Order Number 9215006

Error-control schemes for broadcast channels

Chandran, S. Ram, Ph.D.

University of Hawaii, 1991

U·M·I

300 N. Zeeb Rd.
Ann Arbor, MI 48106

ERROR-CONTROL SCHEMES FOR BROADCAST CHANNELS

A DISSERTATION SUBMITTED TO THE GRADUATE DIVISION OF THE UNIVERSITY OF HAWAII IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

ELECTRICAL ENGINEERING

DECEMBER 1991

By

S. Ram Chandran

Dissertation Committee:

Shu Lin, Chairman

Norman Abramson

N. Thomas Gaarder

E.J. Weldon

Hugh M. Hildon

Acknowledgements

I would like to thank Prof. Shu Lin for his help and encouragement during the course of this work. His boundless energy and drive to conduct research has been a unique motivating factor for me. Prof. Norman Abramson has been a constant source of encouragement and guidance during my stay in Hawaii. His intuitive explanations of very complex problems made them seem easy to me in spite of my limitations. I learnt a small fraction of Prof. Gaarder's deep knowledge of probability theory, random processes and communication theory in his class. I also learnt the importance of rigor and consistent terminology from him. During a course on Coding Theory, Prof. Weldon shared his vast knowledge with us, especially the practical issues that cannot be found in books or papers. I first met Prof. Mike Hilden as a fellow student in some of the graduate courses in the Electrical Engineering department and later as a member in my committee. I thank him for his kindness. In fact, all the aforementioned people shared their knowledge with kindness and generosity for which I am grateful.

Graduate studies away from home would not be bearable if one did not have good friends with whom to share good and bad times. I was fortunate to have the support of so many good friends. My special thanks go to Lynette Wageman, Jose Garcia-Luna, Vijay Kumar, Ed Webman, Willa Valdez, Fernando Santiago, Ann Harada, Ann Palmer, Amit Mathur, Bill Rivard and Surya Tewari.

It is not possible to fully express my gratitude to all members of my family, especially Amma and Appa for their love, strength, encouragement and unconditional support of all my endeavors. I dedicate this dissertation to them.

Abstract

Today's information age has created a virtual explosion in the need for connectivity, resource sharing and transfer of information on a global scale. This has resulted in the mushrooming of data networks. The satisfactory functioning of these networks is becoming increasingly dependent on the *fast* and *reliable* transfer of information over broadcast (one-to-many) and multiple-access (many-to-one) channels. Historically, almost all the efforts towards error-control in data transmission have been for the point-to-point channel. In this dissertation, we have investigated some error-control schemes for the reliable transfer of data over broadcast channels. The goal of this study is to meet a given reliability criterion for the transmission of data while making efficient use of the channel.

Excepting some degenerate cases, most of the interesting broadcast scenarios also include a multiple-access channel in tandem. In other words, many of these broadcast channels have a built-in feedback channel. We have proposed several error-control schemes for the broadcast channels that make use of this feedback channel. The broadcast environments have been classified into two categories: *Homogeneous* and *Non-homogeneous.* Homogeneous broadcast channel (HB) is the case where a single source sends messages to *all* the destinations in the channel. The transmitter tries to deliver the same messages error-free to all the receivers. In the Non-Homogeneous broadcast (NHB) channel, successive messages from the transmitter may be addressed to different subsets of receivers in the broadcast environment and these subsets may not be disjoint.

Several retransmission-error-control schemes have been proposed for these two channels. We introduce criteria for characterizing channel utilization and algorithms

for achieving superior channel throughput. The coding problem for the HB channel is synonymous to that of the point-to-point channel. We have proposed a selective repeat ARQ protocol for the HB channel that uses a dynamic programming optimization technique to choose the optimum number of copies of a message to be transmitted by considering the number of receivers that are yet to acknowledge the message. We have also proposed two hybrid ARQ schemes that use powerful codes combined with parity retransmission. These two schemes are suitable for noisier channels. All these schemes significantly improve upon all the existing schemes in literature both in terms of achievable throughput and reliability.

For the NHB channel, we have examined several retransmission schemes that exploit the broadcast nature of the channel. The proposed schemes are non-time-shared in general. Time-shared schemes can however be realized as special cases of these schemes. We have considered two cases of the NHB channel: Finite Broadcast Population and Infinite Broadcast Population. The finite population case allows us to exploit the broadcast nature of the channel more effectively. Various trade- offs that arise in the choice of these schemes have been investigated. The non-time-shared schemes significantly outperform the time-shared schemes almost always. To the best of our knowledge, this work constitutes the first systematic study of NHB channels. The NHB channels closely model the computer communications networks so prevalent today. The schemes proposed here achieve much higher throughput than the ones being used today. Moreover, these schemes can be used with UEP codes to achieve different levels of protection for the messages intended for different destination groups.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The recent mushrooming of computer communications networks has created a virtual explosion in the need for connectivity, resource sharing and fast data transfer among various hosts. A typical architecture for a computer network is shown in Fig. 1.1, where a file-server is connected to several hosts. Each of these hosts themselves could be connected to several other hosts and terminals. This recursive chain continues indefinitely. These hosts and terminals could be either local or remote. The connection between the hosts could be through the medium of ethernet (as shown in Fig. 1.1), twisted pair cable, packet radio, leased phone lines or a combination of these and other physical media. The satisfactory functioning of these networks is becoming increasingly dependent on the *fast* and *reliable* transfer of information over broadcast (one-to-many) or multiple-access (many-to-one) channels (e.g., file-server to hosts, host to server, host to terminals, host to other hosts etc.). Excepting some very degenerate cases, almost all the interesting communication scenarios involving the broadcast channel have the multiple-access channel as it's front end (or vice-versa). In fact, it would not be an exaggeration if we say that this **broadcast/multiple-access** tandem *is* and *has always been* the most natural mode of communication. Radio communication was perhaps the first full-scale commercial exploitation of broadcast communication. Not so obvious to the average consumer is the extensive use of satellites, computer networks and packet radio networks today, each of which features the

broadcast/multiple-access tandem. Clearly, the example of computer networks (Fig. 1.1), is but one example of several such scenarios that arise in the modern context.

In view of the discussion above, we feel that any references to point-to-point communications should only be made as a special case of the broadcast or multiple-access communications and not the other way around. Historically, the study of point-to-point communication has dominated the communications area. This may be in part due to the simplicity and ease of analysis of the point-to-point channel compared to other channels. It is only recently that we have begun to understand the broadcast channels. Thomas Cover's work [Cover 72] was the first step in that direction. On the other hand, the progress in the study of multiple-access channels has been a little bit better. It may be noted that the capacity regions of some of the simplest broadcast and multiple-access channels are still unknown.

In this dissertation we have attempted to apply the known results in this area to the problem of error-control for broadcast channels. More specifically, we are interested in devising error-control protocols for these channels. We will be concentrating almost entirely on the error-control aspect of the problem. However, from time to time, we will make some comments on the protocol aspect of the problem. Henceforth, all the examples in our discussions will be from the standpoint of computer communications although all the results developed here will also have applications in the other areas of broadcast communications mentioned above.

The corresponding protocols for the point-to-point channel are often referred to as "data link layer protocols" as per the International Standards Organization (ISO) Reference Model of Open Systems Interconnection (OSI) [Stal 87] (Note: This is the only time we will use this rather long name). Not much effort has been made so far to develop broadcast versions of these protocols. For example, the Transmission Control Protocol/Internet Protocol (TCP/IP) makes no provisions for transfer of information

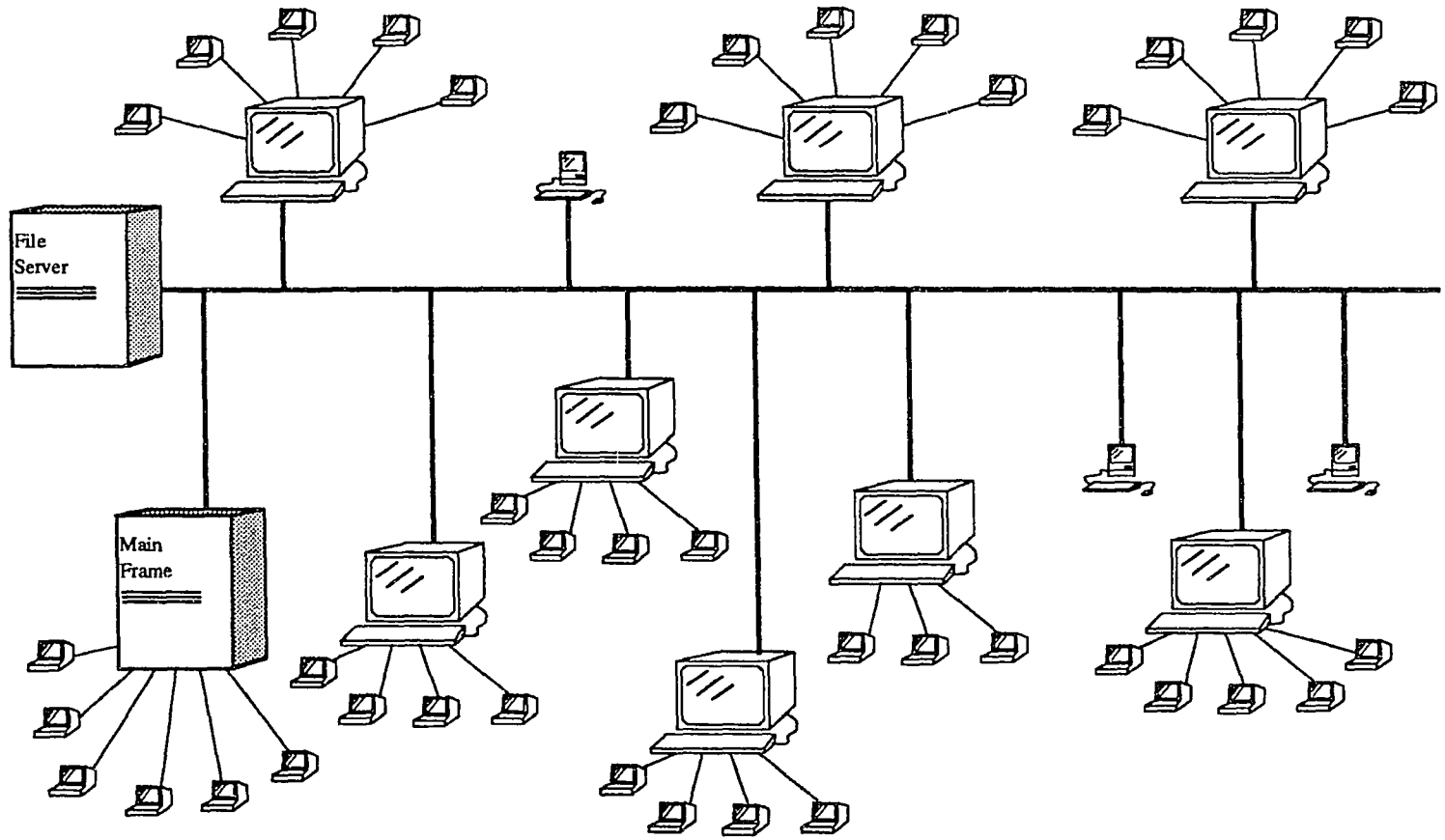

Fig. 1.1 Computer Communications Network

over a broadcast channel. All the broadcast situations are handled by time-sharing the channel or in other words they are handled as a series of point-to-point channels, one at a time. The popular software File Transfer Protocol (FTP) is written based on the TCP/IP protocol. It uses the selective repeat mode of retransmission. However, it has no counterpart for sending files to more than one destination (broadcasting). The UNIX mail system is yet another example where a user often wants to send the same mail to several other users. The current system accommodates such requests by sending the information, packet by packet, to users one at a time. Each packet has to be positively acknowledged separately by each host although a packet transmitted to a particular host (users may be on different hosts) could have been potentially received by other hosts. Clearly this process is extremely wasteful of resources. In this dissertation we have obtained more efficient ways of achieving broadcasting needs while still delivering the information error-free. Given the fact that the need to broadcast information is endemic and still growing at a rapid pace, we believe that the installed capacity of the computer networks will soon be rendered inadequate unless the broadcast nature of information transfer is exploited. In the next section, we give a brief introduction to the error-control problem which is followed by an overview of this dissertation.

1.1 Background

Basically, there are two ways of controlling transmission errors in data communication systems: forward-error correction (FEC) and automatic repeat request (ARQ). In an FEC scheme, an error correcting code is used to correct transmission errors. Whenever the receiver detects the presence of errors in a received codeword (i.e., non-zero syndrome), it tries to determine the error locations and correct them. The

output of the decoder is then delivered to the user. The throughput efficiency of an FEC system is maintained at a constant level regardless of the channel conditions.

