(T^1)} \ln(P_{1\to0}) \tag{2.20}$$

where $T^1 = N_1 T_p + T_w$, and we have used the Mercator series since $|1 - P_{1\to0}| < 1$ for all cases of interest.

For $M > 1$ this expressions are much more complicated. However, expression 2.17 implies that the problem of computing the mean throughput for our scheme is equivalent to that of computing negative moments of the completion time. The problem of computing negative integer moments has been studied previously in [33] and [34]. In particular, we focus in the result of [34] which states that

$$E[X^{-1}] = \int_0^{\infty} M_X(-s)ds \tag{2.21}$$

where $X > 0$ is the random variable, and $M_X(s)$ is the moment generating function of X.

Again, for $M = 1$ we can compute $E[T^{-1}]$ by using expression (2.21):

$$E[T^{-1}] = \int_0^{\infty} M_{T,1}(-s)ds \tag{2.22}$$

$$= \int_0^{\infty} P_{1\to0} \frac{\exp(sT^1)}{1 - P_{1\to1}\exp(sT^1)} ds \tag{2.23}$$

$$= \frac{P_{1\to0}}{T^1 P_{1\to1}} \int_{1-P_{1\to1}}^{1} \frac{du}{u} = \frac{P_{1\to0}}{P_{1\to1}T^1} \ln(\frac{1}{P_{1\to0}}) \tag{2.24}$$

where we have used the fact that $P_{1\to 0} = 1 - P_{1\to 1}$. This result is exactly the same as the one obtained by direct computation of $E[T^{-1}]$ using the Mercator series.

As stated before, the expressions for $E[T^{-1}]$ for the cases of $M > 1$ are too complicated for direct computation. However, it is possible to compute them if we use expression (2.21) and exploit the structure of the moment generating function of our problem (Expression (2.8)). For the case of $M = j$ we get

$$E[T^{-1}] = \int_0^\infty \frac{\exp(-sT^j)}{1 - P_{j\to j}\exp(-sT^j)} \sum_{i=0}^{j-1} P_{j\to i} M_{T,i}(-s) ds. \tag{2.25}$$

Notice that $M_{T,i}(-s), \forall i$ have a multiplying term $\frac{\exp(-sT^i)}{1-P_{i\to i}\exp(-sT^i)}$ which decreases to zero exponentially as $s \to \infty$ and goes to $\frac{1}{1-P_{i\to i}}$ as $s \to 0$. Thus, all terms inside the integral in (2.26) will go to zero exponentially.

Using this characteristic we can numerically compute $E[T^{-1}]$ using numerical integration techniques with the following approximation

$$E[T^{-1}] \approx \int_0^\tau \frac{\exp(-sT^j)}{1 - P_{j\to j}\exp(-sT^j)} \sum_{i=0}^{j-1} P_{j\to i} M_{T,i}(-s) ds. \tag{2.26}$$

where $\tau = \max_{\{i=1,\ldots,j\}} \tau_i$, $\tau_i = C/T^i$, and C is a constant in order to ensure $\exp(-\tau_i T^i)$ is small enough, e.g., $C = 5$ ensures $\exp(-\tau_i T^i) = \exp(-5) \approx 0.0067$.

Although this measure is important, we will define a different throughput measure called η because 1) the mean throughput is computationally demanding, and 2) most of the analysis of typical ARQ schemes is performed using η.

Let us define our measure of throughput η as the ratio between number of data bits transmitted (n) and the time it takes to transmit them. For the case of a block-by-block transmission

$$\eta = \frac{Mn}{T_M} \tag{2.27}$$

where T_M is the mean completion time defined previously.

Note that the mean throughput and η are not equal. For the case of $M = 1$, note

that $E[\frac{Mn}{T}] = \eta \frac{\ln(1/P_{1\to 0})}{P_{1\to 1}}$. More generally, using Jensen's inequality, $MnE[\frac{1}{T}] \geq \frac{Mn}{T_M}$ for $T > 0$. Therefore, η constitutes a lower bound to the mean throughput in our scheme.

Also, note that if M and n are fixed, η is maximized as T_M is minimized. Thus, by minimizing the mean time to complete transmission of a block of M data packets with n bits each, we are also maximizing η for those values. However, we show that the maximal η should be obtained using M and n as arguments in our optimization.

This is important for systems in which the data is streamed. In this case, searching for the optimal values of M and n, in terms of η, provides a way to optimally divide data into blocks of M packets with n bits each before starting communication using our scheme.

2.1.7 Optimal Packet size and packets per block

We have discussed throughput with a pre-determined choice of the number of data bits n and the number of data packets M in each block. However, expression (2.27) implies that the throughput η depends on both n and M. Hence, it is possible to choose these parameters so as to maximize the throughput. We can approach this problem is several ways. The first approach is to look for the optimal n while keeping M fixed:

$$\eta_{opt}(M) = \arg\max_{n} \left\{ \max_{N_M,\ldots,N_1} \eta \right\} \tag{2.28}$$

The second approach is to look for the optimal M while keeping n fixed:

$$\eta_{opt}(n) = \arg\max_{M} \left\{ \max_{N_M,\ldots,N_1} \eta \right\} \tag{2.29}$$

More generally, we could consider the case in which both parameters are variable and we are interested in maximizing η:

$$\eta_{opt} = \arg\max_{n,M} \left\{ \max_{N_M,\ldots,N_1} \eta \right\} \tag{2.30}$$

2.1.8 Performance Evaluation

1)Network coding for TDD optimized for mean completion time (TDD-T): This is our TDD scheme when we choose the N_is to optimize the mean completion time given channel characteristics and system parameters.

2)Network coding for TDD optimized for mean completion energy (TDD-E): This is our TDD scheme when we choose the N_is to optimize the mean completion energy given channel characteristics and system parameters.

3)Network coding for TDD with fixed maximum window: This scheme considers that a fixed number of packets M that have to be transmitted to the receiver, but with a fixed, pre-determined maximal number of coded packets to be transmitted before stopping to listen. We define this maximal value of coded packets as ω. If the number of degrees of freedom i required at the receiver to decode the information is $i \geq \omega$, the transmitter will transmit ω degrees of freedom. If $i < \omega$, the transmitter will transmit i degrees of freedom.

The model for the Markov Chain is derived from the previous case, by setting $N_i = \omega, \forall i \geq \omega$ and $N_i = i, \forall i < \omega$. For $i \geq \omega$, we have that:

$$T_i = \frac{\omega T_p + T_w}{(1 - Pe_{ack})(1 - Pe^{\omega})} + \frac{\sum_{j=1}^{\omega} \binom{\omega}{j} \left(Pe^{\omega - j}(1 - Pe)^j \right) T_{i-j}}{1 - Pe^{\omega}} \tag{2.31}$$

and for $i < \omega$:

$$T_i = \frac{iT_p + T_w}{(1 - Pe_{ack})(1 - Pe^{i})} + \frac{\sum_{j=1}^{i} \binom{i}{j} \left(Pe^{i - j}(1 - Pe)^j \right) T_{i-j}}{1 - Pe^{i}}. \tag{2.32}$$

4) Network coding in full duplex channel: This scheme assumes that nodes are capable of receiving and transmitting information simultaneously, and in that sense it is optimal in light of minimal delay. The sender transmits coded packets back-to-back until an ACK packet for correct decoding of all information (M information packets) has been received. This scheme can be modeled as a Markov chain where, as before, the states represent the number of dofs received. The time spent in each state is the same (T_p). Once the M packets have been decoded, i.e., M dofs have

been received, the receiver transmits ACK packets back-to-back, each of duration T_{ack}. One ACK should suffice, but this procedure minimizes the effect of a lost ACK packet.

The mean time to complete the transmission and get and ACK is:

$$E[T] = T_{rt} + \frac{MT_p}{1 - Pe} + \frac{T_{ack}}{1 - Pe_{ack}} \tag{2.33}$$

where T is the time to complete transmission of M packets.

The mean energy to complete the transmission and get and ACK is:

$$E[\text{Energy}] = \frac{T_{rt}E_p}{T_p} + \frac{T_{rt}E_{ack}}{2T_{ack}} + \frac{ME_p}{1-Pe} + \frac{E_{ack}}{1-Pe_{ack}} \tag{2.34}$$

5)Go-Back-N ARQ for TDD: This is an ARQ scheme developed for a TDD duplex channel studied extensively in [17]. Each transmission contains W data packets sent back-to-back, where W is the window size of our GBN scheme. Reference [17] studied this case and proposed the utilization factor for it. In our notation, the equivalent η is given by

$$\eta_{GBN} = \frac{n(1 - Pe)\left(1 - (1 - Pe)^W\right)}{(WT_p + T_w)Pe}. \tag{2.35}$$

6)Selective repeat ARQ for TDD: This is an ARQ scheme developed for a TDD duplex channel presented in [17]. Each transmission contains W data packets, where W is the window size of our SR scheme. Using the utilization factor studied in Reference [17], we provide the equivalent η for this problem:

$$\eta_{SR} = \frac{Wn(1 - Pe)}{WT_p + T_w}. \tag{2.36}$$

2.1.9 Numerical Examples

This section provides numerical examples that compare the performance of the different network coding schemes we have discussed so far in the case of a link in TDD

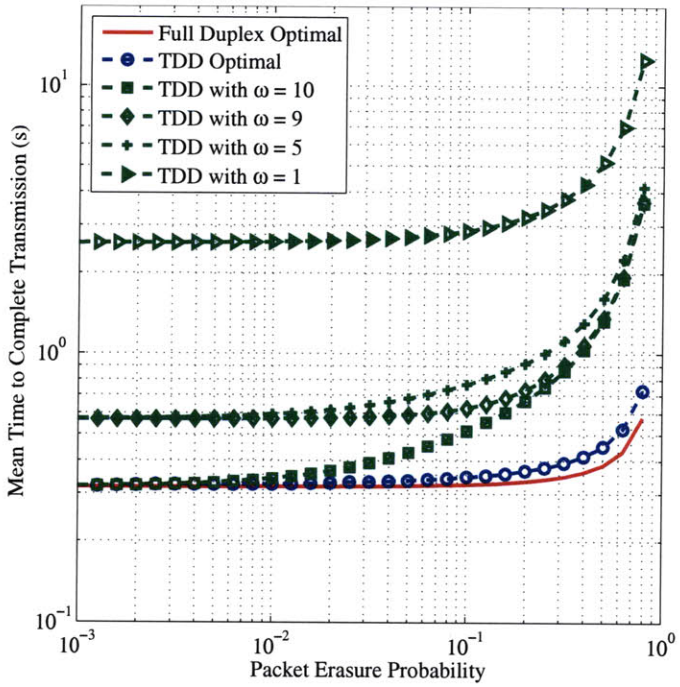

Figure 2-6: Mean time for transmitting M data packets successfully versus Pe in a satellite example. The parameters used are $M = 10$, $T_{rt} = 250$ ms, data rate 1.5 Mbps, $n_{ack} = 100$ bits, $n = 10000$ bits, $g = 100$ bits, $h = 80$ bits, $Pe_{ack} = 0.001$

channels. The comparison is carried out in terms of the mean time to complete a transmission of M data packets through TDD channel under different block error probabilities. We also present results in terms of the measure of throughput η to illustrate its dependence on the values of M and n for varying channel characteristics (erasure probabilities). We use the case of satellite communications as an example of high latency channels.

Figure 2-6 studies the expected time to complete transmission of $M = 10$ data packets of size $n = 10000$ bits, with different packet error probabilities in a GEO satellite link with a propagation delay of 125 ms. We assume a link with parameters specified in the figure. Note that our network coding scheme (TDD optimal) and the network coding full duplex optimal scheme have similar performance over a wide range of block error probabilities. In fact, for the worst case ($Pe = 0.8$) presented in this figure, our scheme has an expected time of completion only 29 % above the full duplex scheme. This is surprising considering that the transmitter in the full duplex scheme sends coded packets non-stop until an ACK packet is received. The explanation for

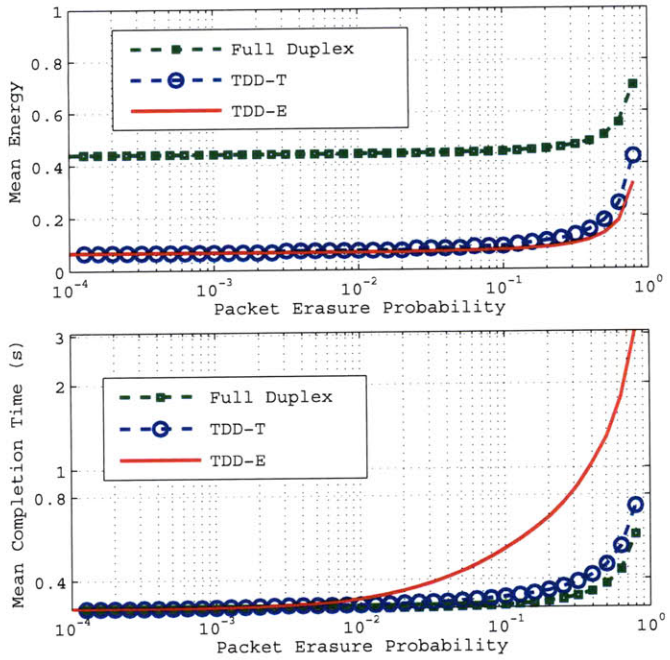

Figure 2-7: Mean Energy and Time to complete transmission. Parameters used: $M = 10$, packet size $n = 10,000$ bits, $R = 1.5$ Mbps, $h = 80$ bits, $g = 20$ bits, $n_{ack} = 100$ bits.

this behavior is that our scheme is sending enough coded packets, given the channel conditions, so that the number of stops to listen (which are very costly) is minimized. Thus, our scheme can have similar performance to that of full duplex optimal scheme, in the sense of expected time to completion. Most importantly, our scheme is very likely to have a much better performance in terms of energy consumption due to the long periods in which the transmitter stops to listen for the ACK packets.

Figure 2-6 also shows the performance of the comparison scheme 3 of Section 2.1.8. Note that when $\omega = 10$, i.e., the transmitter sends at most 10 coded packets before stopping to listen, the performance is comparable to our optimal scheme when the block error probability is low. This fact confirms that, for low block erasure probabilities, the optimal choice of coded packets to transmit when i dofs are required at the receiver (N_i) is simply i. In other words, if $M = 10$ and the block error probability is low, the first transmission contains 10 coded packets. Note that using $\omega = 9$ already suffers from a considerable degradation in performance even for low Pe because the transmitter cannot transmit the minimum number of coded

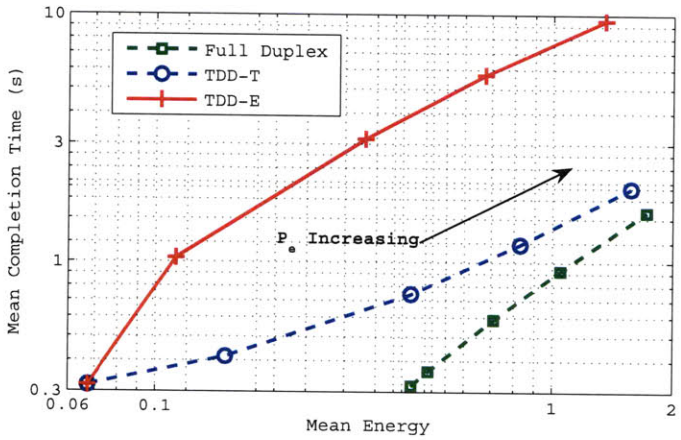

Figure 2-8: Mean Energy and Time to complete transmission trade-off. Parameters used: $M = 10$, packet size $n = 10,000$ bits, $R = 1.5$ Mbps, $h = 80$ bits, $g = 20$ bits, $n_{ack} = 100$ bits, and $Pe = 0.00001, 0.4, 0.8, 0.9, 0.95$.

packets (M) necessary to decode the information after the first transmission, and so it must transmit at least one more coded packet after the first ACK. Note that the performance of $\omega = 5$ and $\omega = 9$ is similar for low block error probability because both of them require at least two stops to listen for ACK packets in order to relay all the information, and it is the stopping time that affects delay the most on a high latency channel. For the case of $\omega > 10$ we would see a degradation for low Pe, with respect to optimum, because more packets than necessary are transmitted.

Finally, note that for the worst data error probability in Figure 2-6, all fixed schemes (TDD with fixed ω) take at least 5 times more time to complete transmission than the network coding full duplex optimal scheme. The case of $\omega = 1$ can be interpreted as the performance of the Stop-and-Wait ARQ scheme under the same channel conditions, which is considerably worse than the other schemes.

Figure 2-7 studies the mean energy and time to complete transmission of $M = 10$ data packets of size $n = 10,000$ bits, with different packet erasure probabilities in a GEO satellite link with a propagation delay of 125 ms, i.e., $T_{rt} = 250$ ms. In the following results, we have considered that coded packets and ACK are transmitted with the same power, and that this value is normalized, i.e., $P = 1$. The link parameters are specified in the Figure.

The first thing to notice in Figure 2-7 is that both TDD schemes have much

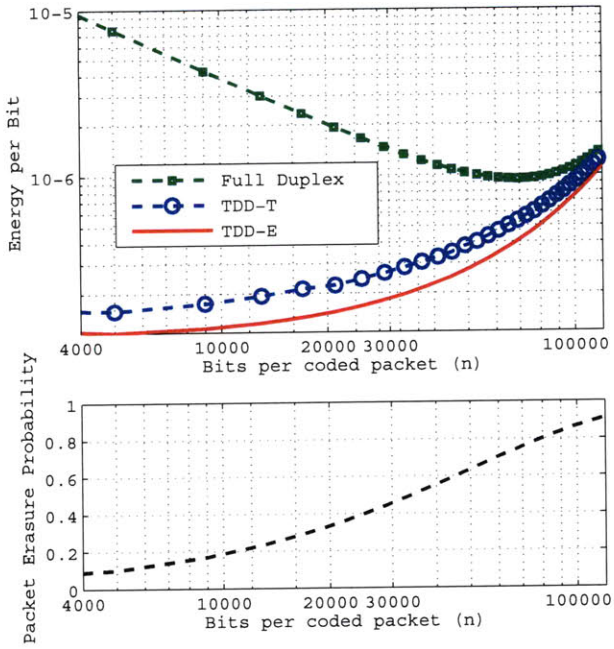

Figure 2-9: Mean Energy per bit to complete transmission with different n bits per coded packet and packet erasure probability for every n. Parameters used: $M = 10$, $R = 10$ Mbps, $h = 80$ bits , $g = 20$ bits, $n_{ack} = 100$ bits, bit error probability is 2.10^{-5}.

better performance with respect to the full duplex scheme, i.e., energy consumption of the full duplex scheme is considerably higher than the TDD schemes given the high latency characteristic of this channel.

Figure 2-7 shows that the gap between our network coding scheme optimized for energy and for completion time. Their performance stays similar over a wide range of packet erasure probabilities. When the packet erasure probability is low, the performance is the same for the two approaches, both in the sense of energy and delay. For high packet erasure probability the performance of both TDD versions is similar in terms of energy, although we observe a clear advantage of TDD-T over TDD-E in mean completion time.

Figure 2-7 also illustrates that our network coding scheme optimized for completion time (TDD-T) and the network coding full duplex optimal scheme have similar performance over a wide range of packet erasure probabilities. In fact, for the worst case ($Pe = 0.8$) presented in this Figure, our scheme has an expected time of com-

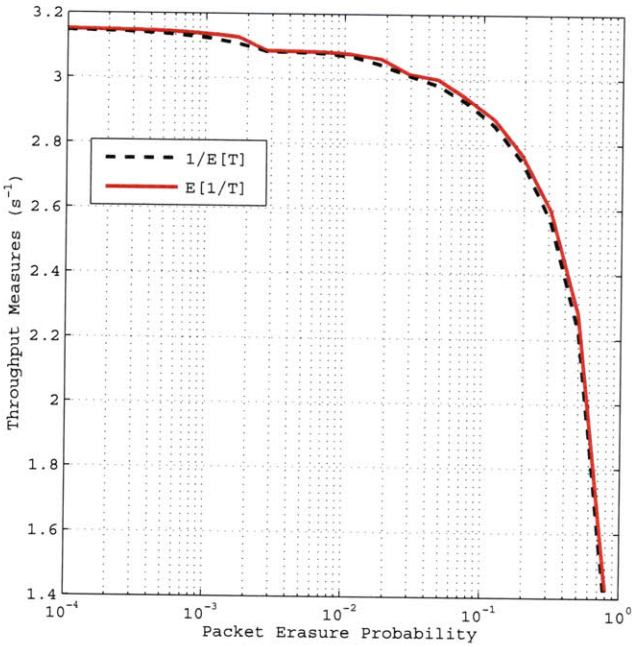

Figure 2-10: Mean throughput and η versus Pe for TDD-T with parameters $g = 20$ bits, $n_{ack} = 100$ bits, $h = 80$ bits, data rate 1.5 Mbps, $T_{rt} = 250$ ms, $Pe_{bit} = 0.0001$, $M = 10$, and $n = 10,000$ bits.

pletion only 30 % above the full duplex scheme. Thus, TDD-T can have similar performance to that of full duplex optimal scheme, in the sense of expected time to completion, while showing similar performance to TDD-E, the version optimized for energy consumption. This means that the TDD-T provides a good trade-off between energy and time to complete transmissions.

Figure 2-8 shows the trade-off curve between time and energy to complete transmission given different packet erasure probabilities Pe (using same parameters as in Fig. 2-7). The mean completion time changes significantly between TDD-T and TDD-E as the Pe increases, but the energy difference is small between both schemes. On the other hand, the mean energy is far greater in the full duplex scheme than both TDD schemes, while the completion time of the TDD-T scheme is close to the full duplex scheme.

Figure 2-9 shows the energy required to successfully transmit one bit of information, i.e., $\frac{E_M}{Mn}$. This Figure considers different values of n in a symmetric channel with parameters in the figure. It also shows the packet erasure probability for the

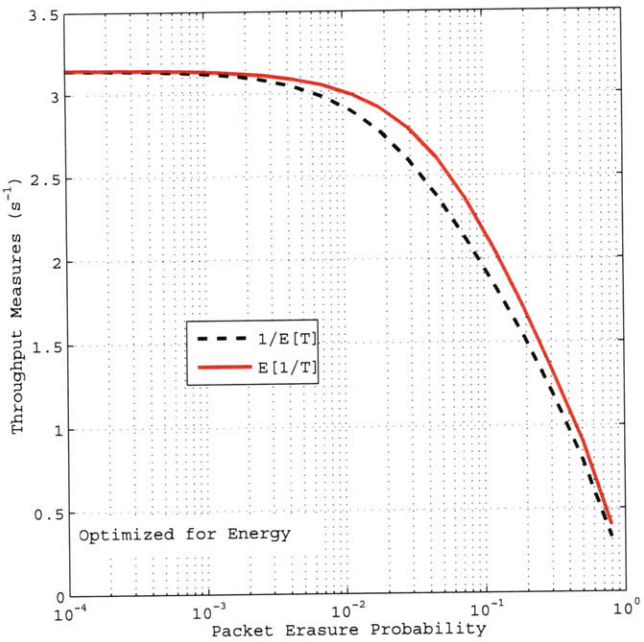

Figure 2-11: Mean throughput and η versus Pe for TDD-E with parameters $g = 20$ bits, $n_{ack} = 100$ bits, $h = 80$ bits, data rate 1.5 Mbps, $T_{rt} = 250$ ms, $Pe_{bit} = 0.0001$, $M = 10$, and $n = 10,000$ bits.

different values of n. First, we notice that the energy per bit required to complete a block transmission is much larger in the full duplex scheme than in both TDD-T and TDD-E. Second, we note that the energy per bit required for both TDD schemes is similar for a wide range of n. Finally, the effect of all three schemes having similar energy per bit consumption when n is large is explained by two factors that reduce the effect of the long propagation delay: 1) an increased packet erasure probability which forces more packet transmissions in the TDD schemes, and 2) a larger packet duration T_p, which causes the number of coded packets in flight to be reduced. Finally, Figure 2-9 shows that the full duplex network coding scheme has a value of n that optimizes the energy per data bit required to transmit the entire information.

Let us study the variance of the TDD-T scheme under different erasure probabilities. Figure 2-12 shows that the variance is very small but it is not a continuous function, showing discontinuities for certain values of Pe. Figure 2-13 shows that these discontinuities are related to a change in the number of coded packets sent in the first transmission of each M blocks, i.e., N_M. The variance decreases when N_M

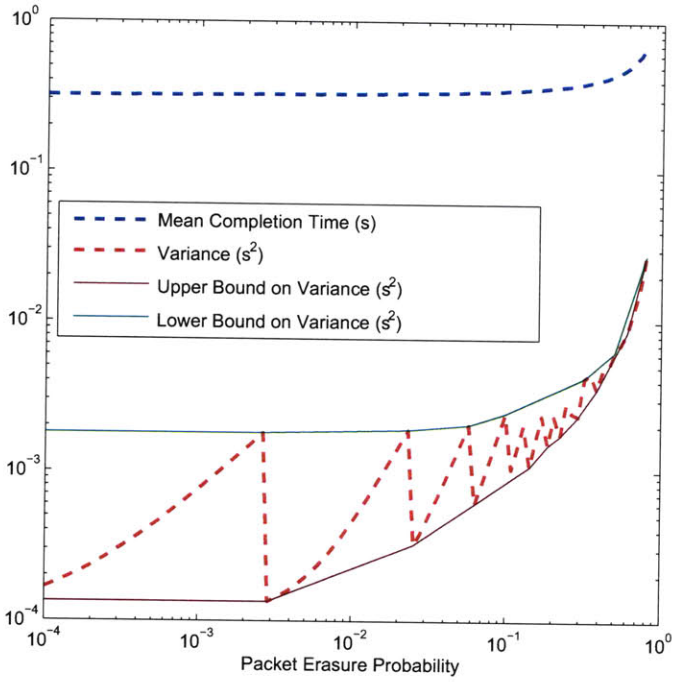

Figure 2-12: Variance and Mean of completion time for TDD-T versus packet erasure probability Pe, with parameters $g = 20$ bits, $n_{ack} = 100$ bits, $h = 80$ bits, data rate 1.5 Mbps, $T_{rt} = 250$ ms, $Pe_{ack} = 0.001$, $M = 10$, and $n = 10,000$ bits.

increases because we are increasing the probability of decoding all M packets after the first transmission. In practice, the Pe is an estimate of the packet erasure probability and these discontinuities can be misleading in terms of expected system performance. Thus, having bounds on the variance for each Pe, as shown in Figure 2-12, is more meaningful from a system's perspective.

Let us compare the mean throughput $MnE[1/T]$ and $\eta = Mn/E[T]$. Figure 2-10 shows that both $E[1/T]$ and $1/E[T]$ are very close when we optimize the N_is in terms of the mean completion time. Thus, choosing the parameters of our scheme to optimize the mean throughput or η will provide very similar results. However, this is not necessarily the case for other choices of N_i, e.g., when we choose them to minimize the mean completion energy as Figure 2-11 shows.

Let us turn our attention now to the problem of maximizing the parameter η, i.e., our mean throughput lower bound. Recall that for this setting we are streaming data which is subdivided into blocks that are transmitted them using our scheme. Considering again a satellite link, given a fixed bit error probability ($Pe_{bit} = 0.0001$)

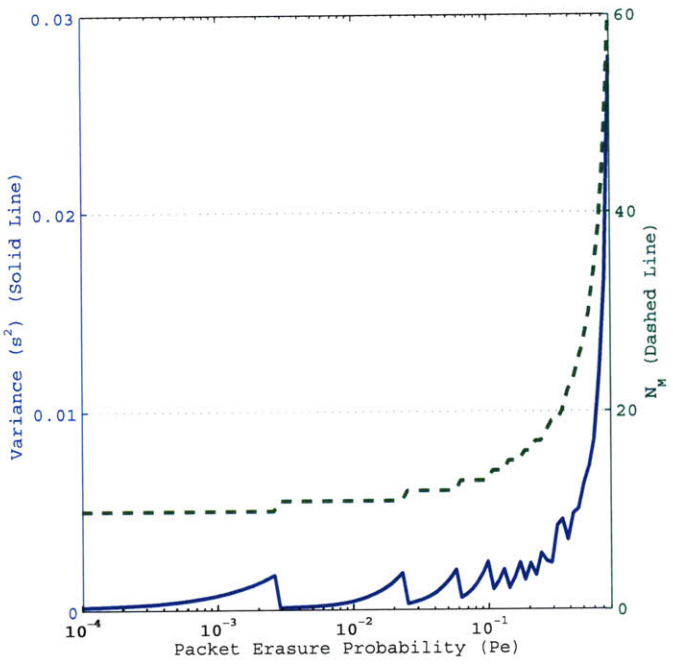

Figure 2-13: Variance of completion time and N_M for TDD-T versus packet erasure probability Pe, with parameters $g = 20$ bits, $n_{ack} = 100$ bits, $h = 80$ bits, data rate 1.5 Mbps, $T_{rt} = 250$ ms, $Pe_{ack} = 0.001$, $M = 10$, and $n = 10,000$ bits.

let us study the problem of computing the optimal number of bits n per packet given some value of M. In these examples, for the case of a symmetric channel with independent bits $Pe = 1 - (1 - Pe_{bit})^{h+n+gM}$ and $Pe_{ack} = 1 - (1 - Pe_{bit})^{n_{ack}}$.

Figure 2-14 illustrates the values of η in Mbps given different choices of M and n. First, note that for each value of M there exists an optimal value of n. Thus, an arbitrary choice of n can produce a considerable degradation in performance in terms of throughput. Secondly, there is a (M, n) pair that maximizes the value of η. Finally, the performance of the full duplex network coding and our TDD optimal scheme is comparable for different values of n and M.

Figure 2-15 shows η in Mbps when we change the round-trip time T_{rt}. As expected, a lower T_{rt} allows more throughput in TDD. Again, we observe that our TDD optimal scheme has comparable performance to the full duplex scheme.

Let us know compare the performance in terms of throughput to other TDD ARQ schemes. Figure 2-16 shows η for the satellite communications setting with a fixed packet size of $n = 10000$ bits, $n_{ack} = 100$ bits, $T_{rt} = 250$ ms, $Pe_{ACK} = 0$ for all

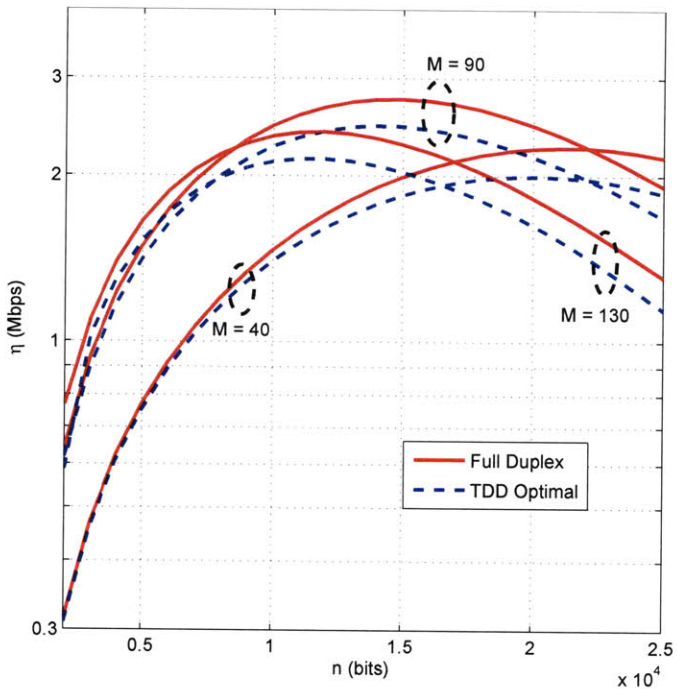

Figure 2-14: Throughput measure η versus the number of bits n in a data packet for a symmetrical channel, for different values of M with parameters $g = 100$ bits, $n_{ack} = 100$ bits, $h = 80$ bits, data rate 100 Mbps, $T_{rt} = 250$ ms, $Pe_{bit} = 0.0001$

schemes, a window size of $W = 10$ for the ARQ schemes, and $g = 20$ bits and $M = 10$ for our network coding scheme. We use different data rates to illustrate different latency scenarios, where higher data rate is related to higher latency. Note that the performance of our scheme is the same as both GBN and SR at low data packet error probability, which is expected because the window size W is equal to the block size of our scheme M and we expect very few errors. Our scheme has a slightly lower η for low Pe because each coded data packet includes gM additional bits that carry the random encoding vectors. This effect is less evident as latency increases. In general, our scheme has better performance than GBN.

Figure 2-16 shows that for low latency (0.1 Mbps) η of our scheme is very close to that of the SR ARQ scheme for all values of Pe, and better than the GBN scheme for high Pe. These results are surprising, because our scheme constitutes a block-by-block transmission scheme which will not start transmission of a new set of M data packets until the previous ones have been received and acknowledged. Note also that, as latency increases, our scheme shows much better performance than the SR scheme

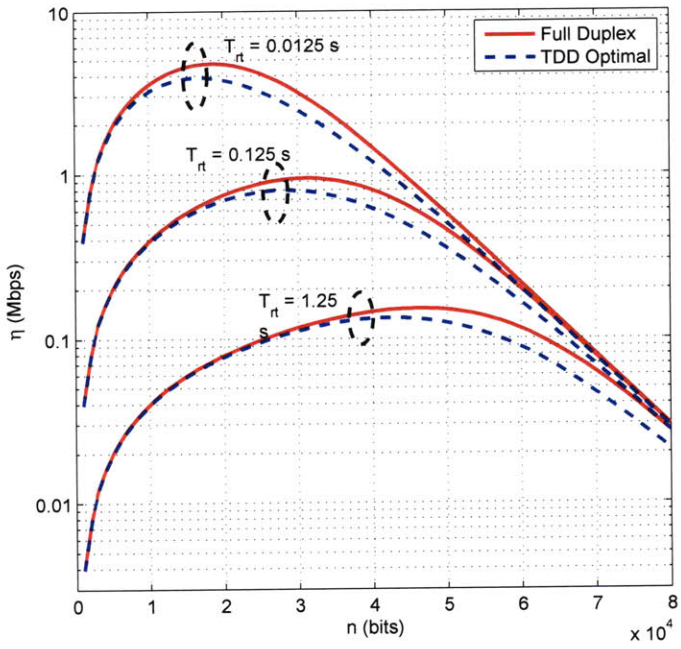

Figure 2-15: Throughput η versus n in a symmetrical channel considering different values of round-trip time T_{rt} with parameters $g = 100$ bits, $n_{ack} = 100$ bits, $h = 80$ bits, data rate 1.5 Mbps, $M = 10$, $Pe_{bit} = 0.0001$

for high Pe. The case of 10 Mbps and $Pe = 0.8$ shows that η of our scheme is more than three (3) times greater than that of SR.

Figure 2-17 shows η for a fixed data rate of 10 Mbps and different T_{rt}. We use a fixed packet size of $n = 10000$ bits, $n_{ack} = 100$ bits, $Pe_{ACK} = 0$ for all schemes, a window size of $W = 10$ for the ARQ schemes, and $g = 20$ bits and $M = 10$ for our network coding scheme. Note that the overhead of transmitting M coefficients of g bits per coded packet is only 2%. Thus, this effect cannot be appreciated in the figures. Again, the performance of our scheme is the same as both GBN and SR at low data packet error probability. Since the data rate is kept fixed, at higher T_{rt} we get higher latency. The throughput performance is similar to that observed in Figure 2-16 if we carry our comparison in terms of latency.

Another advantage of our scheme with respect to SR ARQ is that our scheme relies on transmitting successfully one block of M data packets before transmitting a new one. In fact, our scheme minimizes the delay of every block. In contrast, the SR ARQ does not provide any guarantee of delay for any data packet, e.g., the first packet of a

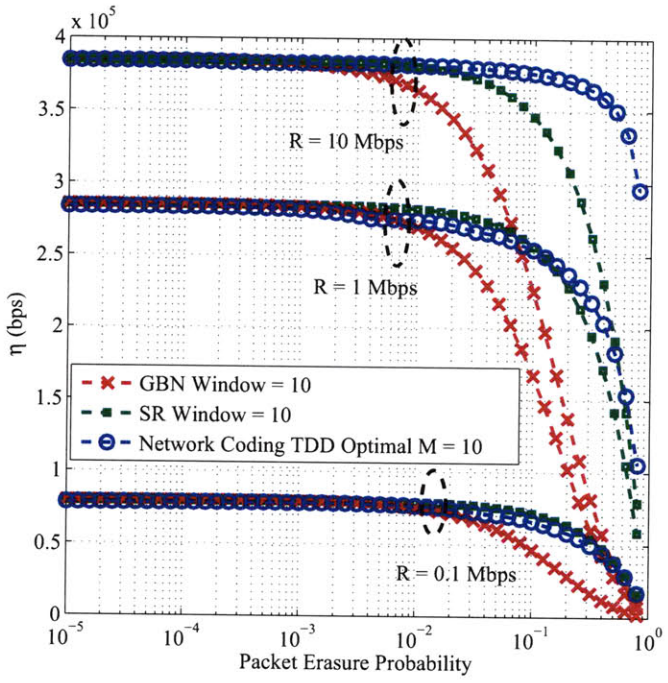

Figure 2-16: η versus data packet error probability with two TDD non-network coding schemes (Go-Back-N and Selective Repeat) and our optimal TDD network coding scheme, with different R. We used as parameters $g = 20$ bits, $n_{ack} = 100$ bits, $n = 10000$ bits, $h = 80$ bits, $T_{rt} = 0.25$ s

file to be transmitted could be the last one to be successfully received. In this sense, our comparison is not completely fair, as it favors the standard schemes. Nonetheless, our scheme is providing similar or better performance than SR but guaranteeing low transmission delays in individual data packets.

2.1.10 Queueing Analysis

The assumption up to this point is that the source had M data packets in its buffer before starting transmission. In a more realistic network setting, this buffer may sometimes empty or contain fewer than M packets awaiting transmission. Then, the source node must choose to either wait for additional packets to arrive, or take those packets in the buffer and start performing random linear coding. Thus, we are interested in studying the performance of our network coding scheme when random packet arrivals occur.

The problem of queueing for network coding systems has been considered pre-

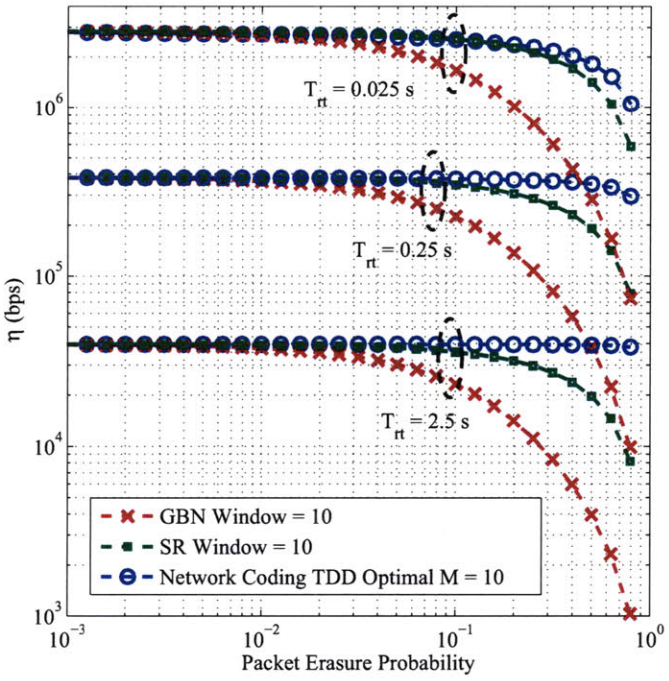

Figure 2-17: η versus data packet error probability with two TDD non-network coding schemes (Go-Back-N and Selective Repeat) and our optimal TDD network coding scheme, with different T_{rt} values. We used as parameters $g = 20$ bits, $n_{ack} = 100$ bits, $n = 10000$ bits, $h = 80$ bits, $R = 10$ Mbps

viously to account for burstiness or losses. Some of this work considered the case in which feedback is available, e.g., [8]. References [16] and [15] studied a system with random linear coding, slotted time, and a Bernoulli arrival process. However, previous work has not considered timing or TDD constraints. Our work considers the problem in which the channel is TDD, where the time is not slotted and, more importantly, the service time depends on the size of the bulk.