In ARQ schemes however, some feedback is involved between the transmitter and receiver through a return channel. This feedback channel is used to improve the reliability of communications. The message is first coded based on a high rate error detecting code and then transmitted over the channel. At the receiver, parity checking is performed on the received message. If the syndrome is zero, the received message is assumed to be error-free and is delivered to the user. At the same time, the receiver sends a positive acknowledgement (ACK) to the transmitter via the return channel. If the the syndrome is non-zero, then the received message is discarded and a negative acknowledgement (NAK) is sent to the transmitter; again through the return channel, requesting a retransmission. Retransmissions continue until the message is successfully received. With this system, erroneous data is delivered to the user only if the receiver fails to detect the presence of errors. Using a proper error-detecting code, the probability of undetected error can be made very small. The ARQ schemes are used widely in data communication systems because they are simple and provide high system reliability. There are three basic types of ARQ schemes: *stop-and-wait, go-back-N* and *selective-repeat.* Selective repeat ARQ is the most efficient of the three but more complex to implement.

However, both FEC and ARQ systems suffer some drawbacks especially for non-stationary channels. In a FEC system, when errors are detected in a received codeword, it must be decoded and the decoded message must be delivered to the user regardless of whether it is correct or incorrect. Since the probability of a decoding error is much greater than the probability of an undetected error, it is hard to achieve high system reliability with FEC. In order to obtain high system reliability, a long powerful code must be used. This calls for high overhead per codeword and decoding

hardware that is complex and expensive. This additional overhead, complexity and hardware may be justified only if the channel has a high noise level constantly. If the noise level is not constant then at low error rates the additional overhead brings the throughput down to below what is possible with pure ARQ schemes. The ARQ systems, on the other hand, suffer from the severe drawback that their throughput efficiency falls rapidly with increasing channel error rate, although they guarantee high reliability. The ARQ schemes perform quite satisfactorily for low error rate channels.

For channels where the error rate is not constant, the drawbacks of both ARQ and FEC schemes could be overcome if the two ideas are properly combined. Such combinations of these two basic error control schemes is referred to as a *hybrid* ARQ. A hybrid ARQ system consists of an FEC subsystem contained in an ARQ system. The function of the FEC system is to correct the error patterns that occur most frequently and thus reduce the frequency of retransmissions. This increases the system throughput. When a less frequent error pattern occurs and is detected, the receiver requests a retransmission rather than delivering the unreliably decoded message to the user. This increases the system reliability. Thus a hybrid ARQ scheme achieves higher reliability than a FEC system alone and higher throughput than a system that uses pure ARQ only, especially when the channel is noisy. Furthermore, since the decoder is designed to correct only a small subset of error patterns, it can be simple.

There are two types of *hybrid* ARQ schemes: *type-1* and *type-2* [Lin-Yu 82]. In a type-1 hybrid ARQ scheme, only one code is used for *simultaneous error correction and detection.* If the number of errors in a received message is within the designed error correcting capability of the code, the errors will be corrected and the decoded message will be passed on to the user. If an uncorrectable error pattern is detected, the receiver discards the received message and requests a retransmission. When a

retransmitted message is received, the receiver again attempts to correct the errors (if any). If the error pattern is uncorrectable, the receiver again requests a retransmission. This process continues until the message is successfully received or decoded. Since this scheme uses the same code for both error correction and detection, the number of parity check bits needed is higher than what would be needed for pure error detection (as in pure ARQ scheme). When the message is being transmitted over a channel with low bit-error-rate (BER), very few frames are received erroneously (hence very few retransmissions). In such a situation, a type-1 hybrid ARQ scheme will have lower throughput than it's corresponding pure ARQ scheme. However, when the channel BER increases, the type-1 hybrid ARQ scheme will have higher throughput than a pure ARQ scheme because the error correction feature reduces the number of retransmissions.

On the other hand, in a type-2 hybrid ARQ scheme, the parity check bits for error correction are sent to the receiver only when they are needed. The message is first coded based on a high rate error detection code and transmitted. When the receiver detects the presence of errors in the received message, it stores the erroneous message in a buffer and requests a retransmission. The transmitter then forms a frame of parity bits based on the original message and an error correction code. This parity frame is then coded based on the same error detection code as before and transmitted. When this parity block is received, the receiver calculates the syndrome of the block based on the error detection code. If the syndrome is zero, then the message is recovered from the parity frame. If the syndrome is non-zero then the parity frame is combined with the original message frame (stored in the buffer) to correct errors based on the error correction code. If the error correction is successful, the corrected message is delivered to the user. However, if the decoding is not successful, the receiver requests a second retransmission. The second retransmission is either a message block or a parity block.

That depends upon the retransmission strategy being used. One possible strategy would be to send retransmissions of message and parity blocks alternately until the message is either successfully received or successfully decoded.

In this dissertation, we study the application of *pure ARQ* and *hybrid* error control systems for broadcast channels. FEC schemes for these channels have been studied elsewhere in literature. We have considered two types of broadcast environments: Homogeneous and Non-homogeneous. Usually, the term *broadcast communications* implies the case of a single source sending out messages to *all* the destinations (users) in the channel. Every message is considered useful for every receiver in the channel. The goal is to deliver the messages error-free to *all* the receivers. This type of broadcast channel will be called *homogeneous broadcast* (HB) channel. There are many broadcast scenarios where the messages sent out by a transmitter are intended for only a subset of the entire population. In the most general case, every successive message may be addressed to a different subset of the broadcast population and these subsets may not be disjoint. Such a communication environment will be called a *non-homogeneous broadcast* (NHB) channel. The NHB channel has also been referred to as the *multi-cast* channel in the literature.

The issues that arise in the study of these schemes can be broadly classified into two categories: *protocol* and *reliability*. The protocol related issues deal with the use of sequence number fields for the proper identification of messages, acknowledgements etc. at the transmitter and receivers. The protocol enables the receivers to deliver the messages to the users in the order in which they were transmitted. In addition to all that, an important part of the protocol is the choice of an appropriate retransmission strategy (e.g., go-back-N, selective repeat etc.) to achieve the best possible channel utilization (throughput). The reliability studies deal with the use of error control coding to meet certain error-performance criteria. For instance, it may be desirable

to achieve a probability of block error that is less than or equal to a given threshold. If we have some prior knowledge of the channel then we can choose an appropriate coding scheme to meet this criterion.

Both the protocol and reliability issues have a direct impact on the memory as well as processing requirements at transmitter and the receivers. Together, they also determine the channel throughput and delay in the delivery of messages to the user. The goal of these measures is to deliver the messages with maximum throughput and minimum delay while meeting the specified reliability criteria.

1.2 Overview of This Thesis

Almost all the work done so far in the area of retransmission- error-control for broadcast channels has been for the HB channel. The coding problem for the HB channel is synonymous to that of the point-to-point channel. Mase et. al.[Mase 83], Gopal et. al. [Gopal 84], Towsley [Tows 85] and Sabnani et. al. [Sabnani 85] have proposed different versions of go-back-N schemes for the HB channel. In this dissertation, we have considered selective repeat retransmission schemes for both of the broadcast environments mentioned earlier. The details of selective repeat protocol implementation for the HB channel has been studied by Sabnani and Schwartz [Sabnani 85], Chandran and Lin [Chandran-Lin 86] and subsequently by others. In [Sabnani 85], Sabnani and Schwartz consider the ideal selective repeat ARQ, where each receiver has infinite buffer capacity. The more realistic case where each receiver has a finite buffer capacity has been analyzed in [Chandran-Lin 86], [Tow-Mit 87], [Wang-Sil 87] and [M-Q-R 88].

We introduce the terminology in Chapter 2 and then propose an ARQ scheme for the HB channel where each receiver has a finite buffer capacity for storing messages. In this scheme, some aspects of implementation that we believe were ignored

by all the previous work has been accounted for. The scheme we propose simplifies the transmitter operation and improves throughput. We have also applied the *dynamic programming optimization* technique of Wang and Silvester [Wang-Sil 87] to our scheme. The resulting *optimum scheme* allows us to choose the *optimum* number of copies of a message, that needs to be transmitted, at every stage of it's retransmission process by considering the number of receivers that are yet to acknowledge a message. Analysis of the schemes show that our schemes outperform all the previous schemes especially when the channel error rate is high. As a special case, our schemes are also applicable to point-to-point channels when the number of receivers in the broadcast environment is set to one. This scheme also outperforms all the existing point-to-point ARQ schemes at higher bit-error-rates (BERs).Thus our schemes extend the useful range of ARQ schemes over such channels. Having established the superiority of the retransmission protocols developed in this chapter, we have used it in all the subsequent schemes proposed for HB channels.

In Chapter 3, we propose and analyze a type-2 hybrid ARQ scheme for the HB channel. The coding scheme used in this scheme is the parity- retransmission scheme due to Lin and Yu [Lin-Yu 82]. The parity retransmission combined with the dynamic programming optimization greatly improves the throughput over pure ARQ schemes when the channel is noisy. When the channel is not noisy, the throughput of this scheme is as good as that of the scheme proposed in Chapter 2.

We have also proposed a very robust error control system using cascaded coding, in Chapter 4. The coding scheme used here is obtained by cascading two error correcting codes: the inner code and the outer code. The inner code is a binary code designed for simultaneous error correction and detection. The outer code is obtained by interleaving a non-binary code with symbols from Galois field $GF(2^l)$. This code is designed for correcting symbol errors and erasures. The interleaving of the outer

code facilitates burst-error correction. There is a parity retransmission feature in our coding scheme that enables the recovery of erroneously received messages. We have analyzed the probabilities of decoding error and undetected error for this scheme. This coding scheme provides high reliability even at bit-error -rates as high as 10^{-2}. We have proposed an optimum type-2 hybrid ARQ scheme for the HB channel using this coding scheme. This scheme is suitable for high-speed file transfer over channels that are very noisy or non-stationary because this scheme provides high throughput over a wide range of bit-error-rates.

In Chapter 5, we study the error-control problem for the NHB channel. Recently Gopal and Rom [Go-Rom 90] have proposed some go-back-N and some ideal selective repeat ARQ schemes for the multi-cast channel. Their protocols are based on time-sharing ideas. However, we have proposed some schemes that are not time-shared. We obtain the time-shared schemes as special cases of the proposed schemes. All the protocols presented in this chapter assume finite buffers for each receiver which makes these schemes very practical for implementation. For comparison, we have also derived the throughput efficiency results for the idealized counterparts for each of these schemes where the receivers have infinite buffer. We have considered two types of NHB channels: *infinite broadcast population channel* and *finite broadcast population channel.* We show that the non-time-shared schemes outperform the time-shared schemes almost always. The case of finite population allows us to exploit the broadcast nature of the channel more effectively. We have also investigated the various trade-offs that arise in the choice of these schemes.The schemes studied in this chapter very closely model the computer communication networks so prevalent today.

We make our concluding remarks and point out several interesting problems for future work in Chapter 6.

Chapter 2

Selective Repeat ARQ Schemes for Homogeneous Broadcast Channels

In this chapter[1] we propose selective repeat ARQ schemes for point-to-multipoint communications. In section 2.2, we propose an ARQ system for the HB channel. This scheme simplifies the transmitter operation and also improves the throughput efficiency compared to the other existing schemes. We give a complete analysis of this scheme in section 2.3. In section 2.4, we apply the dynamic programming optimization technique of Wang and Silvester [Wang-Sil 87] to our scheme. The resulting *optimum* scheme outperforms the scheme in [Wang-Sil 87]. The discussion of computational results appears in section 2.5. In the last section, we point out some implementational aspects of our scheme. We briefly discuss the previous work in this area in the next section and then state the motivation for this work.

2.1 Previous Work

Most of the early work on ARQ systems for point-to-multipoint communications employed the go-back-N strategy. Sabnani [Sabnani 82], Mase et. al. [Mase 83], Towsley [Tows 85] and Gopal et. al [Gopal 84] studied different versions of go-back-N schemes for the point-to-multipoint channel. Sabnani [Sabnani 82] also looked into an ideal selective repeat ARQ scheme for this channel where each receiver has infinite

[1]The results of Chapters 2 and 3 will appear in the Jan.,1992 issue of the IEEE Transactions on Communications [Chandran-Lin 92].

buffer capacity to store messages. The more realistic case where each receiver has only a finite buffer capacity was first analyzed by Chandran and Lin [Chandran-Lin 86]. In [Sabnani 82] and [Chandran-Lin 86] the authors also discuss the details of protocol implementation for the HB channel. More recent work involves the transmission of multiple copies of messages to achieve better throughput performance.

Sending multiple copies of a frame, in case of a non-ideal ARQ system, increases the probability of it's successful reception. This allows the receivers to release that frame and all subsequent frames in sequence that have been received successfully. The released frames free up space in the receiver buffer. This in turn makes *buffer overflow* a less probable and hence less frequent event, thereby increasing throughput. Like all good things in life, the sending of multiple copies is unfortunately a double-edged sword. If too many copies of a frame are sent out, we run the risk of putting out copies of a frame even after all receivers have received at least one copy of the frame successfully. This results in wasted transmission time slots which could have been used to deliver other frames. Hence the throughput is decreased. Clearly, there is a trade-off involved between the number of copies of a frame that needs to be transmitted in transmissions/retransmissions and the throughput. Sabnani's scheme [Sabnani 82] does not allow multiple copies of a frame. However theirs was an ideal ARQ system. It is not very difficult to show that single copy is the optimum choice (in both transmissions and retransmissions) for achieving maximum throughput, in the ideal ARQ system, regardless of the channel noise level. This is so because there is no penalty for sending required number of copies of a frame later since there is *never* a buffer overflow in the ideal case.

Recently, Towsley and Mithal [Tow-Mit 87], Wang and Silvester [Wang-Sil 87] and Mohan et. al. [M-Q-R 88] have proposed different versions of selective repeat ARQ

schemes for the HB channel. Each of these schemes are for receivers with finite buffers and allow multiple copies of frames to be sent in transmissions and retransmissions.

The scheme in [Tow-Mit 87] is based on Weldon's point-to-point scheme [Weldon 82]. Weldon's scheme proposed multiple copies only for retransmissions and single copy for first transmission. Weldon also gave an empirical procedure to calculate the number of copies of a frame that would maximize throughput for the case when the receiver has a buffer size of N frames, where N is the number of frames that can be transmitted in one round trip propagation delay. Later, Chang and Leung [Chang 84] observed that single copy in first transmission is not an optimal choice for maximizing throughput especially when the channel is noisy. They modified Weldon's scheme to accommodate multiple copies in the first transmission. In addition, they also gave a complete procedure to calculate the optimum values of the number of copies of a frame, for a receiver buffer size of N. In [Tow-Mit 87], the authors apply this modified scheme to a point-to-multipoint channel. The scheme in [M-Q-R 88] is based on the point-to-point scheme of Yu and Lin [Yu-Lin 81] and proposes the use of multiple copies for retransmissions only. Wang and Silvester, in [Wang-Sil 87], have outlined a dynamic programming optimization technique for the choice of optimum number of copies of a frame at various stages of it's transmission/retransmission procedure. We will talk more about their scheme in section 2.4.

2.1.1 Motivation for This Work

In the schemes [Weldon 82], [Chang 84], [Tow-Mit 87] and [Wang-Sil 87], the authors have assumed that all the copies of a frame are transmitted consecutively and that the retransmissions must begin exactly $(N-1)$ frames after the transmission of the last copy of frame. In doing so, the transmitter may be required to transmit two or more frames at the same time. In that case the lower numbered (older) frame is sent

first [Weldon 82, Tow-Mit 87]. In our view, the protocol becomes needlessly complex in order to accomplish these objectives. In what follows, we will illustrate this point with an example. We now introduce some terminology which will be used in this as well as later chapters.

A frame is said to be in *state*-0 if it has never been transmitted before. Let us assume that m_0 copies of the frame are transmitted in *state*-0. If all copies of the frame are received in error in it's first transmission of m_0 copies, then the frame is said to have reached *state*-1. If the frame is not received successfully in *state-j* ($j \geq 1$) then it reaches *state*-$(j+1)$, and so on. Let us assume that m_j copies of the frame are transmitted in *state-j* ($j \geq 1$). Let us also assume that the receivers have a buffer that can hold βN frames, where β is an integer and N is the number of frames that can be transmitted in one round trip delay. We now continue with our discussion.

Fig. 2.1 shows the transmission and retransmission procedure of a transmitter under the schemes [Chang 84] and [Tow-Mit 87] for the case of $\beta = 2$ and $N = 10$. This discussion is with reference to the "oldest" frame (sequence number 1). In *state-1*, we see that the transmission of frame-6 is interrupted to retransmit frame-1. Unfortunately, the second copy of frame-6 cannot be transmitted until the transmitter retransmits frames 2 and 4 because each of these two frames are "older". By the time the transmitter transmits second copy of frame-6, it has already received a negative acknowledgement (NAK) for the previous copy that was transmitted nine time-slots ago. In the figure, we have followed the progress of frame-1 till buffer overflow. All the places where conflicts occur are indicated by arrows. We can see that there are eleven places where such conflicts force the transmitter to modify it's input queue. We recall that this is an example where $N = 10$. In practice, it is not unusual to have a round-trip delay of several hundred. In those cases the transmitter will be forced to keep altering it's input queue rather frequently and the operation becomes complex.

We have made a modification [Chandran-Lin 90] in the scheme which simplifies the operation of the transmitter and improves the throughput performance.

In the next section, we precisely define our channel model, introduce some terminology and then describe our scheme.

2.2 The Proposed Scheme

The Communication Environment

The environment for which we propose these retransmission schemes consists of $(R+1)$ stations where a single transmitter broadcasts frames to R receivers. Each receiver can store βN frames for processing, where β is a positive real number. The receivers send acknowledgements back to the transmitter through a separate return channel. For simplicity, we consider a system which has only one transmitter and R receivers. However, in a real broadcast environment, all the stations may be transmitting and receiving simultaneously. In that case we may have to specify the channel access mechanism used by various stations.

Transmitter Operation

Messages are sent in fixed length frames. Each frame has a sequence number that can be used to distinguish between $\lceil(\beta + 1)\rceil$ different frames. These sequence numbers are cyclically reusable [Sabnani 82, Chandran-Lin 86]. After a frame is transmitted, it is stored in the retransmission buffer. A frame becomes a time-out frame if $(N-1)$ frames have been transmitted following that frame. When a frame becomes a time-out frame, the transmitter expects either a positive acknowledgement (ACK) or a negative acknowledgement (NAK) for that frame from each of the receivers. Corresponding to each frame in the retransmission buffer, the transmitter maintains a list of all the receivers that are yet to acknowledge the frame. Whenever a receiver acknowledges a

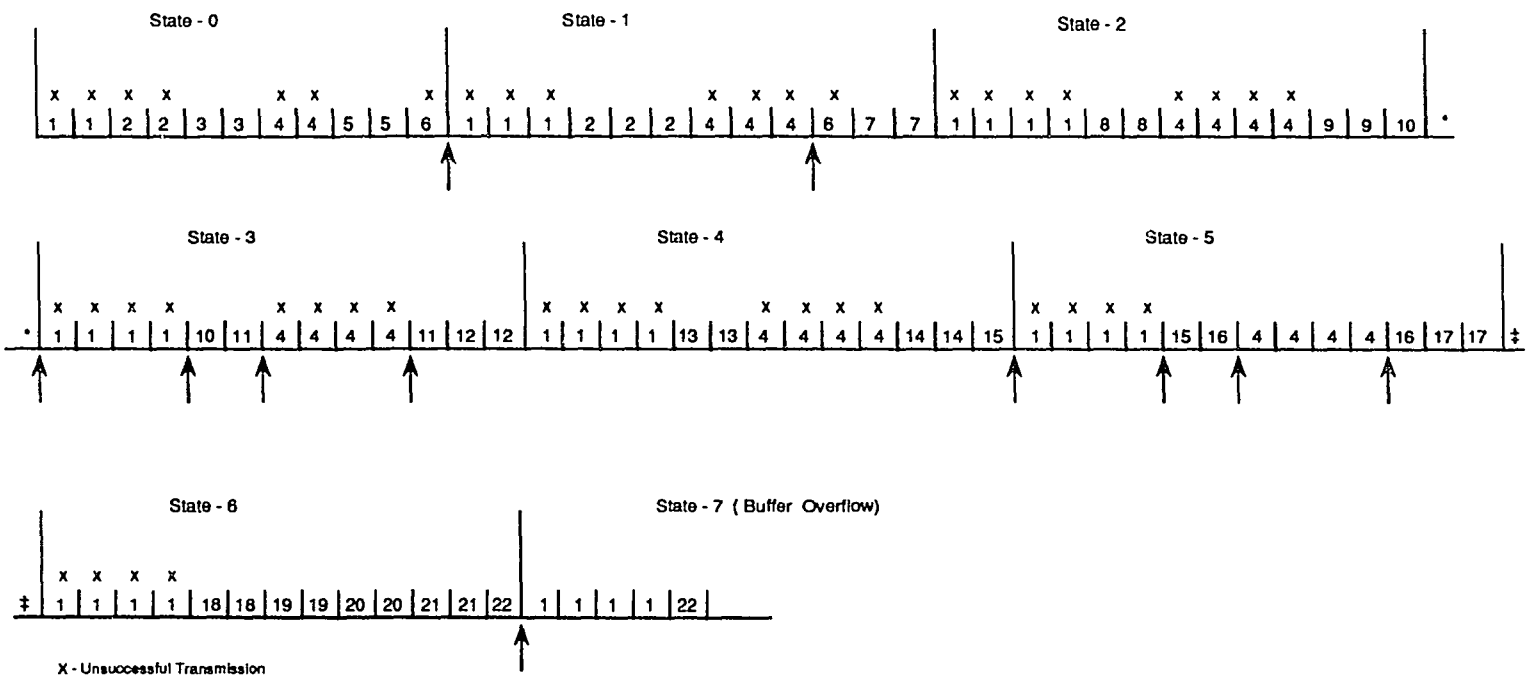

Fig. 2.1 Transmitter Operation for the Weldon / Chang - Leung / Towsley - Mithal Scheme
($m_0 = 2$, $m_1 = 3$, $m_2 = m_3 =$ -- -- $= 4$, $N = 10$, $\beta = 2$)

frame, that receiver is deleted from the list associated with the corresponding frame. If fewer than R ACKs are received at the transmitter, then a retransmission of that frame is initiated. If a receiver fails to send either an ACK or NAK, it is assumed that the receiver did not receive the frame.

The main difference between our transmitter and that in other schemes is that we insist on sending all copies of a frame, for both transmissions and retransmissions, *consecutively.* In *state-j* ($j \geq 0$), the transmitter sends out m_j copies of the frame *consecutively.* As an immediate consequence, we may not be able to start retransmissions exactly $(N-1)$ slots after the last copy of the frame has been transmitted. Whenever a retransmission is required, the transmitter finishes the transmissions (if it is in the middle of one) and then sends out the retransmissions. In case of a conflict between retransmissions, the priority is given to "older" frames (lower sequence numbers), just as in [Weldon 82, Chang 84, Tow-Mit 87]. This prioritizing makes sense because the older frames are the ones holding up the delivery of error-free frames that are received subsequently. Retransmissions continue until all R receivers successfully receive the frame. At that point the frame is discarded from the retransmission buffer. In case of a buffer overflow, the frames lost due to buffer overflow are put back in the input queue. These frames are still considered to be in the same *state* as before.

Since this protocol is driven by positive acknowledgements alone just like the Chandran-Lin scheme [Chandran-Lin 86], it is technically possible to operate the protocol with ACKs alone.

Receivers' Operation

All receivers are assumed to be identical. Upon reception of a frame, the receivers compute the syndrome for the received word. If the syndrome is non-zero, then the frame is assumed error-free and an ACK is sent to the transmitter. If the channel is not

noisy, then the frames are received successfully at the receiver and they are delivered to the user in sequence. The receiver buffer is empty under such circumstances. If a frame is received erroneously, it is discarded and a NAK is sent requesting a retransmission. A space is reserved in the receiver buffer for this frame. After the erroneous reception of a frame, all the frames that are received successfully following that frame cannot be delivered to the user until this frame is recovered. When the "oldest" frame is received successfully, then this frame and all subsequent frames that were received successfully are released from the receiver buffer. The receiver stops delivering frames to the user when it encounters yet another unsuccessful frame. The receiver operation continues in this fashion. All the ACKs and NAKs contain the identity of the receiver and the sequence number of the frame being acknowledged. If the receiver buffer overflows, all the subsequently received frames are discarded (NAKed), because there is no space in the receiver buffer to store these frames. These frames will have to be retransmitted.

The operation of this scheme is shown in Fig. 2.2. For the sake of comparison, we have assumed the same parameters as in our previous example (see Fig. 2.1).

In the next section, we give a complete throughput analysis of the scheme described above.