Furthermore, the analysis in [15] showed to have a general problem: the transition probability matrix proposed is not independent on previous transitions. In our case, we do not claim a fixed transition probability matrix, which depends on the previous transmission. Instead, we study the problem through a hidden Markov chain model in which the transition probability presented constitutes a mean behavior of the system and is useful in determining the stationary probability and the mean delay of a packet entering the system. This problem was solved in [35] prior to our work, and is consistent with other results in the area, e.g., [37].

We study our random linear network coding (RLNC) scheme for TDD channels when the data packets arrive randomly at the source node according to a Poisson process. This problem can be modeled as a bulk service queue with a general service process. Bulk or batch service queues have been studied widely, e.g., [36]. However, the service time for RLNC TDD will depend on the transmission time of both the coded packets and ACK packets, on the number of coded packets that are sent, and on the propagation time. This means that we have a bulk queue with a general service time, where the service time depends on the size of the bulk, which is not common in the existing studies. Reference [35] recently studied this problem with Poisson arrivals, calling it the $M/G^{(m,K)}/1$ queue, where the size of the bulk can range between m and K. We build on this work to develop the queueing model of RLNC TDD.

Queueing Model

We consider each data packet to be of fixed-length, arriving to a source node through a Poisson process with rate λ packets/s. Upon arrival, the data packet is placed in a buffer to await encoding and transmission to the receiver, as in Figure 2-18. The buffer forms a first-in-first-out (FIFO) queue. The service time of the queue is given by the time it takes to transmit a group of M packets taken from the queue using random linear network coding for TDD channels. The size of the group of packets is variable, where $m \leq M \leq K$. The pair (m, K) constitutes the range of the bulk size or number of packets taken to perform random linear network coding [4]. If the buffer has fewer than m data packets, the system will wait until m packets arrive before providing service. If the buffer contains more than K packets, the system will service exactly K packets. Finally, if the buffer has M packets with $m \leq M \leq K$, then the system will service M packets. The service time depends on the number of data packets taken from the queue at any time, i.e., the service time distribution is general but it depends on the size of the batch being transmitted. Thus, we can use the bulk queueing model $M/G^{(m,K)}/1$ developed in [35] to study the problem.

This bulk queueing model considers Poisson arrivals and a general service time

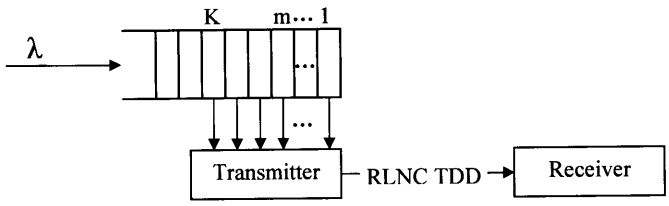

Figure 2-18: Queue model studied in this work.

that depends on the bulk size. The transition probability of the number of packets in the queue is given by [35]

$$P = \begin{bmatrix} a_0^{(m)} & a_1^{(m)} & \cdots & a_K^{(m)} & a_{K+1}^{(m)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_0^{(m)} & a_1^{(m)} & \cdots & a_K^{(m)} & a_{K+1}^{(m)} & \cdots \\ a_0^{(m+1)} & a_1^{(m+1)} & \cdots & a_K^{(m+1)} & a_{K+1}^{(m+1)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_0^{(K)} & a_1^{(K)} & \cdots & a_K^{(K)} & a_{K+1}^{(K)} & \cdots \\ 0 & a_0^{(K)} & \cdots & a_{K-1}^{(K)} & a_K^{(K)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

where $a_k^{(j)}$ is the probability of k arrivals during a service of type j.

Let us define

$$A^{(j)}(z) = \sum_{k=0}^{\infty} a_k^{(j)} z^k. \tag{2.37}$$

We can use a similar analysis as that of Reference [35] to prove that

$$A^{(j)}(z) = M_{T,j}(\lambda(z-1)) \tag{2.38}$$

and that

$$a_k^{(j)} = \frac{1}{k!} \frac{\partial^k}{\partial z^k} M_{T,j}(\lambda(z-1))\Big|_{z=0}. \tag{2.39}$$

The system is stable if and only if $\lambda < K\mu_K$, where $1/\mu_K$ is the mean service time when the bulk size is $M = K$, where $1/\mu_j = \frac{\partial}{\partial z} M_{T,j}(z)\Big|_{z=0}$.

Let us denote by $\Pi(z) = \sum_{i=0}^{\infty} \pi_i z^i$ the corresponding generating function of the

stationary probabilities. Reference [35] showed that $\Pi(z)$ can be expressed as

$$\Pi(z) = \frac{A^{(K)}(z)\sum_{i=0}^{K}\pi_i z^i - z^K A^{(m)}(z)\sum_{i=0}^{m}\pi_i}{A^{(K)}(z) - z^K} - \frac{\sum_{i=m+1}^{K}\pi_i A^{(i)}(z)}{A^{(K)}(z) - z^K} \tag{2.40}$$

which provides an expression for $\Pi(z)$ in terms of its first $K+1$ coefficients $\pi_0, ..., \pi_K$. Determining these $K+1$ coefficients provides a full characterization of the stationary probabilities [35]. Reference [35] proves that $A^{(K)}(z) - z^K$ has exactly K zeros satisfying $|z| \leq 1$ assuming that $A^{(K)}(z)$ has a radius of convergence greater than one. Denoting the roots as $1, z_1, ..., z_{K-1}$ and assuming that they are different, note that the numerator of (2.40) has to vanish for $z_1, ..., z_{K-1}$ which gives us $K-1$ linear equations

$$A^{(K)}(z_k)\sum_{i=0}^{K}\pi_i z_k^i - z_k^K A^{(m)}(z_k)\sum_{i=0}^{m}\pi_i - \sum_{i=m+1}^{K}\pi_i A^{(i)}(z_k) = 0 \tag{2.41}$$

for $k = 1, ...K-1$. Also, the numerator vanishes trivially for $z = 1$ for both the numerator and the denominator in (2.40). We thus need one more linear equation. To obtain this we use l'Hôspital's rule to exploit the fact that $\Pi(1) = 1$. This translates to

$$1 = \sum_{i=0}^{m}\left[1 + \frac{i - \lambda/\mu_m}{\lambda/\mu_K - K}\right]\pi_i + \sum_{i=m+1}^{K}\left[\frac{\lambda/\mu_K + i - \lambda/\mu_i}{\lambda/\mu_K - K}\right]\pi_i$$

where we have used the fact that $\left.\frac{\partial A^{(i)}(z)}{\partial z}\right|_{z=1} = \lambda/\mu_i$. Since, $\lambda/\mu_K \neq K$ in general, the denominator will not become zero as $z \to 1$. In fact, if the system is stable this condition will be satisfied. The final linear equation to fully characterize $\Pi(z)$ given in Reference [35] is

$$(a_0^m - 1)\pi_0 + a_0^m\pi_1 + \cdots + a_0^m\pi_m + a_0^{m+1}\pi_{m+1} + a_0^K\pi_K = 0 \tag{2.42}$$

Queue of Finite Capacity

The general solution requires the calculation of the roots of $A^{(K)}(z) - z^K$, which can result in numerical inaccuracies in practice because $A^{(K)}$ has exponential terms. Also, calculating the roots is increasingly difficult when the decision variable K assumes a larger value. For these reasons, we simplify the problem considering that the system has a capacity of B packets waiting to be serviced, i.e., without considering those that are being transmitted. The transition probability for this case is

$$P = \begin{bmatrix} a_0^{(m)} & a_1^{(m)} & \cdots & a_{B-1}^{(m)} & R(B-1,m) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_0^{(m)} & a_1^{(m)} & \cdots & a_{B-1}^{(m)} & R(B-1,m) \\ a_0^{(m+1)} & a_1^{(m+1)} & \cdots & a_{B-1}^{(m+1)} & R(B-1,m+1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_0^{(K)} & a_1^{(K)} & \cdots & a_{B-1}^{(K)} & R(B-1,K) \\ 0 & a_0^{(K)} & \cdots & a_{B-2}^{(K)} & R(B-2,K) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & a_0^{(K)} & \cdots & a_{B-K}^{(K)} & R(B-K,K) \end{bmatrix}$$

where $R(k,l) = 1 - \sum_{j=0}^{k} a_j^{(l)}$.

In order to compute the stationary distribution, it suffices to solve $\bar{\pi} = P\bar{\pi}$, with $\bar{\pi} = [\pi_0, \pi_1, ..., \pi_B]^T$, under the constraint that $\sum_{i=0}^{B} \pi_i = 1$.

Performance Analysis

We will consider two metrics in order to study performance of the system. First, the mean queue size defined as

$$E[Q] = \sum_{i=0}^{B} i\pi_i \tag{2.43}$$

for the case in which the queue capacity is B. If there is no constraint on the capacity, we simply let $B \to \infty$.

The second metric is the mean batch size in steady state, which can take values

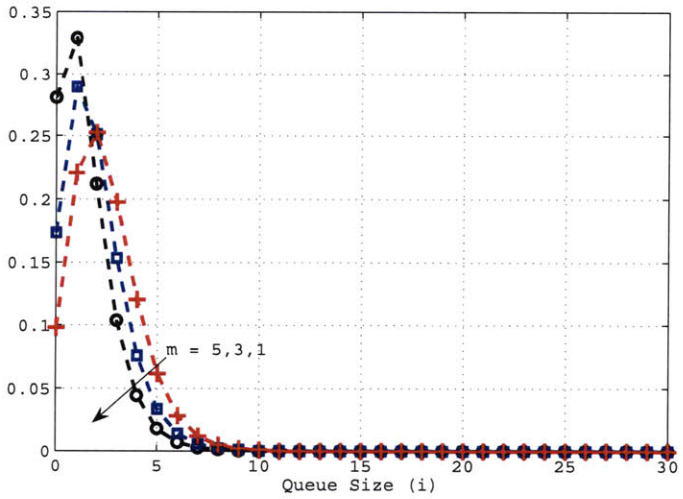

Figure 2-19: Stationary distribution $\pi_i, i = 0, 1, ..., B$ for $\lambda = 30$ packets/s, $K = 5$, $B = 30$, and different values of m.

$\{m, m+1, ..., K\}$. Defining $Z_{(m,K)}$ as the batch size for a choice of (m, K), then

$$E[Z_{(m,K)}] = m \sum_{i=0}^{m} \pi_i + \sum_{i=m+1}^{K-1} i\pi_i + K \sum_{i=K}^{B} \pi_i, \tag{2.44}$$

where $\sum_{i=K}^{B} \pi_i = 1 - \sum_{i=0}^{K-1} \pi_i$. If there is no constraint on capacity, we let $B \to \infty$.

Numerical Results

This section provides numerical examples that show the performance of our network coding scheme for different settings of (m, K) and arrival rate λ. We use the mean queue size defined in the previous section as our metric of interest. We use a high latency channel with packet erasure probability $Pe = 0.2$, propagation time of 12.5 ms, data packets of 10,000 bits, $g = 20$ bits, a rate $R = 1.5$ Mbps, a header of $h = 80$ bits, and the ACK packet has 100 bits. We assume that the number of coded packets to be sent back-to-back (N_i) are chosen to minimize the mean completion time.

Figure 2-19 shows the stationary distribution for $\lambda = 30$ packets/s, $K = 5$, $B = 30$ and different values of m. This figure shows that the high probability states correspond to small number of packets in the queue. The probability of large queue sizes when the system operates in steady state is very low. Figure 2-19 also shows that

for low values of m, the stationary distribution is concentrated in the lower values of the queue size i. As m increases, the stationary distribution spreads over larger values of the queue size i.

Table 2.1 shows the mean queue size when $\lambda = 1$ packet/s under different configurations of the pair (m, K), with $m \leq K$. We observe that the mean queue size shows greater dependence on the value of m than on the value of K. For example, increasing the value of K when $m = 1$ shows little variation in the mean queue size, while increasing m with any value of $K > 1$ increases the mean queue size. In terms of minimizing the mean queue size for low values of λ, this means that we should allow transmission of bulks of size 1. This is the case because waiting for additional packets before transmitting is costly, considering that the time between packet arrivals might be larger than the mean service time for a single data packet. For the case of bulks of size 1, our scheme transmits several copies of the packet back-to-back before stopping for an ACK, which is similar to the idea presented in [20].

Table 2.2 shows the mean queue size when $\lambda = 30$ packet/s under different configurations of the pair (m, K), with $m \leq K$. For this λ, we do not consider the setting $m = K = 1$ because it is not stable for the infinite capacity case $(B \to \infty)$ and will present very high packet drops when the capacity B is finite. Again, there is an advantage of allowing $m = 1$ for the studied cases, in terms of reducing the mean queue size.

Tables 2.1 and 2.2 also show the mean batch size for $\lambda = 1$ packet/s and $\lambda = 30$ packet/s, respectively. The mean batch size is biased by the value of m and K. However, it gives us some intuition about the operation of the system. For example, Table 2.1 shows that when m is too large with respect to the arrival rate, the mean batch size is close to m. This means that on average the system services the batches much faster than the time it takes the queue to fill with m new packets. Thus, the system will be idle for long periods of time just waiting for the queue to fill to the required m. Only with small probability the batches will contain more than m packets. Of course, if $m = K$ the batch size will always be m as seen in the tables.

Let us consider the case of a fixed batch size, i.e., $m = K$. Tables 2.1, 2.2 and

Table 2.1: Mean Queue Size for different (m, K) configurations. The parameters used are $\lambda = 1$ packet/s, $B = 30$

Mean Queue Size	$K = 1$	$K = 2$	$K = 3$	$K = 4$	$K = 5$
$m = 1$	0.0408	0.0398	0.0397	0.0397	0.0397
$m = 2$	-	0.0495	0.0495	0.0495	0.0495
$m = 3$	-	-	0.0595	0.0595	0.0595
$m = 4$	-	-	-	0.0696	0.0696
$m = 5$	-	-	-	-	0.07844
Mean Batch Size	$K = 1$	$K = 2$	$K = 3$	$K = 4$	$K = 5$
$m = 1$	1.0000	1.0009	1.0009	1.0009	1.0009
$m = 2$	-	2.0000	2.0000	2.0000	2.0000
$m = 3$	-	-	3.0000	3.0000	3.0000
$m = 4$	-	-	-	4.0000	4.0000
$m = 5$	-	-	-	-	5.0000

Table 2.2: Mean Queue Size and Mean Batch size for different (m, K) configurations, with $\lambda = 30$ packet/s, $B = 30$

Mean Queue Size	$K = 2$	$K = 3$	$K = 4$	$K = 5$
$m = 1$	2.2972	1.5904	1.4499	1.4085
$m = 2$	2.5720	1.8114	1.6542	1.6092
$m = 3$	-	2.1548	1.9433	1.8766
$m = 4$	-	-	2.2397	2.1575
$m = 5$	-	-	-	2.4345
Mean Batch Size	$K = 2$	$K = 3$	$K = 4$	$K = 5$
$m = 1$	1.5504	1.6442	1.6664	1.6710
$m = 2$	2.0000	2.2645	2.3301	2.3468
$m = 3$	-	3.0000	3.1455	3.1893
$m = 4$	-	-	4.0000	4.0769
$m = 5$	-	-	-	5.0000

Figure 2-20 show that the optimal choices are $m = K = 1$, $m = K = 2$, and $m = K = 3$ for $\lambda = 1$ packet/s, $\lambda = 10$ packet/s, and $\lambda = 30$ packet/s, respectively. However, we notice that the mean queue size is larger than other configurations of (m, K) without the fixed batch size restriction. For example, the optimal fixed batch size configuration for $\lambda = 30$ packet/s is 53 % larger than the $(m, K) = (1, 5)$ configuration presented in the table. Thus, we observe that having a fixed batch size is not the optimal configuration in general.

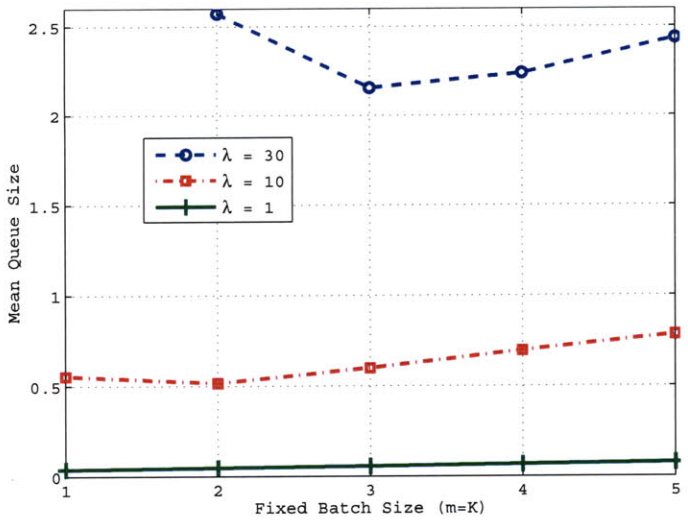

Figure 2-20: Mean queue size for the fixed batch size case ($m = K$) with $B = 30$.

2.2 Random Linear Network Coding for One-to-all Broadcast in Time Division Duplexing Channels

We now analyze the problem of one-to-all broadcast under the TDD constrain and using a similar random linear network coding scheme to that of the link case, i.e., we provide an extension of the scheme for the case of several receivers. In particular, this section studies the mean completion time and energy of the one-to-all broadcast scheme. We provide simple and useful heuristics to determine the number of coded data packets to be transmitted, with a considerable reduction in the computation time. We compare the proposed schemes to optimal scheduling policies and find considerable improvement in terms of completion time.

2.2.1 Model

A sender wants to broadcast M data packets at a given data rate R [bps] to N receivers as in Figure 2-21. We assume an independent packet erasure channel for each of the receivers, for simplicity. However, this is a valid assumption in a satellite scenario, in which a satellite is transmitting information to several geographically

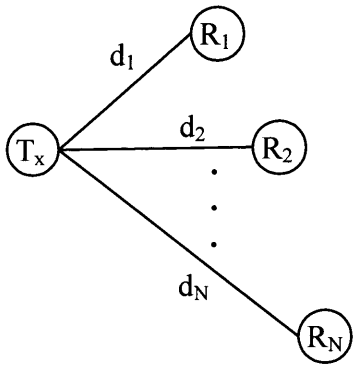

Figure 2-21: Broadcast network.

separated receivers. We also assume that the receivers cannot cooperate or share information. Nodes can transmit and receive, but not both at the same time. The sender uses random linear network coding [4] to generate coded data packets. Each coded data packet contains a linear combination of the M data packets of n bits each, as well as the random coding coefficients used in the linear combination as in the case of a link.

As in previous results, we assume that the field size q is large enough so that the expected number of successfully received packets at the receiver, in order to decode the original data packets, is approximately M.

2.2.2 Description of Scheme

The sender can transmit coded packets back-to-back before stopping to wait for an ACK packet from each receiver. Each ACK packet feeds back the number of degrees of freedom (dofs), that are still required to decode successfully the M data packets to a particular receiver.

Transmission begins with M information packets, which are encoded into a number of random linear coded packets, and transmitted. If all M packets are decoded successfully by all receivers, the process is completed. Otherwise, each ACK informs the transmitter how many dofs are missing, say $i_1, i_2, ..., i_N$ for receivers $1, 2, ..., N$, respectively.

As in the case of the link, the process is modelled as a Markov chain. The states

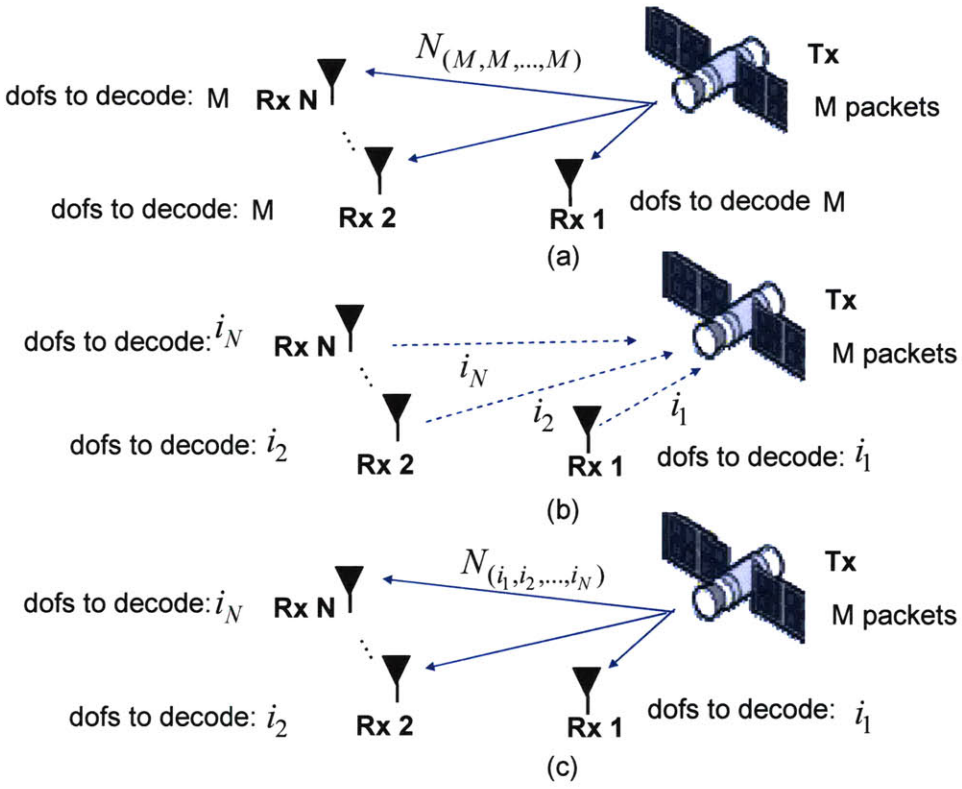

Figure 2-22: Optimal selection of coded packets in our network coding TDD scheme for one-to-all broadcast. (a) The transmitter initially generates $N_{(M,...,M)}$ coded packets from the M packets in its queue and sends them to the receivers before stopping to wait for the ACK packets. (b) The receivers have some dofs useful to decoding the original packets, but some dofs are still missing. Each receiver k send an ACK packet indicating that i_k dofs are needed to decode. (c) Upon reception of the ACK packet, the transmitter updates its knowledge of the receiver and generates $N_{(i_1,...,i_N)}$ coded packets and sends them to the receivers.

$(s_1, s_2, ..., s_N)$ are defined by the number of dofs required s_k at receiver k to decode successfully the M packets. Thus, the states range from $(M, M, ..., M)$ to $(0, 0, ..., 0)$. This is a Markov chain with $(M+1)^N - 1$ transient states and one recurrent state (state $(0, 0, ..., 0)$).

The optimal scheme would require us to allow for a variable, e.g., $N_{(s_1,s_2,...,s_N)}$ representing the number of coded packets to be transmitted given state $(s_1, s_2, ..., s_N)$, for each transient states in our Markov chain. The communication process is illustrated in Figure 2-22. At the beginning, each receiver requires M dofs to decode the information. The transmitter starts by sending $N_{(M,...,M)}$ coded packets before stop-

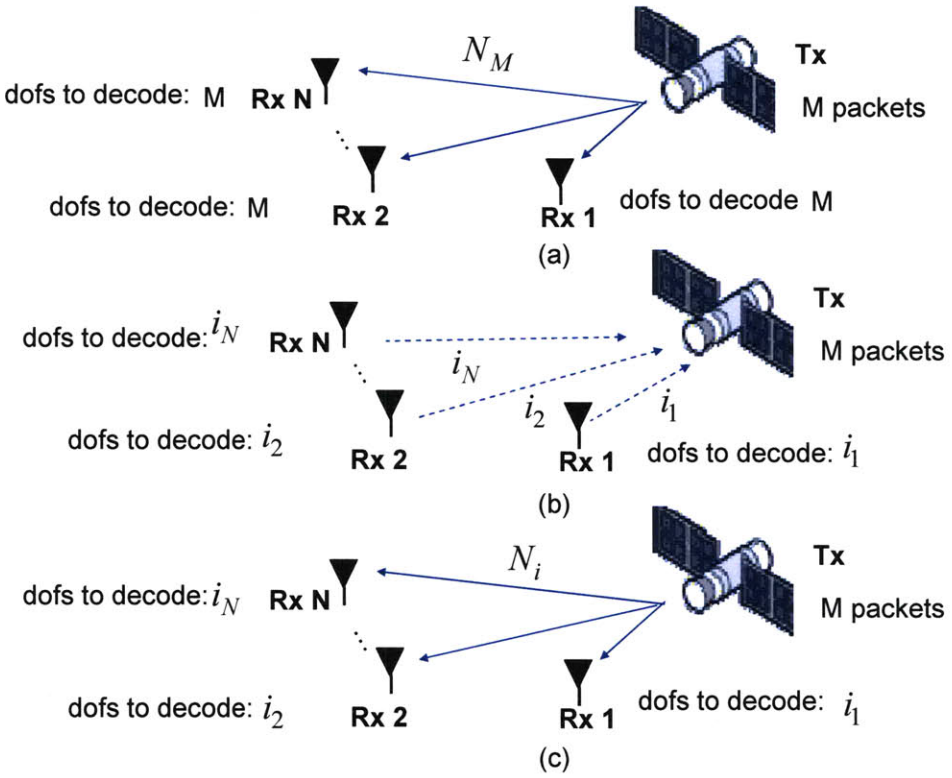

Figure 2-23: Suboptimal selection of coded packets in our network coding TDD scheme for one-to-all broadcast. (a) The transmitter initially generates N_M coded packets from the M packets in its queue and sends them to the receivers before stopping to wait for the ACK packets. (b) The receivers have some dofs useful to decoding the original packets, but some dofs are still missing. Each receiver k send an ACK packet indicating that i_k dofs are needed to decode. (c) Upon reception of the ACK packet, the transmitter updates its knowledge of the receiver and generates N_i coded packets, where $i = \max_{k=1,2,...,N} i_k$, and sends them to the receivers.

ping to listen for ACKs (Figure 2-22 (a)). Each receiver then sends an ACK packet indicating how many dofs it requires to decode, say $i_1, i_2, ..., i_N$ for receivers $1, 2, ..., N$, respectively. (Figure 2-22(b)). Then, the transmitter sends $N_{(i_1,i_2,...,i_N)}$ coded packets before stopping (Figure 2-22(c)), and the process is repeated until all receivers have decoded all the information.

Since this optimal scheme implies an exponentially growing number of variables as we increase the number of receivers, we propose a suboptimal scheme that requires only M variables to be optimized.

This suboptimal scheme considers that the transmitter sends N_i coded packets, where $i = \max_{j=1,2,...,N} i_j$. If an ACK is lost, the transmitter assumes the previ-

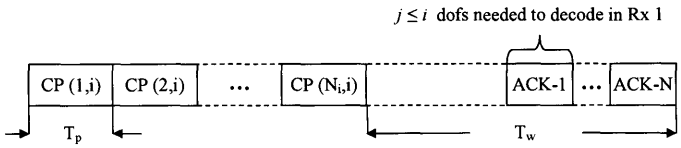

Figure 2-24: Network coding TDD scheme in one-to-all broadcast.

ous state for the corresponding receiver. The communication process is illustrated in Figure 2-23. At the beginning, each receiver requires M dofs to decode the information. The transmitter starts by sending N_M coded packets before stopping to listen for ACKs (Figure 2-23 (a)). Each receiver then sends an ACK packet indicating how many dofs it requires to decode, say $i_1, i_2, ..., i_N$ for receivers $1, 2, ..., N$, respectively. (Figure 2-23(b)). Then, the transmitter sends N_i coded packets before stopping (Figure 2-23(c)). This process is repeated until all M packets have been decoded successfully by all receivers. We are interested in the optimal number N_i of coded packets to be transmitted back-to-back. Note that $N_i \geq i$.

Figure 2-25 provides an example for 2 receivers and a block size of 3 packets. We have highlighted in this figure the states in which at least one receiver requires 3 coded packets in order to decode. Note that not all possible transitions from one state to the others have been included in this figure.

Figure 2-24, illustrates the time window allocated to the system to transmit N_i coded packets. Each coded packet $CP(1,i)$, $CP(2,i)$, etc. is of duration T_p. The waiting time T_w is chosen so as to accommodate the propagation delay and time to receive the ACKs from each receiver. We discuss this in more detail when we study the mean completion time.

The transition probabilities from state $(s_1, s_2, ..., s_N)$ to state $(s'_1, s'_2, ..., s'_N)$ are

$$P_{(s_1,s_2,...,s_N)\to(s'_1,s'_2,...,s'_N)} = P\left(X_1(n)=s'_1,...,X_N(n)=s'_N | X_1(n-1)=s_1,...,X_N(n-1)=s_N\right) \quad (2.45)$$

where $X_i(n)$ is the number of dofs required at receiver i at the end of transmission n. For simplicity of notation, let us say $P\left(X_1(n)=s'_1,...,X_N(n)=s'_N | X_1(n-1)=s_1,...,X_N(n-1)=s_N\right) = P\left(s'_1,...,s'_N | s_1,...,s_N\right)$. Similarly we consider that $P\left(X_i(n)=s'_i | X_1(n-1)=s_1,...,X_N(n-1)=s_N\right) =$

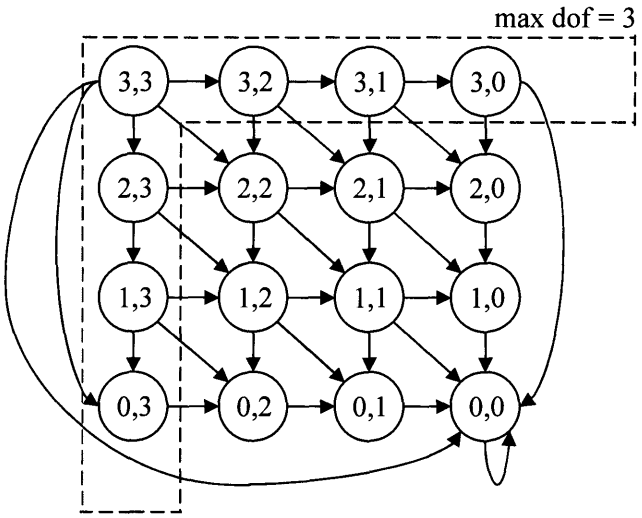

Figure 2-25: Markov chain for the case of $N = 2$ receivers and a block size of $M = 3$.

$P\left(s_i'|s_1,...,s_N\right)$ and $P\left(X_i(n)=s_i'|X_i(n-1)=s_i,\max_{j=1,2,...,N} s_j\right) = P\left(s_i'|s_i,\max_{j=1,2,...,N} s_j\right)$.

If we consider independent packet erasure channels for each of the receivers,

$$P_{(s_1,...,s_N)\to(s_1',...,s_N')} = P\left(s_1'|s_1,...,s_N\right)...P\left(s_N'|s_1,...,s_N\right). \tag{2.46}$$

The dependence on the previous state $(s_1, s_2, ..., s_N)$ can be translated into a dependence on the state with maximum dofs required to transmit, i.e., $i = \max_{j=1,2,...,N} s_j$, because i determines N_i, the number of coded data packets sent by the transmitter. Thus,

$$P_{(s_1,s_2,...,s_N)\to(s_1',s_2',...,s_N')} =$$
$$P\left(s_1'|s_1,\max_{j=1,2,...,N} s_j\right)...P\left(s_N'|s_N,\max_{j=1,2,...,N} s_j\right) = P\left(s_1'|s_1,N_i\right)...P\left(s_N'|s_N,N_i\right). \tag{2.47}$$

where $P\left(s_j'|s_j,N_i\right)$ has a similar structure to the transition probabilities studied for the case of a link. The main difference is that the value of N_i is no longer associated with the starting state of a particular receiver, but with a value determined from all starting states. For $0 < s_j' < s_j$, this can be translated into

$$P\left(s_j'|s_j,N_i\right) = (1 - Pe_{ack-j})f(s_j, s_j')(1 - Pe_j)^{s_j - s_j'} {Pe_j}^{N_i - s_j + s_j'} \tag{2.48}$$

where

$$f(s_j, s'_j) = \begin{cases} \binom{N_i}{s_j - s'_j} & \text{if } N_i \geq s_j, \\ 0 & \text{otherwise} \end{cases} \tag{2.49}$$

and Pe_j and Pe_{ack-j} represents the erasure probability of a coded packet and of an ACK packet for the erasure channel of receiver j, respectively.

For $s_j = s'_j > 0$ the expression for the transition probability reduces to:

$$P\left(s_j|s_j,N_i\right) = (1 - Pe_{ack-j}){Pe_j}^{N_i} + Pe_{ack-j}. \tag{2.50}$$

Note that for $P\left(0|0,N_i\right) = 1$. Finally, for $s'_j = 0$ $P\left(s'_j=0|s_j,N_i\right) = 1 - \sum_{s'_j=1}^{s_j} P\left(s'_j|s_j,N_i\right)$.

We can define P as the transition probability for our system. The speed of convergence from state $(M, ..., M)$ to state $(0, ..., 0)$ can be summarized in the following Lemma.

Lemma 2. *Let λ_2 be the second largest eigenvalue of P, and assume that there is only one eigenvalue with this magnitude. Then, the number of stops to wait for ACKs preceeded by transmissions of back-to-back coded packets $\mathcal{N}$ to transit from state $(M, ..., M)$ to state $(0, ..., 0)$ with probability at least $1 - \epsilon$ is*

$$\mathcal{N} \geq \frac{\ln G - \ln \epsilon}{-\ln |\lambda_2|} \tag{2.51}$$

where G is a constant.

Proof. We use a similar method to the proof of Lemma 2 in [38]. Let us assume, without loss of generality, that our transition probability P has the following structure

$$P = \begin{bmatrix} P_{(M,...,M)\to(M,...,M)} & \cdots & P_{(M,...,M)\to(0,...,0)} \\ \vdots & \vdots \quad \vdots & \vdots \\ 0 & \cdots \quad 0 & 1 \end{bmatrix}$$

and

$$P^{\mathcal{N}} = \begin{bmatrix} a_{11}(\mathcal{N}) & a_{12}(\mathcal{N}) & \cdots & a_{1\eta}(\mathcal{N}) \\ : & : & : & : \\ 0 & \cdot\cdot & 0 & 1 \end{bmatrix}$$

where $a_{ij}(\mathcal{N})$ is the probability of transitioning from the i-th state in the matrix to the j-th state in the matrix in $\mathcal{N}$ iterations (transmissions followed by stop to receive ACKs), and η are the number of columns in the matrix.

We are interested in determining the number of stops to wait for ACKs $\mathcal{N}$ so that $|a_{1\eta}(\mathcal{N}) - 1| \leq \epsilon$ with $\epsilon > 0$. In general, we could write this as $|q_0 P^{\mathcal{N}} - \Pi| < T$, where q_0 is the starting state, Π is the steady state probability, and T is our performance target. In our case, $q_0 = [1, 0, ..., 0]$ since we are interested in studying convergence when we start in state $(M, ..., M)$, $\Pi = [0, ..., 0, 1]$ because state $(0, ..., 0)$ is the only absorbing state, and $T = [T_1 ... T_{\eta-1} \epsilon]$ where ϵ is our target performance, i.e., we have not imposed conditions for convergence from state $(M, ..., M)$ to the other states $(T_1, ..., T_{\eta-1})$.

By the Cayley-Hamilton theorem, for $\mathcal{N} \geq \eta$, $P^{\mathcal{N}} = \sum_{l=0}^{\eta-1} \phi_l(\mathcal{N}) P^l$ for some constants $\phi_l(\mathcal{N})$. Denoting the eigenvalues by $1, \lambda_2, ..., \lambda_\eta$, and using Lagrange's interpolation formula as in [38], we can write

$$P^{\mathcal{N}} = \sum_{i=1}^{\eta} F_i^{\mathcal{N}}(P) \lambda_i^{\mathcal{N}} \tag{2.52}$$

where

$$F_i^{\mathcal{N}}(P) = \lambda_i^{\mathcal{N}} \frac{\prod_{j=1, j\neq i}^{\eta} (P - \lambda_j I)}{\prod_{j=1, j\neq i}^{\eta} (\lambda_i - \lambda_j)} \tag{2.53}$$

where I is the identity matrix, and $\lambda_1 = 1$.

Since P is a stochastic matrix, $|\lambda_i| < 1$ for $i > 1$. Since $q_0 P^{\mathcal{N}} \to \Pi$, as $\mathcal{N} \to \infty$,

we have that $q_0 F_1^{\infty}(P) = q_0 F_1^0(P) = q_0 F_1(P) = \Pi$. Thus,

$$|q_0 P^{\mathcal{N}} - \Pi| \leq |\lambda_2|^{\mathcal{N}} [1, 0, ..., 0] \sum_{i=2}^{\eta} |F_i(P)| \tag{2.54}$$

$$= |\lambda_2|^{\mathcal{N}} [g_{11}, g_{12}, ..., g_{1\eta}] \tag{2.55}$$

where $F_i(P) = F_i^0(P)$, $|F_i(P)|$ denotes a matrix whose elements are the magnitudes of the elements of $F_i(P)$, and

$$\sum_{i=2}^{\eta} |F_i(P)| = \begin{bmatrix} g_{11} & \cdots & g_{1\eta} \\ : & : & : \\ g_{\eta 1} & \cdots & g_{\eta\eta} \end{bmatrix}.$$

Since we are interested in $|a_{1\eta}(\mathcal{N}) - 1| \leq \epsilon$, this translates to

$$|\lambda_2|^{\mathcal{N}} g_{1\eta} \leq \epsilon \tag{2.56}$$

which concludes the proof. □

2.2.3 Mean Completion Time

The mean time for completing the transmission of the M data packets constitutes the mean time of absorption, i.e., the time to reach state $(0, ..., 0)$ for the first time, given that the initial state is $(M, ..., M)$. This can be expressed in terms of the mean time for completing the transmission given that the Markov chain is in state is $(s_1, ..., s_N)$, $T_{(s_1,...,s_N)}$, $\forall s_i = 0, 1, ..M-1, \forall i = 1, ..., N$. Let us denote the transmission time of a coded packet as T_p, and the waiting time to receive an ACK packet as T_w. For our scheme, $T_p = \frac{h+n+gM}{R}$, as in the case of a link, but the expression of T_w changes slightly to consider the transmission of multiple ACK packets from the receivers to the transmitter.

Let us define d_i as the distance between the transmitter and node i, as in Figure 2-21. We assume that the nodes have been numbered so that $d_1 \leq d_2 \leq ... \leq d_N$. Let us define t_{btA}^i as the time node i has to wait before starting to transmit after

he has received the last coded packet from the transmitter. The choice of t^i_{btA} depends on characteristics of the link between the receivers and the transmitter and interference that a receiver could generate in other receivers at the time of transmitting its ACK. If the nodes do not generate interference over other nodes, e.g., a satellite link which typically has a highly directional antenna, then we could use $t^i_{btA} = \max\left(t^{i-1}_{btA} + T_{ack} - T_{rt-i} + T_{rt-(i-1)}, 0\right)$, where $T_{ack} = n_{ack}/R$, n_{ack} is the number of bits in the ACK packet, R is the link data rate, and T_{rt-i} is the round trip time for node i. Thus, $T_w = T_{rt-N} + t^N_{btA} + T_{ack}$ and $t^1_{btA} = 0$. If the transmission of the ACK packets can create interference in transmissions to other receivers, the first ACK could be sent after all data packets have been correctly received. In this case, $t^1_{btA} = \left(T_{rt-N} - T_{rt-1}\right)/2$ and we can use the previous recursive formula for t^i_{btA} and the expression for T_w.

We can define T^i as the time it takes to transmit N_i coded data packets and receive the ACK packets from the different receivers. It is easy to show that $T^i = N_i T_p + T_{rt-N} + t^N_{btA} + T_{ack}$.

The mean completion time when the system is in state $(s_1, ..., s_N)$ is given by

$$T_{(s_1,...,s_N)} = T^i + \sum_{(s_1,...,s_N),(s'_1,...,s'_N)} P_{(s_1,...,s_N)\to(s'_1,...,s'_N)} T_{(s'_1,...,s'_N)} \tag{2.57}$$

where $i = \max_{j=1,...,N} s_j$. We can express this in vector form as

$$\bar{T} = [I - P]^{-1}\bar{\mu}. \tag{2.58}$$

where $\bar{T} = [T_{(s_1,...,s_N)}]$, $\bar{\mu} = [T^i]$ and P is the corresponding transition probability.