2.3 Throughput Analysis

In the analysis, the return channels are assumed to be error-free. The noise process that causes the frames to be received erroneously is assumed to be independent and identically distributed from receiver to receiver (as in the down-link of a satellite). The probability of undetected error is ignored in the analysis because by choosing an appropriate error-detecting code, this probability can be made arbitrarily small

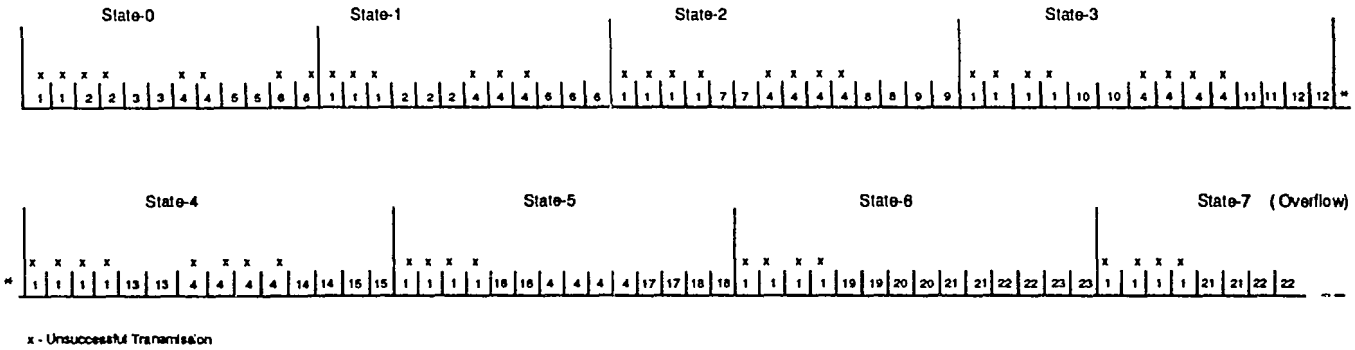

Fig. 2.2 Transmitter Operation in the Proposed Scheme ($m_0=2$, $m_1=3$, $m_2= m_3= \cdots = 4$, $N=10$, $\beta=2$).

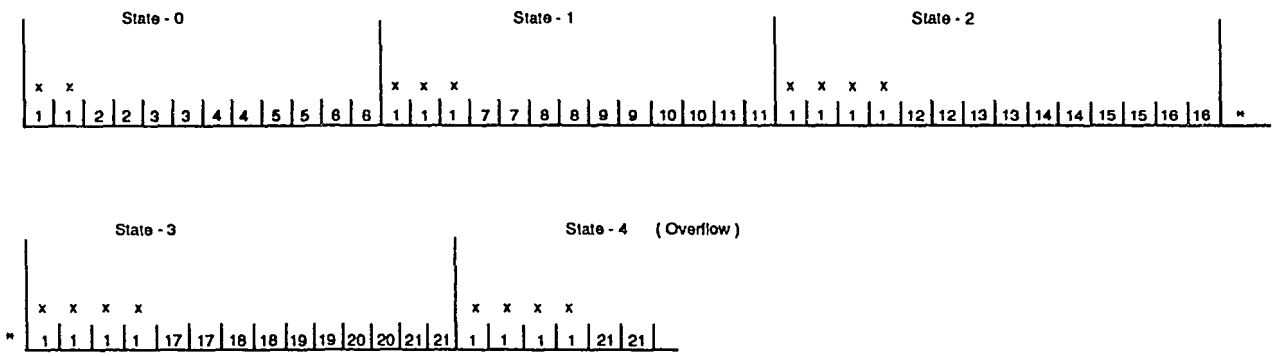

Fig. 2.3 Transmitter Operation for the "worst case" ($m_0=2$, $m_1=3$, $m_2= m_3= \cdots = 4$, $N=10$, $\beta=2$).

[Lin-Cos 83]. We also assume that there is always a frame waiting to be transmitted and that the transfer of frames from receivers to users is instantaneous.

The calculation of throughput efficiency hinges on the determination of two key parameters. The first is the state, δ, of a message when the buffer overflow occurs. The second parameter is the number of frames, ν_i $(i \geq \delta)$, lost in each state, after the buffer overflow. Exact expressions for these two quantities are not possible. Instead, we will obtain a lower bound on δ and an upper bound on $\nu_i, i \geq \delta$. For obtaining a lower bound on δ, we will consider the *worst case* scenario for a random message. The worst case is obtained when the message in question is the oldest message in the retransmission buffer (sequence number 1) and all the subsequently transmitted messages are correctly received. This situation is illustrated in Fig. 2.3. In the analysis, all the lost frames that occur due to buffer overflow are attributed to the transmission of *frame*-1. We will calculate the state in which buffer overflow occurs for this case. Later on, we will verify that this indeed is the worst case as far as buffer overflow is concerned.

For the worst case, the number of distinct messages transmitted in state-0 is given by,

$$1 + \lceil \tfrac{N-1}{m_0} \rceil. \tag{2.1}$$

In each subsequent state, an additional $\lceil \frac{N-1}{m_0} \rceil$ distinct messages are transmitted. For some γ, the "instant" in terms of the number of slots where the number of distinct messages equals the buffer capacity is given by,

$$1 + \gamma \lceil \tfrac{N-1}{m_0} \rceil = \beta N. \tag{2.2}$$

Therefore,

$$\gamma = \frac{\beta N - 1}{\lceil \frac{N-1}{m_0} \rceil}. \tag{2.3}$$

In general, γ is a real number. It is not very difficult to see that the buffer overflow occurs when the random message reaches state-δ, where

$$\delta = \lfloor \gamma \rfloor + 1. \tag{2.4}$$

Also, in the worst case, the number of total frames transmitted in any state-j, is given by

$$m_j + \lceil \tfrac{N-1}{m_0} \rceil m_0. \tag{2.5}$$

The number of frames lost in state-δ, ν_δ is given by

$$\begin{aligned} \nu_\delta &= (N-1), && \text{if } m_0 = 1 \text{ and } \beta = 1. \\ &= \left\lceil (\lceil \gamma \rceil - \gamma) \lceil \tfrac{N-1}{m_0} \rceil m_0 \right\rceil && \text{otherwise.} \end{aligned} \tag{2.6}$$

In every subsequent state, the number of lost frames will be $\lceil \frac{N-1}{m_0} \rceil m_0$. So we see that at least for the worst case, we are able to precisely pinpoint the instant (in terms of the number of time slots) when the buffer overflows. Now we can generalize these two results for the general case, when not all subsequent messages after the random message are received successfully. It is fairly straightforward to see that the right hand side of equation (2.4) represents a lower bound on δ for the general case. Because if some of the subsequent messages require retransmissions, then the buffer overflow is going to occur later than that given by equation (2.4). For instance, in Fig. 2.2, the value of δ is 7 whereas for the worst case δ is 4 (see Fig. 2.3). Since we are sending fewer new frames in a general case compared to the worst case, the true value of γ is going to be more than that in equation (2.3). Because of this reason, the right hand side of equation (2.6) is also an upper bound on ν_δ. We let

$$\nu_i = 0, \quad 0 \leq i < \delta. \tag{2.7}$$

Now we wish to find an upper bound on ν_j $(j > \delta)$. After the retransmission of the random message in state-j, the subsequently transmitted messages during the random message's j^{th} state could be in any state s such that $0 \leq s \leq j$. Let

$$\tau_i = \lceil \frac{N-1}{m_i} \rceil, \quad i \geq 0. \tag{2.8}$$

We can upper bound ν_i $(i > \delta)$ as follows

$$\nu_i = max\{\tau_0 m_0, \tau_1 m_1, \cdots, \tau_i m_i\}. \tag{2.9}$$

This is a reasonable bound for this case because the function $\tau_j m_j$ is a very irregular function of m_j. We now derive a lower bound on the throughput efficiency of this scheme. We define the following terms:

- ϵ — Bit-error-rate (BER) of the channel.
- p — The probability that a frame is received successfully by a receiver. Assuming that each frame has n bits, we have

$$p = (1 - \epsilon)^n. \tag{2.10}$$

- $M_j = \sum_{i=0}^{j} m_i$.
- ξ_j — The probability that a frame is successfully received by a particular receiver within j retransmissions. Here, 0^{th} retransmission simply refers to the first transmission of m_0 copies. Clearly,

$$\xi_j = 1 - (1-p)^{m_0 + m_1 + \cdots m_j} = 1 - (1-p)^{M_j}, \quad j \geq 0. \tag{2.11}$$

- q_j $(j \geq 0)$ — The probability that a frame is successfully received by all R receivers in the j^{th} retransmission and at least one receiver did not recover the frame successfully after $(j-1)$ retransmissions. Then

$$q_j = \xi_j^R - \xi_{j-1}^R, \qquad j \geq 0. \tag{2.12}$$

- T' — Total number of transmissions required to successfully transmit a frame to all R receivers in the worst case. Clearly, T' is a random variable.
- T — *Average* number of transmissions required to successfully transmit a frame to all R receivers.
- $\eta = 1/T$ — Throughput Efficiency of the Scheme.

We now compute the mass function of the random variable T' (the worst case statistics). The total number of transmissions required to successfully deliver the frame to all R receivers will be m_0 if it is received in state-0 and M_s if it is received in state-s, $s < \delta$. For states $s \geq \delta$, we have to account for buffer overflow. In general, in $state - s$, we obtain

$$Pr\left\{T' = M_s + \sum_{i=0}^{s} \nu_s\right\} = q_s. \tag{2.13}$$

The equation above can be used to obtain the expected value of the random variable T'. After some simplification, we can write

$$\mathbf{E}[T'] = \sum_{k=0}^{\infty} q_k M_k + \sum_{k=0}^{\infty} q_{\delta+k} \left(\sum_{i=0}^{k} \nu_{\delta+i}\right). \tag{2.14}$$

The above equation can be simplified considerably. Let us consider the expression one term at a time. Rewriting the first term, we get

$$\begin{aligned} \text{First Term} &= M_0 q_0 + M_1 q_1 + \cdots M_\infty q_\infty. \\ &= m_0 \xi_0^R + M_1(\xi_1^R - \xi_0^R) + M_2(\xi_2^R - \xi_1^R) + \cdots. \\ &= \lim_{n\to\infty} \{M_n \xi_n^R - m_n \xi_{n-1}^R - m_{n-1}\xi_{n-2}^R \cdots - m_1 \xi_0^R\}. \end{aligned}$$

$$= \lim_{n\to\infty} \{m_0 \xi_n^R + \sum_{i=1}^{n} m_i(\xi_n^R - \xi_{i-1}^R)\}.$$

Taking the limit and letting $\lim_{n\to\infty} \xi_n = 1$ (because a frame will almost certainly be received after infinitely many transmissions), we get

$$\begin{aligned} \text{First Term} &= m_0 + \sum_{i=1}^{\infty} m_i(1 - \xi_{i-1}^R). \\ &= m_0 + \sum_{i=1}^{\infty} m_i[1 - \{1 - (1-p)^{M_{i-1}}\}^R]. \end{aligned}$$

If we let $M_{-1} = 0$, we can absorb m_0 into the summation. We get

$$\text{First Term} = \sum_{i=0}^{\infty} m_i[1 - \{1 - (1-p)^{M_{i-1}}\}^R]. \tag{2.15}$$

The second term in equation (2.14) can be simplified in a similar manner. We can now write

$$\mathbf{E}[T'] = \sum_{i=0}^{\infty} m_i[1 - \{1 - (1-p)^{M_{i-1}}\}^R] + \sum_{i=0}^{\infty} \nu_{\delta+i} \, [1 - \{1 - (1-p)^{M_{\delta+i-1}}\}^R]. \tag{2.16}$$

The first term in the above equation gives the number of slots used to actually transmit the message. If we let $m_i = 1$ $(i \geq 0)$, this term will give us the total number of slots required to successfully transmit a message in the ideal ARQ scheme. The second term represents the number of frames lost due to buffer overflow. Since $\mathbf{E}[T'] \geq T$, the above equation can be used to obtain an upper bound on the average number of transmissions necessary for a message to be received successfully by all receivers. The throughput efficiency η is then lower bounded as follows.

$$\begin{aligned} \eta &= \frac{1}{T}, \\ &\geq \frac{1}{\mathbf{E}[T']}. \end{aligned} \tag{2.17}$$

2.4 Optimum Retransmission Scheme

One of the difficulties in implementing the schemes discussed in the previous sections as well as the ones in [Weldon 82, Chang 84, Tow-Mit 87] is that the schemes do not provide a satisfactory method for the choice of the optimum number of frames that needs to be transmitted in each transmission. An empirical solution, for the single destination case, was provided by Weldon [Weldon 82] which was later modified by Chang and Leung [Chang 84]. One way to find these numbers, for the multi-destination case, would be by exhaustive search. However, even that approach won't work because with each transmission, the number of receivers that are yet to acknowledge a message is changing too and the optimum number of copies depends upon the number of receivers also. Towsley and Mithal extended the schemes in [Weldon 82, Chang 84] for the multi-destination environment without considering the number of receivers that are yet to acknowledge a message. Recently, Wang and Silvester [Wang-Sil 87] have proposed a dynamic programming approach based on Weldon's scheme, to optimize the throughput. This scheme takes into account the number of receivers that are yet to acknowledge the message in the optimization process. In this section, we apply their approach to our scheme with a slight modification. Results indicate that our optimal scheme outperforms the scheme in [Wang-Sil 87].

We first make the following definitions:

- $T(s,r),\ s \geq 0,\ 0 < r \leq R$ — Residual average number of frames required to successfully deliver a message when it is in $state - s$ and r receivers have not yet responded positively. Clearly $T(\cdot,0) = 0$.

- $m_s(r),\ s \geq 0,\ 0 < r \leq R$ — Number of copies of a message transmitted when the message is in $state - s$ and r receivers are yet to acknowledge the message.

- $\alpha_s(r)$, $s \geq 0$, $0 < r \leq R$ — The probability that a message in state-s is successfully recovered within $m_s(r)$ copies by a particular receiver. Then we can write,

$$\alpha_s(r) = 1 - (1-p)^{m_s(r)}. \tag{2.18}$$

According to the definition above, $T(0, R)$ gives the average number of total frames required to deliver a message to all R receivers. Our objective is to find the optimum values of $m_s(r)$ $(0 \leq s \leq \infty\ ,\ 0 < r \leq R)$. This is an infinite dimensional optimization problem. The scheme in [Wang-Sil 87] is based on the schemes of [Weldon 82, Chang 84] where it is assumed that the receiver buffer *will* overflow in state-$(\beta+1)$ and that the number of lost frames in each state after the buffer overflow is $(N-1)$. Therefore the strategy for all states $s \geq (\beta+1)$, will be the same because the penalty due to buffer overflow is the same. Therefore Wang and Silvester only had to find the values of $m_s(r)$ for $s \leq (\beta+1)$. Hence, theirs was a finite dimensional optimization problem.

However, our scheme is different in two ways. First, the buffer overflow in our scheme occurs in state-δ, given by equation (2.4). This quantity is dependent on the value of m_0 (see equation 2.3). Secondly, the number of lost frames due to buffer overflow is not a constant from state to state (see equation 2.9). In order to reduce this to a finite dimensional problem, we make the following assumption. Let

$$m_J(r) = m_{J+1}(r) = \cdots, \tag{2.19}$$

for some $(J \geq \delta + 1)$ and $(0 < r \leq R)$.

The equation above implies that

$$T(J, r) = T(J+1, r) = T(J+2, r) = \cdots. \tag{2.20}$$

Later, we verify from computation that indeed the throughput is not very sensitive to the variation of m_j for $(j > \delta + 1)$.

Therefore for $s = 0$, we have

$$T(0,R) = \sum_{j=0}^{R} Pr\{\text{Only } j \text{ receivers receive the message successfully in } m_0(R) \text{ copies}\} \cdot \{m_0(R) + T^*(1, R-j)\}, \quad (2.21)$$

where $T^*(\cdot,\cdot)$ is the optimized value of $T(\cdot,\cdot)$. We get

$$T(0,R) = \sum_{j=0}^{R} \binom{R}{j} \alpha_0(R)^j (1-\alpha_0(R))^{R-j} \{m_0(R) + T^*(1, R-j)\}. \quad (2.22)$$

If we let $m_0(R) = m_0$ and $\alpha_0(R) = \alpha_0$ then we have

$$T^*(0,R) = \min_{m_0} \left[m_0 + \sum_{j=0}^{R} \binom{R}{j} \alpha_0^j (1-\alpha_0)^{R-j}\, T^*(1, R-j) \right]. \quad (2.23)$$

In general, for $s < J$

$$T^*(s,r) = \min_{m_s(r)} \left[\sum_{j=0}^{r} \binom{r}{j} \alpha_s(r)^j (1-\alpha_s(r))^{r-j} \{m_s(r) + \nu_s + T^*(s+1, r-j)\} \right]. \quad (2.24)$$

After some simplification we obtain

$$T^*(s,r) = \min_{m_s(r)} \left[m_s(r) + \nu_s + \sum_{j=0}^{r} \binom{r}{j} \alpha_s(r)^j (1-\alpha_s(r))^{r-j} T^*(s+1, r-j) \right]. \quad (2.25)$$

And for $s \geq J$, after some simplification, we get

$$T^*(J,r) = \min_{m_J(r)} \left\{ \frac{Nr}{Dr} \right\}, \quad (2.26)$$

where

$$Dr = 1 - (1 - \alpha_J(r))^r. \quad (2.27)$$

and

$$Nr = m_J(r) + \nu_J + \sum_{j=1}^{r} \binom{r}{j} \alpha_J(r)^j (1-\alpha_J(r))^{r-j} T^*(J, r-j). \quad (2.28)$$

The throughput efficiency is then given by

$$\eta \geq \frac{1}{T^*(0,R)}. \tag{2.29}$$

In order to solve this optimization problem, we first obtain $m_J^*(r)$ $(1 \leq r \leq R)$ from equation (2.27) and then recursively solve for $m_s^*(r)$ for $s \leq J$ and $1 \leq r \leq R$. The algorithm can be written briefly as follows:

1. Set $m_0(R) = 1$ (This fixes the value of δ).
2. Compute $m_J^*(r)$ $(1 \leq r \leq R)$ and the corresponding $T^*(m_0(R), J, r)$, $1 \leq r \leq R$.
3. For $0 \leq s < J$, compute $m_s^*(r)$ and corresponding $T^*(m_0(R), s, r)$ for $0 \leq r \leq R$.
4. $m_0(R) = m_0(R) + 1$. Go to Step-2 and repeat steps 2 through 4 till $m_0(R)$ reaches some predetermined limit.
5. $T^*(0,R) = \min_{m_0(R)}\{T^*(m_0(R), s, r)\}$ and $\eta = 1/T^*(0,R)$.

2.5 Discussion of Results

We have plotted the throughput efficiency of the proposed schemes for $n = 1024$ and $N = 128$. Performance for other values of parameters are similar. We have compared the throughput of our schemes to that of the ideal selective repeat ARQ scheme and the Wang-Silvester scheme. The Wang-Silvester scheme is chosen for comparison because it provides the envelope of all the achievable throughput of schemes in [Weldon 82, Chang 84, Tow-Mit 87, M-Q-R 88]. Similarly, the throughput of the optimum ARQ scheme of section 2.4 is the envelope of all the achievable throughput curves from the scheme in section 2.2 (by varying parameters m_j). The following abbreviations are used in the graphs: "ARQ" for the optimum ARQ scheme, "ARQ(m_0, m_1, m_3)" for the ARQ scheme of section 2.2 with parameters

m_0, m_1 and m_3 etc., "Id" for the Ideal Selective Repeat Scheme and "WS" for the Wang-Silvester scheme.

The figures 2.4 - 2.6, show throughput versus the number of receivers for different bit-error-rates. The curves show several interesting things. For low error rates ($BER = 10^{-5}$), the optimum ARQ and the Wang- Silvester schemes perform equally well. However, as the BER increases, the optimum ARQ scheme gives significant improvement in throughput over the WS scheme. We also see in Figs. 2.4 - 2.7 that the throughput of the proposed scheme can be made very close to that of ideal scheme by increasing the buffer to twice and thrice respectively. Clearly, there is a trade-off between buffer size and error-correction. However, when the channel becomes noisy ($BER = 10^{-3}$), even the ideal scheme does not provide satisfactory throughput. If a system must operate at such noise levels, it would appear that some form of forward error correction (FEC) is required. We will see the effect of introducing FEC in the next chapter. In Fig. 2.5, we see that the optimum ARQ scheme is the envelope of the curves ARQ(2,1,2) and ARQ(1,3,3). The parameters for these two curves were obtained by exhaustive search.

As mentioned earlier, we can obtain point-to-point ARQ schemes from the proposed schemes by letting $R = 1$. Fig. 2.7 shows the variation of throughput with bit-error-rate, for a point-to-point environment. The proposed schemes give improved throughput for this environment also.

The main reason why the proposed ARQ schemes outperform the previous schemes can be explained as follows. For low BERs, the optimum value of m_0 is one and if $m_0 = 1$, the value of δ is $\beta + 1$ (same as in WS scheme). However, for $m_0 > 1$, the value of δ can be greater than $\beta + 1$. So not only that the buffer overflows later than in the WS scheme, but the larger number of parameters m_0, $m_1, \cdots m_{\delta+1}$ over which the throughput is optimized, gives a second order effect on throughput. Our

scheme facilitates better utilization of the receiver buffer making it seem as though the available buffer were larger. Moreover, we have obtained a tighter bound on the number of frames lost in state-δ.

Since the optimum schemes, proposed here, give the envelope of all achievable throughputs, henceforth we will only discuss the optimum scheme. In the next section, we discuss the details of implementing the optimum scheme.

2.6 About Implementation

The implementation of the optimum retransmission scheme can be done using a table look-up procedure. The optimum number of copies of a frame, that needs to be transmitted/retransmitted, can be computed ahead of time and stored in memory. Every time a frame in state-s is to be transmitted to say P receivers, the value of $m_s(P)$ can be retrieved from the stored table. Such information can be stored for more than one value of BER, thus the scheme affords adaptability. Our computation shows that the value of $m_s(r)$ changes very slowly with BER. Therefore, by carefully selecting the stored information, such a scheme can be used adaptively over non-stationary channels.

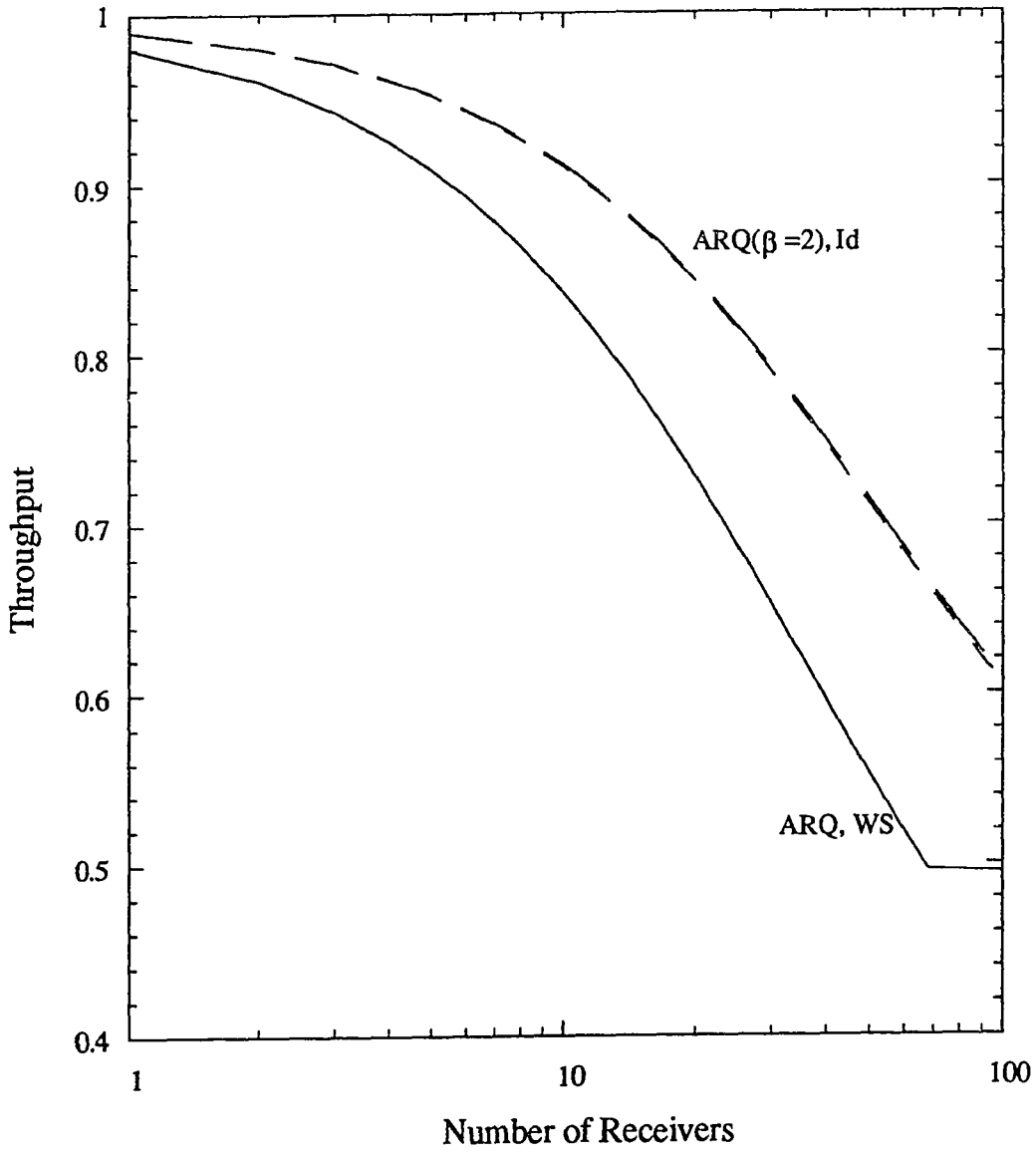

Figure 2.4: Throughput versus number of receivers ($\epsilon = 10^{-5}$).