Since we are interested in the mean completion time when we start at state $(M, ..., M)$, we can use Cramer's rule to determine

$$T_{(M,...,M)} = \frac{\det\left(\Gamma \leftarrow_{(M,...,M)} \bar{\mu}\right)}{\det\left(\Gamma\right)} \tag{2.59}$$

where $\Gamma = I - P$, and the notation $\Gamma \leftarrow_{(M,...,M)} \bar{\mu}$ represents a matrix that has all columns as the Γ matrix except the column corresponding to state $(M, ..., M)$ which

is substituted by the vector $\bar{\mu}$. Due to characteristics of the Markov chain, Γ is a triangular matrix. Thus, computing $\det(\Gamma)$ reduces to multiplying the elements in the main diagonal of the Γ matrix.

The mean time for each state depends on all the mean times for the previous states. However, optimizing the values of all N_i is not as straight forward as the recursive method used in the case of a link.

Also, note that there are $(M+1)^N$ states in our Markov chain. This implies that we have to compute $\left((M+1)^N - 1\right) \times \left((M+1)^N - 1\right)$ transition probabilities to fill the P matrix and then solve the determinants of matrices of the same dimensions for each iteration of a search algorithm. Thus, the computational demands increases significantly, specially when we increase the number of receivers.

2.2.4 Heuristic for the Number of Coded Packets to Transmit

Given this limitation, let us consider some heuristics to estimate the values of $N_i, \forall i = 1, ..., M$, either to use them directly as an approximate solution or as an initial point of a search algorithm. These heuristics rely on solving the link case considering as packet erasure probability of the link a function of the packet erasure probabilities of the different channels in broadcast.

1) Worst Link Channel: In this heuristic we approximate the system as a link to the receiver with the worst channel, i.e., $Pe = \max_j Pe_j$. Then, we compute $N_i, \forall i = 1, ..., M$ to minimize the mean completion time with the network coding scheme studied for a link using the current values of T_p, T_w, and $Pe_{ack} = \max_j Pe_{ack-j}$.

2) Combined Erasure Effect: In this heuristic we approximate the system as a link to a receiver with $Pe = 1 - \prod_j (1 - Pe_j)$, i.e., assuming that a coded packet suffers an erasure in the link when it is seen as an erasure by at least one receiver. Then, we compute $N_i, \forall i = 1, ..., M$ to minimize the mean completion time with the network coding scheme studied for a link using the current values of T_p, T_w, and $Pe_{ack} = 1 - \prod_j (1 - Pe_{ack-j})$.

Determining P for a link requires computing $O\left(M^2\right)$ transition probabilities, while solving the same problem for broadcast requires a computation of $O\left(\left(N(M+1)\right)^{2N}\right)$ equivalent transition probabilities. This does not include the savings provided by inverting considerably smaller matrices.

Also, note the first heuristic is optimistic, disregarding the effect of nodes with better channels, while the second heuristic is pessimistic, concentrating the effect of all losses in one link. Since the N_i's increase as the probability of erasure increases, the 'Worst Link Channel' and 'Combined Erasure Effect' heuristics provide a lower and upper bound on the optimal values of $N_i, \forall i$, respectively.

Finally, it is important to emphasize that the N_i's do not need to be computed in real time. As in the case of a link, they can be pre-computed and stored in the receiver as look-up tables to reduce the computational load on the nodes. The nodes only have to choose the appropriate N_i's from the tables, considering channel conditions at the time of transmission.

2.2.5 Performance Evaluation

In this section, we extend the work in [39] to determine the mean completion time for optimal scheduling policies for broadcast. These policies consider no coding of the data packets, no channel state information, and nodes that only ACK when they have received all M data packets. As in [39], we restrict the analysis to independent symmetric channels, i.e., $Pe_1 = ... = Pe_N$, and no erasures in the ACKs for tractability. Note that we had no such restrictions in our network coding scheme. Our contribution includes 1) considering the effect of T_{rt}, T_p, and T_{ack}, and 2) the characterization for full duplex and TDD channels.

1) Broadcast with Round Robin in Full Duplex Channel (RR Full Duplex): The objective is to transmit M data packets to all users. Since the channels are independent and identically distributed over time and users, one of the optimal policies is Round Robin (RR). Thus, packet k in the block is transmitted every $(mM+k)T_p$ time units for $m = 0, 1, 2, ...$ until all the receivers get all M packets [39]. Using a

similar analysis as in [39],

$$E[T] = T_w + T_p M \left(\gamma + E[\max_{i,k} X_k^i] \right) \tag{2.60}$$

where $1 + X_k^i$ is the number of transmissions of packet k needed to reach node i, $\gamma \in (1/2, 1)$, and

$$E[\max_{i,k} X_k^i] = \sum_{t=1}^{\infty} \left[1 - (1 - Pe^t)^{MN} \right] \tag{2.61}$$

where $Pe = Pe_1 = ... = Pe_N$. Note that $\gamma = 1$ and $\gamma = 1/2$ give us an upper and lower bound on the mean completion time, respectively.

2) Broadcast with Round Robin in TDD (RR TDD): This scheme assumes limited feedback due to the TDD constraint. We assume that the transmitter broadcasts all M packets back-to-back, then stops to receive ACK packets that indicate completion of the entire file. If there are nodes that have not acknowledged the block of packets, the transmitter repeats the process, i.e., sends all M packets and stops to listen for ACKs. We can express the mean completion time of this scheme as

$$E[T] = (T_w + T_p M)\, E[\max_{i,k} X_k^i]. \tag{2.62}$$

2.2.6 Numerical Results

This Section provides numerical examples that compare the performance of our network coding scheme for broadcast in TDD channels, considering a satellite example. In particular, we compare the performance of the scheme when the N_i's are 1) chosen to minimize the mean completion time, 2) chosen using the 'Worst Link Channel' heuristics, and 3) chosen using the 'Combined Erasure Effect' heuristics. The comparison is carried out in terms of the mean completion time of M data packets under different packet erasure probabilities. We show that the 'Worst Link Channel' provides close-to-optimal performance with the advantage of reducing the computational load on the search algorithm. For simplicity, we consider that there are no erasures of ACK packets and that the distance between the transmitter and each receiver is

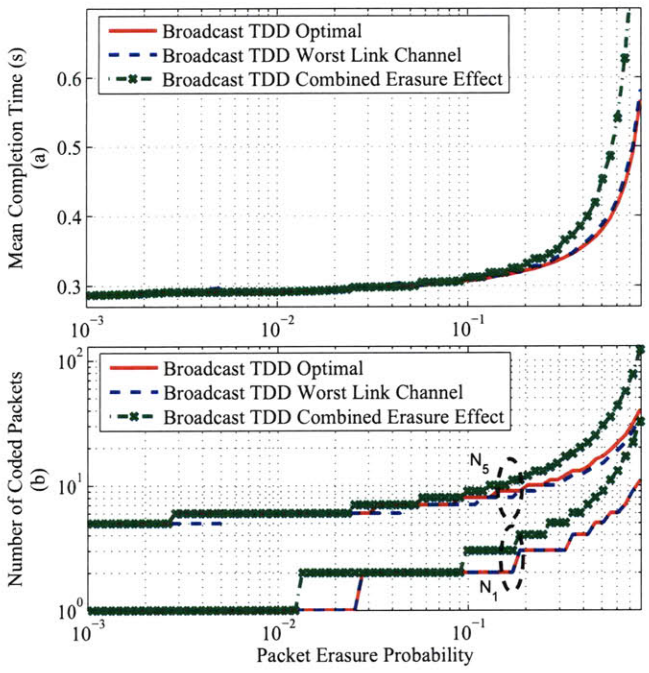

Figure 2-26: (a)Mean completion time and (b) number of coded packets N_5 and N_1, for the optimal choice of N_i's and two heuristics, for $N = 2$ receivers at the same distance from the transmitter, $M = 5$, packet erasure probability is the value for the two independent channels, $R = 1.5$ Mbps, $h = 80$ bits, $g = 20$ bits, $n_{ack} = 50$ bits

the same. The latter is a good approximation in many satellite scenarios. Finally, we compare our broadcast scheme with RR TDD and RR Full Duplex.

Figure 2-26 shows (a) the mean completion time and (b) number of coded packets N_5 and N_1, for the optimal choice of N_i's and our two heuristics when we have independent channels with a common packet erasure probability, i.e., $Pe_1 = Pe_2$. We consider data packets of size $n = 10,000$ bits in a GEO satellite link with a propagation delay of 125 ms, and the parameters specified in the figure.

Figure 2-26(a) illustrates that choosing N_i's using the 'Worst Link Channel' heuristics provides close-to-optimal performance in terms of mean completion time for a wide range of packet erasure probabilities. Although, the 'Combined Erasure Effect' heuristics provides a better estimate for low packet erasure probabilities in this case, its choice of N_i's for high packet erasures produces considerably higher completion times. This fact is explained because the 'Combined Erasure Effect' heuristic is pessimistic in terms of the amount of coded packets that are successfully received.

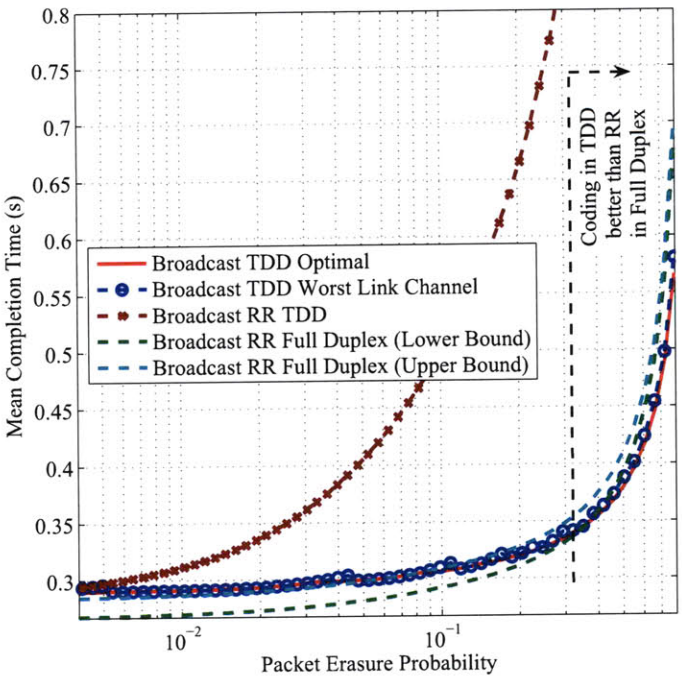

Figure 2-27: Mean completion time for the optimal choice of N_i's, 'Worst Link Channel' heuristic, and Round Robin Broadcast schemes with Full duplex and TDD channels. We use as parameters $N = 2$ receivers at the same distance from the transmitter, $M = 5$, packet erasure probability is the value for the two independent channels, $R = 1.5$ Mbps, $h = 80$ bits, $g = 20$ bits, $n_{ack} = 50$ bits

Figure 2-26(b) shows that the optimal choice of N_i's is bounded by the choices of N_i's using our two heuristics. As explained in Section III, this is not surprising because one of the heuristics is optimistic in its approximation ('Worst Link Channel') and the other is pessimistic ('Combined Erasure Effect'). This result is interesting at the time of developing an algorithm to search for the optimal value, because we could limit the search to the values given by the heuristics.

Since the choice of N_i's using the 'Worst Link Channel' heuristics provides a performance that is close to the optimal, we could use it as an initial choice of the N_i's so that a search algorithm finds the optimal N_i's, or we could use them directly. However, for large values of N and M a full search procedure might become exceedingly expensive in terms of computation time. Computing the N_i's using the 'Worst Link Channel' heuristics is easily performed even for large values of M, because it approximates the system as a link. We have shown some examples for cases of $M = 90$ and

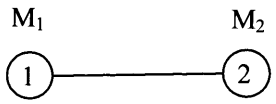

Figure 2-28: Nodes with disjoint information. Node i has M_i disjoint data packets (or independent random linear combinations of packets). Both nodes want to have all the information at the end of the exchange.

$M = 130$ in Section 2.1.9. In practice, using the heuristics provides a good trade-off between complexity and accuracy.

Figure 2-27 compares the performance of our Broadcast TDD scheme with N_i's computed optimally and with the 'Worst Link Channel' heuristics, and compares it to the performance of RR TDD and RR Full Duplex. First, note that for the range of packet erasures considered, our coding scheme performs at least as good as the RR TDD, and considerably better at high erasures. Second, the performance of our scheme is very close to that of the RR Full Duplex for low erasures. However, our coding scheme performs better at high packet erasures ($Pe_1 = Pe_2 > 0.3$), e.g., at $Pe_1 = Pe_2 = 0.8$ the RR Full Duplex scheme takes 20% more time to complete transmissions. Thus, we can perform better than a scheduling full duplex scheme, even with a single channel for data and feedback, i.e., half of the resources, by tailoring coding and feedback appropriately.

2.3 Random Linear Network Coding for All-to-all Broadcast in TDD channels

Previous sections have considered the case in which only one node has information to transmit, i.e., without considering that more than one node might need to transmit or share its information with others. We now analyze the problem of networks of N nodes in which all nodes want to share disjoint information under the TDD constrain. In a sense, we present an extension to the random linear network coding scheme. In fact, the problem studied so far constitutes a subset of the work presented in this Section.

We begin by studying the performance of a simple network with two nodes, in terms of the mean completion time. This is the simplest case of data sharing with more than one node transmitting information. This case provides useful insight to assess the general case of N nodes sharing information. We provide a simple algorithm to determine the number of coded data packets to be transmitted back-to-back before stopping for the case of two nodes. Finally, we extend the analysis and algorithm presented for the general case of N nodes.

2.3.1 The Case of Two Nodes

Model

Two nodes want to share information through a TDD channel, i.e., a channel in which nodes can transmit and receive, but not both at the same time. Each node i has M_i data packets or disjoint random linear combinations that he wants to share with the other node, as in Figure 2-28. A node i can transmit information at a given data rate R_i [bps] to the other node. We assume independent erasure channels for the data packets, where Pe_j represents the erasure probability of a coded packet sent from node j to the other node. Also, there is a propagation time T_{prop} associated to the time that elapses beteween a packet being transmitted and it being received at the other node.

Each node will act as both sender and receiver of information. When a node operates as the sender, it uses random linear network coding [4] to generate coded data packets. Each coded data packet contains a linear combination of the M data packets of n bits each, as well as the random coding coefficients used in the linear combination. Each coefficient is represented by g bits. For encoding over a field size q, we have that $g = \log_2 q$ bits. A coded packet is preceded by an information header of size h. Thus, the total number of bits per packet is $h + n + gM$. Figure 2-1 shows the structure of each coded packet considered in our scheme.

The node acting as a sender at some point can transmit coded packets back-to-back before stopping to wait for the other node to transmit its own packets and

acknowledge how many degrees of freedom (dofs) it still requires to have all the information. In general, there is no explicit ACK packet. The acknowledgement to node i comes piggybacked in the header of each coded packet sent from node j, unless no coded packet has to be sent. We assume that the acknowledgement suffers no erasures. We assume that the field size q is large enough so that the expected number of successfully received packets at the receiver, in order to decode the original data packets, is approximately M_i for transmissions from node i.

Description of Scheme

Transmission begins at one of the nodes, say node 1 with M_1 information packets, which are encoded into $N_{(M_1,M_2,0)} \geq M_1$ random linear coded packets, and transmitted as in Figure 2-29 (a). After receiving the coded packets, node 2 which has M_2 information packets, generates $N_{(i_1,M_2,1)} \geq M_2$ random linear coded packets and transmits them, as in Figure 2-29 (b). The state (i_1, i_2, t) represents the i_1 dofs required by node 2 to decode the information of node 1, the i_2 dofs required by node 1 to decode the information of node 2, and the node that is acting as transmitter t, with $t = 0, 1$ for nodes 1 and 2, respectively. Node 2 includes in the header of each coded packet an ACK of the dofs needed by 2 to decode the information that 1 is trying to send, i.e., i_1. If all M_1 packets are decoded successfully by node 2 ($i_1 = 0$), node 1 becomes a receiver and will send only an explicit ACK packet in the following rounds stating how many dofs it requires to decode, because node 1 does not need to send any more data. Otherwise, node 1 sends $N_{(i_1,i_2,0)}$ coded packets and piggybacks an ACK in each header informing node 2 about how many dofs are missing at node 1, i.e., i_2, as in Figure 2-29 (c). This process is repeated until all packets have been shared successfully by both nodes. We also assume that if node i has completed its transmission to another another j, and i gets enough information to decode the data from j, then node i can stop the transmission process, unless new information is available for transmission. These is a consequence of the ACKs suffering no erasures. We are interested in the optimal number $N_{(i_1,i_2,t)}$ of coded packets to be transmitted back-to-back.

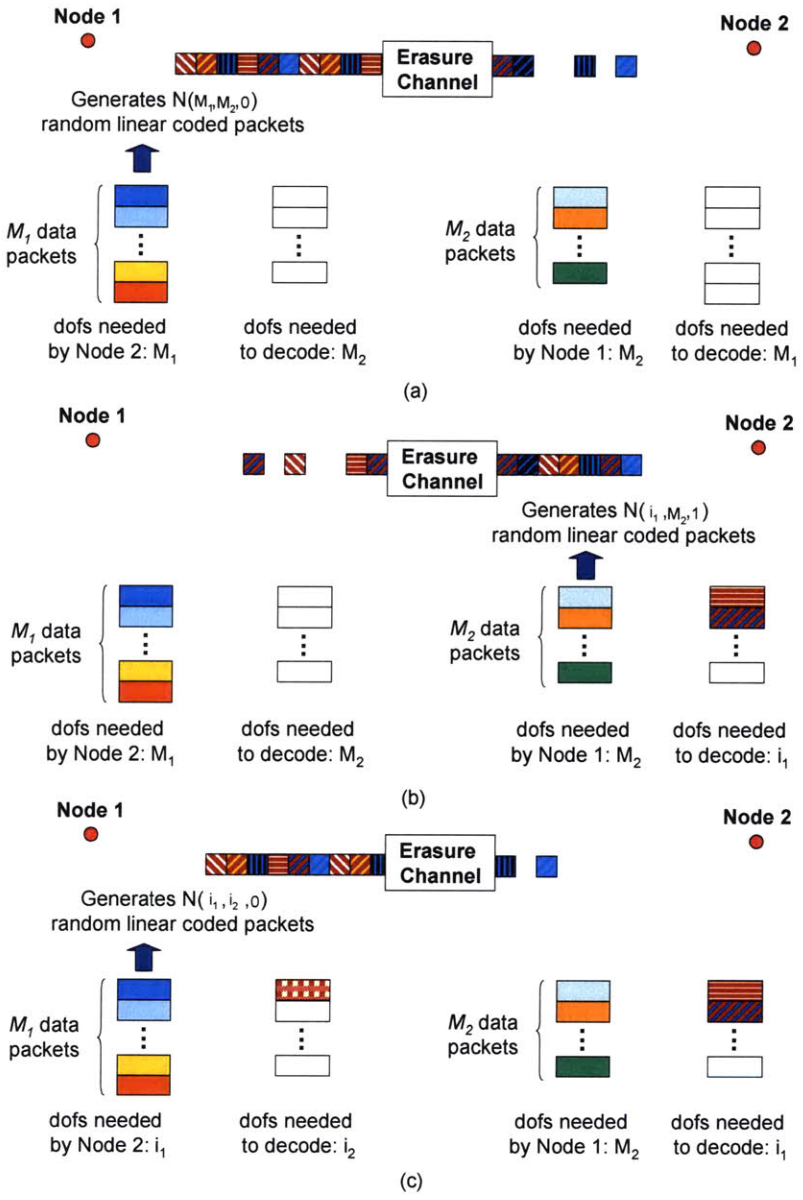

Figure 2-29: Network coding TDD scheme for sharing packets between two nodes. (a) Node 1 generates and sends $N_{(M_1,M_2,0)}$ coded packets from its own M_1 packets. (b) Node 2 generates and sends $N_{(i_1,M_2,1)}$ coded packets from its own M_2 packets and ACKs that that it needs i_1 dofs to decode the M_1 packets. (c) Node 1 updates its information. Node 1 generates and sends $N_{(i_1,i_2,0)}$ coded packets from its own M_1 packets and ACKs that it needs i_2 dofs to decode the M_2 packets.

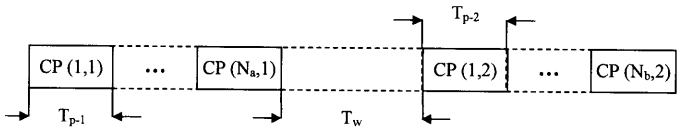

Figure 2-30: Network coding TDD scheme for sharing packets between two nodes. Note that the feedback comes piggybacked in the coded packets CP, and that $CP(\cdot, i)$ corresponds to a coded packet sent from node i.

Figure 2-30 illustrates the time window allocated to the system to transmit $N_{(i_1,i_2,1)}$ coded packets. Each coded packet $CP(1,1)$, $CP(2,1)$, etc. is of duration T_{p-1} and each coded packet $CP(1,2)$, $CP(2,2)$ is of duration T_{p-2}, for packets sent from nodes 1 and 2, respectively. The waiting time T_w is equivalent to the propagation time T_{prop} in this problem.

The process is modelled as a Markov chain. The states (s_1, s_2, t) are defined by the number of dofs required, s_k at receiver k, to successfully decode all packets, and the node t that acts as transmitter in this state. Thus, the states range from $(M_1, M_2, 1)$ to $(0,0,0)$. This is a Markov chain with $(M_1+1)(M_2+1)-2$ transient states and two recurrent states (state $(0,0,0)$ and $(0,0,1)$). Finally, note that a transition occurs every time that a batch of back-to-back coded packets is received at one receiver.

Figure 2-31 provides an example for a block size of 2 packets at each node. We have highlighted in this figure the absorbing states. Note that not all possible transitions from one state to the others have been included in this figure. However, any transient state $(s_1, s_2, 0)$ can only have transitions to a state $(s_1', s_2', 1)$. Similarly, any transient state $(s_1, s_2, 1)$ can only have transitions to a state $(s_1', s_2', 0)$. In Figure 2-31, this translates into no transitions from transient dashed states to other dashed states, or from transient solid-line states to other solid-line states. This observation is crucial in the development of an algorithm to compute the values of $N_{(s_1,s_2,t)}$.

The transition probabilities from state (s_1, s_2, t) to state (s_1', s_2', t') are

$$P_{(i,j,t)\to(i',j',t')} = P\left(X_1(n)=i', X_2(n)=j', T(n)=t' \,|\, X_1(n-1)=i, X_2(n-1)=j, T(n-1)=t\right)$$

where $X_i(n)$ is the number of dofs required at receiver i at the end of transmission

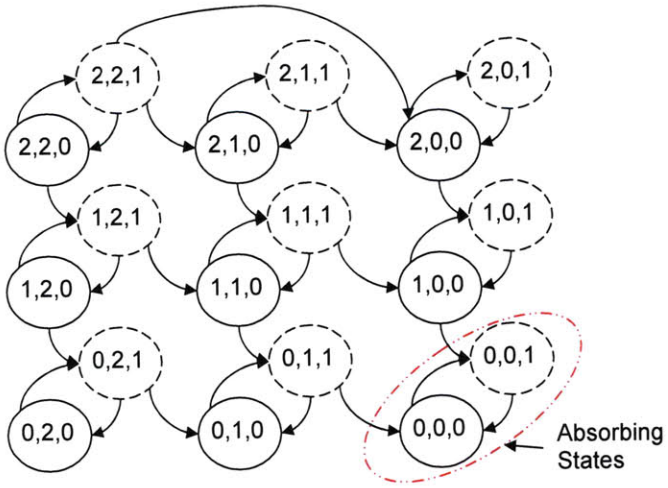

Figure 2-31: Example of the Markov chain for a block size of $M_1 = M_2 = 2$.

n, and $T(n)$ is the designated transmitter at time n.

Given the characteristics of the Markov chain, we have that

$$P_{(s_1,s_2,t)\to(s_1',s_2',t')} = \begin{cases} P\left(s_1'|s_1,N_{(s_1,s_2,t)}\right) & \text{if , } s_2 = s_2',\ t = 0,\ t' = 1 \\ P\left(s_2'|s_2,N_{(s_1,s_2,t)}\right) & \text{if , } s_1 = s_1',\ t = 1,\ t' = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.63}$$

where $N_{(s_1,s_2,t)}$ represents the number of coded packets sent to produce the transition. Note that $P\left(s_j'|s_j,N_{(s_i,s_j,t)}\right)$ have a similar structure to the transition probabilities studied in the link case. The main difference is that the value of the number of coded packets sent back-to-back is no longer associated with the starting state of a particular node, but with a value determined from the state of the system, i.e., $N_{(s_i,s_j,t)}$, which is redolent of the one-to-all broadcast case.

For $0 < s_j' < s_j$, we use

$$f(s_j, s_j') = \begin{cases} \binom{N_{(s_1,s_2,t)}}{s_j - s_j'} & \text{if } N_{(s_1,s_2,t)} \geq s_j, \\ 0 & \text{otherwise} \end{cases} \tag{2.64}$$

to determine the transition probability

$$P\left(s'_j|s_j,N_{(s_1,s_2,t)}\right) = f(s_j,s'_j)(1-Pe_j)^{s_j-s'_j}Pe_j{}^{N_{(s_1,s_2,t)}-s_j+s'_j}. \tag{2.65}$$

For $s_j = s'_j > 0$ the expression for the transition probability reduces to

$$P\left(s_j|s_j,N_{(s_1,s_2,t)}\right) = Pe_j{}^{N_{(s_1,s_2,t)}}. \tag{2.66}$$

Note that for $P\left(0|0,N_{(s_1,s_2,t)}\right) = 1$. Finally, for $s'_j = 0$

$$P\left(s'_j=0|s_j,N_{(s_1,s_2,t)}\right) = 1 - \sum_{s'_j=1}^{s_j} P\left(s'_j|s_j,N_{(s_1,s_2,t)}\right). \tag{2.67}$$

Mean Time for Completing Transmission

The mean time for completing the sharing process of all data packets between the nodes constitutes the mean time of absorption, i.e., the time to reach states $(0,0,0)$ or $(0,0,1)$ for the first time, given that the initial state is either $(M_1,M_2,0)$ or $(M_1,M_2,1)$, depending on which node starts transmitting. This can be expressed in terms of the mean time for completing the transmission given that the Markov Chain is in state $(s_1,,s_2,t)$ is $T_{(s_1,s_2,t)}$, $\forall s_1 = 0,1,..M_1, \forall s_2 = 0,1,..M_2, \forall t = 0,1$. For our scheme, $T_{p-i} = \frac{h+n+gM_i}{R_i}$, and the waiting time T_w shown in Figure 2-30 is $T_w = T_{prop}$.

Let us define $T^{(i,j,t)}$ as the time it takes to transmit $N_{(i,j,t)}$ coded data packets and reach the other node. It is easy to show that $T^{(i,j,t)} = N_{(i,j,t)}T_{p-t} + T_w$. The mean completion time when the system is in state (i,j,t) is given by

$$T_{(i,j,t)} = T^{(i,j,t)} + \sum_{(i',j',t')} P_{(i,j,t)\to(i',j',t')}T_{(i',j',t')}. \tag{2.68}$$

We can express this in vector form as $\bar{T} = \bar{\mu} + P\bar{T}$, where $\bar{T} = [T_{(i,j,t)}]$, $\bar{\mu} = [T^(i,j,t))]$ and P is the corresponding transition probability. Thus, the mean completion time starting as every state is given by $\bar{T} = [I-P]^{-1}\bar{\mu}$. Since we are interested

in the mean completion time when we start at states $(M_1, M_2, 0)$ or $(M_1, M_2, 1)$, we can use Cramer's rule as

$$T_{(M_1,M_2,t)} = \frac{\det\left(\Gamma \leftarrow_{(M_1,M_2,t)} \bar{\mu}\right)}{\det\left(\Gamma\right)} \tag{2.69}$$

where $\Gamma = I - P$, and the notation $\Gamma \leftarrow_{(M_1,M_2,t)} \bar{\mu}$ represents a matrix that has all columns as the Γ matrix except the column corresponding to state (M_1, M_2, t) which is substituted by the vector $\bar{\mu}$, t can take value 0 or 1.

Minimizing The Mean Completion Time

The expected time for each state depends on all the expected times for the previous states. However, optimizing the values of all $N_{(i,j,t)}$ is not as straightforward as the recursive method used in the case of a link. Also, note that there are $2(M_1+1)(M_2+1)$ states in our Markov chain, and that we have $2(M_1+1)(M_2+1)-2$ integer variables that we need to optimize.

Let us consider exploiting the structure of the problem to determine an algorithm to estimate the values of $N_{(i,j,t)}, \forall i = 1, ..., M_1, \forall j = 1, ..., M_2, \forall t = 0, 1$. Let us express the mean completion time for states with $t = 0$ as

$$T_{(i,j,0)} = T^{(i,j,0)} + \sum_{i'} P_{(i,j,0)\rightarrow(i',j,1)} T_{(i',j,1)} \tag{2.70}$$

and the mean completion time for states with $t = 1$ as

$$T_{(i,j,1)} = T^{(i,j,1)} + \sum_{j'} P_{(i,j,1)\rightarrow(i,j',0)} T_{(i,j',0)}. \tag{2.71}$$

Notice that the mean completion time for any state of the form $(i, j, 0)$ depends on the mean completion time of states of the form $(i, j, 1)$ but not on states of the form $(i, j, 0)$. Thus, we can substitute equation (2.71) into (2.70), and use the fact

that $T^{(i,j,t)} = N_{(i,j,t)}T_{p-t} + T_{prop}$ to obtain

$$T_{(i,j,0)} = N_{(i,j,0)}T_{p-0} + 2T_{prop} + T_{p-1}\left[\sum_{i'} N_{(i',j,1)}P_{(i,j,0)\to(i',j,1)}\right] \quad (2.72)$$

$$+ \sum_{i',j'} T_{(i',j',0)}P_{(i,j,0)\to(i',j,1)}P_{(i',j,1)\to(i',j',0)}. \quad (2.73)$$

A similar expression can be found for $T_{(i,j,1)}$.

Note that these expressions are redolent of the mean completion time for a link. In fact, we have shown that the cost for transitioning from the current state to other states was of the form $N_iT_p+2T_{prop}+T_{ack}$. In expression (2.73), we have a similar cost with some changes, namely there is no cost for transmitting an ACK packet because the ACKs are piggybacked in the coded packets. Also, we have an additional term, i.e., $T_{p-1}\left[\sum_{i'} N_{(i',j,1)}P_{(i,j,0)\to(i,j,1)}\right]$, which represents the mean additional waiting time for one transmitter due to the transmission of the other node.

Let us define $\hat{N}_{(i,j,t)}(n)$ as the estimate for $N_{(i,j,t)}$ at step n of the algorithm, and $P_{(i,j,0)\to(i',j,1)}(n)$ and $P_{(i,j,0)\to(i,j',1)}(n)$ are the transition probabilities based on the estimates $\hat{N}_{(i,j,t)}(n)$ for the n-th step.

Algorithm 1. *Search algorithm for case of two nodes*

- *STEP 1: INITIALIZE*

  -Set $\hat{N}_{(i,j,1)}(0) = j$ *and* $\hat{N}_{(i,j,0)}(0) = i$.

  -Set $n = 1$.

- *STEP 2: TRANSMISSION FROM NODE 1 TO NODE 2:*

  FOR $j' = 1, 2, ..., M_2$

  Compute $\hat{N}_{(i,j',0)}(n), \forall i = 1, ..., M_1$ *to minimize the completion time of a TDD link, with transition cost*

$$\hat{N}_{(i,j',0)}(n)T_{p-0} + 2T_{prop} + T_{p-1}\left[\sum_{i'} \hat{N}_{(i',j',1)}(n-1)P_{(i,j',0)\to(i',j',1)}(n)\right] \quad (2.74)$$

  END FOR

- *STEP 3: TRANSMISSION FROM NODE 2 TO NODE 1:*

  FOR $i' = 1, 2, ..., M_1$

  Compute $\hat{N}_{(i',j,1)}(n), \forall j = 1, ..., M_1$ *to minimize the completion time of a TDD link, with transition cost*

$$\hat{N}_{(i',j,1)}(n)T_{p-0} + 2T_{prop} + T_{p-1}\left[\sum_{j'} \hat{N}_{(i',j',0)}(n)P_{(i',j,0)\rightarrow(i',j',0)}(n)\right] \qquad (2.75)$$

  END FOR

- *STOPPING CRITERIA:*

  IF $\hat{N}_{(i,j,t)}(n) = \hat{N}_{(i,j,t)}(n-1)$, $\forall i, j, t$

  Stop

  ELSE

  $n = n + 1$, *and go to Step 2.*

  END IF

The proposed algorithm computes the $N_{(i,j,t)}$ by using the search algorithm for a link with the appropriate costs. The algorithm is iterative and has two phases. The first one tries to solve the problem of a link for the case in which the first transmitter is operating. This means that we are looking to optimize the variables $N_{(i,j,0)}, \forall i$ assuming the variables $N_{(i',j,1)}$ to be fixed and j also fixed. We repeat the process for every value of j. After finding the optimal values for $N_{(i,j,0)}, \forall i, j$, the algorithm proceeds to the second phase, which is to compute the optimal values of $N_{(i,j,1)}$ given the new values of $N_{(i,j,0)}$, keeping these last values fixed. The algorithm stops when $N_{(i,j,t)}$ becomes stable, i.e., when the current iteration provides the same result as the previous iteration.

Let us emphasize that the $N_{(i,j,t)}$'s do not need to be computed in real time. Again, they can be pre-computed and stored in the receiver as look-up tables. Thus, the computational load on the nodes is minimal, because they only have to choose

the appropriate $N_{(i,j,t)}$'s from the tables considering channel conditions at the time of transmission.

2.3.2 The Case of N nodes

The problem for N nodes constitutes a natural and simple extension from the case of two nodes. Let us formalize the model and notation before starting our analysis.

Model

Let us assume that each node i has M_i data packets or disjoint random linear combinations that it wants to share with the other node. We assume that the nodes transmit following a round robin assignment, where the order of transmission has been predefined. Each node i can transmit information at a given data rate R_i [bps] to the other nodes. We assume a memoryless packet erasure channel $Pe_{i,j}$ for transmissions from node i to node j, and that the channels are independent. We also assume that each transmission from a node can be received by each of the other nodes. Finally, we assume that the next transmitter node will wait for all nodes to receive the previous information. For later analysis, let us define T_{prop-i} as the propagation time from node i to the node that is farthest from it.

Again, the process is modelled as a Markov chain. We consider that the states of the Markov chain are given by $((s_{1,2},..,s_{1,N}),(s_{2,1},s_{2,3},..,s_{2,N}),..,(s_{N,1},..,s_{N,N-1}),t)$, where $s_{a,b}$ constitutes the number of dofs required by node b to successfully decode all packets from node a, and the node t that acts as transmitter in this state. Note that $(s_{a,1},...,s_{a,N})$ represents the degrees of freedom that other nodes require from node a in order to decode his information. In order to simplify notation, let us define $S = ((s_{1,2},..,s_{1,N}),(s_{2,1},s_{2,3},..,s_{2,N}),..,(s_{N,1},..,s_{N,N-1}))$, so that (S,t) represents a state of the Markov chain, and $S_a = (s_{a,1},s_{a,2},..,s_{a,N})$ being the state of the receivers of node a. This is a Markov chain with $N(M_1+1)^{N-1}(M_2+1)^{N-1}..(M_N+1)^{N-1}-N$ transient states and N recurrent states. Finally, note that a transition occurs every time that a batch of back-to-back coded packets is received at one receiver.

The number of variables to be optimized in order to provide an optimal solution increases exponentially with the number of nodes N, because we would have to consider a variable per state, i.e., $N_{((S_1,S_2,...,S_N),t)}, \forall S_i, t$. However, we can use a similar approach to the case of one-to-all broadcast to reduce the number of variables, i.e., consider only the maximun degrees of freedom $s_a = \max_b s_{a,b}$ that the receivers of a need in order to completely decode the information. This reduces the number of variables to optimize to $N(M_1+1)(M_2+1)...(M_N+1)-N$ and we will rename them $N_{((s_1,s_2,...,s_N),t)}$, which represent the number of coded packets to send. Given the characteristics of the Markov chain, the transition probabilities from state (S,t) to state (S',t') are

$$P_{(S,t)\to(S',t')} = \begin{cases} P\left(S'_a|S_a,N_{((s_1,...,s_N),t)}\right) & \text{if,} S_b = S'_b, \forall b \neq a, \\ & t = a,\ t' = t_{next}(a),\ \forall a \\ 0 & \text{otherwise} \end{cases}$$

where $t_{next}(a)$ represents the next node that should transmit after node a has transmitted, and $P\left(S'_a|S_a,N_{((s_1,...,s_N),t)}\right)$ is the probability of transitioning from S_a to S'_a when node a has transmitted $N_{((s_1,...,s_N),t)}$ coded packets. Assuming that a transmission from any node to the other $N-1$ nodes goes through independent channels, we have that

$$P\left(S'_a|S_a,N_{((s_1,...,s_N),t=a)}\right) = \prod_{j\neq a} P\left(s'_{a,j}|s_{a,j},N_{((s_1,...,s_N),t=a)}\right) \tag{2.76}$$

where $P\left(s'_{a,j}|s_{a,j},N_{((s_1,...,s_N),t)}\right)$ has the same distribution studied for the case of two nodes, and represents the transition probability related to the knowledge of node j with respect to the data node a has, when node a sends $N_{((s_1,...,s_N),t)}$ coded packets. For ease of notation, we will substitute $N_{((s_1,...,s_N),t)}$ for Nt. For $0 < s_{a,j'} < s_{a,j}$, this can be translated into

$$P\left(s'_{a,j}|s_{a,j},Nt\right) = f(s_{a,j},s_{a,j'})(1-Pe_{a,j})^{s_{a,j}-s_{a,j'}} {Pe_{a,j}}^{Nt-s_{a,j}+s_{a,j'}}. \tag{2.77}$$

For the case of $s_{a,j} = s_{a,j'} > 0$ the expression for the transition probability reduces to $P\left(s_{a,j}|s_{a,j},Nt\right) = Pe_{a,j}{}^{Nt}$. Note that for $P\left(0|0,Nt\right) = 1$. Finally, for $s'_j = 0$ $P\left(s_{a,j'=0}|s_{a,j},Nt\right) = 1 - \sum_{s_{a,j'=1}}^{s_{a,j}} P\left(s_{a,j'}|s_{a,j},Nt\right)$.

We can define P as the transition probability for our system.

Mean Time for Completing Transmission

Similar to previous cases, the mean time for completing the sharing process of all data packets between the nodes constitutes the mean time of absorption, i.e., the time to reach any state $((0,...0),...,(0,...0),t)$ for some t for the first time, given that the initial state is $((M_1,...,M_1),...,(M_N,...,M_N),t)$ for some t, depending on which node starts transmitting. This can be expressed in terms of the expected time for completing the transmission given that the Markov chain is in state (S,t), $T_{(S,t)}$, $\forall S \forall t$. For our scheme, we consider that the transmission time of a packet from node i is given by $T_{p-i} = \frac{h+n+gM_i}{R_i}$.

Let us define $T^{(S,t)}$ as the time it takes to transmit $N_{(s_1,...,s_N,t)}$ coded data packets and reach the node that is farthest away from t. It is easy to show that $T^{(S,t)} = N_{(s_1,...,s_N,t)}T_{p-t} + T_{prop-t}$.

The mean completion time when the system is in state (S,t) is given by

$$T_{(S,t)} = T^{(S,t)} + \sum_{(S',t')} P_{(S,t)\to(S',t')}T_{(S',t')} \tag{2.78}$$

which can be expressed in vector form as $\bar{T} = [I - P]^{-1}\bar{\mu}$, where $\bar{T} = [T_{(S,t)}]$, $\bar{\mu} = [T^{(S,t)}]$, and P is the corresponding transition probability. Since we are interested in the mean completion time when we start at states $((M_1,...,M1,...,(M_N,...,M_N),t)$ for some t, we can use Cramer's rule.

Minimizing the Mean Completion Time

Note that even after reducing the number of variables to optimize, the optimization becomes computationally prohibitive because the number of states in the Markov chain increases exponentially with the number of nodes. However, we can compute

the variables of interest by using a slightly modified version of the algorithm for two nodes. This new algorithm uses heuristics to obtain good estimates of the variables while reducing computation. Since we have assumed that only one node transmits at each time and that this transmitter will broadcast the information to all other nodes, we can use similar heuristics to those presented for the case of broadcast. These heuristics rely on computing the number of coded packets to be transmitted from a particular node by reducing the problem of broadcast from one node to all other nodes to that of a link. The equivalent packet erasure probability of the link constitutes a function of the packet erasure probabilities of the different channels in broadcast. This approximation will allow us to use a similar algorithm to that proposed for the case of two nodes, with only slight modifications.