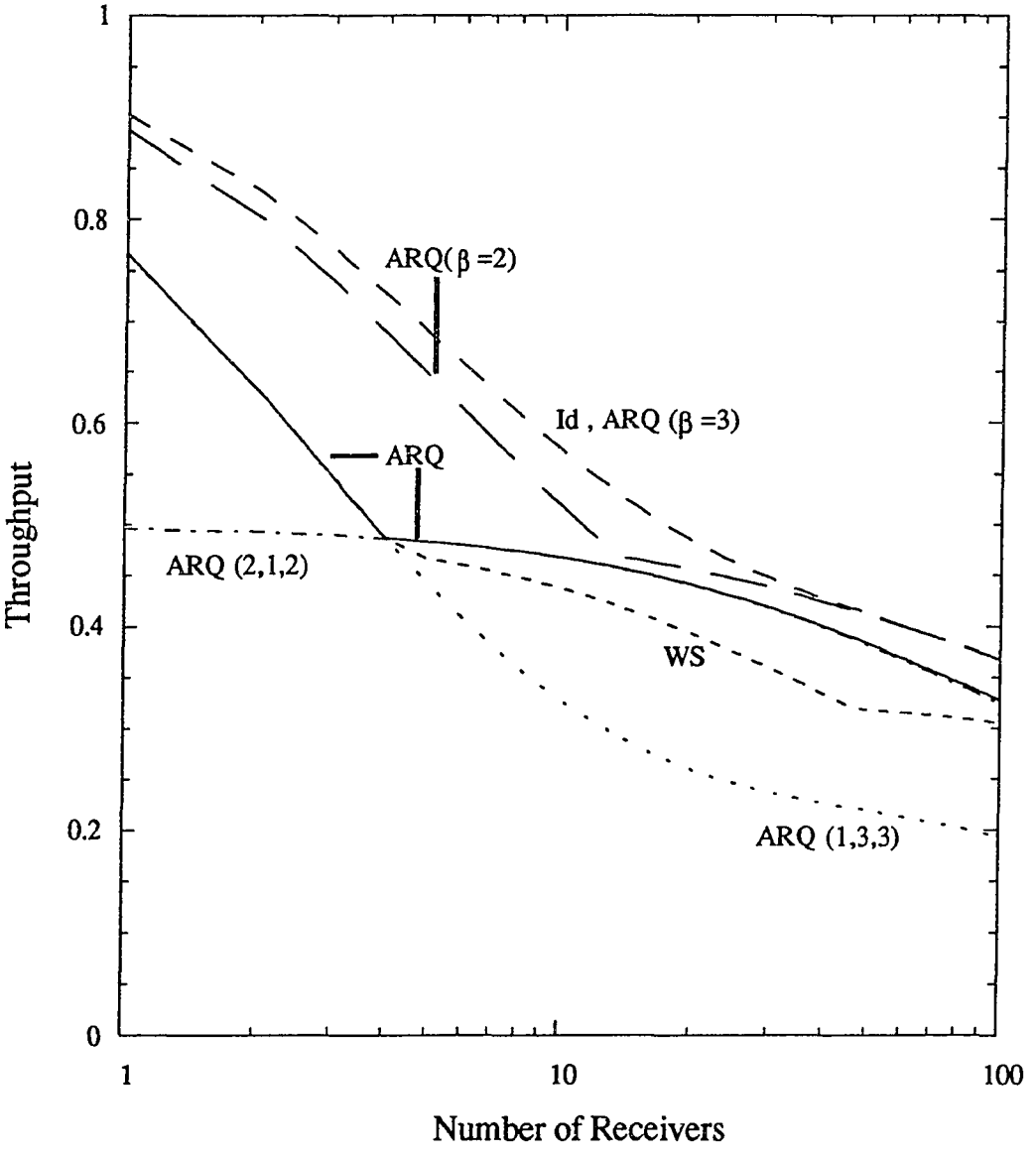

Figure 2.5: Throughput versus number of receivers ($\epsilon = 10^{-4}$).

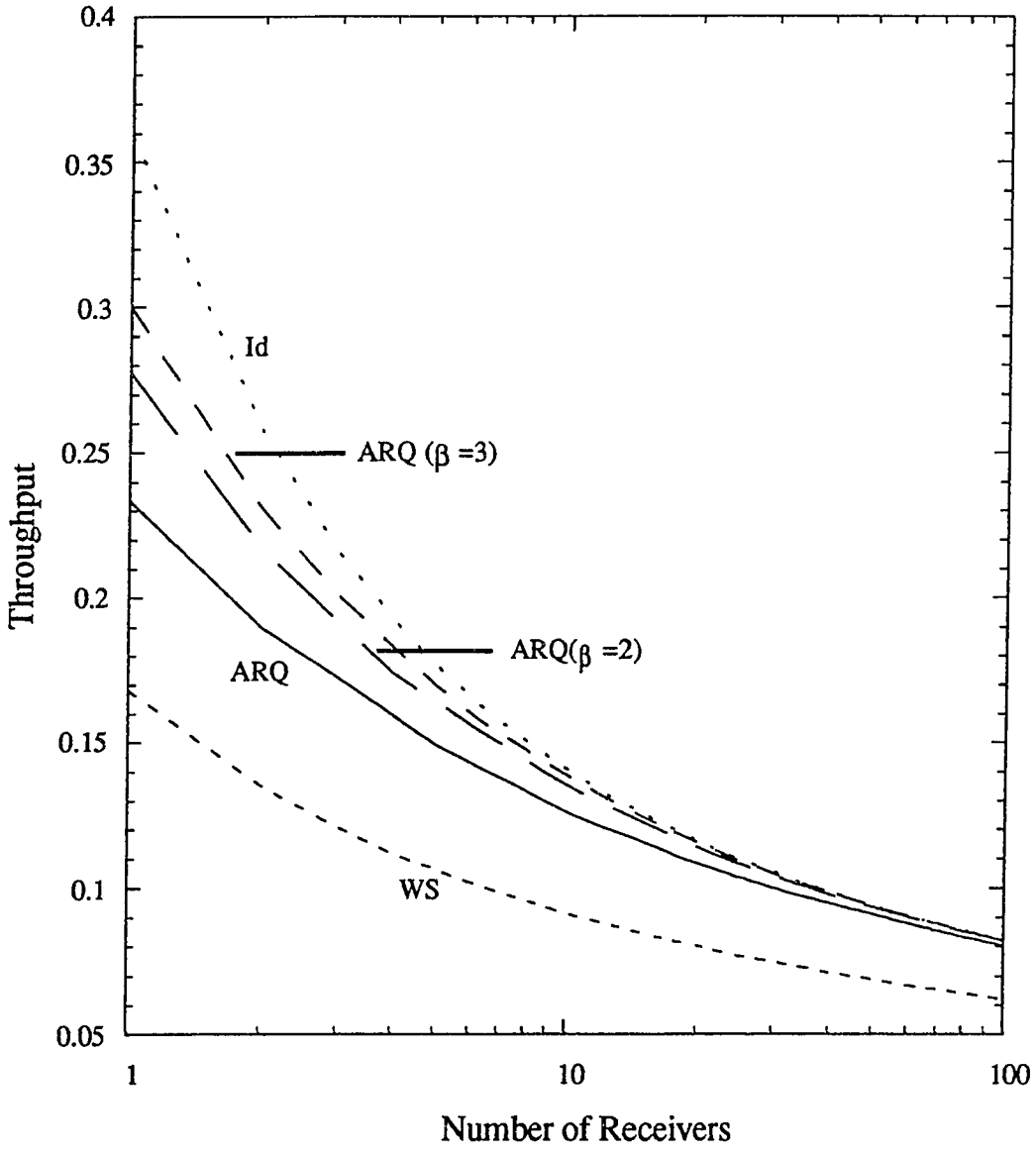

Figure 2.6: Throughput versus number of receivers ($\epsilon = 10^{-3}$).

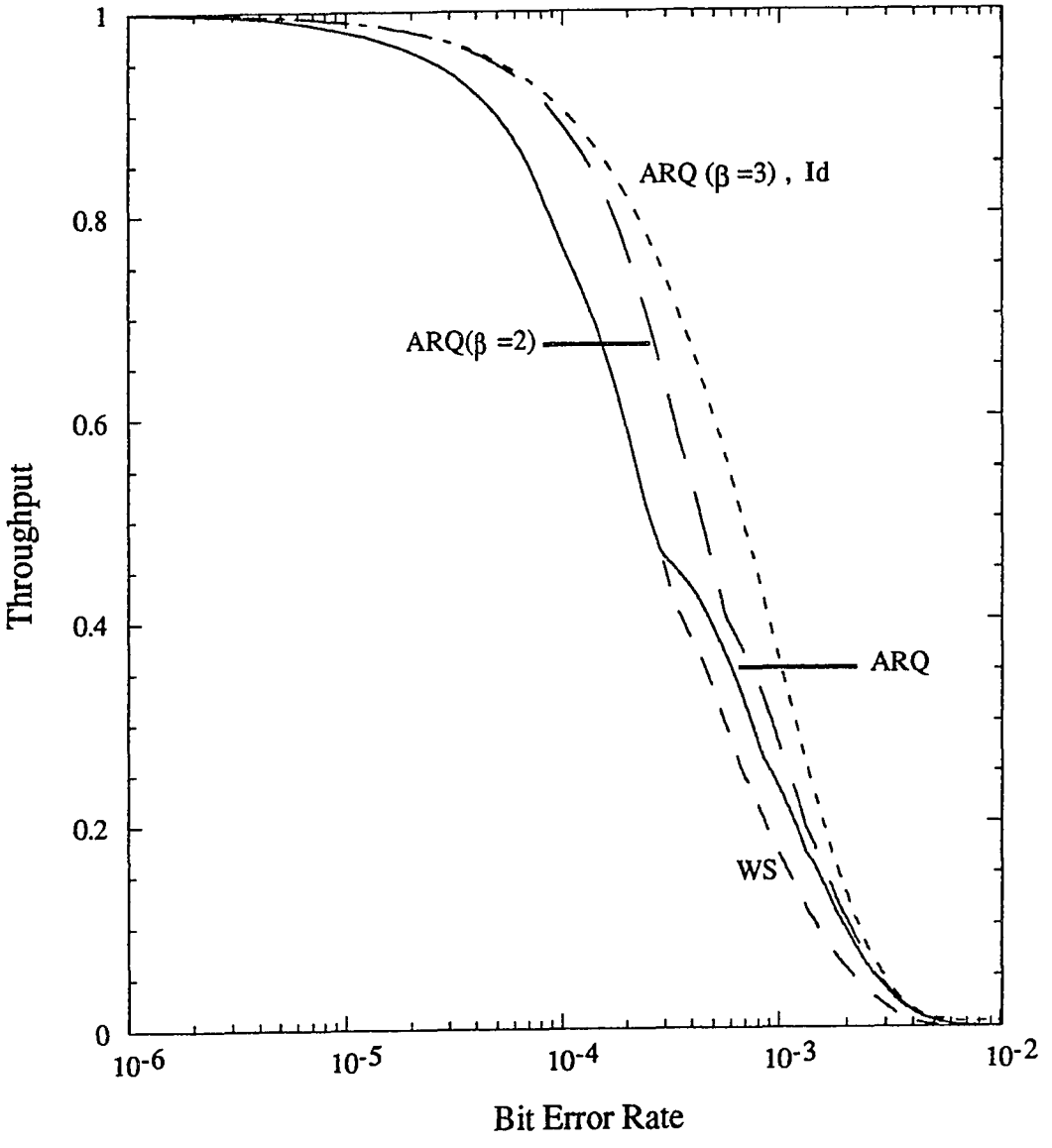

Figure 2.7: Throughput versus Bit-Error-Rate for a point-to-point channel.

Chapter 3

A Hybrid ARQ System for Homogeneous Broadcast Channels

In the last chapter, we saw an ARQ scheme for homogeneous broadcast channels that gives improved performance under noisy conditions. We saw that when the channels become noisy ($BER \geq 10^{-3}$), the throughput performance of even the best ARQ schemes fall far short of desirable levels. It is natural to aim for an error-control system that would provide uniformly high throughput as well as high reliability over a wide range of noise levels. As mentioned in chapter 1, such a scheme is possible by suitably combining the ideas of feedback and FEC. The result of such a combination is what are now known as hybrid ARQ schemes. In this chapter, we will discuss a type-2 hybrid ARQ system that uses parity retransmission. The basic idea of parity retransmission schemes is that the parity check bits for error-correction are sent to receivers only when they are needed. This concept for the case of point-to-point communications was first introduced by Mandelbaum [Mandel 74] then extended by Metzner [Metz 79], Ancheta [Anch 79], Lin and Ma [Lin-Ma 79] and Lin and Yu [Lin-Yu 82]. The scheme presented here [Chandran-Lin 88] is based on the point-to-point scheme of Lin and Yu [Lin-Yu 82].

In the next section, we briefly describe the Lin-Yu scheme. In section 3.2, we describe the proposed hybrid ARQ scheme. Analysis of the proposed scheme appears in section 3.3. We discuss our results in section 3.4.

3.1 The Lin-Yu Scheme

In this scheme, two linear codes are used: one is a high rate (n, k) code C_0 which is used for error detection only and the other is a half-rate invertible $(2k, k)$ code C_1 which is designed for correcting t or fewer errors and simultaneously detecting d $(d \geq t)$ or fewer errors. The code C_1 is invertible in the sense that k information bits of a codeword can be uniquely determined by knowing the k-parity check bits by an inversion process. The design and implementation of such a coding scheme is discussed in [Lin-Yu 82]. The scheme works as follows.

Each incoming message is assumed to have k bits of information. The k bits of incoming data D is first encoded into a codeword (D, Q) of n bits based on the code C_0, where Q represents the $(n - k)$ parity bits. (D, Q) represents the data frame. The transmitter also computes the codeword $(D, P(D))$ based on the message D and the half-rate invertible $(2k, k)$ code C_1. The code C_1 is invertible in the sense that the message D can be recovered uniquely from the $k - bit$ parity $P(D)$ by inversion.

For an incoming message D, the codeword (D, Q) is first transmitted. Let $(\tilde{D}, \tilde{Q})$ denote the received word corresponding to (D, Q). Upon receiving $(\tilde{D}, \tilde{Q})$ the receiver computes the syndrome based on C_0. If the syndrome is zero, then $\tilde{D}$ is assumed to be error-free and will be accepted by the receiver. If the syndrome is non-zero, $(\tilde{D}, \tilde{Q})$ is considered erroneous. The erroneous message $\tilde{D}$ is then saved in the receiver buffer and a NAK is sent to the transmitter. Upon receiving a NAK, the transmitter encodes the $k - bit$ parity block $P(D)$ into a codeword $(P(D), Q^{(1)})$ of n bits based on the error detecting code C_0, where $Q^{(1)}$ denotes the $(n - k)$ parity check digits for $P(D)$. The parity frame $(P(D), Q^{(1)})$ is then transmitted. Let $(\tilde{P}(D), \tilde{Q}^{(1)})$ denote the corresponding received word. The receiver then computes the syndrome of $(\tilde{P}(D), \tilde{Q}^{(1)})$ based on C_0. If the syndrome is zero, then $\tilde{P}(D)$ is assumed to be error-

free and the message D is then recovered from $\tilde{P}(D)$ by inversion. If the syndrome is non-zero, then $\tilde{P}(D)$ and the erroneous message $\tilde{D}$ (stored in the receiver buffer) together are used for error correction based on the code C_1. If the errors in $(\tilde{D}, \tilde{P}(D))$ form a correctable error pattern, they will be corrected. The decoded message D is then accepted by the receiver and an ACK is sent to the transmitter. However, if the errors in $(\tilde{D}, \tilde{P}(D))$ are detectable but not correctable, $\tilde{D}$ is discarded and the erroneous parity block $\tilde{P}(D)$ is stored in the receiver buffer; also a NAK is sent to the transmitter.

Upon receiving a NAK for the message D, the transmitter resends the codeword (D, Q) . When $(\tilde{D}, \tilde{Q})$ is received, it's syndrome is again computed by the receiver, based on C_0. If the syndrome is zero, $\tilde{D}$ is assumed to be error-free and is accepted by the receiver; the erroneous parity frame $\tilde{P}(D)$ is then discarded. If the syndrome is non-zero, then $\tilde{D}$ and the erroneous parity frame $\tilde{P}(D)$ (stored in the receiver buffer) together are used for error correction based on C_1. If the errors in $(\tilde{D}, \tilde{P}(D))$ form a correctable error pattern, they will be corrected. The decoded message D is then accepted by the receiver and an ACK is sent to the transmitter. However, if the errors in $(\tilde{D}, \tilde{P}(D))$ are detectable but not correctable, $\tilde{P}(D)$ is discarded; $\tilde{D}$ is then stored in the receiver buffer and a NAK is sent to the transmitter. The next retransmission will be the parity word $(P(D), Q^{(1)})$. Therefore, the retransmissions are alternate repetitions of the parity word $(P(D), Q^{(1)})$ and the information codeword (D, Q) . The receiver stores the message $\tilde{D}$ and the received parity frame $\tilde{P}(D)$ alternately. The retransmissions continue until the message D is finally recovered either by inversion or by the decoding process.

In what follows, we have generalized the Lin-Yu hybrid scheme to send multiple copies of the message and extended it for the point-to-multipoint environment.

3.2 Description of the Scheme

This scheme is also for the homogeneous broadcast environment described in chapter 2. The protocol for this scheme is almost the same as the one described in sec. 2.2. In state-0, we send m_0 copies of the message. This first transmission, consisting of m_0 copies of the message, starts with a data frame (D, Q) . The remaining $(m_0 - 1)$ frames consist of parity frames $(P(D), Q^{(1)})$ and data frames (D, Q) alternately. The receivers try to recover the message from the first data frame. If they are unsuccessful they temporarily store the erroneous data frame, send a NAK and look for the next copy of the message which is a parity frame. If the parity frame is error-free, the message is recovered by inversion and an ACK is sent to the transmitter. If the parity frame is also erroneous, then the receivers combine the parity frame and the previously stored data frame to correct errors based on code C_1. An ACK is sent to the transmitter if the error correction is successful or else the message is NAKed and the receivers store the parity frame in place of the data frame, which is discarded. The message recovery procedure continues in this fashion and at each stage, the receiver stores one previous copy of the erroneous message which could be a data or a parity frame. We will call this a *single memory* scheme. A retransmission is initiated if all m_0 copies of the message are NAKed by one or more receivers. At this point, the receivers that have not recovered the message after the first transmission have the last copy of the message stored in their buffers. The message is said to have reached the $state - 1$. In this state, the transmitter first finishes the transmission process (if it is in the middle of one) and then sends out m_1 copies of the message. This retransmission starts with a data frame if the last copy of the first transmission was a parity frame or else it is a parity frame. Again the rest of the copies are data and parity frames alternately. The receivers try to recover the message from

these incoming frames. If one or more receivers NAK the message after the first retransmission, then a second retransmission is initiated. We send m_j copies of the message in j^{th} retransmission, $j \geq 1$. Retransmissions continue in this fashion until all receivers ACK the message. A message is stored in the retransmission buffer at the transmitter until all R receivers have ACKed the message. The receivers that have already received a message successfully, ignore retransmitted copies of the message and simply send an ACK for each. In the absence of an ACK or a NAK, the transmitter assumes that the message was unsuccessful. Each frame has a sequence number from a sequence number field that can distinguish between $(\beta+1)N$ different frames, where β is a positive real number.

3.3 Throughput Efficiency Results

We derive a lower bound on the *optimum* achievable throughput efficiency of the hybrid scheme described in the previous section. We use the dynamic programming optimization technique discussed in chapter 2. Although the coding scheme used in our hybrid ARQ system uses single memory, we will give a general analysis that can be used to analyze schemes of arbitrary memory order. All the assumptions and definitions made in sections 2.3 and 2.4 will be used here also.

Let $q^{(1)}$ be the conditional probability that a frame D_j is successfully recovered by a receiver from the first received parity frame $\tilde{P}(D_j)$ either by inversion or by decoding based on code C_1, given that errors are detected in the received data frame $(\tilde{D}_j, \tilde{Q}_j)$. Then from [Lin-Yu 82],

$$\begin{aligned} q^{(1)} &= p + \frac{1}{1-p}\left\{\sum_{i=0}^{t}\binom{2k}{i}\epsilon^i(1-\epsilon)^{2k-i} + (1-\epsilon)^{2n} - 2(1-\epsilon)^n \right. \\ &\quad \left. \cdot\left[(1-\epsilon)^k + \sum_{i=1}^{t}\binom{k}{i}\epsilon^i(1-\epsilon)^{k-i}\right]\right\}. \end{aligned} \tag{3.1}$$

Here again, the quantity p is given by equation 2.10.

Let $q^{(2)}$ be the conditional probability that a frame D_j is successfully recovered by a receiver from the second received data frame $(\tilde{D'}_j, \tilde{Q'}_j)$ given that errors are detected in the received parity frame $(\tilde{P}(D_j), \tilde{Q}_j^{(1)})$ and the first received data frame $(\tilde{D}_j, \tilde{Q}_j)$ and that the first parity frame $\tilde{P}(D_j)$ fails to recover D_j from the first received erroneous frame $\tilde{D}_j$, but detects the presence of errors in $(\tilde{D}_j, \tilde{P}(D_j))$. Then from [Lin-Yu 82],

$$q^{(2)} = p + \frac{1}{(1-p)(1-q^{(1)})} \left\{ \sum_{i=1}^{t-1} \Delta_i S_{t-i}(1 - \Delta_0 - S_{t-i}) \right\}, \tag{3.2}$$

where

$$\Delta_i = \binom{k}{i} \epsilon^i (1-\epsilon)^{k-i} \tag{3.3}$$

$$S_j = \sum_{l=1}^{j} \Delta_l. \tag{3.4}$$

Let $q^{(j)}$ be the conditional probability that a frame is successfully recovered by a receiver from the j^{th} copy, given that the receiver in question is unsuccessful in recovering the message (both by decoding or inversion) up to the $(j-1)^{th}$ copy. We let $p = q^{(0)}$. If the channel is stationary, the conditional probability $q^{(j)}$ for $j \geq 2$ does not change with each successive retransmission, because the receiver stores only one previous frame (data or parity) in each case. This frame was already involved in an unsuccessful inversion and decoding. In this case we get

$$q^{(2)} = q^{(3)} = q^{(4)} = \cdots = q^{(j)} = q^{(j+1)}, j > 2. \tag{3.5}$$

Let $\alpha_0(R)$ be the probability that a message is successfully received by one receiver within $m_0(R)$ copies. Then $\alpha_0(R)$ is given by

$$\alpha_0 = 1 - (1-p)(1-q^{(1)})(1-q^{(2)}) \cdots (1-q^{(m_0(R)-1)}). \tag{3.6}$$

In general, let $\alpha_s(r)$ represent the probability that a message in $state - s$ is successfully recovered within $m_s(r)$ copies by a particular receiver. Then we can obtain $\alpha_s(r)$ as follows,

$$\alpha_s(r) = 1 - \prod_{i=0}^{m_s(r)-1} (1 - q^{m_{s-1}(r)+i}). \tag{3.7}$$

In state-1, we need the value of $q^{m_0(R)}$. If $m_0 = 1$, then the quantity $q^{(m_0)}$ is $q^{(1)}$. If $m_0 > 1$ then q^{m_0} is $q^{(2)}$. For the case of $s \geq 2$, at least two copies of the message have already been transmitted, hence

$$\alpha_s(r) = 1 - (1 - q^{(2)})^{m_s(r)}. \tag{3.8}$$

From here on, the throughput efficiency of this scheme can be obtained by following the procedure outlined in section 2.4.

Two special cases of this scheme can be obtained. The first one is the case when we remove the FEC portion from this hybrid system. This will give us the optimum ARQ scheme proposed in section 2.4. The second special case is obtained by letting $R = 1$ in our hybrid scheme. We obtain an optimum point-to-point hybrid ARQ scheme. We have compared the scheme proposed here and the point-to-point special case to other schemes proposed in literature. A brief discussion of results follows.