The heuristic that showed best performance in the case of one-to-all broadcast was the 'Worst Link Channel' heuristic. This heuristic approximates the system as a link to the receiver with the worst channel, i.e., the worst packet erasure probability. Then, we compute $N_{(s_1,\ldots,s_N),t}$, for any s_i and for any t to minimize the mean completion time as in the case of a link with similar modifications to that in the algorithm for two nodes, namely using as round trip time that depends on both the physical round trip time and the transmissions of other nodes in the system.

Note that using a similar procedure to that of subsection 2.3.1, we can obtain that

$$\begin{aligned} T_{(S,t_1)} &= N_{(s_1,..,s_N,t_1)} T_{p-t_1} + \sum_{b=1}^{N} (T_{prop-t_b}) \\ &+ \sum_{n=2,..,N} T_{p-t_n} E_{t_2,..,t_n} \left[N_{(i_1,..,i_N,t_n)} | (S,t_1) \right] \\ &+ \sum_{S',S^{(2)},..,S^{(N)}} T_{(S',t_1)} \mathbf{P}_{(S,t_1)\to(S',t_1)} \end{aligned}$$

where

$$\mathbf{P}_{(S,t_1)\to(S',t_1)} = P_{(S,t_1)\to(S^{(2)},t_2)} P_{(S^{(2)},t_2)\to(S^{(3)},t_3)} \cdots P_{(S^{(N)},t_N)\to(S',t_1)}, \tag{2.79}$$

and

$$E_{t_2,\ldots,t_n}\left[N_{(i_1,\ldots,i_N,t_n)}|(S,t_1)\right] = \sum_{S^{(2)},\ldots,S^{(n)}} N_{(S^{(n)},t_n)}\mathbf{P}_{(S,t_1)\to(S^{(n)},t_n)} \tag{2.80}$$

where

$$\mathbf{P}_{(S,t_1)\to(S^{(n)},t_n)} = P_{(S,t_1)\to(S^{(2)},t_2)}\ldots P_{(S^{(n-1)},t_{n-1})\to(S^{(n)},t_n)}. \tag{2.81}$$

Let us define $\hat{N}_{(s_1,s_2,\ldots,s_N,t_k)}(n)$ as the estimate for $N_{(s_1,s_2,\ldots,s_N,t_k)}$ at step n of the algorithm. Then, our algorithm can be written as follows

Algorithm 2. *Search Algorithm for N nodes*

- *STEP 0: INITIALIZE*

  -Set $\hat{N}_{(s_1,s_2,\ldots,s_N,t_k)}(0) = s_k,\ \forall k.$

  -Set $n = 1.$

- *STEP 1: TRANSMISSION FROM NODE 1:*

  -Set $\hat{N}_{(s_1,s_2,\ldots,s_N,t_k)}(n) = \hat{N}_{(s_1,s_2,\ldots,s_N,t_k)}(n-1),\ \forall k.$

  FOR $s_2' = 1, 2, \ldots, M_2$

  ...

  FOR $s_N' = 1, 2, \ldots, M_N$

  Compute $\hat{N}_{(s_1,s_2',\ldots,s_N',t_1)}(n), \forall s_1 = 1, \ldots, M_1$ *to minimize the completion time of a TDD link with* $Pe = \max_b Pe_{1,b}$, *and with transition cost*

$$\hat{N}_{(s_1,s_2',\ldots,s_N',t_1)}(n)T_{p-1} + \sum_{b=1}^{N}(T_{prop-t_b}) + \sum_{y=2,\ldots,N} T_{p-t_y}E_{t_2,\ldots,t_y}\left[\hat{N}_{(i_1,\ldots,i_N,t_y)}(n)|(S,t_1)\right] \tag{2.82}$$

  where $S = (s_1, s_2', \ldots, s_N').$

  END FOR

  ...

END FOR

- *STEP k (=2,...,N): TRANSMISSION FROM NODE k TO NEXT NODE:*

  FOR $s'_1 = 1, 2, ..., M_1$

  ...

  FOR $s'_N = 1, 2, ..., M_N$

  Compute $\hat{N}_{(s'_1,...,s_k,...,s'_N,t_k)}(n), \forall s_k = 1, ..., M_k$ *to minimize the completion time of a TDD link with* $Pe = \max_b Pe_{k,b}$, *and with transition cost*

$$\begin{aligned} &\hat{N}_{(s'_1,...,s'_N,t_k)}(n)T_{p-k} + \textstyle\sum_{b=1}^{N}(T_{prop-t_b}) \\ &+ \textstyle\sum_{y=k+1,..,N,1,..k-1} T_{p-t_y} E_{t_k,..,t_y}\left[\hat{N}_{(i_1,..,t_y)}(n)|(S,t_k)\right] \end{aligned} \tag{2.83}$$

  where $S = (s'_1, ..., s_k, ..., s'_N)$.

  END FOR

  ...

  END FOR

- *STEP N+1: STOPPING CRITERIA*

  IF $\hat{N}_{(s_1,...,s_N,t)}(n) = \hat{N}_{(s_1,...,s_N,t)}(n)(n-1)$, $\forall s_1, ..., s_N, t$

  Stop

  ELSE

  $n = n + 1$, *and go to Step 1.*

  END IF

2.3.3 Performance Evaluation

In this Section, we present two full duplex schemes as a basis for comparison for the case of two nodes. First, we consider a full duplex network coding scheme that minimizes the mean completion time of the sharing process. Secondly, we extend the

work of [39], which deals with broadcast, to determine the mean completion time for a round robin scheduling policy for sharing information between two nodes. This policy considers no coding of the data packets, no channel state information, and nodes that only ACK when they have received all information.

We consider that both schemes send the ACK in the header of the coded data packets that each node sends. As in [39], we restrict the analysis to independent symmetric channels, i.e., $Pe_1 = Pe_2$ and $T_{p-0} = T_{p-1} = T_p$, and no erasures in the ACKs for tractability. Note that we had no such restrictions in our TDD network coding scheme and will not have it for the full duplex network coding scheme. Our contribution to the work of broadcast scheduling policies in [39] includes 1) considering the effect of T_{prop}, T_p, 2) the characterization of a full duplex channel, and 3) resetting the analysis to match the problem of sharing information between two nodes.

1) Data Sharing with Network Coding in Full Duplex Channel (DSNC Full Duplex): Each node generates random linear combinations of its original M_i data packets, and sends those coded packets back-to-back through the channel to the other node. We assume that both nodes start transmitting at the same time to reduce the completion time of the sharing process. This problem can be modelled through a Markov chain with states (i, j), where i and j represent the dofs required by the to decode at node 1 and 2, respectively. Since both nodes start transmitting at the same time, and we assume the packets to take the same time to be transmitted T_p, then transitions occur every arrival of a coded packet. The transition probabilities are modeled as $P_{(i,j)\to(i',j')} = P_{i\to i'} P_{j\to j'}$ where we assume independence of the channels. Note that

$$P_{i\to i'} = \begin{cases} Pe & \text{if } i = i' \neq 0, \\ 1 & \text{if } i = i' = 0, \\ 1 - Pe & \text{if } i = i' + 1, \\ 0 & \text{otherwise} \end{cases} \tag{2.84}$$

where $M \geq i, i' \geq 0$.

Using a similar procedure as in previous sections, we can express the mean number of coded packets from each node to complete the sharing process in vector form as $\bar{T} = \bar{1} + P\bar{T}$, where $\bar{T} = [T_{(i,j)}]$ is the vector of mean completion times starting at each state (i,j), $\bar{1} = [1]$ is a vector of all ones, and P is the corresponding transition probability. Thus, the mean completion time for the sharing process is

$$E[T] = T_w + T_p \frac{\det\left(\Gamma \leftarrow_{(M,M)} \bar{1}\right)}{\det\left(\Gamma\right)} \tag{2.85}$$

where $T_w = 2T_{prop} + T_{delay}$, $T_{delay} = T_h + T_p\left(1 - \mathtt{R}\left(T_{prop}, T_p\right)\right)$, $T_h = h/R$. Note that T_{delay} represents the delay to send the ACK because it is piggybacked to the header of the coded packets. Note that the function $\mathtt{R}(x,y)$ returns the remainder of x/y.

2) Round Robin Data Sharing in Full Duplex Channel (DSRR Full Duplex): The objective is to transmit M_1 data packets from node 1 to node 2, and M_2 data packets from node 2 to node 1. We consider the simpler problem when $M_1 = M_2 = M$. We assume that both nodes start transmitting at the same time. Note that packet k in the block of each node is transmitted every $(mM + k)T_p$ time units for $m = 0, 1, 2, ...$ until the other nodes gets all M packets. The sharing process is completed when both nodes have received all information. Note that this problem is very similar to that in [39]. Using a similar analysis,

$$E[T] = T_w + T_p M\left(\gamma + E[\max_{i,k} X_k^i]\right) \tag{2.86}$$

where $1 + X_k^i$ is the number of transmissions of packet k needed to reach node i from the other node, $\gamma \in (1/2, 1)$, $T_w = 2T_{prop} + T_{delay}$,$T_{delay} = T_h + T_p\left(1 - \mathtt{R}\left(T_{prop}, T_p\right)\right)$, $T_h = h/R$, and

$$E[\max_{i,k} X_k^i] = \sum_{t=1}^{\infty}\left[1 - \left(1 - Pe^t\right)^{2M}\right]. \tag{2.87}$$

Note that $\gamma = 1$ and $\gamma = 1/2$ give us an upper and lower bound on the mean completion time, respectively.

3) Round Robin Data Sharing in TDD Channel: This approach is similar to DSRR Full Duplex but considering a TDD channel. Again, the objective is to transmit M_1

data packets from node 1 to node 2, and M_2 data packets from node 2 to node 1. We consider the simpler problem when $M_1 = M_2 = M$ and that each transmitter sends all M packets before stopping to listen for a transmission of the other. The data packets also contain feedback indicating if the node should keep transmitting or if all packets have been received successfully at the other node. Using a similar analysis, the mean completion time $E[T]$ is bounded by

$$E[T] \leq (T_w + 2T_pM)\left(1 + E[\max_{i,k} X_k^i]\right) \tag{2.88}$$

and

$$E[T] \geq (T_w + T_pM)\left(1 + E[\max_{i,k} X_k^i]\right) + T_pM \tag{2.89}$$

using the same definitions as in DSRR Full Duplex.

2.3.4 Numerical Results

This section provides numerical results that compare the performance of our network coding scheme for sharing disjoint information between two nodes in TDD channels. We consider a GEO satellite example where the propagation time $T_{prop} = 125$ ms, and data packets of size $n = 10,000$ bits. We assume symmetric uplink and downlink channels, i.e., $Pe_1 = Pe_2 = Pe$ and $R_1 = R_2 = R = 1.5$ Mbps. We compare the performance of the scheme in terms of mean completion time when the $N_{(i,j,t)}$'s are chosen to minimize the mean completion time using the proposed search algorithm under different packet erasure probabilities Pe. We consider that both nodes have he same number of packets at the start of the process, which means that $M_1 = M_2 = M$ data packets. We show that our TDD scheme can outperform a full duplex round robin scheduling scheme for large Pe. Also, our TDD scheme performs at most 3 dB above that of a full duplex network coding scheme. For small Pe, the difference between our scheme and the full duplex network coding is much less than 3 dB. Finally, we show that the number of iterations required for our algorithm to convergence is very small for a wide range of Pe.

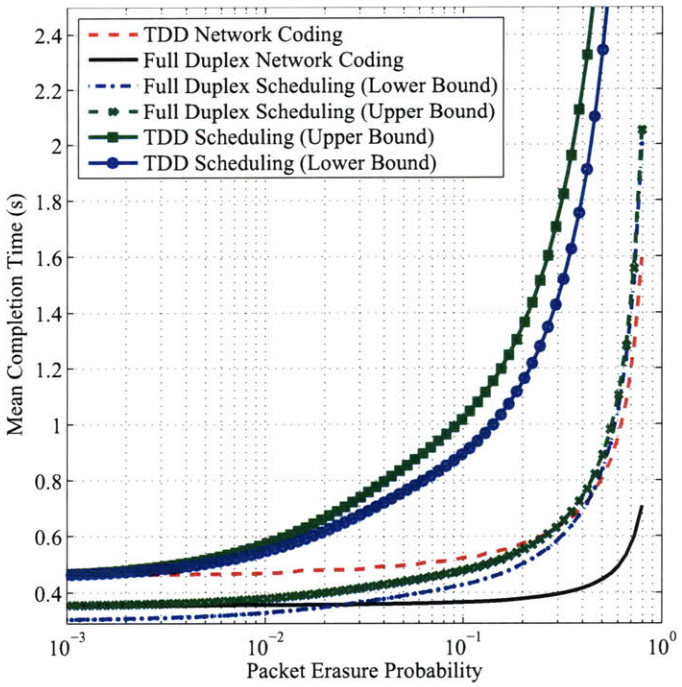

Figure 2-32: Mean completion time for the TDD scheme choosing the $N_{(i,j,t)}$'s through the search algorithm proposed in Section 2.3.1, two full duplex schemes, and a TDD scheme with no coding. We use the following parameters $R = 1.5$ Mbps, $h = 80$ bits, $g = 20$ bits, a block size per node of $M = 15$.

Figure 2-32 shows the mean completion time for the TDD scheme choosing the $N_{(i,j,t)}$'s through the search algorithm proposed in Section 2.3.1, two full duplex schemes, and a TDD scheme with no coding.

Figure 2-32 illustrates that choosing $N_{(i,j,t)}$'s using the search algorithm provides very good performance in terms of mean completion time for a wide range of packet erasure probabilities, when compared to full duplex schemes and the TDD scheme with no coding. In general, our scheme outperforms the TDD with no coding. This difference is most noticeable at large Pe, but even at moderate schemes there is a clear advantage of our scheme. Also, note that at low packet erasure probabilities, our TDD scheme is only 1 dB away from the performance of the DSNC full duplex scheme, which is the optimal scheme in terms of mean completion time. Since we have packets to be transmitted from both nodes, we expected the difference between these two schemes to be around 3 dB, i.e., twice the completion time for the TDD scheme because it has half of the channels that the network coding full duplex scheme has.

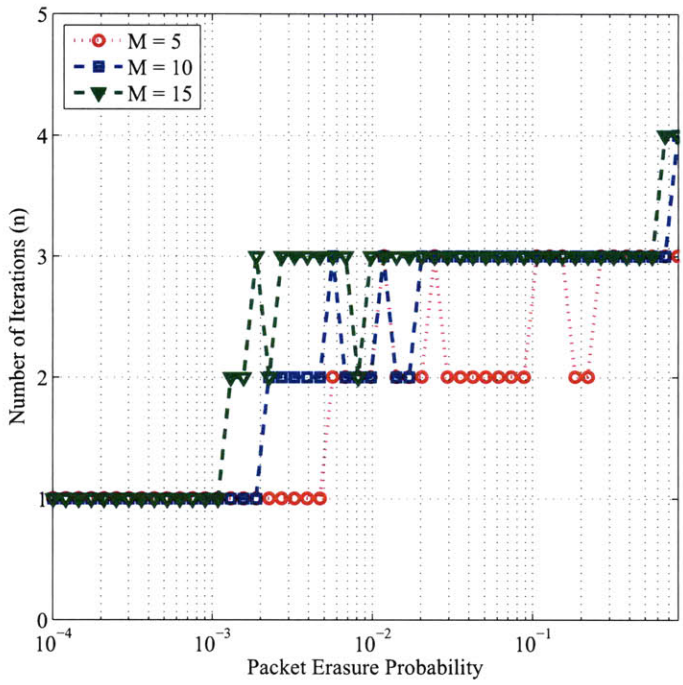

Figure 2-33: Number of iterations of the algorithm to reach a stable solution under different packet erasure probabilities. We use the following parameters $R = 1.5$ Mbps, $h = 80$ bits, $g = 20$ bits.

The main reason for this improvement is related to the relatively small number of coded packets that are being transmitted. Thus, the main cost in the completion time is the T_{prop} of the packets. For very high packet erasure probabilities, our TDD scheme is closer to the expected 3 dB. In this case, the number of coded packets transmitted in order to decode the information is the dominant cost to the completion time.

Figure 2-32 also shows that for high packet erasures, $Pe \geq 0.4$, our TDD scheme using the proposed search algorithm to choose the $N_{(i,j,t)}$'s, outperforms a full duplex scheduling scheme (SDRR full duplex). This is clear because the lower bound on the mean completion time of the SDRR full duplex exceeds the mean completion time of our scheme at around $Pe \geq 0.4$. It is possible that we could outperform SDRR full duplex with our TDD scheme even for $Pe > 0.2$, which is the point at which the upper bound of SDRR full duplex intersects with the mean completion time of our scheme. Note that for $Pe = 0.8$ our TDD scheme outperforms SDRR full duplex by about 1.1 dB. Thus, even with a single channel for data and feedback, i.e., half of the resources, we can perform better than scheduling by tailoring coding and feedback

appropriately.

Figure 2-32 shows that if we have a full duplex system for two nodes to share data they clearly should do so using network coding, specially for high packet erasures. Note that for $Pe = 0.8$ SDNC full duplex more than 4 dB better in terms of the completion time performance than the scheduling scheme SDRR full duplex. It is important to point out that for low packet erasures the lower bound on the mean completion time of SDRR full duplex is loose while the upper bound is tight. In fact, the performance of SDRR full duplex is always equal or worse to that of the full duplex network coding scheme.

Finally, Figure 2-33 illustrates the number of iterations that the search algorithm proposed in this work before it reaches a stable solution. We observe that for different values M and a wide range of Pe, the number of iterations is very low, always lower or equal to 5 iterations in the example. Thus, the algorithm converges very fast, especially if the Pe is low. Thus, we have an algorithm that converges fast in a search that involves a very large amount of integer variables, e.g., with $M = 5$ and $M = 15$ initial packets in each node, we need to optimize 70 and 510 variables, respectively, for every value of Pe.

Chapter 3

Network Coding for Data Dissemination in Arbitrary Networks

This chapter considers the problem of data dissemination in more complex networks than those proposed in Chapter 2, namely, networks in which nodes may not directly communicate with every other node in the network. In fact, we study data dissemination in arbitrary networks with no specific structure, which means that a node may be at more than one hop away from other nodes in the network. Although we maintain a time division duplexing or half-duplex constraint on the nodes, we will restrict our study to cases in which latency is very low.

The problem of data dissemination has been widely studied for routing scenarios, focusing on theoretical analysis, e.g., [26], and protocol design, e.g., [32]. More recently, reference [27] studied the effect of using network coding showing significant improvement over routing in terms of completion time.

Reference [28] studied a wireless medium access control combined with network coding for multi-hop content distribution. The authors focus on a protocol that uses a content-directed medium access control (MAC), through which transmission priority is given to those nodes based on the rank of the coefficient matrix associated with the coded content the node holds, i.e., nodes with more information are given higher

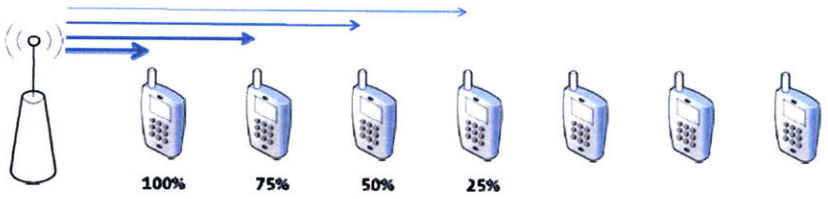

Figure 3-1: Network Example: J mobile devices require M data packets, but not all of them can obtain them directly from the Base Station. Each node can transmit to I nodes.

priority.

We advocate for the combination of network coding and medium access strategies, similar to the idea in Reference [28]. However, we illustrate that giving priority to the nodes with the most information in the network is not necessarily going to promote a faster dissemination of the data, specially in multihop networks.

We focus on the problem of minimizing the completion time to disseminate information assuming a time slotted system as described in Section 3.2. We show in Section 3.3 that the problem can be stated as a scheduling problem which is in hard to solve in general. We then propose a greedy algorithm in Section 3.4 to solving the problem, in which the nodes with the greatest impact on the network at each time slot should transmit, instead of choosing the node with the most information. Starting with a toy example for a linear meshed network, i.e., nodes deployed in a line, different medium access strategies are compared with each other in terms of the mean completion time in Section 3.5. We show that our scheme can obtain considerable gains with respect to choosing transmitters in terms of their knowledge. Even in small networks and moderate number of data packets to transmit, roughly a twofold improvement can be obtained. Although the examples and simulation results focus on linear meshed networks, we emphasize that the description and analysis of our algorithm is valid for any network and any starting distribution of (coded) packets of the nodes. In fact, our analysis considers routing as a particular case. Also, the problem of linear meshed networks is interesting in itself for some applications, e.g., underwater acoustic networks [29] [30].

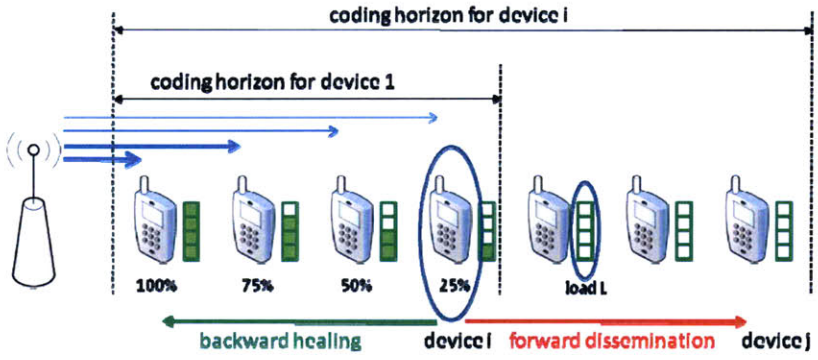

Figure 3-2: Network Example: J mobile devices require M data packets, but not all of them can obtain them directly from the Base Station. Each node can transmit to I nodes.

3.1 Motivating Examples

In order to understand the combination of network coding and medium access strategies, we assume the following scenario: A set of J mobile devices wants to receive the same number of M packets from the base station. The mobile devices are lined up with different distances to the base station as shown in Figure 3-1. The coverage of the base station is sufficient to reach $I < J$ mobile devices. Furthermore, we assume that the devices with longer distances to the base station will receive less information. For illustration we assume that the packet receiving probability, after the base station is broadcasting a number of coded packets, varies between 25% and 100% in our example with $J = 8$ and $I = 4$. The question at this point is how to continue disseminating the information, as different strategies will have an impact on the overall number of transmitted packets to satisfy all mobile devices. Let us discuss some simple possibilities.

- Base Station Centered: The base station continues the transmission of coded packets until all stations in its coverage range have understood the full information (all 100%). Note that nodes closer to the base station will need less time to gather all information, but the base station needs to continue to satisfy all devices in its coverage range. Once all devices in the coverage of the base station are satisfied, optimally the device with full knowledge that is far-

ther downstream (farther away from the base station) will start to relay the information to the rest. Network coding helps in this example to compensate for packet erasures, as the base station does not need to know about which packets have been received so far. It only has to focus on delivering enough linear combinations to the nodes. Thus, the base station transmits random linear combinations until all devices have a sufficiently high number to decode all packets.

- Progressive Base Station: The first mobile device that receives the full information will start to transmit and the base station will stop automatically. The mobile deice with full information will act as base station until another device farther downstream has all information. Obviously this approach has the advantage that more devices can be reached and that those missing information are now closer to the source, which in turn will lead to lower packet loss probability. This scheme will perform as well as or better than the Base Station Centered scheme.

- Greater Impact at each time slot (Greedy Algorithm): For the last approach, it is not the node closest to the base that goes first, in general. We look at the received packets so far by each mobile device to determine which nodes should transmit. It is important to note that the received packets in the example are uncorrelated , i.e., the packets marked as 25 % are not necessarily contained in those marked as 50%. This assumption opens the door for a new strategy. For the following discussion we introduce the term coding horizon, which is roughly the number of devices one transmitting device can reach by broadcasting information. The very first mobile device in the line has the coding horizon of 4, while mobile device i (the last one that received information from the base station) has a coding horizon of 7 (Figure 3-2). In this simple example, there is clearly a drift of information from left to right. Note that in this example, devices with a small coding horizon have collected more packets so far, but devices with fewer packets have a larger coding horizon and will therefore reach

more neighbors. We observe that node i has more impact on the network in this time slot because it can benefit more nodes with the transmission of a single coded data packet. Therefore, each relaying action is divided into a backward healing and a forward dissemination part. In case mobile device i is sending a packet, obviously all devices to the right of it are seeing that information for the first time (forward dissemination). Simultaneously, there might be devices to the left that are also interested in these packets (backwards healing). This is the main component of our heuristic scheme.

Note that the Progressive Base Station scheme will perform as well as or better than the Base Station Centered scheme. For this reason, we will focus on comparing Progressive Base Station scheme to the Greater Impact scheme.

Before starting with a formal analysis of the problem, let us consider two simple examples that illustrate the advantage of 1) choosing the transmitter node in order to provide the greatest impact to the system at each time slot versus choosing the node that has the most information, and 2) the advantage of breaking ties between sets of transmitters with the same impact on the network by choosing the one that includes nodes with the least information.

- Example 1: Let us consider a network with no packet erasures where each node wants to receive M data packets. Each node can contact 2 neighbors to the left and 2 to the right, as in Figure 3-3. The leftmost node has all M packets while a middle node has $M/2$ linear combinations. Figure 3-3 (a) shows the data dissemination process in time if we choose the node with the most information and that is further downstream (Progressive Base Station scheme). This scheme takes $\frac{5M}{2}$ time slots to complete the dissemination. Figure 3-3 (b) shows the same procedure when the node with the highest impact is chosen at the beginning. We break ties without considering parallel transmissions and trying to keep as close a behavior to the first approach. This is a simplified version of the Greater Impact scheme. Note that this very naive scheme requires only $2M$ time slots to disseminate all information to the nodes, just by choosing node

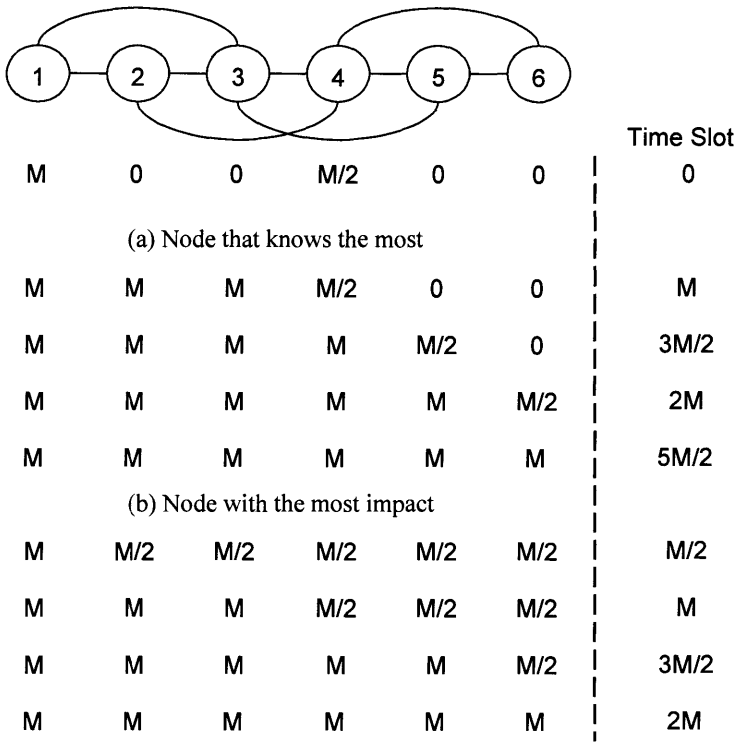

Figure 3-3: Motivating Example 1: Data dissemination when choosing (a) node with the most knowledge, and (b) node with the most impact

4 as the first transmitter. Thus, choosing the node with the most information requires 25% more time to complete transmission in this simple example, even with a vanilla version of our scheme. If we performed the dissemination with a scheme similar to 1 of the introduction, $3M$ time slots would be required to complete the transmission, i.e., 50% more time than choosing the middle node, which has greater impact to the network, at the beginning.

- Example 2: We consider a similar setup as the previous example. However, we study the case of a network with K nodes in which only one node has all the information. Assuming no packet erasures, choosing the node with the most knowledge at every time slot will transmit all of its information to its N nodes further downstream. At this point, the node further downstream starts transmitting to its N neighbors until those neighbors have all information. This process is repeated until all nodes have all data. The time to complete

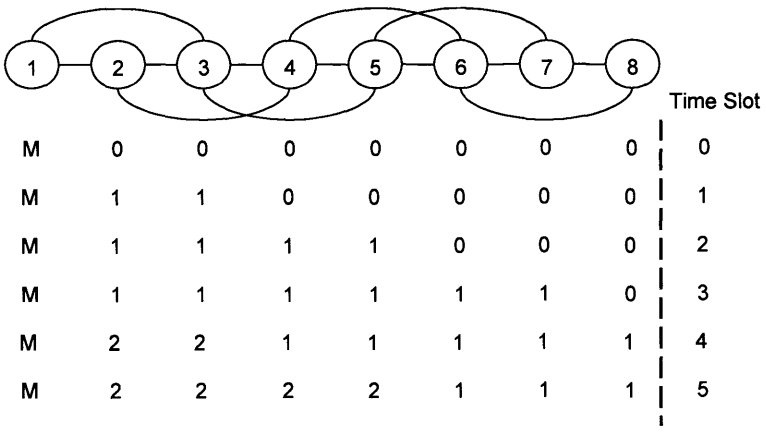

Figure 3-4: Motivating Example 2: Data dissemination exploiting parallel transmission

transmission $T_c^{(1)}$ under this scheme is

$$T_c^{(1)} = M \left\lceil \frac{K-1}{N} \right\rceil . \tag{3.1}$$

However, if we use a scheme that chooses the node with the greatest impact to the network but that breaks ties in favor of sets of transmitters that will benefit nodes with the least information, there will be a considerable reduction in the completion time. This happens because the system will be able to take advantage of parallel non-interfering transmissions. Figure 3-4 illustrates the effect of such a scheme when $N = 2$ and $K = 8$. We observe that, every 3 time slots, the same sequence of transmitting events occurs. This is valid for larger K. Using this insight, the time to complete transmission $T_c^{(2)}$ under this scheme is

$$T_c^{(2)} = 3(M-1) + \left\lceil \frac{K-1}{N} \right\rceil . \tag{3.2}$$

It is simple to show that $T_c^{(2)} \leq T_c^{(1)}$ for every value of K, N and M of importance. However, $T_c^{(2)}$ is strictly less than $T_c^{(1)}$ for $K > 4N$. Let us define the gain G in this case as

$$G = \frac{T_c^{(1)}}{T_c^{(2)}} = \frac{M \left\lceil \frac{K-1}{N} \right\rceil}{3(M-1) + \left\lceil \frac{K-1}{N} \right\rceil} \tag{3.3}$$

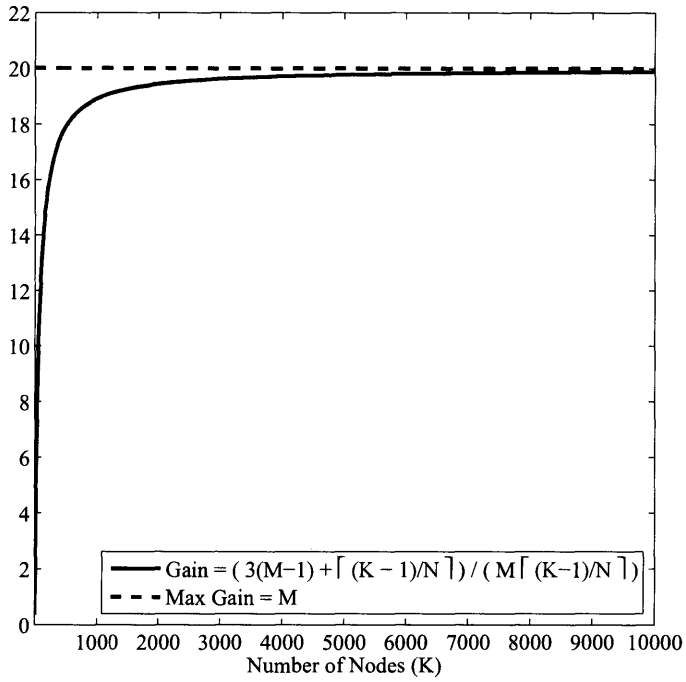

Figure 3-5: Gain G when M is fixed and the size of the network is increased. Parameters used are $M = 20$ and $N = 1$

which represents how much more time it takes to complete the dissemination when we use a scheme that chooses the node with greater knowledge instead of trying to take advantage of the spatial diversity. We can show that G can be made arbitrarily large. For example,

$$\lim_{M \to \infty} G = \frac{1}{3} \left\lceil \frac{K-1}{N} \right\rceil \tag{3.4}$$

for a fixed value of K. This is related to the case of a fixed network and a large number of packets that need to be disseminated. On the other hand, if we have a fixed number of packets but our network is large with respect to the number of packets

$$\lim_{K \to \infty} G = M. \tag{3.5}$$

Figure 3-5 illustrates the gain for a fixed value of M when we increase the size of the network. For this example, $M = 20$ and $N = 1$. Figure 3-6 illustrates the gain for a fixed network size K when we vary the number of packets to be transmitted.

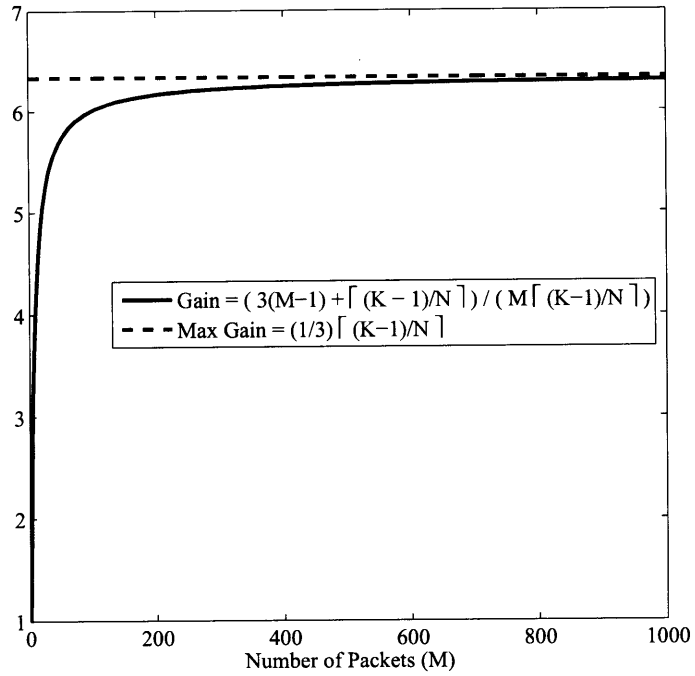

Figure 3-6: Gain G when K is fixed and we change the number of transmitted packets M. Parameters used are $K = 20$ and $N = 1$

3.2 Model

Let us formalize our problem. In particular, we focus in networks where each of the nodes has some information but it wants all M data packets present in the network. In our previous examples, we generally considered a single node, e.g., node 1 in Figure 3-7, that wanted to transmit M data packets to all other $K - 1$ nodes in the network. Let us assume that time is slotted. A data packet or coded packet is transmitted in a single slot. Each node i has a vector space $V_i(t)$ at time t. If we used no coding, each vector space would be spanned by a subset of individual packets $P_a, \forall a = 1, ..., M$ where M is the total number of packets to disseminate. If we use coding, each vector space is spanned by a set of linear combinations of $P_a, \forall a = 1, ..., M$.

We consider the network to be modeled as a hypergraph $G = (N, A)$, where N is the set of nodes and A is the set of hyperarcs. This is an extension of the graph, in which we are capturing the broadcast nature of the channel. A hyperarc (i, J) represents a connection between a node i (the transmitter) and a set of nodes J (the receivers).

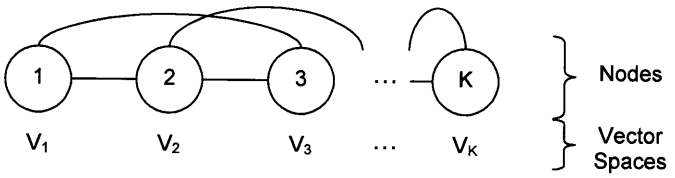

Figure 3-7: Network of interest. Each node i has at the beginning a vector space V_i. If we used no coding, each vector space would be spanned by a subset of individual packets $P_a, \forall a = 1, ..., M$ where M is the total number of packets to disseminate. If we use coding, each vector space is spanned by a set of linear combinations of $P_a, \forall a = 1, ..., M$.

We also assume a perfect knowledge of the information of the nodes and that we can operate under different channel conditions, and have different knowledge about the transmission channel. We assume a centralized controlled that decides which hyperarcs should be active at each slot, i.e., which nodes should transmit and to what set of receiver nodes, and what information should be transmitted through each hyperarc.

3.3 Description of Optimal Scheme

In general, the decision made in a time slot will affect decisions in the following slots. Let us consider the case of no erasures first. We can think about this problem as a scheduling problem, where we have a set of schedules S from which we can choose $s(t)$at time t. Each schedule $s \in S$ is a set of hyperarcs that are active in that schedule, e.g., $s = \left\{(i_1, J^{(1)}), (i_2, J^{(2)}), ..., (i_m, J^{(m)})\right\}$. In general, each schedule should only include hyperarcs with different transmitters, i.e., $i_a \neq i_b, \forall a \neq b$. However, we impose no conditions on the set of receivers J^a of each node a. This allows us to cause collisions in one node during a time slot if this means a higher benefit for the system overall.

Let us define $B^s_{iJ^{(i)}j}(t)$ as the benefit that node j gets from node i when i uses hyperarc $(i, J^{(i)})$ at time t. This benefit $B^s_{iJ^{(i)}j}(t)$ will depend on what coded packet node i transmits at time t, say $q_{i,J^{(i)}}(t) \in V_i(t)$. Let us define $W_{s(t),\bar{q}(t)}$ at time t as the weight for a schedule $s(t)$ given that $\bar{q}(t) = [q_{i,J^{(i)}}(t)]$ coded packets are being

transmitted by the active transmitters of that schedule. This weight represents the impact of that schedule over the system in that time slot, i.e., how many nodes are seeing an increase in dimension of their vector space if that schedule is chosen. We can define $W_{s(t),\bar{q}(t)}$ as

$$W_{s(t),\bar{q}(t)} = \sum_{i=1}^{m(s(t))} \left[\sum_{j \in J^{(i)}} B^{s}_{iJ^{(i)}j} \right] \tag{3.6}$$

where $m(s(t))$ is the number of hyperarcs in schedule $s(t)$. The objective is to find the sequence $U(1), U(2), ...,$ that minimizes the time to disseminate all data packets to all nodes of the network, where $U(t) = (s(t), \bar{q}(t))$.

If we assume that the network started with $W_{ini} = \sum_{k=1}^{K} \dim(V_k)$ and that at the end of the process the system should have a total of MK packets, then the problem can be formulated as

$$\min_{U(1),U(2),...} \quad n$$

subject to

$$U(t) = (s(t), \bar{q}(t)), \forall t$$

$$s(t) \in S, \forall t$$

$$q_{i,J^{(i)}}(t) \in V_i(t), \forall t, i, J^{(i)}$$

$$MK - W_{ini} = \sum_{t=1}^{D} W_{s(t),\bar{q}(t)}, \forall D \geq n$$

Solving this scheduling problem is hard even in the absence of packet erasures or perfect knowledge of the channel. Let us focus on a greedy algorithm that tries to maximize the impact on the network at each time slot.

3.4 Greedy Algorithms

Let us use greedy algorithms that only take into account the current state of the system to make a decision, i.e., we try to find the set of hyperarcs s that will have the node with the most information or the set s that has greater impact in the network

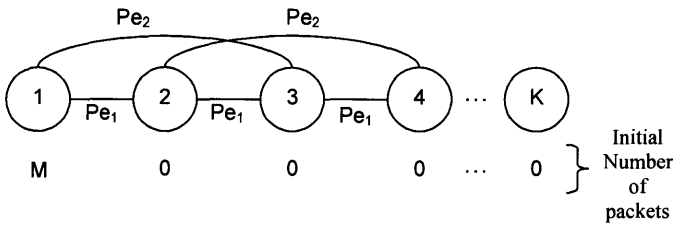

Figure 3-8: Simulation Setup: linear meshed network, with each node with at most $N = 2$ neighbors upstream (closer to node 1) and $N = 2$ downstream. Node 1 has all M data packets at the beginning. The objective is to disseminate those packets to every receiver in the least amount of time.

in the current slot for the Progressive Base Station and Greater Impact schemes, respectively. From its perspective, at any time slot the network can be modelled as a set of nodes N with a vector space V_i associated to each node i (Figure 3-7).

Let us define v_{iJ} as the vector selected for transmission in hyperarc (i, J). This choice is made to maximize the impact of the transmission from i to each of his receivers in J. One way to state this problem is to choose v_{iJ} so that we increase as much as possible the dimensions of the vector spaces of each of the receivers

$$v_{iJ} = \arg\max_{q_{iJ} \in V_i} \sum_{j \in J} \dim\left(\{V_j, q_{iJ}\}\right). \tag{3.7}$$

Note that this is valid for network coding. The simplest way of generating v_{iJ} is to create a random linear coded packet over a large enough field size. This coded packet is generated from the packets or linear combinations that span the vector space of node V_i. If no coding is allowed, we have to impose the additional constrain on q_{iJ} to be a single packet, i.e., the vector q_{iJ} will have a very specific structure. At any time slot we choose a schedule depending on the scheme we use.

The Greater Impact greedy algorithm will choose as schedule

$$s^*_{GI} = \arg\max_{s \in S} W_s \tag{3.8}$$

where W_s is the weight for each schedule given the choice of q_{iJ}.

Similar to the full problem, W_s represents the impact of a schedule s over the system in that slot. Thus,

$$W_s = \sum_{i=1}^{m(s)} \left[\sum_{j \in J^{(i)}} B^s_{iJ^{(i)}j} \right] \tag{3.9}$$

where $m(s)$ is the number of hyperarcs in schedule s, $B^s_{iJ^{(i)}j}$ is the benefit that node j gets from node i when i uses hyperarc $(i, J^{(i)})$.

In the case of no erasures,

$$B^s_{iJ^{(i)}j} = \begin{cases} 1 & \text{if } Z_{iJ^{(i)}j} \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

where $Z_{iJ^{(i)}j} \equiv \dim(V_j) < \dim\left(\{V_j, v_{iJ^{(i)}}\}\right)$, $j \notin J^{(k)} \forall k \neq i$, $j \neq i_a, \forall a$

If we had perfect Channel State Information (CSI),

$$B^s_{iJ^{(i)}j} = \begin{cases} C_{iJ^{(i)}j} & \text{if } Z_{iJ^{(i)}j} \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

where $C_{iJ^{(i)}j}$ is the channel state of the channel from i to j when i transmits through hyperarc $iJ^{(i)}$. $C_{iJ^{(i)}j} = 1$ if the channel will cause no erasure, and $C_{iJ^{(i)}j} = 0$ otherwise.

As a contrast, the Progressive Base Station scheme will choose the schedule such that

$$s^*_{PBS} = \arg\max_{s \in S} \left(\max_{(i,J) \in s} |V_i| \right) \tag{3.12}$$

and we break ties in favor of nodes that have more neighbors with incomplete information. This approach is similar in its essence to the work in [28], because it gives nodes with the most information a priority to access the channel.

If we have no CSI but we have knowledge (or estimates) of the packet erasure probability, the Progressive Base Station scheme remains the same. For the Greater

Impact scheme, we can define

$$B^s_{iJ^{(i)}j} = \begin{cases} 1 - P_{iJ^{(i)}j} & \text{if } Z_{iJ^{(i)}j} \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

where $P_{iJ^{(i)}j}$ is the packet erasure probability of the channel from i to j when i transmits through hyperarc $iJ^{(i)}$.

Note that this last approach is equivalent to choosing the schedule s^* that maximizes the average weight of the schedule, i.e., $s^* = \arg\max_{s \in S} E[W_s]$. Note that

$$E[W_s] = \sum_{i=1}^{m(s)} \left[\sum_{j \in J^{(i)}} E\left[B^s_{iJ^{(i)}j} \right] \right] \tag{3.14}$$

when

$$B^s_{iJ^{(i)}j} = \begin{cases} C_{iJ^{(i)}j} & \text{if } Z_{iJ^{(i)}j} \\ 0 & \text{otherwise} \end{cases} \tag{3.15}$$

and $E\left[C_{iJ^{(i)}j}\right] = 1 - P_{iJ^{(i)}j}$.

3.5 Numerical Results

This section provides numerical examples that compare the performance of two different network coding schemes we have discussed so far. Namely, our proposed scheme which chooses at each time slot the schedule that will provide the greatest impact on the network ('Greater Impact'), and the scheme that chooses a schedule based on the node that has the most knowledge in the network ('Progressive Base Station'). We assume that the available schedules are the same for both schemes. Also, we simplify the problem by only allowing schedules that do not generate interference in some nodes, which could be beneficial in some cases. Note that this is not a restriction of the analysis but a means to simplify the simulation. The comparison between the

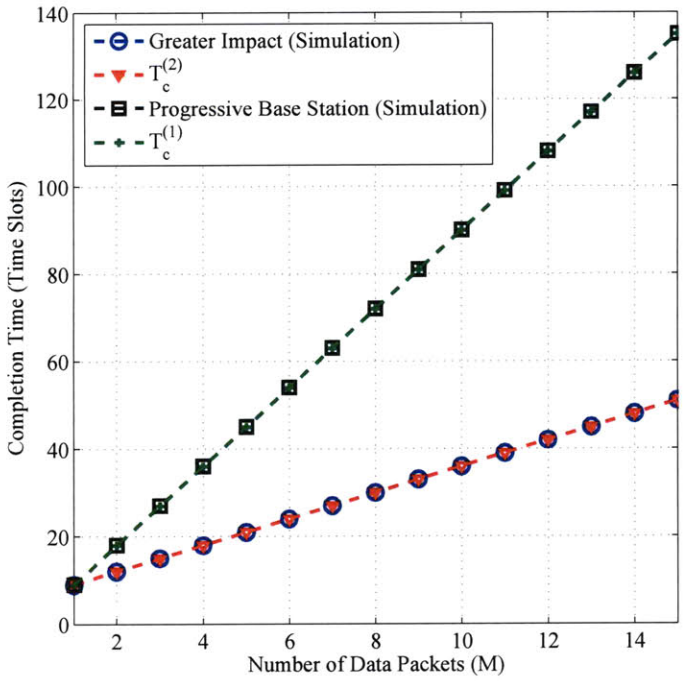

Figure 3-9: Completion time of schemes on a linear meshed network of 10 nodes with different number of data packets to be disseminated. Each node can only contact one neighbor upstream and one downstream but with no packets being erased, i.e., $Pe_1 = 0, Pe_2 = 1$.

schemes is carried out in terms of average time to complete transmission of M data packets under different packet erasure probability scenarios.

Our results focus on linear meshed networks with one or two neighbors upstream and downstream. The packet erasure probability to the closest two neighbors is Pe_1, while Pe_2 corresponds to the packet erasure probability for the two neighbors farther away. Finally, we assume that one node at the edge of the network has all information at the beginning and that all other nodes have no information (See Figure 3-8).

Figure 3-9 shows that the results obtained in Section II for Motivating Example 2 match the simulation results. In particular, we observe that our 'Greater Impact' scheme has the same completion time to that in $T_c^{(2)}$ for $N = 1$ neighbor upstream and 1 neighbor downstream, $K = 10$ nodes in the network, and a range of data packets M. The 'Progressive Base Station' approach shows similar performance to $T_c^{(1)}$. Figure 3-9 shows that a considerable reduction in completion time can be found by choosing the schedule with the greatest impact to the network and by breaking

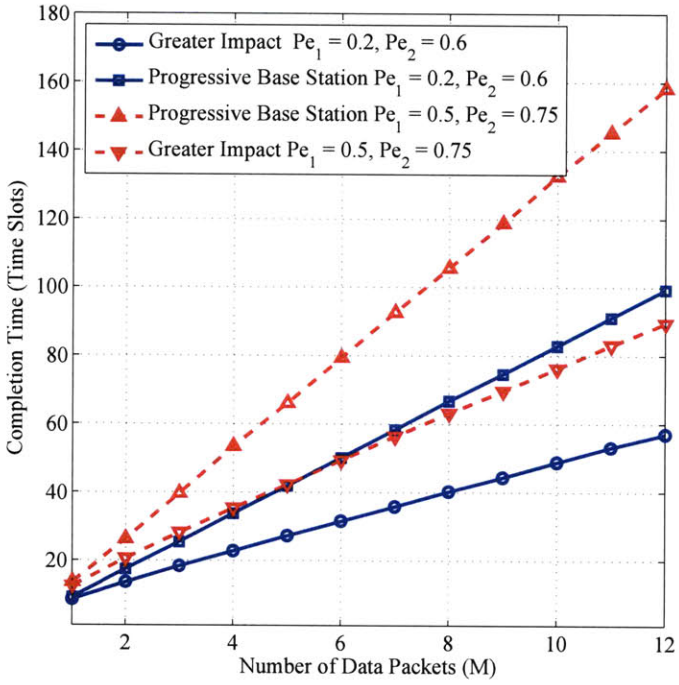

Figure 3-10: Completion time of schemes on a linear meshed network of 10 nodes with different number of data packets to be disseminated. Each node can contact $N = 2$ neighbors upstream and 2 downstream.

ties in favor of schedules that benefit nodes with the least information. This is related to the opportunistic overhearing idea presented in [31] for routing, i.e., if we allow overhearing of the data packets the dissemination time can be considerably reduced, because each transmission will be useful for more than one node. The figure shows the degradation in performance we would get if we performed routing of the coded packets from one node to its neighbor without exploiting the broadcast nature of wireless channels.

Figure 3-10 shows the performance of the two schemes when coded packets can suffer erasures. Each node has $N = 2$ neighbors upstream and downstream, except nodes at the edges of the linear meshed network. This figure shows that for different scenarios our 'Greater Impact' scheme shows much better performance. Note that both schemes are allowed the same schedules. The main difference is the way each scheme chooses amongst the different schedules.

Figure 3-11 illustrates the gain in completion time for different pairs (Pe_1, Pe_2) of packet erasure probabilities. This figure illustrates the gains for a network of $K = 10$

nodes. We expect larger gains if we increase M or the number of nodes K present in the network. However, it is clear that there is a considerable advantage of our scheme even for moderate network size and M.

Figure 3-11 shows that the gain from using the Greater Impact scheme instead of the Progressive Base Station scheme increases with packet erasures for our example with $N = 2$. We observe this by comparing the result for $(Pe_1 = 0, Pe_2 = 0)$ to those with non-zero packet erasures, except the case $(Pe_1 = 0, Pe_2 = 1)$ which corresponds to $N = 1$. The gain for the case of $(Pe_1 = 0, Pe_2 = 0)$ is a propagation gain, i.e., the Greater Impact scheme propagates coded packets faster to other nodes, allowing parallel transmissions. The same applies for $(Pe_1 = 0, Pe_2 = 1)$. For the cases of random erasures, the gain is in part due to the propagation gain and in part because the Greater Impact scheme chooses schedules based on their impact to the network without focusing the decision on a single node. Figure 3-11 shows that for $K = 10$ nodes and $M = 12$ packets with different packet erasure pairs (Pe_1, Pe_2) we observe a gain of about 1.8. This means that Progressive Base Station takes 80% more time to complete the data dissemination process than what it would take if we used the Greater Impact scheme.

Figure 3-11 shows that the case of $(Pe_1 = 0, Pe_2 = 1)$ (in essence $N = 1$) has a larger gain than $(Pe_1 = 0, Pe_2 = 0)$ $(N = 2)$, although the former requires more time complete the transmission for both schemes. This illustrates that allowing overhearing of the coded packets, which is represented by the ability of the nodes to transmit to more than one neighbor, reduces the gain of using Greater Impact with respect to Progressive Base Station, for the same number of nodes in the network K. However, this gain will increase as we increase the network size or as we increase the number of packets to transmit. Even for a small network of $K = 10$ nodes we get a gain of 1.6 when we transmit $M = 12$ packets with no erasures, which is to say that Progressive Base Station takes 60% more time to complete the data dissemination process than what it would take if we used the Greater Impact scheme.

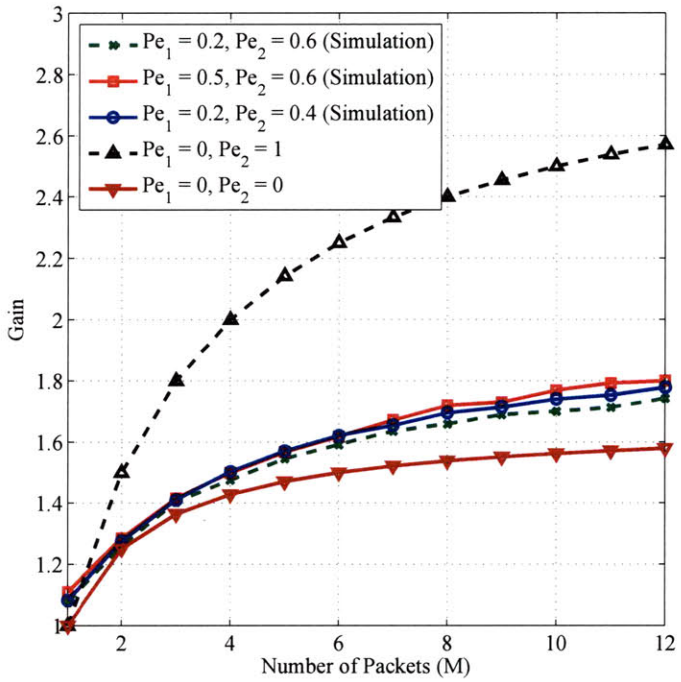

Figure 3-11: Gain in completion time on a linear meshed network of 10 nodes with different number of data packets to be disseminated. Each node can contact $N = 2$ neighbors upstream and 2 downstream.

Chapter 4

Underwater Acoustic Networks

We have so far dealt with the problem of networks that require a fast, reliable and efficient dissemination of data through 1) environments that have physical limitations that can cause considerable delay, e.g., networks with large latency TDD channels studied in Chapter 2, or 2) network deployments that impose connectivity limitations, as in Chapter 3.

This chapter deals with a specific application of delay challenged environment, that of underwater acoustic networks. This particular application suffers from large latency and, in many cases, nodes are considered to have TDD constraints. In this sense, the techniques studied in Chapter 2 can be applied in order to improve delay performance, reliability and efficiency of data transmissions underwater.

Also, many underwater network deployments require information to traverse multiple nodes, that is, it requires transmission over multiple hops, in order to reach an intended destination. Thus, the insights provided in Chapter 3 could considered and adapted for underwater networks, taking into account that a decentralized controller and large latency would be key factors to be considered in a modified dissemination scheme.

However, we focus in this chapter in studying in more detail the underwater channel model in order to 1) derive a lower bound on the transmission power for multicast transmissions, 2) propose and test practical transmission schemes, and 3) determine the gap between the lower bound and the practical schemes.

4.1 Underwater Channel Model

An underwater acoustic channel is characterized by an attenuation that depends on the distance l and the signal frequency f as

$$A(l,f) = \left(\frac{l}{l_{ref}}\right)^{\alpha} a(f)^{l-l_{ref}} \tag{4.1}$$

where l_{ref} is a reference distance (typically 1 m). Note that this model generalizes the free-space model. In practical applications, typically $l >> l_ref$ so that $l - l_{ref} \approx l$.

A common empirical model used for the absorption $a(f)$ is Thorp's formula [47] [56] which captures the dependence on the frequency in dB/km. This absorption $a(f)$ is an strictly increasing function of f measured in kHz. The spreading factor describes the geometry of propagation and is typically $1 \leq \alpha \leq 2$, e.g., $\alpha = 1$ and $\alpha = 2$ correspond to cylindrical and spherical spreading, respectively. A cylindrical spreading (Figure 4-1(a)) corresponds to cases in which the transmission distance l is much larger than the depth of the ocean. In this case, the ocean bottom and the interface between the ocean and the air act as boundaries for the spreading of acoustic waves. This problem can be modelled as a cylindrical wave guide. On the other hand, spherical spreading (Figure 4-1(b)) is considered when the transmission distance is smaller than the depth of the ocean. This type of spreading provides a similar α as the free-space approximation for radio wireless communications.

The noise in an acoustic channel can be modeled through four basic sources: turbulence $N_t(f)$, shipping $N_s(f)$, waves $N_w(f)$, and thermal noise $N_{th}(f)$ [47]. The power spectral densities (psd) of these noise components in dB re μ Pa per Hz as functions of frequency in kHz are presented in [57]. These psd's have two important parameters: 1) the shipping activity s ranging from 0 to 1, for low and high activity, respectively, 2) the wind speed w measured in m/s. Figure 2 in [47] shows $N(f)$ for different values of s and w, and an approximation $10\log N(f) = N_1 - \eta\log(f)$ for $f \leq 100$ kHz, where $N_1 = 50$ dB re μ Pa and $\eta = 18$ dB/dec.

The complete model for a colored Gaussian underwater link was presented in [71] where power was allocated through waterfilling. In the absence of multipath and

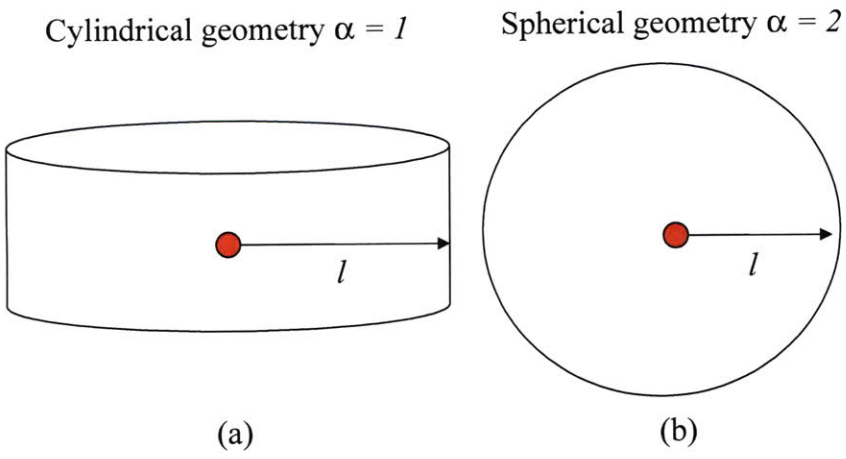

Figure 4-1: Spreading Geometries in Underwater Communications: (a) cylindrical, (b) spherical

channel fading, the relationship among capacity, transmission power, and optimal transmission band of a point-to-point link is given by [71]

$$C = \int_{B(l,C)} \log_2 \left(\frac{K(l,C)}{A(l,f)N(f)} \right) df \tag{4.2}$$

where $N(f)$ is the psd of the noise, $B(l,C)$ is the optimum band of operation and $K(l,C)$ is a constant whose value is determined by the link distance l and the desired capacity C. The transmission power associated with a particular choice of (l,C) is given by

$$P(l,C) = \int_{B(l,C)} S(l,C,f) df \tag{4.3}$$

where the psd of the signal is $S(l,C,f) = K(l,C) - A(l,f)N(f), f \in B(l,C)$.

The corresponding signal-to-noise ratio is given by:

$$SNR = \frac{\int_{B(l,C)} S(l,C,f) A^{-1}(l,f)\, \mathrm{d}f}{\int_{B(l,C)} N(f)\, \mathrm{d}f}. \tag{4.4}$$

We observe that a choice of (l,C) determines implicitly the SNR level. Hence, there is a one-to-one correspondence between the pair (l,C) and the pair (l,SNR). The latter parameterization was used in [47] to compute the transmission power and bandwidth

representation to ensure a preset SNR, which determines the value of C implicitly. The present analysis focuses on using the former parameterization, i.e., on determining the power and transmission band that ensure a pre-set link capacity.

4.1.1 Dependence on the spreading factor

The dependence on the spreading factor α is quite simple. Let us assume that a model for $P(l, C)$ has been developed for a particular value of α, i.e., $P(l, C, \alpha)$. To determine $P(l, C, \alpha')$ for $\alpha' \neq \alpha$, note that a change in α, the product $A(l, f)N(f) = (l/l_{ref})^{\alpha} a(f)^l N(f)$ constitutes a constant scaling factor with respect to f. Therefore, for a link of distance l the term $B(l, C)$ will remain unchanged. Thus, if the same capacity C is required for α and α', equation (4.2) shows that the only other term that can vary is $K(l, C)$, i.e., $K(l, C, \alpha)$. Then, $K(l, C, \alpha') = (l/l_{ref})^{\alpha'-\alpha} K(l, C, \alpha)$. Finally, let us use the equation (4.3) to determine the relationship between $P(l, C, \alpha)$ and $P(l, C, \alpha')$.

$$P(l, C, \alpha') = \int_{B(l,C)} \left(K(l, C, \alpha') - l_m^{\alpha'} a(f)^l N(f) \right) df \tag{4.5}$$

$$= l_m^{\alpha'-\alpha} \int_{B(l,C)} \left(K(l, C, \alpha) - l_m^{\alpha} a(f)^l N(f) \right) df \tag{4.6}$$

$$= l_m^{\alpha'-\alpha} P(l, C, \alpha) \tag{4.7}$$

where $l_m = l/l_{ref}$ to shorten the derivation. Thus, any model generated for some parameter α has a simple extension. Also, note that the transmission band remains the same for any value of α.

4.1.2 Interference Characteristics

The optimal transmission band of a link was shown to change with the distance, under the assumption that the channel is Gaussian and that the capacity of a link is obtained through waterfilling. If the capacity for a link is low, e.g., less than 2 kbps, and the transmission distance is below 10 km, the transmission bandwidth will also be low, and its optimal location in the spectrum will change dramatically with the

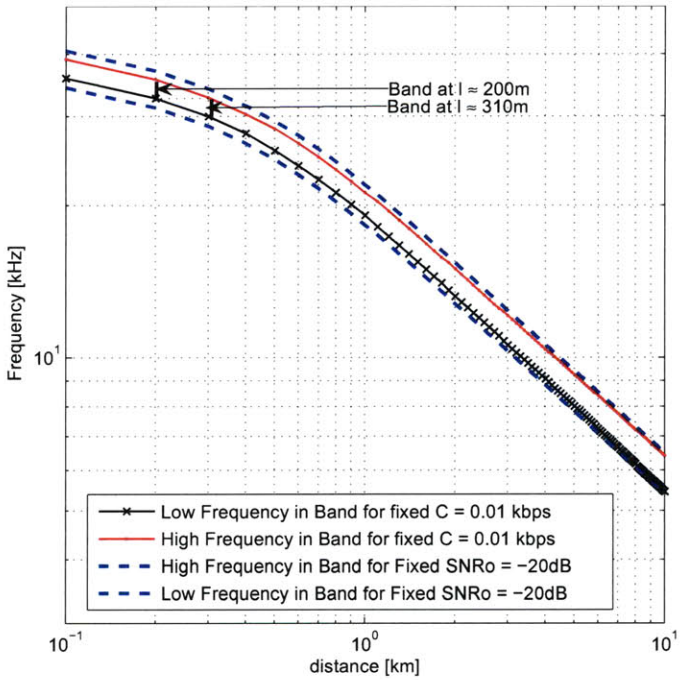

Figure 4-2: High and low band edge frequency of the transmission band for $C = 0.01$kbps, $\alpha = 1.5$, $s = 0.5$, and $w = 0$m/s.

distance. Figure 4-2 shows this effect for $C = 0.01$ kbps. In this figure, the high and low band edge frequencies are plotted. This figure also shows the high and low band edge frequency if an SNR requirement of -20 dB is set, i.e., using the SNR instead of the capacity as the fixed parameter. As noted before, the constraint over the capacity is related to different SNR levels depending upon the distance. It is clear that low values of C are related to a very low SNR value.

Figure 4-2 shows that if two links with the same $C = 0.01$ kbps are established, one with $l \approx 200$ m and the other with $l \approx 310$ m, the optimal transmission bands for these links will not overlap; thus, they do not interfere with one another. This characteristic of the underwater channel suggests that if a network is established in which the nodes are at different distances from one another, and each node has a limited range of transmission when the data rate requirement is very low (all valid assumptions in underwater networks), there will be no interference between transmissions of the various links. If each link allocates its band optimally, this suggests that a form of FDMA is the optimal approach in an underwater network, where transmission band

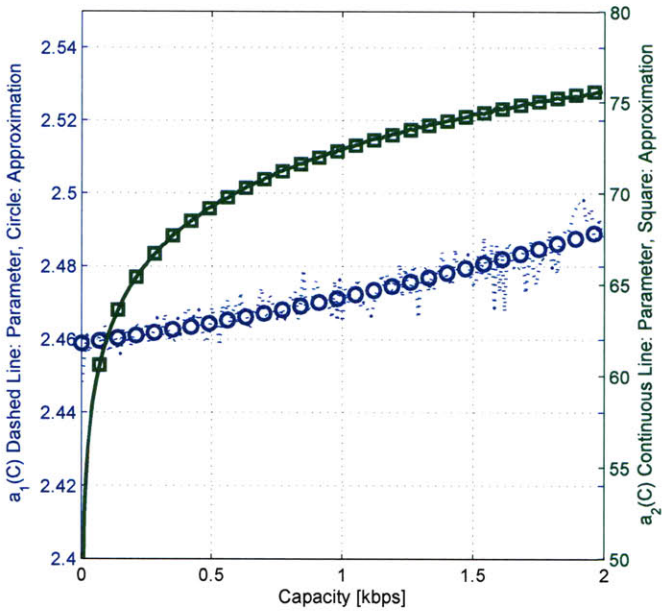

Figure 4-3: Parameters a_1 and a_2 for $P(l, C)$ and approximate model.$l \in [0, 10]$ km, $C \in [0, 2]$ kbps,$\alpha = 1.5$,$s = 0.5$ and $w = 0$ m/s.

is determined by both the distance and the required data rate. From a network optimization view point, the cost function to be minimized is clearly separable under these assumptions, where the channel model for a link can be used as the cost function for each of the separable terms.

4.1.3 Transmission Power in power limited scenarios

A distinguishing feature of the underwater acoustic channel is the dependence of the optimal transmission band on the link distance. Figure 4-6 illustrates the optimal center frequency $f_c(l)$ as a function of distance. The optimal center frequency is defined as the frequency at which $A(l, f)N(f)$ is minimal. This implies that if the transmission power for a link is low, the transmission bandwidth will be low and around the optimal frequency. Thus, the optimal transmission band in the spectrum changes dramatically with the link distance. Figure 4-6 also illustrates that a node transmitting over a short range will optimally be assigned a transmission band at high center frequency, as in case (a), while a node transmitting over a longer distance

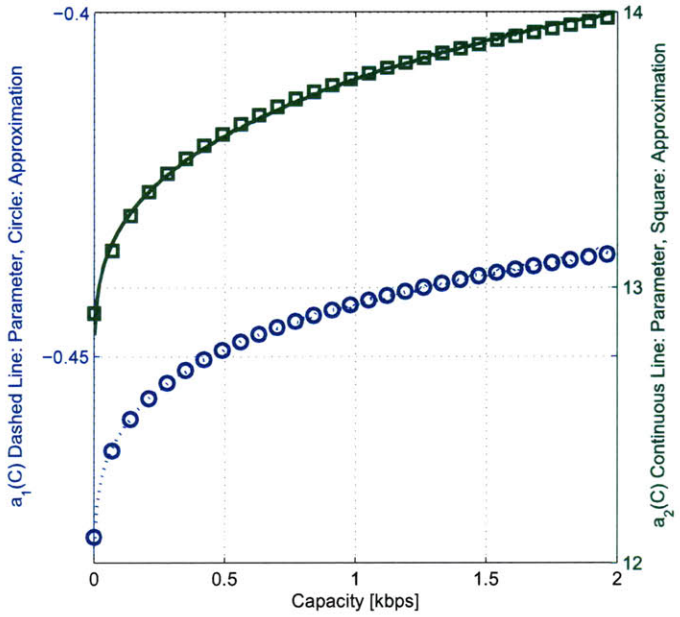

Figure 4-4: Parameters a_1 and a_2 for $\hat{f}_{end}(l, C)$ and approximate model.$l \in [0, 10]$ km, $C \in [0, 2]$ kbps,$k = 1.5$,$s = 0.5$ and $w = 0$ m/s.

will be assigned a different transmission band at lower center frequency, as in case (b).

For the case in which the power available for transmission is low, the bandwidth of the transmission band will also be small. When the bandwidth is low enough $|B(l, C)| = \Delta f$, such that the product $A(l, f)N(f)$ does not change much over that band, one can make a Taylor series approximation around the center frequency $f_c(l)$. This allows us to determine the power P for which the transmission band is narrow owing to our waterfilling argument. Noting that the first derivative of $A(l, f)N(f)$ with respect to f is zero at f_c, the Taylor series approximation has the form

$$A(l, f)N(f) \approx A(l, f_c)N(f_c) + \Upsilon \frac{(f - f_c)^2}{2} \tag{4.8}$$

$\forall f \in (f_{\min}, f_{\max})$, where $\Upsilon = \frac{\partial^2}{\partial f^2} (A(l, f)N(f)) \mid_{f=f_c}$. Substituting this expression (4.8) into expression (4.3), and using the fact that $K(l, C) = A(l, f_{\max})N(f_{\max}) = A(l, f_{\min})N(f_{\min})$, where $f_{\max}$ and $f_{\min}$ are the maximum and minimum frequencies

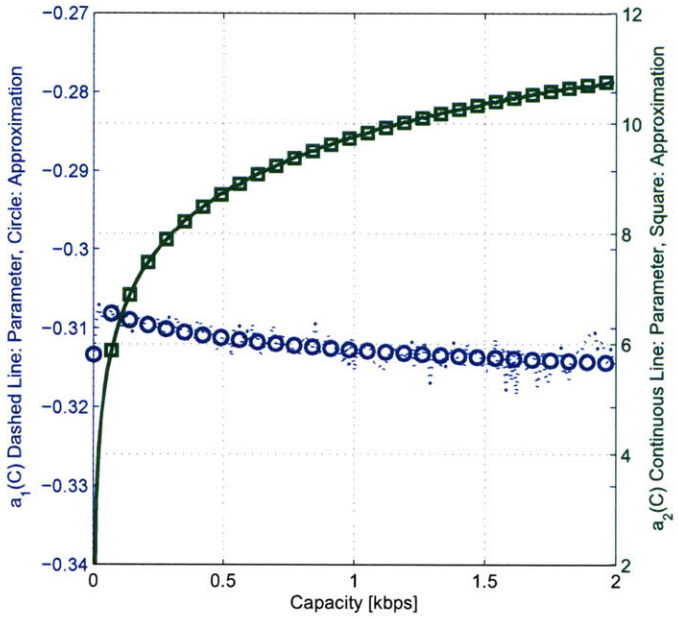

Figure 4-5: Parameters a_1 and a_2 for $\hat{B}(l, C)$ and approximate model.$l \in [0, 10]$ km, $C \in [0, 2]$ kbps, $k = 1.5$,$s = 0.5$ and $w = 0$ m/s.

of the transmission band, we obtain

$$P(l, C) \approx A(l, f_{\max})N(f_{\max})\Delta f - \int_{f_{\min}}^{f_{\max}} \left(A(l, f_c)N(f_c) + \Upsilon \frac{(f - f_c)^2}{2} \right) df$$

where $\Delta f = f_{\max} - f_{\min}$. Considering $f_{\max} - f_c \approx \frac{\Delta f}{2}$ and $f_c - f_{\min} \approx \frac{\Delta f}{2}$, given our quadratic Taylor series approximation of $A(l, f)N(f)$, the above expression reduces to

$$P = \frac{\Upsilon}{12}\Delta f^3. \tag{4.9}$$

4.1.4 Numerical Evaluation Procedure

A numerical evaluation procedure similar to that of [47] is used to compute the value of $P(l, C)$, $\hat{B}(l, C)$ and $\hat{f}_{end}(l, C)$, for a region of values of (l, C). The procedure starts by fixing a target value of the capacity C. Then, for each distance l, the initial value of $K(l, C)$ is set to the minimum value of the product $A(l, f)N(f)$, i.e., $K(l, C) =$

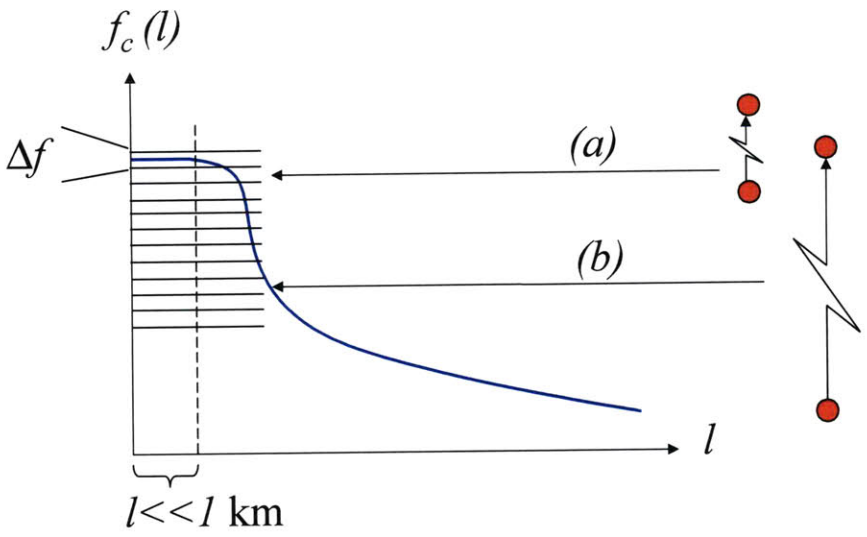

Figure 4-6: Relationship between transmission distance and center frequency in a narrow band system.

$\min_f A(l,f)N(f)$. The frequency at which this occurs, i.e., $f_0 = \arg\min_f A(l,f)N(f)$, is called the optimal frequency. After this, $K(l,C)$ is increased iteratively by a small amount, until the target capacity value C is met. Finally, this procedure is repeated for each value of C in a range of interest. At the n-th step of the procedure, when $K^{(n)}(l,C)$ is increased by a small amount, the band $B^{(n)}(l,C)$ is determined for that step. This band is defined as the range of frequencies for which the condition $A(l,f)N(f) \leq K^{(n)}(l,C)$ holds. Then, the capacity $C^{(n)}$ is numerically determined for the current $K^{(n)}(l,C)$ and $B^{(n)}(l,C)$, using the equation (4.2). If $C^{(n)} < C$, a new iteration is performed. Otherwise, the procedure stops.

4.1.5 Approximate models

Evidently, the expressions for the complete model are quite complicated to be used in a computational network analysis. Also, they provide little insight into the relationship between power consumption, $\hat{B}$ and $\hat{f}_{end}$, in terms of the pair (l,C). This motivates the need for an approximate model to represent these relations for ranges of C and l that are of interest to acoustic communication systems. The model should also provide flexibility to changing other parameters, such as the spreading factor α, wind

speed w and shipping activity s. As shown in Equation (4.7), any approximate model for the transmission power generated for some parameter α has a simple extension to any other value of α. Also, a model for the transmission band remains the same for any value of α.

By applying the numerical procedure in Subsection 4.1.4 for various l and C and fitting the data, it is possible to obtain approximate models for power consumption (Eq. 4.10), band-edge frequency $\hat{f}_{end}(l,C)$ (Eq. 4.12), and for the bandwidth $\hat{B}(l,C) = \hat{f}_{end}(l,C) - \hat{f}_{ini}(l,C)$ (Eq. 4.14). Note that some important properties for these parameters are kept, e.g., $P(l,0) = 0$.

$$\tilde{P}(l,C) = l^{a_1(C)} 10^{\frac{a_2(C)}{10}} \text{ with} \tag{4.10}$$

$$a_1(C) = \alpha_3 + \alpha_2 C + \alpha_1 C^2 \tag{4.11}$$

$$a_2(C) = \beta_3 + \beta_2 10log_{10}C + \beta_1 (10log_{10}(C+1))^2$$

$$\hat{f}_{end}(l,C) = l^{a_1(C)} 10^{\frac{a_2(C)}{10}} \text{ with} \tag{4.12}$$

$$a_1(C) = \alpha_3 + \alpha_2 10log_{10}C + \alpha_1 (10log_{10}C)^2 \tag{4.13}$$

$$a_2(C) = \beta_3 + \beta_2 10log_{10}C + \beta_1 (10log_{10}C)^2$$

$$\hat{B}(l,C) = l^{a_1(C)} 10^{\frac{a_2(C)}{10}} \text{ with} \tag{4.14}$$

$$a_1(C) = \alpha_4 + \alpha_3 10log_{10}C + \alpha_2 (10log_{10}C)^2 + \alpha_1 (10log_{10}C)^3 \tag{4.15}$$

$$a_2(C) = \beta_3 + \beta_2 10log_{10}C + \beta_1 (10log_{10}C)^2.$$

The transmission power, band-edge frequency and bandwidth of transmission band were computed for a variety of values of s, w and two ranges of interest of the pair (l,C): $l \in (0,10]$ km, $C \in [0,2]$ kbps, and $l \in (0,100]$ km, $C \in [0,100]$ kbps. The models proposed fitted these cases quite well. Results are presented for $\alpha = 1.5$, $w = 0$ and $s = 0.5$, for both cases. For the first case, the α and β parameters show almost no dependence on the shipping activity factor s, especially if the wind speed

Table 4.1: a_1 and a_2 approximation parameter values for $P(l,C)$, $\hat{f}_{end}(l,C)$ and $\hat{B}(l,C)$, with $\alpha = 1.5$,$s = 0.5$ and $w = 0$ m/s for Case 1: $l \in [0,10]$ km, $C \in [0,2]$ kbps, and Case 2: $l \in [0,100]$ km, $C \in [0,100]$ kbps.

Case		α_1	α_2	α_3	α_4	MSE
1	$P(l,C)$	-0.00235	0.01565	2.1329	0	2.532e-7
1	$\hat{f}_{end}(l,C)$	4.795e-5	0.00246	-0.44149	0	3.930e-9
1	$\hat{B}(l,C)$	-5.958e-7	-2.563e-5	-0.000305	-0.30694	6.599e-9
2	$P(l,C)$	-5.617e-5	0.02855	2.9305	0	0.00011
2	$\hat{f}_{end}(l,C)$	-0.00019	0.01186	-0.55076	0	1.32e-7
2	$\hat{B}(l,C)$	1.696e-6	4.252e-5	-0.00249	-0.36397	7.29e-7
			β_1	β_2	β_3	MSE
1	$P(l,C)$		0.014798	1.0148	74.175	5.8979e-5
1	$\hat{f}_{end}(l,C)$		0.00171	0.07153	13.738	3.4706e-5
1	$\hat{B}(l,C)$		-5.163e-6	0.33427	9.6752	2.9233e-7
2	$P(l,C)$		0.04317	0.90597	76.156	0.00010115
2	$\hat{f}_{end}(l,C)$		0.0065157	-0.032693	14.739	7.3024e-5
2	$\hat{B}(l,C)$		-0.0018252	0.34788	10.328	0.00019414

is $w > 0$. Thus, the approximate model for this case can be simplified to consider w only as part of the model, instead of the pair (s,w).

Figures 4-3, 4-4 and 4-5 show parameters a_1 and a_2 for $P(l,C)$, $\hat{f}_{end}(l,C)$, and $\hat{B}(l,C)$, respectively, for the first case with $\alpha = 1.5$, $s = 0.5$ and $w = 0$ m/s. The values of α's and β's are shown in Table 4.1 as Case 1, for parameters a_1 and a_2, respectively. These tables also show the mean square error (MSE) of the approximation with respect to the actual parameters. A similar result can be found for the second case with $\alpha = 1.5$, $s = 0.5$ and $w = 0$. The values of α's and β's are shown as Case 2 in Table 4.1, for parameters a_1 and a_2, respectively. For both ranges, the proposed models give a very good approximation to the actual numerical values. Also note that for the $a_1(C)$ parameter of $P(l,C)$, it is possible to use a linear approximation, instead of a quadratic model.

For Case 1, the values for α and β parameters in the approximate $\tilde{P}(l,C)$ model show very little dependence with respect to s while they show a greater dependency on w. This is not unexpected, since the transmission band is at a high frequency (between 5 and 40 KHz) where the noise psd is influenced more by the w ($O(f^{-2})$) than s ($O(f^{-3.4})$). Therefore, a further approximation is to discard s and consider parameters α and β to be functions of w only. In particular, a simple model is $\alpha_i = \gamma_3 + \gamma_2 10log_{10}(w+1) + \gamma_1 (10log_{10}(w+1))^2$, $\forall i = 1,2,3$. A similar relation holds

for $\beta_i, \forall i = 1, 2, 3$. Table 4.2 shows γ parameters for the different α's and β's.

4.1.6 Convexity Analysis

In this section we prove the convexity of the complete model of $P(l, C)$ in the entire region of interest, i.e., positive data rates and $l > 0$. Then we discuss the necessary conditions that the value of l has to fullfill to ensure convexity of the approximate model. This is particularly useful if we use these approximate models in network optimization problems.

Convexity of Complete Model

The convexity of the transmission power of the complete model is stated in the following lemma, which is proven in the Appendix. Lemma 1 assures that $P(l, C)$ is a convex function with respect to C for the ranges of interest of C and l for the case of non-overlapping finite bands.

Lemma 3. *$P(l, C)$ is a convex, increasing function with respect to C, $\forall C > 0$ and $l > 0$, if $A(l, f)N(f) > 0$ and $B(l, C) = \cup_i [f^i_{ini}(l, C), f^i_{end}(l, C)]$, with $f^i_{ini}(l, C) < f^i_{end}(l, C) < \infty, \forall i$ and $f^i_{ini}(l, C), f^i_{end}(l, C) \notin [f^j_{ini}(l, C), f^j_{end}(l, C)], \forall i \neq j$, i.e., a union of non-overlapping finite bands.*

Proof. We consider a set Ξ of bands, each band $i \in \Xi$ having a $f^i_{end}(l, C)$ and $f^i_{ini}(l, C)$ associated to it. Then,

$$P(l, C) = \sum_i K(l, C)(f^i_{end}(l, C) - f^i_{ini}(l, C)) - \sum_i \int_{f^i_{ini}(l,C)}^{f^i_{end}(l,C)} A(l, f)N(f)df \quad (4.16)$$

and

$$C = \sum_i \int_{f^i_{ini}(l,C)}^{f^i_{end}(l,C)} log_2 \left(\tfrac{K(l,C)}{A(l,f)N(f)}\right) df. \quad (4.17)$$

Using the Leibniz Integral rule, the fact that $K(l, C) = A(l, f_{end}(l, C))N(f_{end}(l, C))$ and $K(l, C) = A(l, f_{ini}(l, C))N(f_{ini}(l, C))$, that $A(l, f)N(f)$ is independent of C, and

that the derivative of the sum is the sum of the derivatives,

$$\frac{\partial P(l,C)}{\partial C} = \frac{\partial K(l,C)}{\partial C} \sum_i (f^i_{end}(l,C) - f^i_{ini}(l,C)). \tag{4.18}$$

Taking the second derivative:

$$\frac{\partial^2 P(l,C)}{\partial C^2} = \sum_i \frac{\partial^2 K(l,C)}{\partial C^2} (f^i_{end}(l,C) - f^i_{ini}(l,C)) + \sum_i \frac{\partial K(l,C)}{\partial C} \left(\frac{\partial f^i_{end}(l,C)}{\partial C} - \frac{\partial f^i_{ini}(l,C)}{\partial C} \right) \tag{4.19}$$

Taking the derivative of C with respect to itself, using Leibniz Integration Rule and $K(l,C) = A(l, f_{end}(l,C))N(f_{end}(l,C))$ and $K(l,C) = A(l, f_{ini}(l,C))N(f_{ini}(l,C))$, then:

$$1 = \frac{1}{ln(2)K(l,C)} \frac{\partial K(l,C)}{\partial C} \sum_i \left((f^i_{end}(l,C) - f^i_{ini}(l,C)) \right) \tag{4.20}$$

Since $K(l,C) > 0$ for any $l > 0$ and $C > 0$ by the physics of the channel and $f^i_{end}(l,C) - f^i_{ini}(l,C) > 0, \forall C > 0, l > 0$, and the i bands are non/overlapping and $ln(2) > 0$ this implies that $\frac{\partial K(l,C)}{\partial C} > 0$. Then $\frac{\partial P(l,C)}{\partial C} = ln(2)K(l,C) > 0, \forall l > 0, C > 0$. Taking a second derivative to the C expression with respect to itself:

$$\begin{gathered} (\frac{\partial K(l,C)}{\partial C})^2 \textstyle\sum_i (f^i_{end}(l,C) - f^i_{ini}(l,C)) = \\ \frac{\partial^2 K(l,C)}{\partial C^2} \sum_i (f^i_{end}(l,C) - f^i_{ini}(l,C)) + \frac{\partial K(l,C)}{\partial C} \sum_i \left(\frac{\partial f^i_{end}(l,C)}{\partial C} - \frac{\partial f^i_{ini}(l,C)}{\partial C} \right) \end{gathered} \tag{4.21}$$

Thus,

$$\frac{\partial^2 P(l,C)}{\partial C^2} = (\frac{\partial K(l,C)}{\partial C})^2 \sum_i (f^i_{end}(l,C) - f^i_{ini}(l,C)) \tag{4.22}$$

where $(f^i_{end}(l,C) - f^i_{ini}(l,C)) > 0, \forall C > 0$, finite and non-overlapping and $\frac{\partial K(l,C)}{\partial C} > 0$. Thus, $\frac{\partial^2 P(l,C)}{\partial C^2} > 0$ □

Convexity of Approximate Model

The function $P(l,z)$ represents the minimum power required to transmit at a data rate z over a link of distance l. The function $P(l,z)$ was proven to be a convex function

with respect to z, using l as a parameter (Lemma 1). However, the exact model is complicated from a computational viewpoint. Let us determine the conditions for which the approximate model $\tilde{P}(l,z)$ in equation (4.10) is convex with respect to z, and having l as a fixed parameter. We study the case of $z < 2$ kbps. Since the α and β parameters come from fitting the data, the only variable left to analyze is the distance l. Note that ensuring that $\tilde{P}(l,z)$ is increasing and convex translates into the following inequalities:

$$ln(l)\frac{\partial a_1(z)}{\partial z} + \frac{ln(10)}{10}\frac{\partial a_2(z)}{\partial z} > 0 \tag{4.23}$$

$$ln(l)^2\left(\frac{\partial a_1(z)}{\partial z}\right)^2 + \frac{ln(10)}{10}\frac{\partial^2 a_2(z)}{\partial z^2} + \left(\frac{ln(10)}{10}\frac{\partial a_2(z)}{\partial z}\right)^2 + ln(l)\left(2\frac{ln(10)}{10}\frac{\partial a_1(z)}{\partial z}\frac{\partial a_2(z)}{\partial z} + \frac{\partial^2 a_1(z)}{\partial z^2}\right) \geq 0. \tag{4.24}$$

There is both a linear and a quadratic constraint upon l to ensure convexity. Since these constraints are also functions of z, the range of values of this parameter should be considered. From previous results for the fitting parameters, it is possible to determine some properties of the model for $z < 2$ kbps. In terms of the parameters of interest, $\alpha_1 < 0$, $\alpha_2 > 0$, $2\alpha_1 C + \alpha_2 > 0$, $\beta_1 > 0$ and $\beta_2 > 0$. Thus, for the choices of $a_1(z)$ and $a_2(z)$, the first and second derivatives of these functions with respect to z are $\dot{a_1}(z) > 0$, $\ddot{a_1}(z) < 0$, $\dot{a_2}(z) > 0$ $\ddot{a_2}(z) < 0$. Using these conditions, the constraints (4.23) and (4.24) can be simplified to

$$ln(l) > -\frac{ln(10)}{10}\frac{\dot{a_2}(z)}{\dot{a_1}(z)} + \max\left[0, -\frac{\ddot{a_1}(z)}{2\dot{a_1}(z)^2} + \sqrt{\frac{ln(10)}{10}\left(\frac{\dot{a_2}(z)\ddot{a_1}(z)}{\dot{a_1}(z)^3} - \frac{\ddot{a_2}(z)}{\dot{a_1}(z)^2}\right) + \frac{\ddot{a_1}(z)^2}{4\dot{a_1}(z)^4}}\right] \tag{4.25}$$

where the term under the square root is positive which ensures real values of l. Note that for each value of z there is a minimum value of l. Let us use the values of Case 1 in Table 4.1 to determine the (l,z) region for which the approximate model is convex. For these values, if the distance between to nodes l is at least 13 m, for any value of z <2 kbps the model will be convex. The limitation to l >13 m is related to the sampling of the distance used for computing the parameters of the approximate model. For all practical purposes the approximate model $\tilde{P}(l,z)$ is convex.

Table 4.2: Approximation parameters of α and β for $P(l, C)$, $l \in [0, 10]$ km, $C \in [0, 2]$ kbps, $\alpha = 1.5$,$s = 0.5$

	γ_1	γ_2	γ_3
α_1	5.2669e-6	-0.000157	-0.004575
α_2	-2.971e-5	0.000865	0.029306
α_3	0.000152	0.01809	2.4586
β_1	9.924e-6	-0.00027	0.012288
β_2	7.799e-6	-0.000219	1.0118
β_3	0.068091	1.3659	73.144

4.2 Lower Bound to Transmission Power in Underwater Networks

The problem of achieving minimum-energy multicast using network coding in a wireless network has been studied previously [24]. A wireless network, as presented in [24] can be represented through a directed hypergraph $H = (\aleph, A)$ where $\aleph$ is the set of nodes and A is the set of hyperarcs. A hypergraph is a generalization of a graph, where there are hyperarcs instead of arcs. A hyperarc is a pair (i, J), where i, the start node, is an element of $\aleph$,and J is the set of end nodes is a nonempty subset of A. Each hyperarc (i, J) represents a broadcast link from node i to nodes in the nonempty set J. Let us denote by z_{iJ} the rate at which coded packets are injected into hyperarc (i, J). If the cost function is separable, the optimization problem can be expressed as follows

$$\begin{aligned}
&\min \sum_{(i,J)\in A} \theta f(z_{iJ}/\theta) \\
&\text{subject to } z \in Z \\
&z_{iJ} \geq \sum_{j\in J} x_{iJj}^{(t)}, \forall (i, J) \in A, t \in T \\
&\sum_{\{J|(i,J)\in A\}} \sum_{j\in J} x_{iJj}^{(t)} - \sum_{\{j|(j,I)\in A\}, i\in I} x_{jIi}^{(t)} = \delta_i^{(t)} \\
&x_{iJj}^{(t)} \geq 0, \forall (i, J) \in A, j \in J, t \in T
\end{aligned} \tag{4.26}$$

with

$$\delta_i^{(t)} = \begin{cases} R & \text{if } i = s, \\ -R & \text{if } i = t, \\ 0 & \text{otherwise} \end{cases} \tag{4.27}$$

where T is a non-empty set of sink terminals, a source s, a multicast rate R, and a fixed transmission duty cycle at each link θ. $x_{iJj}^{(t)}$ represents the flow associated with terminal t, sent through hyperarc (i, J) and received by node $j \in J$.

In the underwater scenario this formulation is used to establish a lower bound on the transmission power required to achieve a multicast rate R. Assuming no interference for transmissions in different hyperarcs yields a separable cost function. Note that if interference was taken into account, the power to reach the desired data rate would increase. Then, the cost function $f(z_{iJ})$ for each particular hyperarc corresponds to a link transmission power $P(l, z_{iJ})$ in order to obtain the minimum transmission power required to achieve a data rate of z_{iJ}, where l represents the distance from i to the farthest node $j \in J$. For the lower bound computation, continuous transmission ($\theta = 1$) is assumed. A simplification of this problem can be made under the assumption that transmissions are omnidirectional, and considering the fact that if a node transmits over a certain range, all nodes in that range will be able to receive the information. This is proven in Lemma 4. This lemma assures that if a link between a transmitter i and receiver j at a distance l achieves a certain capacity C, another node k at distance $l' < l$ from node i, will be able to decode the information transmitted from i to j. Note that the transmission band is optimal for the link of distance l.

Lemma 4. *In an underwater acoustic channel with Gaussian noise,* $C(l, B(l, C)) < C(l', B(l, C))$ *for* $l' < l$, *if* $A(l, f) = (l/l_{ref})^{\alpha} a(f)^l$ *with* $\alpha \geq 1$ *and* $a(f) \geq 1, \forall f$.

Proof. Since $A(l, f) = (l/l_{ref})^{\alpha} a(f)^l$ and

$$\frac{\partial A(l, f)}{\partial l} = (\alpha/l_{ref})(l/l_{ref})^{\alpha-1} a(f)^l + (l/l_{ref})^{\alpha} ln(a(f)) a(f)^l > 0 \tag{4.28}$$

since $a(f) \geq 1$ and $l_{ref} > 0$. Then, $A(l,f) > A(l',f), l > l'$. Also, $\frac{K(l,C)}{A(l,f)N(f)} \geq 1, \forall f \in B(l,C)$ which implies $log_2(\frac{K(l,C)}{A(l,f)N(f)}) \geq 0, \forall f \in B(l,C)$. Let us compute the capacity of a link of distance l' when we use the optimum band and spectral density for a link of distance l and capacity C, i.e., $B(l,C)$ and $S(l,C,f) = K(l,C) - A(l,f)N(f), f \in B(l,C)$, respectively. Then,

$$C(l', B(l,C)) = \int_{B(l,C)} log_2 \left(1 + \frac{K(l,C) - A(l,f)N(f)}{A(l',f)N(f)}\right) df \tag{4.29}$$

$$> \int_{B(l,C)} log_2 \left(1 + \frac{K(l,C) - A(l,f)N(f)}{A(l,f)N(f)}\right) df = C \tag{4.30}$$

□

Finally, the model for this channel ensures that any value of z_{iJ} can be achieved if enough power is used. Thus, the constraint set Z can be dropped.

Although the problem for minimum-cost multicast is well known for wireless radio networks, the cost function presented here is different because it represents the minimum transmission power for an hyperarc transmitting at a data rate Z, which is given by the power needed to transmit at capacity $C = Z$, without assumption on technology or, more importantly, a specific transmission band which is usually the case for wireless radio networks. Thus, we are providing a lower bound valid for any acoustic underwater network for the case of Gaussian noise.

4.3 Performance Comparison

For this study, five schemes are considered. The first scheme corresponds to the lower bound to the transmission power using network coding given by solving the problem in Section 4.2 with $\theta = 1$. The second scheme corresponds to solving the problem in Section 4.2 for $\theta < 1$, in order to study the effect of using a duty cycle for link transmissions in underwater networks over interference and transmission power. The third scheme corresponds to using the paths chosen by the optimal scheme but establishing a SNR requirement for the transmission links with the objective of studying interference when the SNR requirement is increased. The links are considered to transmit

continuously. The schemes (4) and (5) consider implementations of network coding in a rateless fashion with the implicit acknowledgment (ACK) [72] and routing with link-by-link ACK using an ALOHA-like MAC layer. Let us explain in more detail each of the schemes.

1)Network coding based lower bound to transmission power: Transmission power is computed by solving the convex optimization problem in 4.2 and it provides a lower bound on the optimal transmission power for networks operating at low data rates. For this computation, continuous transmission, i.e., a duty cycle of $\theta = 1$ is used. This scheme is used as the gold standard to which the remaining schemes are compared. The no-interference assumption is assessed by computing the average percent of the randomly deployed networks that have at least a link which suffers from severe interference, i.e., a signal-to-interference ratio (SIR) below 3 dB.

2)Network Coding with optimal power consumption for links with fixed duty cycle: Transmission power is computed by solving the convex optimization problem in (4.2) for links with a fixed duty cycle, i.e., $\theta < 1$. This value provides a lower bound on the optimum power consumption for networks operating at low data rates when links have a particular duty cycle. By convexity of the cost functions used, this bound will be higher than for the previous scheme. This scheme is used to illustrate the effect upon transmission power and interference when the links transmit at a fixed duty cycle $\theta < 1$ by comparing this scheme to the previous scheme.

3)Network Coding with SNR requirement on link transmission: This scheme is a heuristic scheme that imposes an SNR requirement for transmissions. Using the subgraph selected by solving the problem in (4.2) for $\theta = 1$, it computes the SIR on the different links for a variety of SNR constraints using the models of transmission power and band in [47]. As in scheme (1) and (2), the percent of randomly deployed networks with at least a link with severe interference is computed. Results of this scheme suggest that continuous transmission with a moderate SNR requirement causes severe interference. A solution to this problem is to use of a duty cycle $\theta < 1$, similarly as in scheme (2) when there is an SNR requirement.

4)Network coding in rateless fashion with implicit ACK: For a concate-

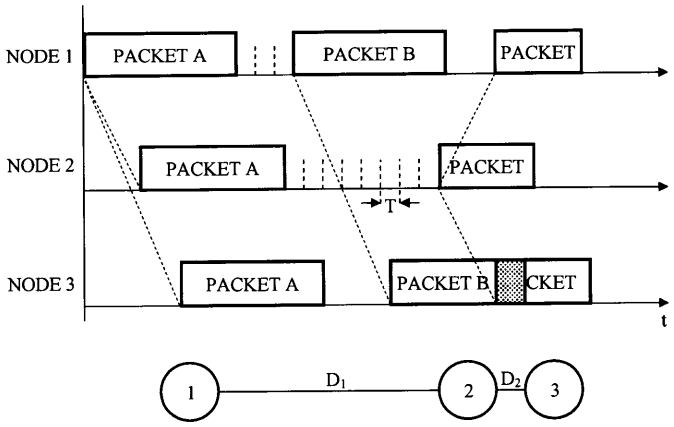

Figure 4-7: Medium Access Protocol for schemes (4) and (5)

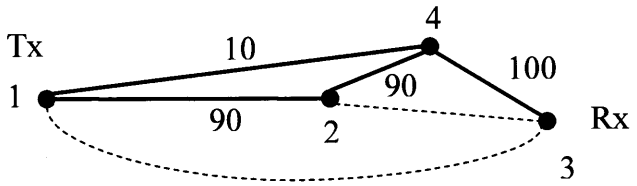

Figure 4-8: Subgraph selection for scheme (4). Selected subgraph with values of z_{iJ} to provide a unicast rate of 100, where dashed lines represent unused transmission ranges. Note that hyperarcs include in the same range, e.g., the possible hyperarcs with node 1 as the staring point are $1\{2\}$,$1\{2,4\}$,$1\{2,3,4\}$.

nated relay network as in Figure 4-7, the path between a source node and sink node is fixed and includes all relay nodes. If a node b is closer than node a to the collecting node, a is said to be upstream with respect to b, and b is said to be downstream with respect to a. For the concatenated relay network this ordering is quite natural. This problem was studied in [72]. For a two-dimensional scenario, subgraph selection [24] with linear and separable cost functions are used to determine the active links in the network and the transmission power required for each link. The cost function of each hyperarc is computed based on the approximate formulas for transmission power and bandwidth for a fixed SNR level [47]. The weight of each link is given by $DP(l, SNR)/B(l, SNR)$, where D is a constant common to all links related to the number of transmitted bits per burst and modulation used. For the performance computation using optimal modulation, i.e., Gaussian signaling, the weight for each link in the path is $DP(l, SNR)/C(l, SNR)$, where $C(l, SNR)$ is the function of capacity

related to the pair (l, SNR). We assume that the coding is over very large number of data packets. This could be extended computing an error probability based on error exponents [65]. Once the subgraph has been selected, if several links share the same transmitting node, this node will randomly choose the link to use. The weight of each link in the random choice is given by the fraction of data rate the optimization problem assigned to each of these links. Let us consider the network in Figure 4-8 as an example, where dashed lines represent unused hyperarcs. If node 1 transmits to both node 2 and 4, but we send a rate of 90 units through hyperarc z_{12}, while we send 10 units through hyperarc $z_{12,4}$, then when node 1 transmits it will do so 90 % of the time to reach node 2 only, and 10 % of time using enough power to reach nodes 2 and 4. Finally, we have to determine which nodes are upstream and downstream to each node in the subgraph. If the subgraph corresponds to a single path, the choice is clear. If there are multiple paths, we use the following heuristics: we start by ordering the nodes starting at the transmitter and looking at the nodes directly connected to it in the optimal subgraph. These nodes are ordered as follows: the node associated with the link with higher data rate from the transmitter is considered to be directly downstream from the source node, the node with the second highest data rate is considered to be downstream with respect to the previous one, and so on. In Figure 4-8, 2 is directly downstream of 1, and 4 is downstream of 2. Once all nodes connected to the transmitter (let us call this set of nodes S) are ordered, we proceed to order the nodes connected to S by a similar procedure as for the case of one node described before. In the example, $S = \{2, 4\}$ and the nodes connected to it are $\{3, 4\}$. If a node connected to one of the nodes in S has already been ordered, like node 4 in the example, the link is discarded keeping the previous order of the nodes. We update S with the nodes that were connected to S and not previously in it, until we reach the receiver. For the network example in Figure 4-8 the ordering is 1,2,4,3.

For this particular scheme, once a relay node gets its first coded packet, i.e., a packet formed by a random linear combination of data packets, it will transmit until the receiving node sends a confirmation that all the information has been received. The same happens at the source node. However, nodes eavesdrop on other trans-

missions. If a node receives a coded packet from a node further downstream with the same, or a greater number of degrees of freedom than what it has, it will stop transmitting and update its information if necessary. Degrees of freedom in this setting represents the number of packets that were linearly combined to form the coded packet as in [72]. The node will resume transmitting if an innovative packet, i.e., a packet with a new random linear combination of data packets useful for decoding the information, is received from a node upstream. The sink node will retransmit a coded packet with its degrees of freedom when a coded packet is received. This strategy assumes that there is a mechanism that informs the collecting node about the number of degrees of freedom that constitute the total message or that this number is fixed *a priori*.

5)Routing using link-by-link acknowledgement: For a concatenated relay network, the path between the source node and the sink node is fixed and includes all relay nodes. This problem was studied in [72]. For a two dimensional scenario, the sink and the source are chosen randomly and the shortest path is computed before starting data transmission in unicast. The weight of each link is computed based on the approximate formulas for transmission power and bandwidth for a fixed SNR level in the same fashion as the cost function per link of scheme (4). In the current scheme, every time a node receives a packet, it will retransmit the packet and send an acknowledgement to the previous node. Once a packet has been acknowledged, the node can start transmitting a new data packet in its queue. If it has no new packets to transmit, it will only transmit if a node upstream sends new information, or sends a previous packet, in which case the node will acknowledge this packet.

In terms of the physical layer, schemes (4) and (5) use both PSK modulation, which implies the use of a data rate in each link that is lower than capacity, and Gaussian signaling assuming that the encoding is over a large number of bits. In order to deal with the SNR requirement, we use an approximate model for the transmission power, high band edge frequency and bandwidth as functions of SNR similar to the work in [58]. When PSK modulation is used, the probability of packet error due to noise over the link from node i to j is obtained from the probability of bit

error by $P_{\text{packet Error}}(i,j) = 1 - (1 - P_{\text{bit error}})^n$, where n is the number of bits in the packet, and $P_{\text{bit error}}$ is computed using the standard PSK bit error probability. Note that nodes farther away from the transmitter have some probability of receiving the packet correctly. For Gaussian signaling, the probability of packet error due to noise is considered to be zero for all nodes in range, and 1 for all nodes further away.

In terms of the MAC layer, schemes (4) and (5) use an ALOHA-like MAC layer. This ALOHA protocol considers a fixed number of bits per data packet and uses the optimal transmission band for an SNR requirement per link. Thus, the duration of the transmitted packet depends on the transmission distance [47]. Every node has a probability to access the medium every T units of time following a Bernoulli process. Transmission delay is considered using a typical value of sound speed (1500 m/s). Figure 4-7 shows an example of using this MAC layer for three nodes with $D_1 >> D_2$. In this example, when node 1 transmit a packet to node 2, this packet also reaches node 3. Note that the duration of the packet transmitted from node 1 to node 2 (Packet A) is large compared to the packet transmitted from node 2 to node 3 because of the relation of distance to bandwidth/capacity for a fixed SNR value mentioned above [47]. Once node 1 has transmitted the packet it will try to transmit again, and it has some probability to start transmission every time slot T. Let us assume that node 2 has a data packet for node 3. Figure 4-7 shows the case when the data packet transmitted from node 2 to node 3 suffers a collision at node 3 with a new packet transmitted from node 1 to node 2. We consider that a collision at any receiver causes a loss of all packets involved in the collision for that receiver.

Let us study some numerical results that correspond to a network in which nodes are deployed randomly in a two dimensional space. Unicast connections of rate R are established, i.e., the network has one transmitter, one receiver chosen randomly, and, possibly, several relay nodes. The number of nodes ranges from 3 to 8. The transmission power lower bound, as an average over random deployments, will be compared with transmission power of schemes for routing and network coding. Also, a comparison between the schemes (1) and (2) in terms of interference is presented. Note that the transmission power lower bound is computed assuming that all nodes

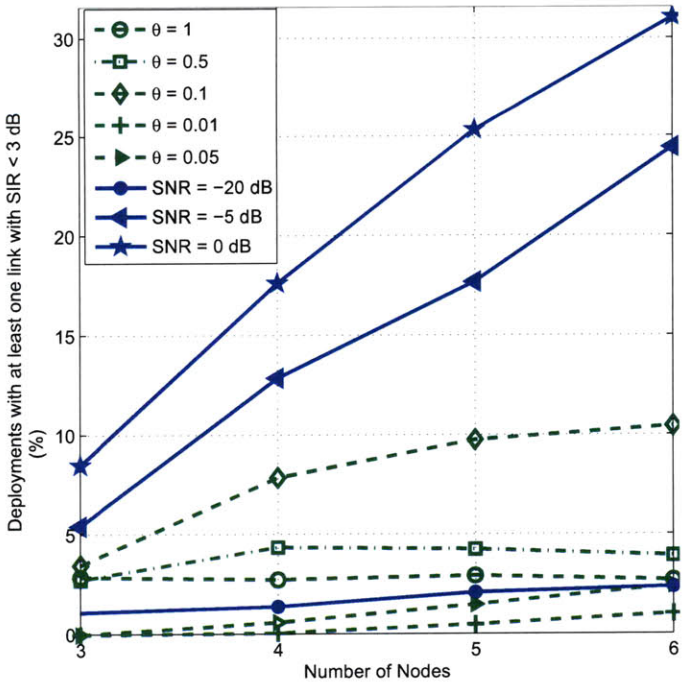

Figure 4-9: Percent of deployments in a fixed square of 5x5 km^2 with $SIR < 3$ dB in at least one link vs number of nodes deployed in that area, for the first three schemes. For scheme 1 $\theta = 1$, while scheme 2 is shown with different values of θ to achieve unicast rate of $R = 0.1$ kbps. Performance for scheme 3 is shown for different SNR values.

are within the transmission range of the others, i.e., full connectivity.

Figure 4-9 illustrates the effect of introducing a duty cycle θ (dashed lines) for link transmission under a random deployment in a 5 x 5 km^2 square. For $\theta = 1$ in Figure 4-9, which corresponds to scheme 1, note that less than 3 % of the random deployments cause severe interference (SIR $<$ 3 dB) over at least one link. This corroborates the no-interference assumption used during the analysis to obtain a lower bound for transmission power. Furthermore, this percentage seems to have little dependence on the number of nodes deployed in the network. When a value of $\theta < 1$ is used, Figure 4-9 shows that the percentage of deployments with SIR $<$ 3 dB increases for the initial decrements of θ, but decreases as θ becomes very small (below 1.5% for $\theta = 0.01$). Although this may seem counter intuitive, introducing a duty cycle causes the link to transmit at a higher data rate when it is active which translates to using more bandwidth and power to achieve that data rate in the underwater channel. Although duty cycle reduces interference by not using the channel continuously, the

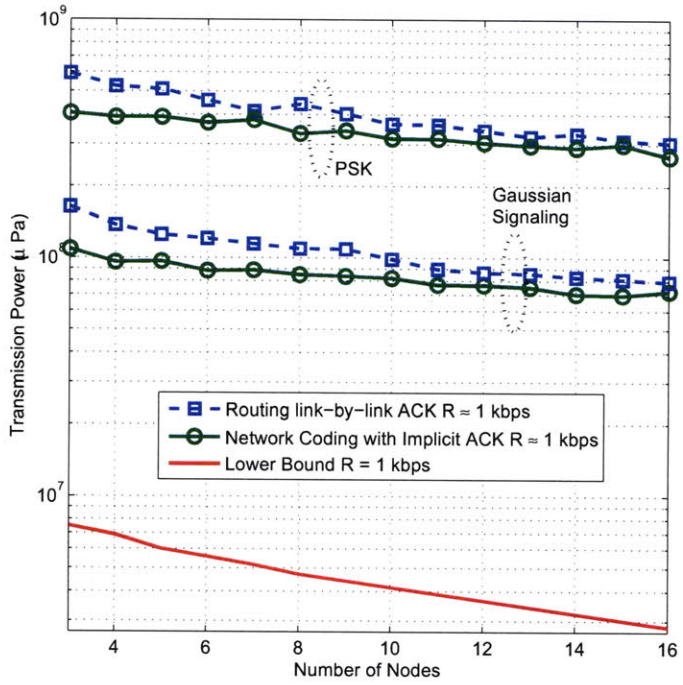

Figure 4-10: Average transmission power of networks deployed randomly on a 1x1 km^2 square. Network schemes operating at SNR = 10 dB and a lower bound for transmission power (scheme 1) is presented. Model used considers $k = 1.5$, $s = 0.5$ and $w = 0$ m/s.

combined effect with the increased transmission bandwidth and power causes more interference for initial decrements on the value of θ. This is a transient effect, and it has a breaking point for a small value of θ when the probability of having interference is small. As the value of θ decreases the transmission power can be shown to increase. This is an expected effect since the cost function is convex and the value of θ is a constant parameter to all links in this problem.

Figure 4-9 shows the results in continuous lines for the third scheme with different SNR requirements. The figure presents the percentage of random deployments that have at least one link suffering from severe interference. For very low SNR, the assumption of no-interference is justified. However, even for SNR = -5 dB the percentage of deployments with severe interference for a unicast connection increases dramatically, especially when the number of nodes in the network increases. A similar effect occurs when SNR = 0 dB. One way to reduce interference while having an SNR requirement is to use a similar approach as is scheme (2), i.e., to have a transmission

duty cycle in each of the links. Schemes (4) and (5) show an implementation using an ALOHA MAC protocol, where every link has an associated duty cycle when it has some data to transmit.

Let us compare the transmission power of scheme (4) and (5) to the lower bound using both PSK and Gaussian signaling in a 1 x 1 km^2. Figure 4-10 shows the average transmission power for different number of nodes in the network, both active and inactive, i.e., before determining the shortest path or solving the subgraph selection problem, with a transmission power computed to obtain a burst SNR = 10 dB. The average data rate for the different schemes is $R \approx 1$ kbps. This figure shows optimal signaling (Gaussian signalling) and a PSK modulation, which illustrates that close to 6 dB in the gap between a PSK modulation and the lower bound is due to the choice of the modulation. Notice that the gap between the average transmission power for Gaussian signaling and the lower bound of $R = 1$ kbps in Figure 4-10 is about 11 dB for scheme (4) and 13 dB for scheme (5). Also, it shows that this gap is maintained as more nodes are deployed. Some part of the gap is related to the MAC protocol used. Another is related to the 10 dB SNR requirement which is usually used for a practical implementation.

Figure 4-11 compares transmission power for different data rates using Gaussian signaling. The number of transmitted bits was kept constant, while the transmission probability over each link was increased to achieve the desired rate. Note that transmission power increases by 3 dB for scheme (4) while it increases by almost 5 dB for scheme (5) when the data rate is increased from 1 kbps to 2 kbps, i.e., the gap between scheme (4) and (5) increases as data rate increases. This figure shows also that the gap between schemes (4) and (5) is very low when the data rate is 0.2 kbps. Note that an increase in data rate is related to an increase in the collision probability in the ALOHA protocol. For the same setting, Figure 4-12 shows transmission energy for both schemes. The energy required for transmitting at the chosen data rates remains constant in the case of network coding, while it increases for routing when high data rates are attempted. Multiple transmissions of one packet are the main cause of the increased energy consumption for scheme (5), caused both by packet losses and long

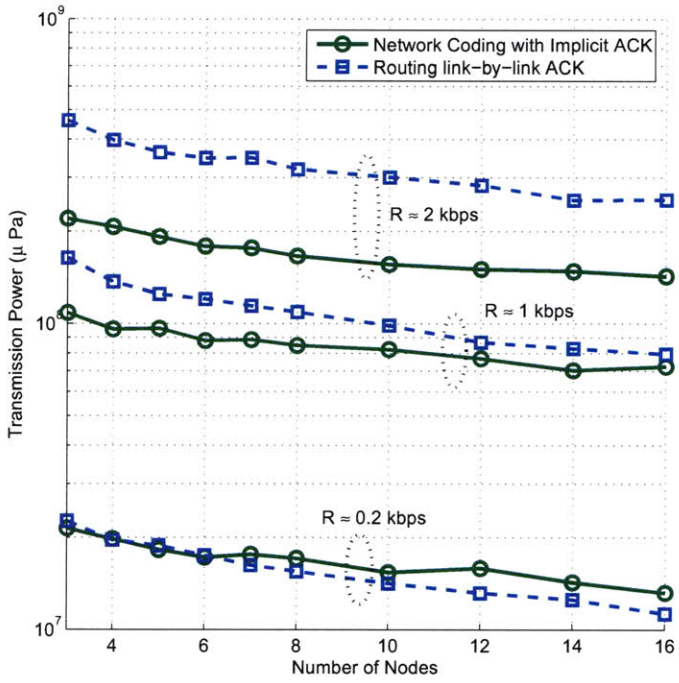

Figure 4-11: Average transmission power of networks deployed randomly on a 1x1 km^2 square. Network schemes operating at SNR = 10 dB using Gaussian signaling. Model used considers $k = 1.5$, $s = 0.5$ and $w = 0$ m/s.

delays in transmitting an ACK packet given the ALOHA MAC layer. While scheme (4) transmits innovative packets at each transmission, scheme (5) tries to retransmit the same packet if no ACK has been received. Consider the case of a long delay in transmitting an ACK, i.e., the packet was correctly received but the ACK is transmitted a long time after reception, scheme (5) can generate several transmissions of the same data packet. This involves an additional energy consumption. For the same number of transmissions, scheme (4) will transmit several innovative packets, which are useful in decoding the information at the receiver.

These results illustrate that coding, subgraph selection and the eavesdropping capabilities associated with network coding allow a better performance when the collision probability increases. However, notice that when transmission rates are low the benefits of network coding are less marked. This is explained by the fact that an implicit ACK might or might not be received by an upstream node. If it is not received, the node will keep transmiting innovative packets. This effect is particularly evident when Gaussian signaling since only nodes in range of transmission

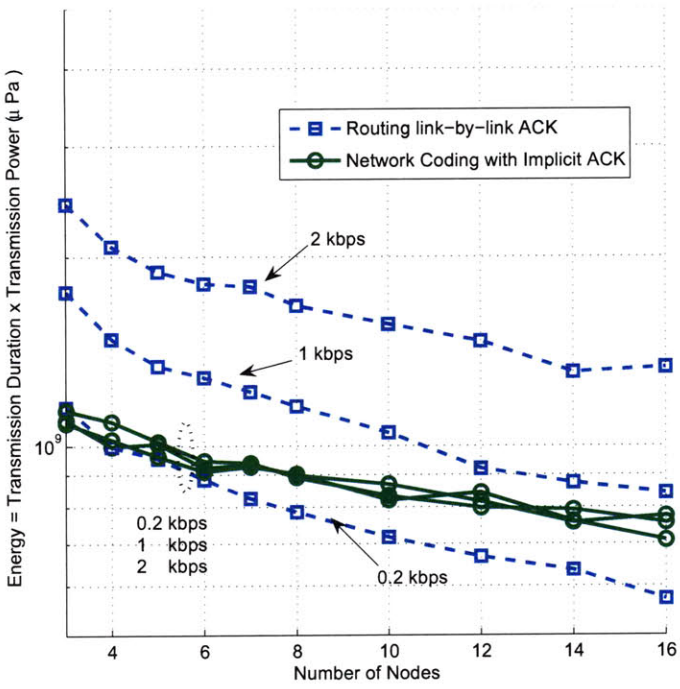

Figure 4-12: Energy for transmission for networks deployed randomly on a 1x1 km^2 square. Network schemes operating at SNR = 10 dB using Gaussian signaling. Model used considers $k = 1.5$, $s = 0.5$ and $w = 0$ m/s.

will correctly receive a packet. If we use a similar example as in Figure 4-7, node 1 will not receive any implicit ACK from node 2, and it will continue to transmit until informed that all information was received at the sink node. Thus, an explicit ACK procedure should be used if node deployments are likely to produce these situations, especially if only one node is actively generating new data packets.

Chapter 5

Capacity Scaling Laws for Underwater Acoustic Networks

This chapter considers the most complex scenario in this study, that of arbitrarily deployed networks with constantly active unicast connections and its relation to the number of nodes in the network. In particular, we are interested in determining the scaling of capacity given the size of the network, in terms of number of nodes deployed.

The seminal work by Gupta and Kumar [66] studied wireless networks, modeled as a set of n nodes that exchange information, with the aim of determining what amount of information the source nodes can send to the destination as the number n grows. The original results obtained for nodes deployed in a disk of unit area motivated the study of capacity scaling laws in different scenarios, ranging from achievability results in random deployments using percolation theory [67] or cooperation between nodes [68], to the impact of node mobility on the capacity of the network, e.g., [69]. Reference [70] provides a good overview of the different assumptions and scaling laws for radio wireless networks.

Existing capacity scaling laws for wireless radio networks correspond to scenarios for which $a(f) = 1$, or a constant greater than one, and $\alpha \geq 2$, e.g., [66], [67]. These results cannot be directly applied to underwater acoustic networks in which the attenuation varies over the system bandwidth and $\alpha \leq 2$. We study the scaling

laws under a model that considers a water-filling argument to assess the minimal transmission power and optimal transmission band as functions of the link distance and desired data rate. In particular, we study the case of arbitrarily deployed networks in a disk of unit area, and follow a similar procedure as in [66] to derive an upper bound on the capacity. In this sense, we provide an extension of the work in [66] under a more complicated power loss model.

We show that the amount of information that can be exchanged by each source-destination pair in an underwater acoustic network goes to zero as the number of nodes n goes to infinity. This occurs at least at a rate $n^{-1/\alpha}e^{-W_0(O(n^{-1/\alpha}))}$, where W_0 represents the branch zero of the Lambert function [25]. We illustrate that this throughput per source-destination pair has two different regions. For small n, the throughput decreases very slowly as n increases. For large n, it decreases almost as $n^{-1/\alpha}$. Thus, for large enough n, the throughput decreases more rapidly in underwater networks than in typical radio networks, because of the difference in the path loss exponent α.

5.1 Fixed Narrowband Model

Let us study the physical model of interference to obtain an upper bound on the transport capacity for transmission in an arbitrarily chosen narrow band in an underwater channel. The narrowband assumption allows us to consider the attenuation as a constant over that band. Although we use similar assumptions in terms of node deployment and connection set up to those in [66], the steps to derive the upper bound change somewhat in order to accomodate a more complex path loss model with different characteristics, e.g., $1 \leq \alpha \leq 2$ and $a(f) \geq 1$, instead of $\alpha \geq 2$ and $a(f) = 1$ considered in [66] for radio channels. In fact, we show that the upper bound is expressed in terms of one of the branches of the Lambert function, which is an implicit function.

We assume that the nodes are arbitrarily deployed in a disk of unit area, as in Figure 5-1, that each node has an intended destination node, and that the requirement

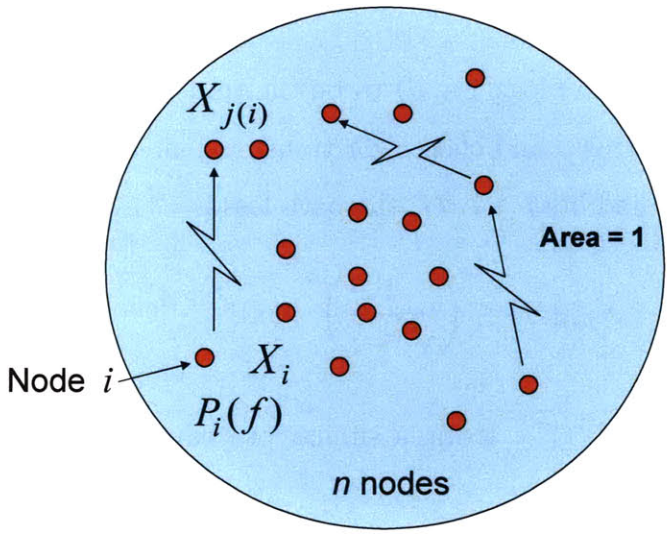

Figure 5-1: Network deployment scenario for the scaling laws of underwater acoustic networks.

for successful reception at node j of a transmission from node i is

$$\frac{\frac{P_i(f)}{A(|X_i - X_{j(i)}|, f)}}{N(f) + \sum_{k \in \tau, k \neq i} \frac{P_k(f)}{A(|X_k - X_{j(i)}|, f)}} \geq \beta \tag{5.1}$$

where X_i is the position of node i, $X_{j(i)}$ is the position of node j to which i is transmitting, and τ is the set of all nodes transmitting simultaneously in the same transmission sub-band and time slot. We assume that all sub-bands are in the narrow band [66], so that the attenuation is only dependent on the central frequency of the narrow band. The above expression can also be writen as

$$\frac{\frac{P_i(f)}{A(|X_i - X_{j(i)}|, f)}}{N(f) + \sum_{k \in \tau} \frac{P_k(f)}{A(|X_k - X_{j(i)}|, f)}} \geq \frac{\beta}{\beta + 1}. \tag{5.2}$$

Note that the parameter f is kept to keep in mind the frequency dependence, and to allow us to use these results in the following sections where we analyze more complex settings. We consider that λ is the throughput [bits/sec] of each node, the network transports λnT bits over T seconds, and that the average distance between source and destination of a bit is $\bar{L}$. As in [66], we define the transport capacity as

$\lambda n \bar{L}$ bits-meters per second.

Let us define $W = \triangle f \log_2(1+\beta)$ to be the transmission rate, where $\triangle f$ is the bandwidth of the narrow band chosen for transmission. Since $|X_k - X_{j(i)}| \leq \frac{2}{\sqrt{\pi}}$ for a disk of unit area, and $a(f) \geq 1, \forall f$, the path loss is

$$A(|X_k - X_{j(i)}|, f) \leq \left(\frac{2}{\sqrt{\pi} l_{ref}}\right)^{\alpha} a(f)^{2/\sqrt{\pi} - l_{ref}} \equiv \frac{\gamma_\alpha}{l_{ref}^{\alpha} a(f)^{l_{ref}}} \tag{5.3}$$

where $\gamma_\alpha = \left(\frac{2}{\sqrt{\pi}}\right)^{\alpha} a(f)^{2/\sqrt{\pi}}$. Using a similar procedure as in [66], from eq. 5.2 we have that

$$A(|X_i - X_{j(i)}|, f) \leq \frac{\beta+1}{\beta} \frac{\gamma_\alpha}{l_{ref}^{\alpha} a(f)^{l_{ref}}} \frac{P_i(f)}{\sum_{k \in \tau} P_k(f)} \,. \tag{5.4}$$

Let us sum over all transmitters $i \in \tau$ and use the definition of the path loss in expression (4.1):

$$\sum_{i \in \tau} |X_i - X_{j(i)}|^{\alpha} a(f)^{|X_i - X_{j(i)}|} \leq \gamma_\alpha \frac{\beta+1}{\beta}. \tag{5.5}$$

We define $r^A(h,b) = l(h,b)^{\alpha} a(f)^{l(h,b)}$, where $l(h,b)$ represents the distance between receiver and transmitter for the h-th hop of bit b, and H is defined as the number of hops performed in T seconds, which can be bounded by $H \leq \frac{WTn}{2}$ [66]. Summing over all sub-bands and time slots and dividing both sides by H, we obtain

$$\frac{1}{H} \sum_{b=1}^{\lambda n T} \sum_{h=1}^{h(b)} r^A(h,b) \leq \gamma_\alpha \frac{\beta+1}{\beta} \frac{WT}{H} \tag{5.6}$$

where $h(b)$ represents the h-th hop of a bit b. Since the function $r^A(l) = l^{\alpha} a(f)^l$ is increasing and convex for $l \geq 0$, $\alpha \geq 1$ and $a(f) \geq 1$, we have that

$$\left(\frac{\ln a(f)}{H} \sum_{b=1}^{\lambda n T} \sum_{h=1}^{h(b)} l(h,b)\right)^{\alpha} \exp\left(\frac{\ln a(f)}{H} \sum_{b=1}^{\lambda n T} \sum_{h=1}^{h(b)} l(h,b)\right) \leq (\ln a(f))^{\alpha} \gamma_\alpha \frac{\beta+1}{\beta} \frac{WT}{H}.$$

Let us define $\psi = (\ln a(f))^{\alpha} \gamma_\alpha \frac{\beta+1}{\beta} \frac{WT}{H}$, and note that $\psi \geq 0$. Noticing that the left

hand side of the above inequality is a Lambert function of the form $W^{\alpha}\exp W$, which is an increasing function when $W \geq 0$, is one of the key steps in our proof that are different to the work in [66]. Hence,

$$\frac{\ln a(f)}{H}\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h(b)} l(h,b) \leq \psi^{1/\alpha}\exp\left(-W_0\left(\frac{\psi^{1/\alpha}}{\alpha}\right)\right) \tag{5.7}$$

where $W_0(\cdot)$ is the branch zero of the Lambert function, using the nomenclature of [25]. This fact implies that

$$\lambda n\bar{L} \leq \frac{H}{T\ln a(f)}\psi^{1/\alpha}\exp\left(-W_0\left(\frac{\psi^{1/\alpha}}{\alpha}\right)\right). \tag{5.8}$$

Substituting for ψ in (5.8), we obtain

$$\lambda n\bar{L} \leq \frac{H^{\frac{\alpha-1}{\alpha}}}{T^{\frac{\alpha-1}{\alpha}}}\left(\gamma_{\alpha}\frac{\beta+1}{\beta}W\right)^{1/\alpha}\exp\left(-W_0\left(\frac{\psi^{1/\alpha}}{\alpha}\right)\right). \tag{5.9}$$

Since $H^{\frac{\alpha-1}{\alpha}}$ is an increasing function for $\alpha > 1$, and constant for $\alpha = 1$, then $H^{\frac{\alpha-1}{\alpha}} \leq \left(\frac{WTn}{2}\right)^{\frac{\alpha-1}{\alpha}}$. Another important step of our proof, different to [66], is to note that $W_0(\cdot)$ is an increasing function. Hence, we have that

$$W_0\left(\frac{\psi^{1/\alpha}}{\alpha}\right) \geq W_0\left(\frac{2\ln a(f)a(f)^{\frac{2}{\alpha\sqrt{\pi}}}}{\alpha\sqrt{\pi}}\left(\frac{\beta+1}{\beta}\right)^{1/\alpha}\frac{2^{1/\alpha}}{n^{1/\alpha}}\right).$$

Substituting these inequalities into expression (5.9), we obtain the scaling law:

$$\lambda n\bar{L} \leq \Phi W n^{\frac{\alpha-1}{\alpha}}\exp\left(-W_0\left(\Phi\frac{2\ln a(f)}{\alpha}\left(\frac{1}{n}\right)^{1/\alpha}\right)\right) \tag{5.10}$$

where

$$\Phi = \frac{2^{1/\alpha}}{\sqrt{\pi}}\left(\frac{\beta+1}{\beta}\right)^{1/\alpha}\left(a(f)^{\frac{2}{\sqrt{\pi}}}\right)^{1/\alpha}.$$

Since the zero-branch of the Lambert function satisfies $W_0(x) \geq 0, \forall x \geq 0$, the exponential term $\exp\left(-W_0\left(O\left(n^{-1/\alpha}\right)\right)\right)$ has values between 0 and 1. Note that as $n \to \infty$, the exponential term in the scaling law goes to 1. This implies that the

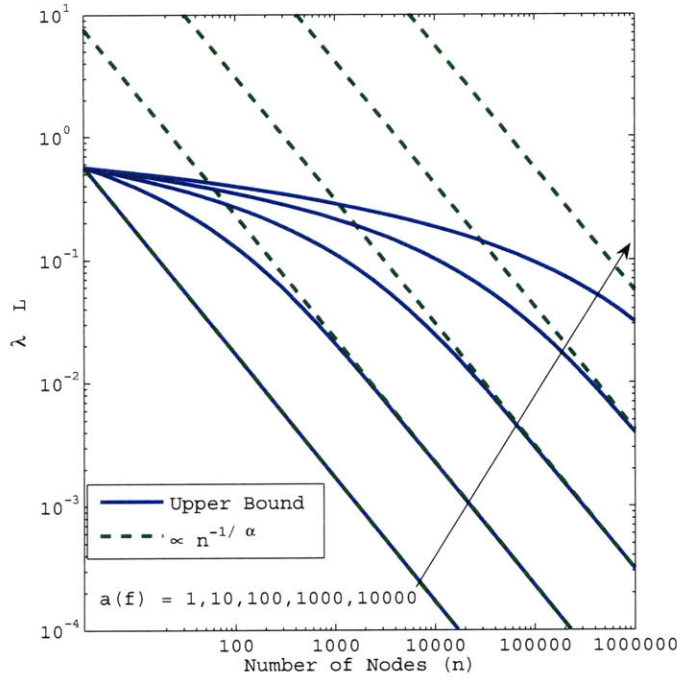

Figure 5-2: Upper bound on $\lambda\bar{L}$ for an arbitrarily chosen narrow band and different values of $a(f)$. $W = 1$ bps, $\alpha = 1$, $\beta = 2$, area $= 1$ km^2.

exponential term influences the scaling for small n, while for large enough n, the upper bound is $O(n^{\frac{\alpha-1}{\alpha}})$.

If we consider $a(f) = 1$, i.e., the same path loss model as in [66], and recall that $W_0(0) = 0$, we have that

$$\lambda n\bar{L} \leq \frac{1}{\sqrt{\pi}}\left(\frac{2\beta+2}{\beta}\right)^{1/\alpha} W n^{\frac{\alpha-1}{\alpha}} \tag{5.11}$$

which is the original result of [66]. We have thus proved that the result in [66] is valid for $\alpha \geq 1$.

Figure 5-2 and Figure 5-3 illustrate the upper bound on $\lambda\bar{L}$ for different values of $a(f)$ ranging from 1 to 10,000, which are characteristic of an underwater environment at different frequencies with l in [km]. For example, $a(f) = 1{,}000$ corresponds to a frequency of around 100 KHz. We have used $\alpha = 1$ and $\alpha = 2$ and the parameters specified in the figure for Figure 5-2 and Figure 5-3, respectively. We also plot dashed lines proportional to $n^{-1/\alpha}$. As expected, as n becomes larger, the exponential term of the upper bound becomes negligible, making the bound scale as $O(n^{-1/\alpha})$. However,

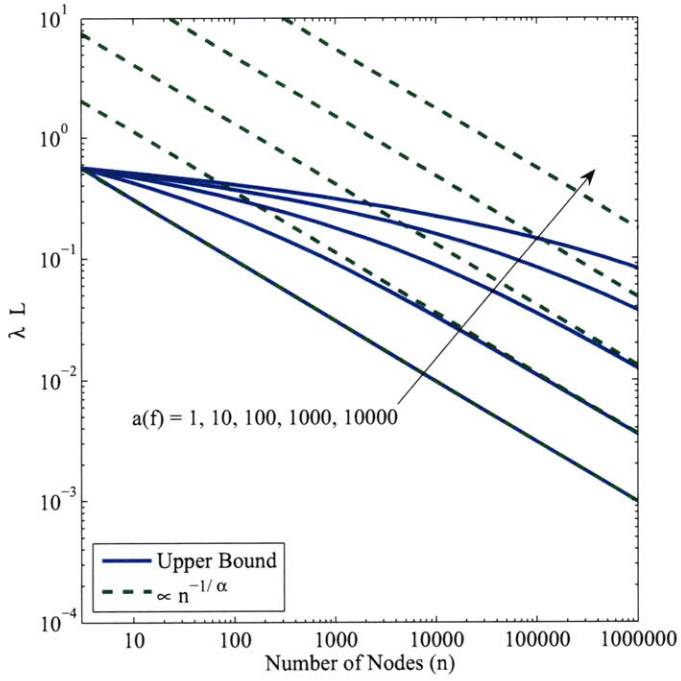

Figure 5-3: Upper bound on $\lambda\bar{L}$ for an arbitrarily chosen narrow band and different values of $a(f)$. $W = 1$ bps, $\alpha = 2$, $\beta = 2$, area $= 1$ km^2.

for small values of n, the bound begins at a common point for the different $a(f)$ values, and decays very slowly. Figure 5-2 and Figure 5-3 also illustrate that the value of $a(f)$ determines the transition between these two operating regions: the larger $a(f)$, the greater n has to be before transitioning. Of course, if we use a transmission band with high $a(f)$ each node will have to be able to transmit at higher power to reach its destination. In the underwater channel, this also means that a higher center frequency is required because $a(f)$ is an increasing function of f.

Finally, Figure 5-2 and Figure 5-3 show that $\lambda\bar{L}$ remains almost constant for $n \leq$ 100 nodes, $a(f) > 100$ and a disk area of 1 km^2, which corresponds to densities of up to 100 nodes per km^2. Note that the expected density of nodes in an underwater network is usually much lower given the current applications for which they are deployed, e.g., environmental measurements. Thus, $\lambda\bar{L}$ is almost constant for all practical purposes.

5.2 Low power - Narrowband Case

As mentioned in Chapter 4, one of the characteristics of the underwater acoustic channel is that the optimal transmission band using the waterfilling principle depends strongly on the distance of a link [47]. In particular, if the transmission power of a node is very low, then nodes will optimally transmit in different bands corresponding to different transmission distances. Thus, interference will come only from nodes transmitting in the same band. We have derived an expression for the power under these assumptions in Chapter 4. In order to assign disjoint transmission bands, we divide the total transmission band of the system into non-overlapping bands of width Δf. We use $f_c(l)$ as the mapping between the transmission distance and the corresponding transmission band for a low-power/narrow-band scenario. Thus, if a node transmits to another node at a distance l, we assign the transmission band centered at the frequency $f_c(l)$ as in Figure 4-6.

The capacity analysis is similar to Section 5.1 if $a(f)$ is replaced by $a(f_m)$ for each of the bands, where f_m is the central frequency of transmission band m. Note that this analysis is inherently different to that in [66], which does not consider any frequency dependence of the path loss model. Let us assume that each node is capable of transmitting at $\triangle W$ bps in each band, where $\triangle W = \triangle f \log_2(1+\beta)$, and $\triangle f$ is the bandwidth of each non-overlapping band.

5.2.1 Multi-Node Hopping

Note that the definition of H changes slightly when we allow multi-node hopping. In this case, $H \leq \frac{T|\Gamma|\triangle Wn}{2} = \frac{TWn}{2}$, where Γ is the set of sub-bands used by the network, and $W = |\Gamma|\triangle W$.

For each of the different bands, the analysis is as before up to eq. (5.5). At this point, we define $\gamma_\alpha(f_m)$ as γ_α for band m. Summing over all sub-bands and time slots, we obtain

$$\sum_{s\in S}\sum_{m\in\Gamma}\sum_{i\in\tau}|X_i - X_{j(i)}|^\alpha a(f_m)^{|X_i - X_{j(i)}|} \leq \frac{\beta+1}{\beta}\triangle WT\sum_{m\in\Gamma}\gamma_\alpha(f_m) \tag{5.12}$$

where S is the set of time slots. We can use the fact that $a(f_m) \geq a_{min}$, where $a_{min} = \min_{m\in\Gamma} a(f_m)$. In the underwater scenario, $a_{min} = a(f_{min})$ because $a(f)$ is an increasing function of f. Defining $r^A(h,b,f_{min}) = l(h,b)^\alpha a(f_{min})^{l(h,b)}$, and following similar steps that lead to eq. (5.6) we get

$$\frac{1}{H}\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h(b)} r^A(h,b,f_{min}) \leq \frac{\beta+1}{\beta}\frac{\triangle WT}{H}\sum_{m\in\Gamma}\gamma_\alpha(f_m)\,.$$

Defining $\psi = (\ln a(f_{min}))^\alpha \frac{\beta+1}{\beta}\frac{\triangle WT}{H}\sum_{m\in\Gamma}\gamma_\alpha(f_m)$, we can use a similar procedure as in the previous Section to show that

$$\lambda n\bar{L} \leq \frac{H^{\frac{\alpha-1}{\alpha}}}{T^{\frac{\alpha-1}{\alpha}}}\left(\frac{\beta+1}{\beta}W\sum_{m\in\Gamma}\gamma_\alpha(f_m)\right)^{1/\alpha}\exp\left(-W_0\left(\frac{\psi^{1/\alpha}}{\alpha}\right)\right). \tag{5.13}$$

Finally, using the inequality $H \leq \frac{TWn}{2}$, we obtain

$$\lambda n\bar{L} \leq \frac{2^{1/\alpha}}{\sqrt{\pi}}\left(\frac{\beta+1}{\beta}\right)^{1/\alpha}\left(\sum_{m\in\Gamma}a(f_m)^{\frac{2}{\sqrt{\pi}}}\right)^{\frac{1}{\alpha}} M^{\frac{\alpha-1}{\alpha}}Wn^{\frac{\alpha-1}{\alpha}}\exp\left(-W_0\left(O\left(n^{-1/\alpha}\right)\right)\right).$$

Thus, the scaling law now becomes

$$\lambda n\bar{L} \leq \Phi Wn^{\frac{\alpha-1}{\alpha}}\exp\left(-W_0\left(\Phi\frac{2\ln a(f_{\min})}{\alpha}\left(\frac{1}{n}\right)^{1/\alpha}\right)\right)$$

where

$$\Phi = \frac{2^{1/\alpha}}{\sqrt{\pi}}\left(\frac{\beta+1}{\beta}\right)^{1/\alpha}\left(\frac{1}{|\Gamma|}\sum_{m\in\Gamma}a(f_m)^{\frac{2}{\sqrt{\pi}}}\right)^{1/\alpha}.$$

The scaling law is similar in structure to the one obtained in Section 5.1. However, the constant Φ depends on the *average* of a function of the absorption coefficients at $f_m, \forall m \in \Gamma$ instead of a particular value. Again, if $a(f) = 1, \forall f$ the result reduces to that of [66].

5.2.2 Direct Transmissions

If we constrain our system to perform direct transmissions only (single-hop), using the fact that there is an assignment of frequency bands in terms of the distance, we can consider that $h(b) = 1, \forall b$, i.e., only one hop. Given the distance-band separation property mentioned in previous sections, the problem can be thought of as solving for several networks that lie on top of each other, in different layers with no cross-layer interference. Membership to the layers is based on the distance of the connection. In other words, each transmission band m will have n_m transmitters, where $n = \sum_{m \in \Gamma} n_m$ constitutes the total number of nodes in the network since each transmitter has only one intended destination.

These facts cause a different capacity scaling for each of the transmission bands, i.e., the scaling for each transmission band will have the form of expression (5.10) with $\triangle W$ instead of W and n_m instead of n to obtain the scaling for band m.

5.3 High Power - Wide Band Case

In this scenario, nodes have enough power to transmit in a wide band B, which implies that the absorption cannot be considered to be constant over the band. The band B is again chosen using a waterfilling argument. The SINR requirement can now depend on the frequency, that is

$$\frac{\frac{P_i(f)}{A(|X_i - X_{j(i)}|, f)}}{N(f) + \sum_{k \in \tau, k \neq i} \frac{P_k(f)}{A(|X_k - X_{j(i)}|, f)}} \geq \beta(f) \, . \tag{5.14}$$

We define W as the data rate over the entire band, computed as

$$W = \int_{f \in B} \log_2(1 + \beta(f)) df \, . \tag{5.15}$$

If we assign a transmission rate to every sub-band df of $dW = \log_2(1 + \beta(f))df$, the analysis for each frequency is similar as in Section IV. Letting $\triangle f \rightarrow 0$, renaming

$\triangle f$ as df and replacing the sums by integrals, we have that

$$\frac{1}{H}\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h(b)} Wr^{A}(h,b,f_{min}) \leq \frac{T}{H}\int_{W}\frac{\beta(f)+1}{\beta(f)}\gamma_{\alpha}(f)dW$$

$$= \frac{T}{H}\int_{B}\frac{\beta(f)+1}{\beta(f)}\gamma_{\alpha}(f)\log_2(1+\beta(f))df \tag{5.16}$$

where $f_{min} = \arg\min_f a(f)$ and H can be shown to have the bound $H \leq \frac{TWn}{2}$ using the definition of W (eq. (5.15)). Following the procedure of Section IV, we show that the scaling law for the high power - wide band case has the form

$$\lambda n\bar{L} \leq \Theta W n^{\frac{\alpha-1}{\alpha}} \exp\left(-W_0\left(\Theta\frac{2\ln a(f_{\min})}{\alpha}\left(\frac{1}{n}\right)^{1/\alpha}\right)\right)$$

where

$$\Theta = \frac{2^{1/\alpha}}{\sqrt{\pi}}\left(\frac{1}{W}\int_{B}\frac{(\beta(f)+1)\,a(f)^{\frac{2}{\sqrt{\pi}}}\log_2(1+\beta(f))}{\beta(f)}df\right)^{1/\alpha}.$$

Chapter 6

Practical Considerations

After discussing different network coding scenarios, we must answer the question of wether this results are relevant and applicable in practice. There are two main objections to the use of random linear network coding. The first is the perception that very large field sizes are necessary in order to achieve good performance. The use Galois fields with a larger field size requires more complex basic operations, e.g., multiplication, in order to code/decode. Other fountain codes typically operate with binary symbols, i.e., a field size of 2.

The second objection is that the decoding complexity of random network coding, which is $O(M^3)$ for decoding a batch of M data packets, is larger than other fountain codes. For example, Raptor codes requires $O(M \log(1/\epsilon))$ operations to recover the original data with $M(1+\epsilon)$ packets being received [41], and LT codes require $O(k \log(k/\delta))$ operations to decode from $M + O(\sqrt{M}\log^2(M/\delta))$ coded packets with probability $1-\delta$ [40].

We try to address these objections in this chapter. We deal first with the problem of field size, showing that the use of a small field size causes very little degradation on performance from a receiver's perspective, especially if the number of data packets M to be combined is moderately large. Even if a $GF(2)$ is used, i.e., a Galois field of size 2, which only requires XOR operations in orther to code the data packets, a receiver at most needs $M+2$ coded packets in average in order to decode. Note that at least M coded packets have to be received in order to decode the original M data

packets.

We then deal with the problem of decoding complexity. In particular, we propose the use of systematic random linear network coding as an approach that allows us to reduce decoding complexity and rely on small field sizes, while maintaining almost the same performance as a random linear network coding approach that uses large field sizes. Finally, we illustrate these results through an example that adapts the problem of coding in large latency TDD channels, studied in Chapter 2.

6.1 Effect of Field Size

Let us consider the effect of the field size from a receiver's perspective. This allows us to model the effect of the field size separately from other effects and to avoid making many assumptions about the channel or network topology. We simply assume that M data packets are combined using random linear network coding and that the network also uses random linear network coding. We also assume that the coded packets traverse a channel/network in which coded packets can be lost (suffer erasures) before being received. However, we do not make any other assumptions about the nature of the channel/network. For example, it is not necessary the event of loosing a packet is independent from the event of loosing others. In this sense, our modelling of the effect of the field size is useful to any random linear network coding network in which packet losses occur.

Using random linear network coding arguments, we can model the process of decoding M packets from the random linear coded packets received at a node as a Markov chain, as in Figure 6-1. A transition occurs when a new coded packet is successfully received at a node, while the states in the Markvo chain represent how many dofs are needed in order to decode all M data packets. Note that the arrival of a new coded packet can cause one of two effects: a self-transition, because the coded packet provides a combination of the original packets that is linearly dependent on the combinations that have been previously received at the receiver, or a transition to the next state, when the new coded packet provides an independent linear combination

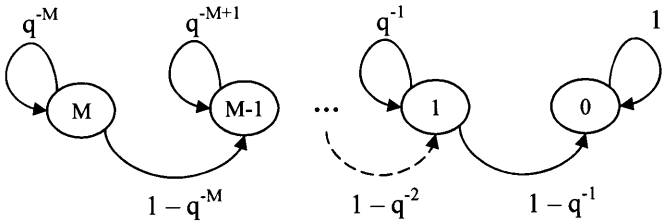

Figure 6-1: Markov chain of the degrees of freedom required to decode. Transitions occur when a new coded packet is successfully received by a node.

(a new dof).

The transition probability matrix for this problem is

$$P_q = \begin{bmatrix} q^{-M} & 1-q^{-M} & 0 & \cdots & 0 & 0 \\ 0 & q^{-M+1} & 1-q^{-M+1} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ 0 & 0 & 0 & \cdots & q^{-1} & 1-q^{-1} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix}.$$

Let us provide a full characterization of this problem by obtaining the moment generating function of the number of coded packets that need to be received before successfully decoding the information. We state this result in the following lemma.

Lemma 5. *The moment generating function $M_n(s)$ of the number of coded packets that need to be received before successfully decoding all the data when n linearly independent coded packets are needed to decode is given by*

$$M_n(s) = \frac{e^s}{1 - P_{n\to n}e^s} P_{n\to(n-1)} M_{n-1}(s) \tag{6.1}$$

with $M_0(s) = 1$

Proof. It follows the same steps as the proof of Lemma 1. □

Now that we have a full characterization of the problem, let us bound the average number of coded packets that need to be received before successfully decoding the

M packets in order to gain some insight as to what to expect in terms of average performance. Clearly, at least M coded packets must be received before being able to decode. Thus, a trivial lower bound is M. The upper bound is given by the following lemma.

Lemma 6. *If M data packets are encoded using random linear network coding with a field size q, then the mean number of coded packets that have to be received before completely decoding the original packets is upper bounded by*

$$\min\left\{M\frac{q}{q-1}, M+1+\frac{1-q^{-M+1}}{q-1}\right\} \tag{6.2}$$

Proof. Let us define the minimum number of coded packets received to decode as N_c. Then $E[N_c] = \sum_{k=1}^{M} \frac{1}{1-q^{-k}}$.

Since $q^{-k} \leq q$ for $q \geq 2$ and $k \geq 1$, then $E[N_c] \leq \sum_{k=1}^{M} \frac{q}{q-1} = M\frac{q}{q-1}$ which shows the first bound, proved in Reference [39].

The second bound comes from

$$E[N_c] = M + \sum_{k=1}^{M} \frac{1}{q^k-1} \leq M + \sum_{k=0}^{M-1} q^{-k} \tag{6.3}$$

$$= M + \frac{1-q^{-M}}{1-q^{-1}} = M+1+\frac{1-q^{-M+1}}{q-1} \tag{6.4}$$

where we have used the fact that $q^k - 1 \geq q^{k-1}$ for $k \geq 1$ and $q \geq 2$. □

One important conclusion of this Lemma is that $E[N_c] \leq M+2, \forall q \geq 2$, i.e., on average the number of coded packets needed to decode the M original packets will be between M and $M+2$ for any field size. Note that, if $M >> 2$, we expect that a scheme using $q = 2$ and one using larger q will have a small difference in performance.

Figure 6-2 illustrates the upper and lower bounds for a wide range of field sizes. It also shows that the upper bound $E[N_c] \leq M\frac{q}{q-1}$ becomes the dominant bound for large q, while the $E[N_c] \leq M+1+\frac{1-q^{-M+1}}{q-1}$ is the dominant bound for small q. Finally, Lemma 3 shows that the new bound on $E[N_c]$, i.e., $E[N_c] \leq M+1+\frac{1-q^{-M+1}}{q-1}$, will be the dominant bound up to a field size q that increases as M increases.

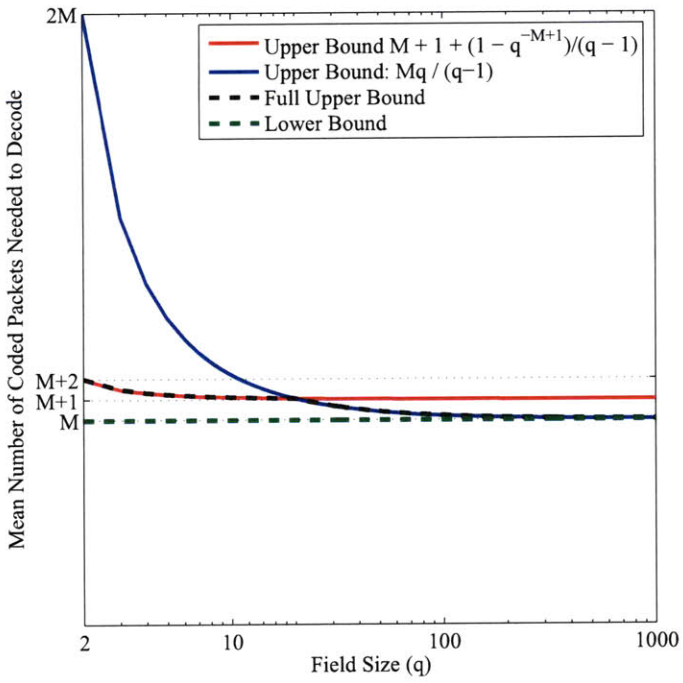

Figure 6-2: Upper and Lower bounds on the mean number of coded packets needed at the receiver in order to decode versus the field size q.

Lemma 7. *Exists* q^* *for which* $M\frac{q}{q-1} \leq M+1+\frac{1-q^{-M+1}}{q-1}, \forall q \geq q^*, M \geq 1$. q^* *scales as* $O(M)$.

Proof. Since both bounds are decreasing functions and $\lim_{q\to\infty} M\frac{q}{q-1} = M$ and $\lim_{q\to\infty} M+1+\frac{1-q^{-M+1}}{q-1} = M+1$, q^* exists. Note that $M+1+\frac{1-q^{-M+1}}{q-1} \in [M+1, M+2], \forall q \geq 2$. Thus, the bounds must cross in this region. Let us consider $\alpha \in [1,2]$ and find q_α that satisfies $M\frac{q_\alpha}{q_\alpha-1} = M+\alpha$, i.e., $q_\alpha = M/\alpha+1$. Thus, $q_2 = M/2+1 \leq q^* \leq M+1 = q_1$, which concludes the proof. □

6.2 Systematic Network Coding: Benefits in Decoding Complexity

Systematic network coding consist of sending the original packets initially, and transmitting random linear combinations of the packets in subsequent transmissions. This is an idea that has been considered in previous work. Reference [42] considers the implementation of systematic and random linear network coding on battery constrained

mobile devices with low computational capabilities using a field size of 2. Reference [43] uses systematic network coding as a MAC layer mechanism for WiMAX, called MAC layer Systematic Network Coding (MSNC), which transmits the packets once uncoded, and employs random network coding for retransmissions. The authors showed that it achieves the optimum performance for delay sensitive applications while achieving the same overhead level as an equivalent MAC layer random network coding scheme [44]. Reference [45] proposes variants of the systematic network coding idea for online applications, showing that they are both throughput optimal and have better decoding delay performance than other online network coding approaches. However, these references have not fully characterized and modelled the effects on the decoding complexity of systematic network coding.

We study the gains on average decoding complexity of using systematic network coding as an alternative to pure random linear network coding. Although we focus our analysis on the case of an erasure channel in which packets can suffer erasures independently from other packets (IID Bernoulli with parameter Pe), the results apply to more general networks which can be translated into an equivalent erasure channel. Also, the techniques and mechanisms used to derive the average decoding complexity results can be extended to characterize the case of time-dependent erasure channels.

6.2.1 Model

Systematic network coding is used to transmit the information reliably. Initially, the original packets are transmitted. Subsequente transmissions consist of random linear combinations of the original packets. Random linear coded packets are transmitted until the receiver has enough independent linear combinations in order to decode the original packets. We assume a packet erasure channel where erasures are IID Bernoulli with parameter Pe.

We also assume that the decoder can recognize uncoded packets and use this knowledge to speed up the decoding process. Note that, in vector form, each packet CP_j, coded or not, can be expressed as a linear combination of the original packets

as $CP_j = [a_{j1}a_{j2}...a_{jM}][P_1...P_M]^T$ where a_{ji}'s are the random linear coefficients and P_i's are the original data packets. Clearly an original packet P_j can be represented as $[0..010..0][P_1..P_{j-1}P_jP_{j+1}..P_M]^T$.

It can be shown that if a receiver has $D \leq M$ uncoded packets, it can decode the remaining $M - D$ coded packets in $O((M-D)^3)$ operations if it uses Gaussian elimination. The decoding procedure requires performing Gaussian elimination in a matrix that can be reordered as

$$[CP_j]=\begin{bmatrix} 1 & 0 & .. & 0 & .. & 0 \\ 0 & 1 & .. & 0 & .. & 0 \\ : & : & : & : & & \\ 0 & 0 & .. & 1 & .. & 0 \\ a_{(D+1)1} & a_{(D+1)2} & .. & a_{(D+1)D} & .. & a_{(D+1)M} \\ : & : & : & : & & \\ a_{M1} & a_{M2} & .. & a_{MD} & .. & a_{MM} \end{bmatrix} P'$$

where $[CP_j]$ constitutes the vector of (coded) packets received, P' is the vector of original packets in the appropriate order to have the adequate matrix structure.

Note that the uncoded packets will be used to perform a forward elimination only in the coded packets that are received and not in the other uncoded packets. Furthermore, the operation will be restricted to a single column in the matrix corresponding to the equivalent uncoded packet, i.e., no operations have to be performed for the columns that are known to be zero in the uncoded packet. Finally, no backward substitution step is needed for the uncoded packets.

6.2.2 Decoding Complexity

Let us compute the average decoding complexity of using systematic network coding. Considering the characteristics of the channel, we can formulate the following Lemma.

Lemma 8. *The average number of operations required to decode using Gaussian elim-*

ination on a full rank matrix when systematic network coding is used to transmit M original data packets where each transmitted coded packet undergoes erasures that are IID Bernoulli with parameter Pe, grows as $O(M^3Pe^3)$.

Proof. Note that the number of uncoded packets D that are received depends only on the channel and the number of original data packets M that are transmitted, and it is given by a binomial distribution with probability Pe of a packet being erased. Thus, $E[D] = M(1-Pe)$, $E[D^2] = (1-Pe)M + (1-Pe)^2M(M-1)$, and $E[D^3] = (1-Pe)M + 3(1-Pe)^2M(M-1) + (1-Pe)^3M(M-1)(M-2)$ which are necessary for the computation of the average number of operations.

Assuming that the matrix of coefficients is full rank, and that the total number of operations of Gaussian elimination is given by $An^3 + Bn^2 + Cn$, for some constants A, B, C, for a procedure on an $n \times n$ matrix. The number of operations required to decode the M original data packets is given by two effects: 1) elimination of contribution of uncoded packets in the linear combinations of coded packets, which requires $A_1(M-D)D$ operations, where A_1 is a constant, and 2) a full Gaussian elimination in the remaining $(M-D)\times(M-D)$ matrix, which requires $A(M-D)^3 + B(M-D)^2 + C(M-D)$ operations. Then,

$$E[A_1(M-D)D] = (A_1Pe(1-Pe))(M^2-M) \tag{6.5}$$

and

$$E[A(M-D)^3 + B(M-D)^2 + C(M-D)] = A(MPe)^3 + (3A+B-3APe)(MPe)^2 + (A-3APe+2APe^2+B-BPe+C)(MPe). \tag{6.6}$$

The average number of operations comes from adding these two terms, which is $O(Pe^3M^3)$. □

This result shows that systematic network coding allows us to reduce computational complexity by a factor of Pe^3 on average with respect to pure random linear network coding. Note that decoding random linear network coded packets using

Gaussian elimination on a full rank matrix takes $O(M^3)$ operations. This can be a considerably large value, e.g., in a channel with $Pe = 0.1$ on average we will require 1000 times fewer operations in order to decode.

Let us consider the case in which Gaussian elimination is used not just to decode the information but also to determine if a newly received coded packet is useful, i.e., linearly independent of the information already at a receiver. As before, we assume that no decoding operations are performed when uncoded packets are received. If only D out of M uncoded packets are received, all the remaining packets will be linear combinations of the original packets. Let us consider the particular case of multiplication operations, although only small adjustments are necessary to consider the case of additions. The number of multiplication operations to determine if a coded packet is linearly independent (and stored) or if it should be discarded is given by

$$D\,(1+K) + \mathbf{1}_{\{\beta>D\}} \sum_{u=1}^{\beta-D} (M - D - u + 1 + K)\,, \tag{6.7}$$

where β indicates the number of independent linear combinations previously received, K represents the size of the data in the packet in number of symbols of size $log_2 q$ bits, and $\mathbf{1}_{\{s\in S\}}$ is 1 when $s \in S$ and zero otherwise. The following Lemma shows that our previous result holds even when this Gaussian elimination procedure is performed to determine if coded packets constitute new dofs, regardless of the field size q.

Lemma 9. *The average number of operations required to decode, when systematic network coding is used and Gaussian elimination is performed on every new packet to determine linear independence, is upper bounded by a function that is $O(M^3 Pe^3)$, regardless of the field size q, when the erasures are IID Bernoulli with parameter Pe.*

Proof. We shall prove for the case of multiplication operations (# Mult). Only small changes are necessary for the case of addition operations, and they are related to finding an equivalent expression to expression (6.7). We can use the Markov Chain model that considers the effect of field size to solve this problem. The number of operations required to process a newly received packet depends on whether the packet is uncoded (no operations) or not, and how many uncoded packets were received, say

D, and how many coded packets were previously received, say $\beta - D$. This number of operations could be thought of as the time or penaly for a transition when the system is in a particular state. In order to upper bound the average number of operations, let us first consider the case in which D uncoded packets are received, i.e.,

$$E[\#\text{ Mult}|D] = \sum_{m=1}^{M-D} \frac{D+DK}{1-q^{-m}} + \kappa \sum_{m=1}^{M-D-1} \frac{1}{1-q^{-m}} + (-K-1/2) \sum_{m=1}^{M-D-1} \frac{m}{1-q^{-m}} + (-1/2) \sum_{m=1}^{M-D-1} \frac{m^2}{1-q^{-m}} \tag{6.8}$$

where $\kappa = \frac{-D}{2}(2M - D + 2K + 1) + M^2/2 + MK + M/2$, and the q^{-m} factors come from a random linear network coding argument.

We have shown in Section 6.1 that $\sum_{m=1}^{X} \frac{1}{1-q^{-m}} \leq X + 2$ for any integer $X > 0$ and $q \geq 2$. Using similar manipulations, we can show that $\sum_{m=1}^{M-D-1} \frac{m}{1-q^{-m}} \leq (M - D - 1)(M - D)/2 + 4$. Finally, $\sum_{m=1}^{M-D-1} \frac{m^2}{1-q^{-m}} \geq \sum_{m=1}^{M-D-1} m^2$, which is a well known series.

Let us compute $E[\#\text{ Mult}] = \sum_d E[\#\text{ Mult}|d]P(D = d)$, using the fact that $P(D = d)$ is characterized by a binomial distribution. After some manipulations, it can be shown that $E[\#\text{ Mult}] = M^3Pe^3/3 + o(M^3)$. The proof concludes by noting that the operations related to the backward substitution step in Gaussian elimination grow as $O(M^2Pe^2)$ and are performed only once after enough dofs have been received, regarless of the field size q. □

6.3 An Example of Systematic Network Coding in TDD channels

We study the case of one node transmitting information to a single receiver as an example to illustrate the benefits of systematic network coding and the use of the model for the effect of field size into a previously discussed problem. Then, our main contribution is to include the effect of systematic network coding and the effect of the field size into the transition probabilities in the problem of a link discussed in Section

2. We present an overview of the scheme before starting our study of the transition probabilities.

Similarly as in Section 2, the sender can transmit packets back-to-back before stopping to wait for an ACK packet from each receiver. However, in this work we consider that in the first transmission the system will transmit back-to-back all M original packets first without coding followed by random linear coded packets. Every ACK packet returns the number of degrees of freedom (dof) that a particular receiver still requires to decode successfully the M original data packets. Note that the uncoded packets represent a dof with a particular structure.

The transmission process starts with M data packets being encoded into $N_s \geq M$ packets, M original data packets followed by $N_s - M$ random linear coded packets, and transmitted to the receiver. If all M packets are decoded successfully by the receiver, the process is completed. Otherwise, the receiver sends an ACK packet that informs the transmitter how many dofs are missing, say i. At this point, the transmitter sends N_i coded packets. The process is repeated until the M data packets are successfully decoded. As in previous work, we are interested in the optimal number N_i of coded packets to be transmitted back-to-back in order to minimize a specific metric, e.g., mean completion time.

The process can be modelled as a Markov Chain where we have an initial 'systematic' state (State S in Figure 6-3) which is the starting point of the system. This state is only visited once to represent the fact that uncoded packets are only sent during the first transmission. The remaining states represent the knowledge of the sender in terms of the number of dofs that the receiver needs in order to decode.

6.3.1 Transition Probabilities

Let us compute the transition probabilities of the Markov chain in Figure 2-5. The transition probabilities from state i to state i' of the Markov chain in Figure 2-5 are given by

$$P_{i \to i'} = P\left(X(n)=i' | X(n-1)=i\right) \tag{6.9}$$

where $X(n)$ is the number of dof required at the receiver at the end of transmission n. The previous state i determines N_i, the number of coded data packets sent by the transmitter given that i dofs are required at the receiver. Let us study in detail the transition probabilities $P_{i \to i'}$. First, we consider the transitions from the systematic state S to all other states. Note that there is no self-transition to S. The transition probability to go from state S to state i is given by the probability of receiving $M-i$ dofs given that N_s coded packets were sent and the first M packets are uncoded, i.e.,

$$P_{S \to i} = P\left(_{M-i|S}\right) = \sum_{j=0}^{M-i} P\left(_{M-i|j \text{ uncoded},S}\right) P\left(j \text{ uncoded}|S\right) \tag{6.10}$$

where $P\left(_{M-i|S}\right)$ indicates the probability that $M-i$ dofs are correctly received, given that the system is in state S,

$$P\left(j \text{ uncoded}|S\right) = \binom{M}{j} (1-Pe)^j Pe^{M-j} \tag{6.11}$$

indicates the probability of j uncoded data packets have been received given that the system is in state S, and

$$\begin{aligned} &P\left(_{M-i|j \text{ uncoded},S}\right) = P\left(_{M-i-j \text{ coded}|S}\right) \\ &= \sum_{l=M-i-j}^{N_s-M} P\left[_{M-i-j \text{ coded}|S,l \text{ received}}\right] P\left[_{l|S}\right] \end{aligned} \tag{6.12}$$

constitutes the probability that $M-i$ dofs are correctly received, given that the system is in state S and j uncoded packets have been received, and it is equivalent to the probability of $M-i-j$ coded packets being received given that the system is in state S, $P\left(_{M-i-j \text{ coded}|S}\right)$. Note that

$$P\left[_{l|S}\right] = \binom{N_s-M}{l} (1-Pe)^l Pe^{N_s-M-l} \tag{6.13}$$

constitutes the probability of l coded packets being received, but with some probability of being linearly dependent of the information at the receiver due to the effect of the field size. Then, $P\left[_{M-i-j \text{ coded}|S,l \text{ received}}\right]$ can be found by computing P_q^l, using

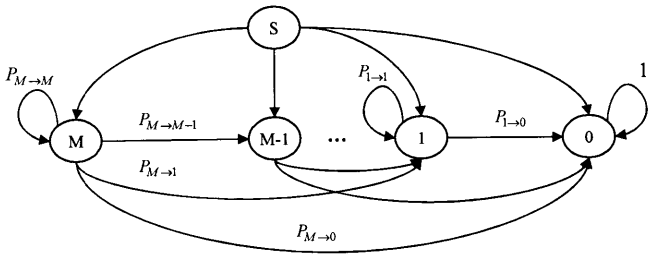

Figure 6-3: Makov chain for systematic network coding TDD scheme.

the transition probability matrix P_q computed in Section 6.1, and searching in the appropriate column and row corresponding to starting state S and end state $M-i-j$.

Let us now consider the transition probabilities starting from states other than the systematic state S. We assume that $N_i \geq i$, which means that there is some probability of transitioning from any i to $i' = 0$. For $i > i'$ we have that

$$P_{i\to i'} = (1 - Pe_{ack}) \sum_{k=\max\{1,i-i'\}}^{N_i} P\left[i'|i,k\right] P\left[k|i\right]$$

where k represents the number of coded packets that have been received, i.e., that have not been erased when going through the channel.

Note that $P\left[i'|i,k\right]$ represents the probability of starting at state i in the field size Markov Chain and transitioning to state i' in k transitions or hops. This can be found by computing P_q^k, and searching in the appropriate column and row corresponding to starting state i and end state i'. For the case of $i = i' > 0$,

$$P_{i\to i} = (1 - Pe_{ack}) \left[\sum_{k=0}^{N_i} P\left[i|i,k\right] P\left[k|i\right]\right] + Pe_{ack}$$

and that $P_{0\to 0} = 1$. Finally, note that

$$P\left[k|i\right] = \binom{N_i}{k} (1 - Pe)^k Pe^{N_i - k} \tag{6.14}$$

which completes the characterization of the problem.

6.3.2 Mean Completion Time

The mean time for completing the transmission of the M data packets to the receiver is again the constitutes the expected time of absorption, i.e., the time to reach state 0 for the first time, given that the initial state is S. If we define T_i as the mean completion time when the system is in state i, then we are interested in determining T_S.

We can define T^i as the time it takes to transmit N_i coded data packets and receive the ACK packets from the different receivers. It is easy to show that $T^i = N_i T_p + T_w$, where T_w combines the effect of the propagation delay and the duration of the ACK packet.

The mean completion time when the system is in state i is given by

$$T_i = T^i + \sum_{j \leq i} P_{i \rightarrow j} T_j$$

by exploiting the structure of the Markov Chain.

Our objective is to minimize the value of the expected transmission time T_S, that is,

$$\min_{N_s, N_M, .., N_1} T_M = \min_{N_s} (T^S + \sum_{i=1}^{M} P_{S \rightarrow i} \min_{N_i, .., N_1} T_i)$$

where $T^i = N_i T_p + T_w$. Similarly to the result in Chapter 2, regardless of the assumption on N_i, the problem of minimizing T_S in terms of the variables $N_s, N_M, .., N_1$ can be solved iteratively. First, we compute $\min_{N_1} T_1$, then use this results in the computation of $\min_{N_2, N_1} T_2$, and so on.

As an important remark, note that if we were interested in obtaining the mean completion time of our random linear network coding of Chapter 2 with the effect of field size, we could use the analysis as is. We would only need to optimize for the value T_M instead of T_S. Thus, this analysis has solved the problem for both systematic and purely random linear network coding taking into account the effect of field size.

Finally, note that if the values of $N_M, .., N_1$ were previously computed for the case

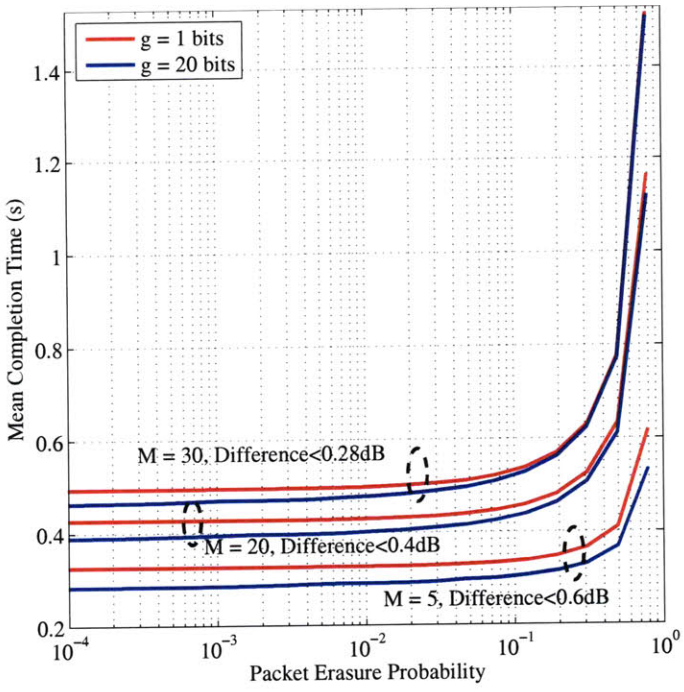

Figure 6-4: Mean completion time for the TDD scheme with different M and field sizes $q = 2$ and $q = 1048576$, for $g = 1$ and $g = 20$, respectively. We use the following parameters $R = 1.5$ Mbps, $h = 80$ bits, $n_{ack} = 100$ bits.

of purely random linear network coding with the effect of field size, we could use these values and only focus on computing N_s.

6.4 Numerical Results

This section provides numerical results that compare the performance of our network coding scheme in TDD channels, considering the effect of different field size. We consider a GEO satellite setting with a propagation time $T_{prop} = 125$ ms, and data packets of size $n = 10,000$ bits.

We compare performance of a random linear network coding scheme in terms of mean completion time under different packet erasure probabilities. We show that using $q = 2$ shows a small degradation in performance with respect to higher field sizes. Also, the gap in performance between $q = 2$ and higher q reduces as M increases, as expected. Finally, if the performance of $q = 2$ is not sufficiently good for small M, we can get very close to the performance of high field sizes with small

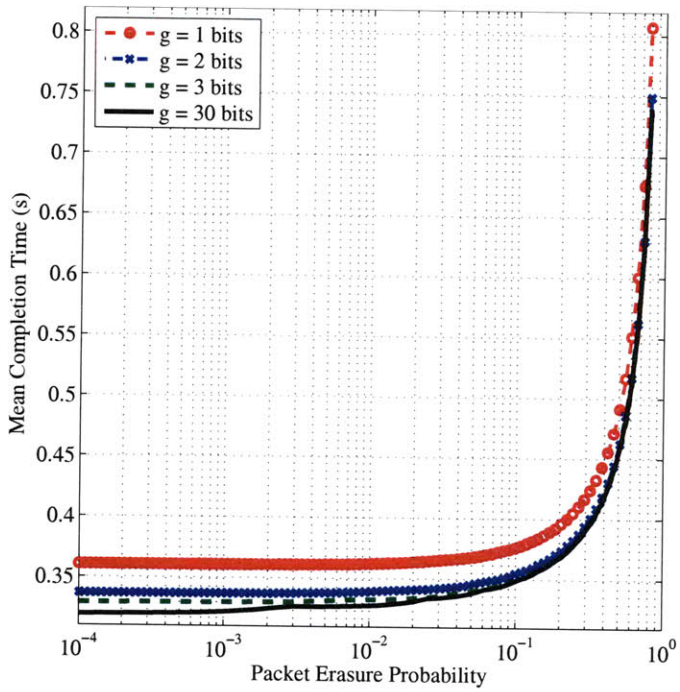

Figure 6-5: Mean completion time for the TDD scheme with a single receiver with different field sizes $q = 2^g$. We use the following parameters $R = 1.5$ Mbps, $h = 80$ bits, $n_{ack} = 100$ bits, $M = 10$.

increases of the field size, e.g., $q = 4$ or $q = 8$.

Also, we compare performance of our systematic network coding scheme to that of random linear network coding (RLNC) schemes in order to show that using the systematic network coding with field size $q = 2$, i.e., XORs for the coded packets, has very little or no degradation in performance with respect to RLNC with high field size. This means that using systematic network coding allows us to reduce the complexity of the operations (only performing XORs) and reducing the total number of operations, in average by a factor of Pe^3, while maintaining close to the same performance to the high field size RLNC proposed in previous work.

Figure 6-4 shows the mean completion time for the TDD scheme for a single receiver for $q = 2$ and $q = 2^{20}$ for various block sizes M and a wide range of packet erasure probabilities. Figure 6-4 illustrates that the gap between field sizes $q = 2$ and $q = 2^{20}$ is very small. For $M = 5$ the gap is smaller than 0.6 dB for the range of packet erasure probabilities considered, which ranges from 10^{-4} to 0.8. This means that the completion time is increased by at most 15 % on average for $M = 5$. For

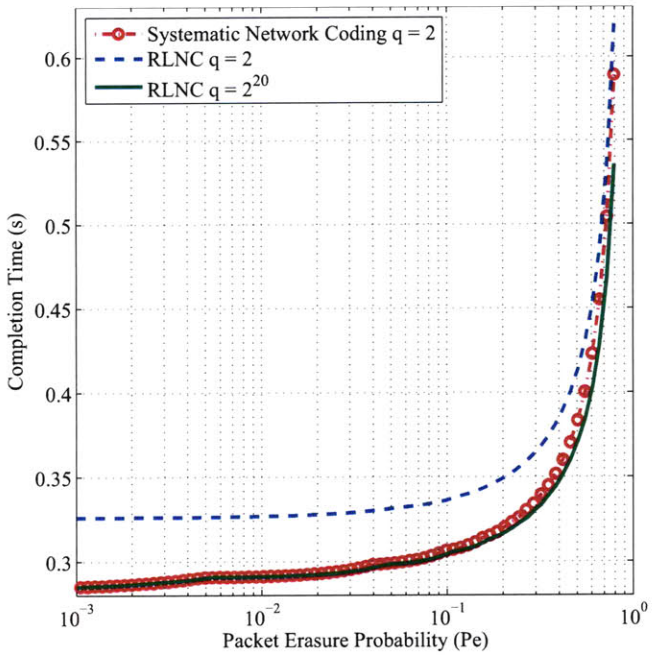

Figure 6-6: Mean completion time for the TDD scheme with $M = 5$. We use the following parameters $R = 1.5$ Mbps, $h = 80$ bits, $n_{ack} = 100$ bits, $M = 5$ packets

$M = 20$ and $M = 30$ we observe that the gap reduces to less than 0.4 dB and 0.28 dB, respectively. In other words, the completion time is increased on average by at most 10 % and 6.6 %, respectively. The importance of this result is two-fold. First, the degradation in performance due to the use of $q = 2$ is very small, even for small values of M where the effect of small field size is more noticeable. Also, the degradation in performance reduces as M, the number of data packets that are being randomly combined, increases. This effect was predicted by the result in Lemma 2. Since we expect to need between M and $M + 2$ coded packets on average in order to decode, then the effect of the additional coded packets is clearly reduced if M increases because proportionally more resources are being used to transmit the first M coded packets than the additional coded packets needed to finally decode.

Second, we can rely on considerably simpler coders and decoders. Note that for $q = 2$, random linear network coding is basically performing an XOR of those packets that were chosen from the pool of M original packets. Note that each packet has a probability of $1/2$ to be chosen to be XORed in each coded packet that is being

generated. Also, the overhead on the coded packet is reduced because the coefficient size $g = 1$ bit. For large enough M and n fixed, $q = 2$ could outperform cases where q is larger than 2 because larger q and larger M involve an increased overhead in the coded packet, i.e., cases in which we are using more resources sending information about coefficients than sending the actual information.

Figure 6-5 illustrates that if M is small, e.g., $M = 10$ in the figure, and the performance of $q = 2$ is insufficient, we can get considerable improvements with small field sizes. Figure 6-5 considers a single receiver and the cases of $q = 4$ and $q = 8$, which correspond to $g = 2$ and $g = 3$ bits, and compares it to the performance of $q = 2^{30}$, i.e., coefficients of $g = 30$ bits. Note that $q = 8$ is extremely close to the performance of $q = 2^{30}$, especially for $Pe > 0.01$ which is a range of common Pe values for wireless systems. We observe that for $Pe > 0.1$, the performance of $q = 4$ is essentially the same to that of our scheme using a field size of $q = 2^{30}$. Note that for a GEO satellite example the range of $Pe > 0.1$ are typical values. Thus, for wireless systems we could expect similar performance if we use small or large field sizes, even if M is not too large.

Figure 6-6 shows that the performance of systematic network coding is essentially the same as RLNC with a high field size for moderate values of Pe and the difference in performance at high Pe is very small. Figure 6-6 also shows the performance of RLNC with field size $q = 2$. The performance in terms of mean completion time for RLNC with $q = 2$ constitutes an upper bound to the mean completion time of the systematic network coding scheme with the same field size $q = 2$. Since an increase in M causes a reduction in the performance gap between RLNC with large field size and with $q = 2$, the difference in performance between our systematic approach and RLNC with high field sizes should also decrease as the number of original packets M increases.

Chapter 7

Conclusions

This work analyzes different scenarios of networks in delay challenged environments and the role of network coding as a means to provide viable and meaningful solutions to these scenarios. Chapter 2 provides a study of random linear network coding schemes for reliable communications in large latency time-division duplexing channels. We analyze three main cases of interest: a link, one-to-all broadcast, and all-to-all broadcast.

For the case of a link we provide a full characterization of the problem by providing a recursive expression for the moment generating function. This moment generating function is valid for both the completion time and energy using the appropriate substitutions. We present an analysis and numerical results that show that transmitting the optimal number of coded data packets sent before stopping to listen for an ACK in terms of mean completion time, provides a good trade-off between energy consumption and completion time. In fact, its performance in completion time is close to the optimal full duplex network coding scheme.

We also provide a queueing model for random linear network coding scheme for time division duplexing channels with Poisson arrivals. The analysis considers that the size of the batch that is sent using random linear coding can be in a range of values, say (m, K). At the time of completing service to a batch, if the queue size is below the minimum allowable value of the batch m, the system will wait until the queue size becomes m. If the queue size is in the range of batch sizes, all data packets

are serviced at that time. Finally, if the queue size is greater than the maximum allowable batch size K, the first K packets of the queue are serviced. We have used this analysis to study the mean queue size of the system and to choose the (m, K) pair that minimizes it under different λ. The analysis is useful if we are interested in choosing the optimal (m, K) given a different objective function.

Numerical results suggest that the mean queue size shows greater dependence on the value of minimum batch size m than on the value of the maximum batch size K for low values of λ. In general, K determines the maximum serviceable λ of the system, while m should be allowed to have small values. In our examples, $m = 1$ provided the best performance in terms of minimizing the mean queue size. Also, numerical results suggests that having a fixed batch size, i.e., $m = K$, is not the optimal configuration in general.

Future research should consider extensions of the principles proposed for one link to the general problem of wireless networks. These extensions are possible within the framework of random linear network coding. Also, future research should consider a queueing model for online network coding *à la* Sundararajan et al [8] in the case of large latency time-division duplexing channels, e.g., studying the delay for a packet to be "seen" (in the framework of [8]) and the time between decoding packets after being "seen".

For the case of one-to-all broadcast, we provide an extension of the principles studied in a link. Due to the exponentially increasing number of states in our Markov chain model, which increases the computation time of the optimal number of coded packets to be transmitted before stopping, we provide simple heuristics to compute this number of coded packets to be sent before stopping. This heuristic computation achieves close to optimal performance with the advantage of a considerable reduction in the search time. Numerical results show that our coding scheme outperforms a Round Robin broadcast scheme in an TDD channel. More importantly, for high packet erasures, our coding scheme for TDD outperforms a RR scheme operating in a full duplex channel.

From a practical implementation perspective, we provided a general model of

the effect of field size for random linear network coding networks that transmit a finite number of original data packets and for channels/networks that have packet losses. We provided bounds on $E[N_c]$, the mean number of coded packets that a receiver needs to receive successfully in order to decode the information. These bounds are valid for random linear network coding in general. A trivial lower bound is that $E[N_c] \geq M$. We prove an insightful upper bound, which states that $E[N_c] \leq \min\left\{M\frac{q}{q-1}, M+1+\frac{1-q^{-M+1}}{q-1}\right\}$. This bound implies that $E[N_c]$ can become arbitrarily close to M as the field size q increases, but more importantly, we showed that $E[N_c] \leq M+2$ for any $q \geq 2$. This means that as M increases, the effect of the additional coded packets that have to be sent due to a "bad" random selection of the coefficients, will be negligible. A "bad" random selection is a choice of coefficients that does not provide innovative information.

We present numerical results that illustrate that the gap between using $q = 2$ and larger values of q is small, specially when M is large. Note that we can rely on considerably simpler coders and decoders. For $q = 2$, random linear network coding is basically performing an XOR of those packets that were chosen from the pool of M original packets, each packet is chosen to be combined with probability $1/2$. Finally, if the performance of $q = 2$ is not sufficiently good for small M, we can get very close to the performance of high field sizes with small increases of the field size, e.g., $q = 4$ or $q = 8$.

We have shown that there is a considerable reduction in computational complexity by using systematic network coding. The average number of operations required to decode using systematic network coding in an IID Bernoulli packet erasure channel with parameter Pe is reduced by a factor of Pe^3 with respect to using a random linear network coding approach with the same field size. These results apply not only to the case of a point-to-point link or broadcast system, but also to any network that uses systematic network coding. The key problem there will be to identify the equivalent Pe of such a system. As an example, a daisy chain that preserves the systematic structure of the code and that shows an erasure probability of Pe_i in link i, will have an equivalent $Pe = 1 - \prod_i (1 - Pe_i)$.

We present numerical results that illustrate that systematic network coding using XORs can provide almost the same performance as the random linear network coding procedure that uses large field sizes. This fact implies that the complexity of the operations at both encoder and decoder can be considerably reduced with small or no degradation in performance.

Future research should consider extending the use of systematic network coding to more general networks, where one of the key design challenges is to determine what nodes and through which links there should be pure coding and which links can maintain a systematic structure.

Chapter 3 presents an analysis and numerical results that show that choosing a schedule based on its impact on the network a each time slot provides considerably better performance than schemes that choose schedules giving priority to nodes that know the most information. In fact, even for small networks and moderate number of packets to transmit we can expect large gains in terms of completion time.

Future research will consider an extension of the principles proposed in this work to the case of a random arrivals of new packets to the system, e.g., from a base station, that have to be disseminated to all nodes in the network. One metric of interest is the throughput performance of our system. More importantly, future work will focus on using the principles proposed in this paper to provide distributed MAC protocols that improve performance of practical systems. In particular, we will consider distributed protocols that provide a higher priority to a node based on the number of neighbors that require some of the information of that node. One of the main challenges will be to develop a protocol that can allow nodes to quickly and efficiently gather information of the degrees of freedom that their neighbors have or require, without incurring into much overhead.

Chapter 4 presented complete and approximate models of the underwater acoustic channel under a Gaussian noise assumption and used them to propose a network coding based lower bound on transmission power for multicast connections in underwater networks. This lower bound was used to determine the gap of different medium access protocols and network schemes for some multicast rate in underwater networks. This

comparison is carried out for several routing and network coding schemes.

Network coding with implicit acknowledgements has better performance than routing with link-by-link ACK in terms of transmission power, especially when the probability of collision is increased. The gap between routing with link-by-link ACK and network coding with implicit ACK in terms of transmission power was shown to increase as the transmission probability increased, which is closely related to the probability of packet collision. However, study of a network coding scheme with an explicit ACK should be considered to improve performance. Also, network coding with implicit acknowledgement compared to common rateless network coding schemes allows to save resources, e.g., memory required in the nodes, and rate adaptation following a similar analysis as in [62]. Also, there has been previous work on network coding based Ad-Hoc protocols, such as CODECAST [61], which could be extended to use implicit acknowledgements, and adapted to the underwater acoustic channel.

Chapter 5, shows that the amount of information that can be exchanged by each source-destination pair in an underwater acoustic network goes to zero as the number of nodes n goes to infinity, at least at a rate $n^{-1/\alpha}e^{-W_0(O(n^{-1/\alpha}))}$. This rule is valid for the different scenarios in general, requiring only changes in the scaling constants. The throughput per source-destination pair has two different regions. For small n, the throughput decreases very slowly as n increases. For large n, it decreases as $n^{-1/\alpha}$. Given that $1 \leq \alpha \leq 2$ in an underwater acoustic channel, the available throughput for large n decays more rapidly than in typical radio wireless networks. However, typical node densities in underwater networks correspond to the small n regime. In a narrowband example with values of $a(f)$ characteristic of an underwater channel, we showed that the upper bound on the throughput remains almost constant for densities up to 100 nodes per km^2. Most underwater networks have node densities in this range owing to the applications for which they are deployed.

Finally, we have identified some important characteristics of the underwater acoustic channel useful for future studies. For example, we could allow cooperation between nodes *à la* Ozgur et al [68] taking advantage of the distance-band separation property of the underwater channel. Namely, instead of performing time division between long

and short transmissions, we could simply transmit in different bands that do not interfere with one another. This is important because acoustic transmissions have long propagation delays due to the speed of sound underwater ($\sim$1500 m/s), which reduces the usefulness of a time-division scheme. Another important extension is to consider scaling the frequency of transmission with the number of nodes, i.e., increasing the carrier frequency as the number of nodes increases, in order to take advantage of the increasing absorption coefficient $a(f)$.

On a more general note, the problem of coding in delay challenged networks warrants further study. In this thesis, we have identified several important facets of it, e.g., high latency, limited feedback, half-duplex constraints on the nodes, application requirements, in a variety of environments and conditions. However, many other interesting environments and applications exist where coding for delay can be relevant. One interesting example is to consider that the packets have deadlines after which they are no longer useful to the intended receivers, as in real-time voice transmission applications. In this case, the coding must focus on reducing delay of the packets, as a means to ensure that packets are not lost unnecessarily, but the transmitter and relays must also prioritize the coding of packets based on the different deadlines so that the coding does not impact negatively in the system. A transmitter may choose to discard or delay the transmission of a packet with an urgent deadline, if this can avoid other packets to exceed their own deadlines.

Another general comment is that the importance of half-duplex constraints on the nodes has not been thoroughly studied. In fact, it is not considered in many theoretical results and applications. A half-duplex constraint is quite natural from a system design perspective and, although the conclusion for some of our sections is that a half-duplex constrained system could reach close to or the same performance as a system operating in full-duplex, we observe that it required a channel-adaptive coding scheme to achieve this. Any given scheme is not guaranteed to have this trait. Studying the impact of half-duplex constraints on existing theoretical results and as a part of practical schemes is an interesting and relevant aspect for future research.

Bibliography

[1] R. Ahlswede, N. Cai, S. Y. R. Li, R. W. Yeung, "Network Information Flow," IEEE Trans. Inf. Theory, vol. 46, no. 4, pp. 1204-1216, Jul. 2000.

[2] S. Y. R. Li, R. W. Yeung, N. Cai, "Linear Network Coding," IEEE Trans. Inf. Theory, vol. 49, pp. 371, Feb. 2003.

[3] R. Koetter, M. Médard, "An algebraic Approach to network coding," IEEE/ACM Trans. Netw., vol. 11, no. 5, pp. 782-795, Oct. 2003.

[4] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, B. Leong,"A Random Linear Network Coding Approach to Multicast," Trans. Info. Theory, vol. 52, no. 10, pp.4413-4430, Oct. 2006.

[5] D. S. Lun, M. Médard, R. Koetter, M. Effros, "On coding for reliable communication over packet networks," Physical Comm., Vol. 1, Issue 1, pp. 3-20, Mar. 2008.

[6] P. Maymounkov, N. J. A. Harvey, D. S. Lun, "Methods for Efficient Network Coding," In Proc. 44-th Allerton Conference'06.

[7] D. S. Lun, P. Pakzad, C. Fragouli, M. Médard, R. Koetter, "An Analysis of Finite-Memory Random Linear Coding on Packet Streams," In Proc. WiOpt'06, Boston, MA, USA, Apr. 2006.

[8] J. K. Sundararajan, D. Shah, M. Médard, "On Queueing for Coded Networks - Queue Size Follows Degrees of Freedom," In Proc. IEEE ITW 2007, Bergen, Norway, Jul. 2007.

[9] S. Chachulski, M. Jennings, S. Katti, D. Katabi, "Trading Structure for Randomness in Wireless Opportunistic Routing," In Proc. COMM'07, pp. 169-180, Kyoto, Japan, Aug. 2007.

[10] A. Eryilmaz, A. Ozdaglar, M. Médard, "On Delay Performance Gains from Network Coding," In Proc. CISS'06, pp. 864-870, Princeton, NJ, USA, Mar. 2006.

[11] E. Ahmed, A. Eryilmaz, M. Médard, A. Ozdaglar, "On the Scaling Law of Network Coding Gains in Wireless Networks," In Proc. MILCOM'07, Orlando, FL, USA, Oct. 2007.

[12] M. Ghaderi, D. Towsley, J. Kurose, "Network Coding Performance for Reliable Multicast," In Proc. MILCOM'07, Orlando, FL, USA, Oct. 2007.

[13] J. K. Sundararajan, D. Shah, M. Médard, "ARQ for Network Coding," In Proc. ISIT'08, pp. 1651-1655, Toronto, Canada, Jul. 2008.

[14] A. F. Dana, R. Gowaikar, R. Palanki, B. Hassibi, M. Effros, "Capacity of wireless erasure networks," IEEE Trans. on Info. Theory, vol. 52, no. 3, pp. 789-804, Mar. 2006.

[15] B. Shrader, A. Ephremides, "A queueing model for random linear coding," In Proc. MILCOM'07, Orlando, USA, Oct. 2007.

[16] B. Shrader, A. Ephremides, "On the queueing delay of a multicast erasure channel," In Proc. IEEE Info. Theory Workshop (ITW), Oct. 2006.

[17] T. Ozugur, M. Naghshineh, P. Kermani, J. A. Copeland,"On the performance of ARQ protocols in infrared networks," Int. Jour. Commun. Syst., vol. 13, pp. 617-638, 2000.

[18] A. M. Shah, S. S. Ara, M. Matsumoto,"An Improved Selective-Repeat ARQ Scheme for IrDA Links at High Bit Error Rate," In Proc. 4th int. conf. on Mobile and ubiquitous multimedia, Christchurch, New Zealand, pp. 37-42, 2005.

[19] M. Stojanovic, "Optimization of a Data Link Protocol for an Underwater Acoustic Channel," In Proc. Oceans 2005 - Europe, pp. 68- 73, Jun. 2005

[20] A. R. K. Sastry, "Improving Automatic Repeat-Request (ARQ) Performance on Satellite Channels Under High Error Rate Conditions," IEEE Trans. on Comms., vol. 23, no. 4, pp. 436-439, Apr. 1975.

[21] P. Yu, S. Lin, "An Efficient Selective-Repeat ARQ Scheme for Satellite Channels and Its Throughput Analysis," IEEE Trans. on Comms., vol. 29, no. 3, pp. 353-363, Mar. 1981.

[22] I. F. Akyildiz, O. B. Akan, J. Fang, "TCP-Planet: A Reliable Transport Protocol for InterPlaNetary Internet," JSAC, vol. 22, no 2, pp. 348361, 2004.

[23] S. Lin, D. J. Costello, M. J. Miller,"Automatic-Repeat-Request Error Control Schemes," IEEE Commun. Mag., vol. 22, n. 12, pp. 5-17, Dec. 1984.

[24] D. S. Lun, N. Ratnakar, M. Médard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, F. Zhao, "Minimum-Cost Multicast Over Coded Packet Networks," IEEE Trans. on Info. Theory, vol. 52, no. 6, pp.2608-2623, Jun. 2006.

[25] F. Chapeau-Blondeau, A. Monir, "Numerical Evaluation of the Lambert W Function and Application to Generation of Generalized Gaussian Noise With Exponent 1/2," IEEE Trans. on Signal Proc., Vol. 50, No. 9, Sept. 2002.

[26] S. Boyd, A. Ghosh, B. Prabhakar, D. Shah," Gossip Algorithms: Design, Analysis, and Applications," In Proc. IEEE Infocom'05, Miami, Mar. 2005.

[27] S. Deb, M. Médard, C. Choute,"Algebraic Gossip: A Network Coding Approach to Optimal Multiple Rumor Mongering," Trans. Info. Theory, vol. 52, no. 6, pp.2486-2507, Jun. 2006.

[28] C. M. Cheng, H. T. Kung, C. K. Lin, C. Y. Su, D. Vlah,"Rainbow: A Wireless Medium Access Control Using Network Coding for Multi-hop Content Distribution," In Proc. MILCOM'08, San Diego, CA, Nov. 2008.

[29] M. Stojanovic,"Capacity of a Relay Acoustic Link," in Proc. IEEE Oceans'07 Conference, Vancouver, Canada, Oct. 2007.

[30] W. Zhang, M. Stojanovic, U. Mitra, "Analysis of A Simple Multihop Underwater Acoustic Network," in Proc. Third ACM International Workshop on Underwater Networks (WUWNeT'08) / MobiCom 2006, San Francisco, CA, Sept. 2008.

[31] S. Biswas, R. Morris,"ExOR: Opportunistic MultiHop Routing for Wireless Networks," In the Proc. SIGCOMM 2005, Philadelphia, PA, Aug. 2005.

[32] J. Kulik, W. Heinzelman, H. Balakrishnan,"Negotiation-Based Protocols for Disseminating Information in Wireless Sensor Networks," Wireless Networks, 8, pp. 169185, 2002.

[33] M. T. Chao, W. E. Strawderman, "Negative Moments of Positive Random Variables," Jour. of the American Statistical Asso., vol. 67, no. 338, pp. 429-431, Jun. 1972.

[34] N. Cressie, A. S. Davis, J. L. Folks, G. E. Policello II, "The Moment-Generating Function and Negative Integer Moments," The American Statistician, vol. 35, no. 3, pp. 148-150, Aug. 1981.

[35] S. K. Bar-Lev, M. Parlar, D. Perry, W. Stadje, F. A. Van der Duyn Schouten, "Applications of Bulk Queues to Group Testing Models with Incomplete Identification," European Journal of Operational Research, no. 183, pp. 226237, 2007.

[36] M. L. Chaudhry, J. G. C. Templeton,"A First Course in Bulk Queues", Wiley, 1983.

[37] L. E. N Delbrouck, "A Feedback Queueing System with Batch Arrivals, Bulk Service, and Queue-Dependent Service Time," Journal of the Association for Computing Machinery, Vol. 17, No. 2, pp. 314-323, Apr. 1970.

[38] M. Médard, R. Srikant, "Capacity of Nearly Decomposable Markovian Fading Channels Under Asymmetric Receiver-Sender Side Information," IEEE Trans. on Info. Theory, vol. 52, no. 7, pp. 3052-3062, Jul. 2006.

[39] A. Eryilmaz, A. Ozdaglar, M. Médard, "On Delay Performance Gains from Network Coding," In Proc. CISS'06, pp. 864-870, Princeton, NJ, USA, Mar. 2006.

[40] M. Luby, "LT-codes," In Proc. 43rd Annu. IEEE FOCS, Vancouver, BC, Canada, pp. 271280, Nov. 2002.

[41] A. Shokrollahi, "Raptor Codes," IEEE Trans. Info. Theory, vol. 52, no. 6, pp. 25512567, 2006.

[42] J. Heide, M. V. Pedersen, F. H. P. Fitzek, T. Larsen, "Network Coding for Mobile Devices - Systematic Binary Random Rateless Codes," In Proc. Workshop on Cooperative Mobile Networks 2009, Dresden, Germany, Jun. 2009.

[43] A. A. Yazdi, S. Sorour, S. Valaee, R. Y. Kim, "Optimum Network Coding for Delay Sensitive Applications in WiMAX Unicast," In Proc. INFOCOM'09, Rio de Janeiro, Brazil, Apr. 2009.

[44] J. Jin, B. Li, T. Kong, Is Random Network Coding Helpful in WiMAX?, In Proc. IEEE INFOCOM'08, Apr. 2008.

[45] J. Barros, R, A. Costa, D. Munaretto, J. Widmer, "Effective Delay Control in Online Network Coding," In Proc. INFOCOM'09, Rio de Janeiro, Brazil, Apr. 2009.

[46] J. Partan, J. Kurose, B. N. Levine, "A Survey of Practical Issues in Underwater Networks," In Proc. WUWnet '06, pp. 17-24, Los Angeles, Sept. 2006.

[47] M. Stojanovic, "On the Relationship Between Capacity and Distance in an Underwater Acoustic Communication Channel," ACM SIGMOBILE Mobile Computing and Communications Review (MC2R), pp.34-43, vol.11, Issue 4, Oct. 2007.

[48] D. Pompili, T. Melodia, I. F. Akyildiz, "Routing Algorithms for Delay-Insensitive and Delay-Sensitive Applications in Underwater Sensor Networks," In Proc. MobiCom 2008, pp. 298-309, Los Angeles, CA, USA, Sept. 2006.

[49] E. A. Carlson, P.-P. Beaujean, E. An, "Location-Aware Routing Protocol for Underwater Acoustic Networks," In Proc. OCEANS 2006, pp. 1-6, Boston, MA, USA, Sept. 2006.

[50] H. Yan, Z. J. Shi, J. H. Cui, "DBR: Depth-Based Routing for Underwater Sensor Networks," In Proc. IFIP Networking'08, pp. 1-13, Singapore, May. 2008.

[51] E. M. Sozer, M. Stojanovic, J. G. Proakis, "Underwater Acoustic Networks," IEEE Jou. of Ocea. Eng., vol. 25, no. 1, pp. 72-83, 2000.

[52] I. F. Akyildiz, D. Pompili, T. Melodia, "Underwater Acoustic Sensor Networks: Research Challenges," Elsevier Ad Hoc Net., vol. 3, no. 3, pp.257279, 2005

[53] I. F. Akyildiz, D. Pompili, T. Melodia, "State-of-the-art in Protocol Research for Underwater Acoustic Sensor Networks," In Proc. of WUWNet '06, pp. 7-16, Los Angeles, Sept. 2006.

[54] S. Chachulski, M. Jennings, S. Katti, D. Katabi, "Trading Structure for Randomness in Wireless Opportunistic Routing," In Proc. COMM'07, pp. 169-180, Kyoto, Japan, Aug. 2007.

[55] Z. Guo, P. Xie, J. H. Cui, B. Wang. "On Applying Network Coding to Underwater Sensor Networks," In Proc. of WUWNet '06, pp. 109-112, Los Angeles, Sept. 2006.

[56] L. Berkhovskikh, Y. Lysanov, "Fundamentals of Ocean Acoustics," New York, Springer, 1982.

[57] R. Coates, "Underwater Acoustic Systems," New York, Wiley, 1989.

[58] M. Stojanovic, Capacity of a Relay Acoustic Link," in Proc. IEEE Oceans'07 Conference, Vancouver, Canada, Oct. 2007.

[59] M. Chiang, S. H. Low, A. R. Calderbank, J. C. Doyle, "Layering as Optimization Decomposition: Questions and Answers," In Proc. MILCOM 2006, pp. 23-25, Washington, DC, USA, Oct. 2006.

[60] M. Chiang, S. H. Low, A. R. Calderbank, J. C. Doyle, "Layering as Optimization Decomposition: A Mathematical Theory of Network Architectures," Proc. of the IEEE, vol. 95, no. 1, Jan. 2007.

[61] J. S. Park, M. Gerla, D. S. Lun, Y. Yi, M. Médard, "CODECAST: a Network-Coding-Based AD HOC Multicast Protocol," IEEE Wireless Comms. Magazine, pp. 76-81, Oct. 2006.

[62] C. Fragouli, D. S. Lun, M. Médard, P. Pakzad, "On Feedback for Network Coding," In Proc. CISS 2007, Baltimore, USA, Mar. 2007.

[63] S. C. Ergen, P. Varaiya, "TDMA Scheduling Algorithms for Sensor Networks," Technical report, Department of Electrical Engineering and Computer Sciences, UC Berkeley, Jul. 2, 2005.

[64] T. M. Cover, J. A. Thomas, "Elements of Information Theory," 2nd Edition, New Jersey, Wiley, 2006.

[65] R. G. Gallager, "Information Theory and Reliable Communications," New York, Wiley, 1968.

[66] P. Gupta, P.R. Kumar, "The Capacity of Wireless Networks," IEEE Trans. Inf. Theory, vol 46, no 2, pp. 388-404, Mar. 2000.

[67] M. Franceschetti, O. Dousse, D. N. C. Tse, P. Thiran, "Closing the Gap in the Capacity of Wireless Networks Via Percolation Theory," IEEE Trans. Inf. Theory, vol. 53, no. 3, pp. 1009-1018, Mar. 2007.

[68] A. Ozgur, O. Leveque, D. N. C. Tse, "Hierarchical Cooperation Achieves Optimal Capacity Scaling in Ad Hoc Networks," IEEE Trans. Inf. Theory, vol. 53, no. 10, pp. 3549-3672, Oct. 2007.

[69] M. Grossglauser, D. N. C. Tse, "Mobility Increases the Capacity of Ad Hoc Wireless Networks," IEEE Trans. Inf. Theory, vol. 10, no. 4, pp. 477-486, Aug. 2002.

[70] M. Vu, N. Devroye, V. Tarokh, "An Overview of Scaling Laws in Ad Hoc and Cognitive Radio Networks," Wireless Pers. Comm., vol 45, no 3, May 2008.

[71] D. E. Lucani, M. Médard, M. Stojanovic, "Underwater Acoustic Networks: Channel Models and Network Coding based Lower Bound on Transmission Power for Multicast," IEEE Journal on Selected Areas in Communications (JSAC), Special Issue on Underwater Wireless Communications and Networks, pp.1708 - 1719, Dec. 2008.

[72] D. E. Lucani, M. Médard, M. Stojanovic, "Network Coding Schemes for Underwater Networks: The Benefits of Implicit Acknowledgement," In Proc. WUWnet '07, pp.25-32, Montreal, Quebec, Canada, Sept. 2007.