3.4 Discussion of Results

We have plotted the throughput efficiency of the optimum hybrid ARQ scheme for $n = 1024$, $N = 128$ and three different values of error-correcting capability ($t = 3$, 5 and10). Figures 3.1 to 3.3 show these curves for BERs of 10^{-5} to 10^{-3}. For comparison, we have also plotted the ideal ARQ and the optimum ARQ obtained in chapter 2. We see that the hybrid scheme gives substantial improvement in throughput for noisy channels ($BER \geq 10^{-4}$). The hybrid scheme outperforms even the ideal

ARQ scheme for BERs$\geq 10^{-5}$. It is clear that sufficient gains can be obtained by using the hybrid scheme, if the channel is noisy.

The throughput of the ideal ARQ scheme can be bounded by the capacity of M-ary erasure broadcast channel with feedback, where M is the number of choices for data frames. However, if we use coding, then it becomes a discrete memoryless broadcast channel with feedback (whose capacity is unknown). Thus it is possible for the hybrid schemes to outperform even the ideal selective repeat scheme.

We have also plotted the throughput efficiency of optimum point-to-point hybrid scheme as a function of BER in Fig. 3.4. We see that the proposed hybrid scheme gives much improved throughput over other schemes when the channel becomes noisy.

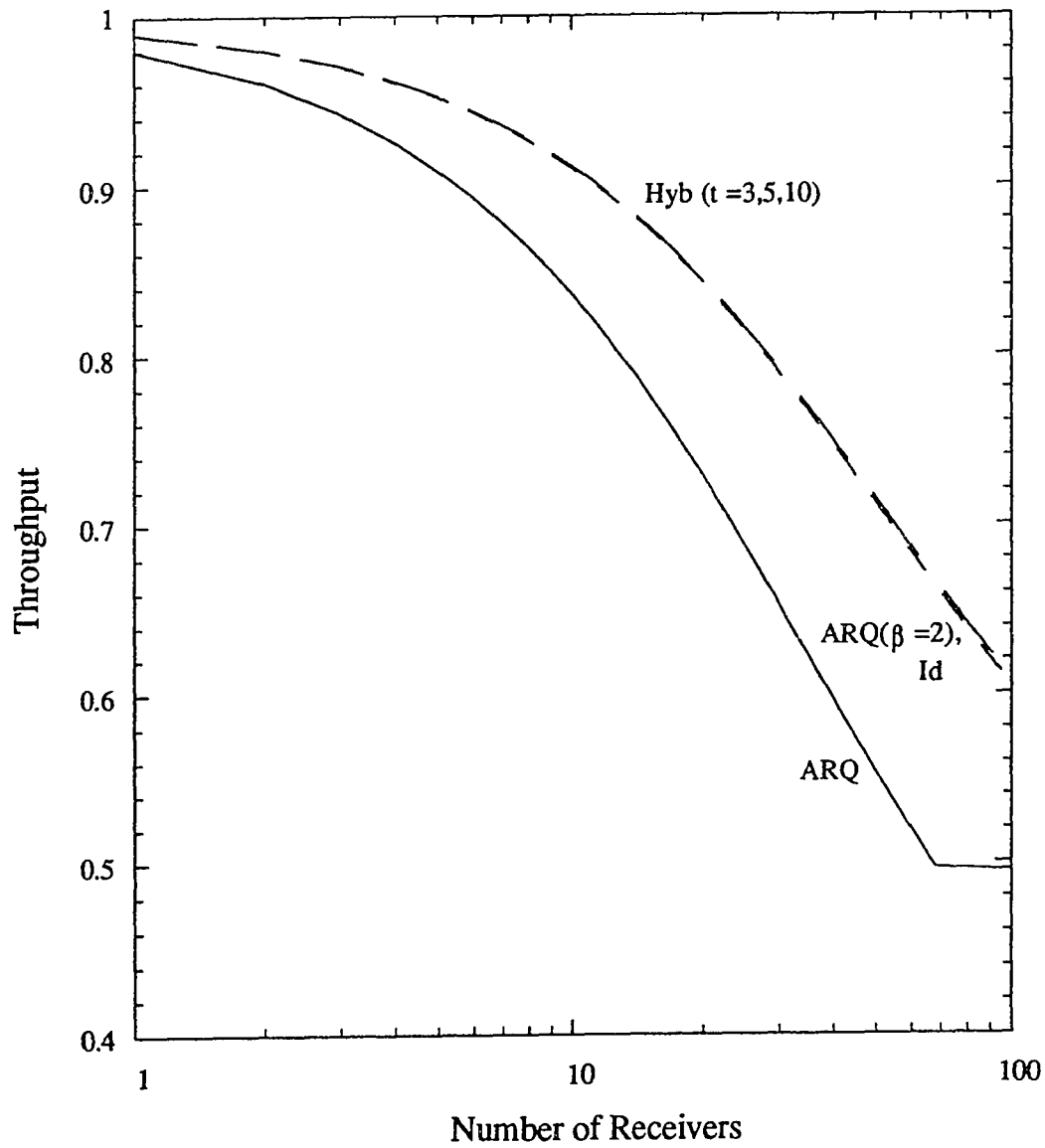

Figure 3.1: Throughput versus number of receivers ($\epsilon = 10^{-5}$).

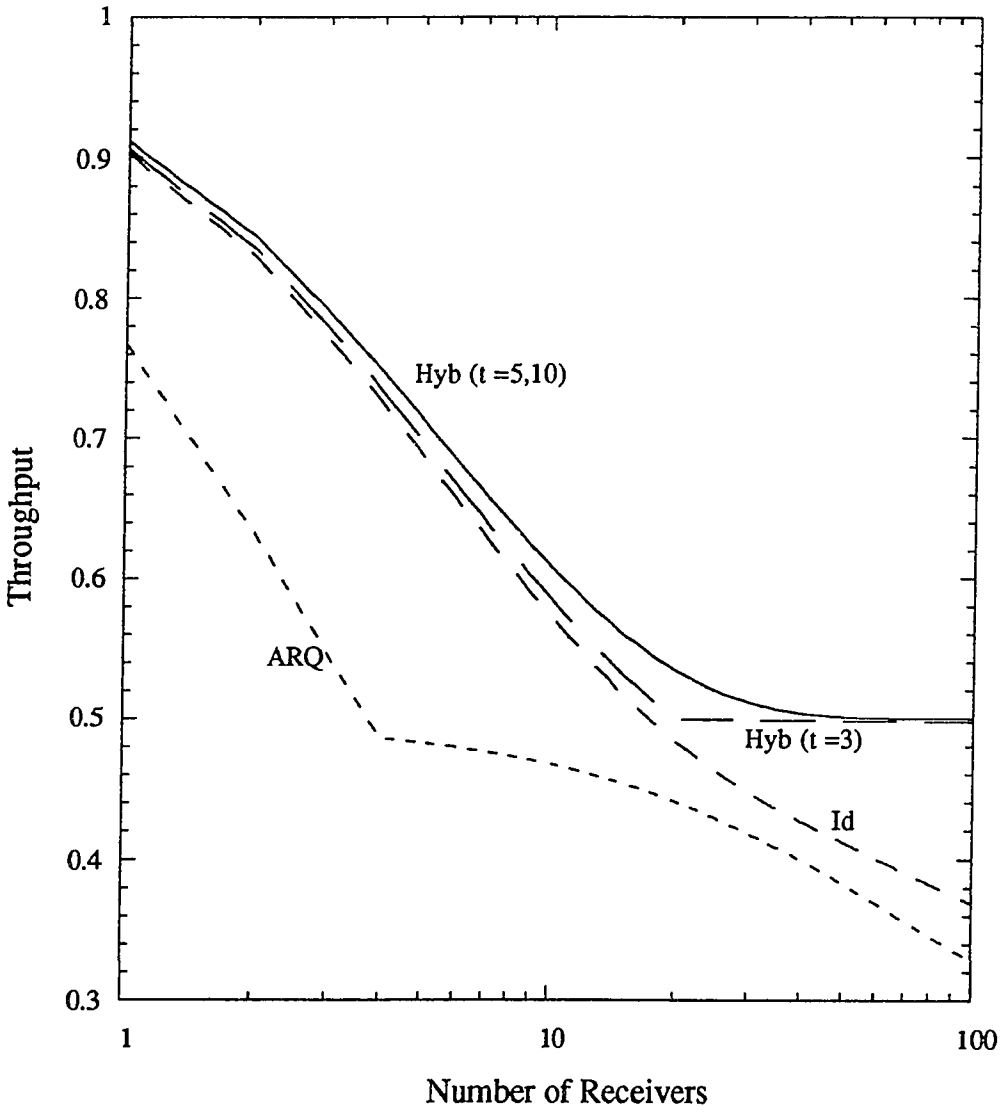

Figure 3.2: Throughput versus number of receivers ($\epsilon = 10^{-4}$).

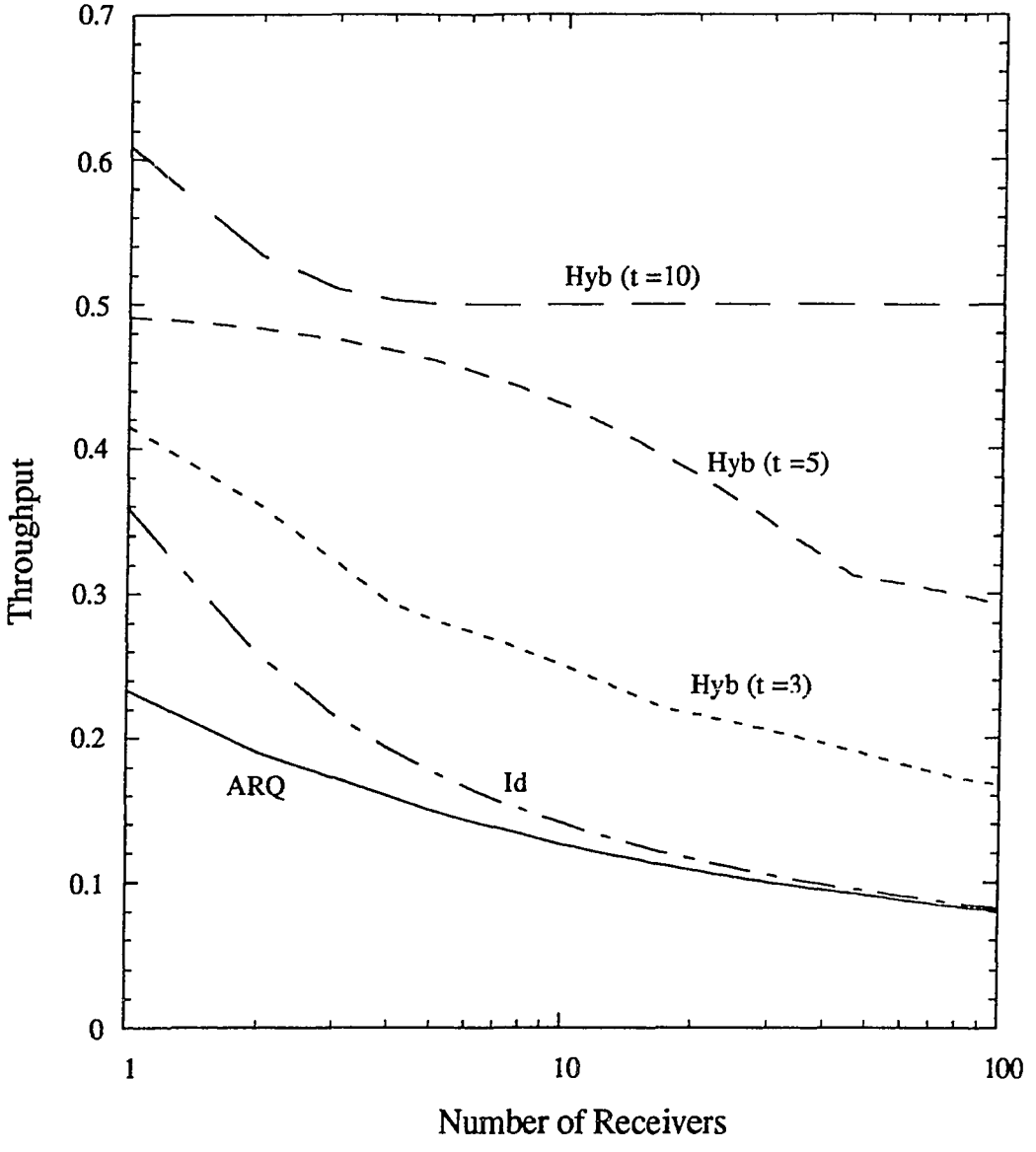

Figure 3.3: Throughput versus number of receivers ($\epsilon = 10^{-3}$).

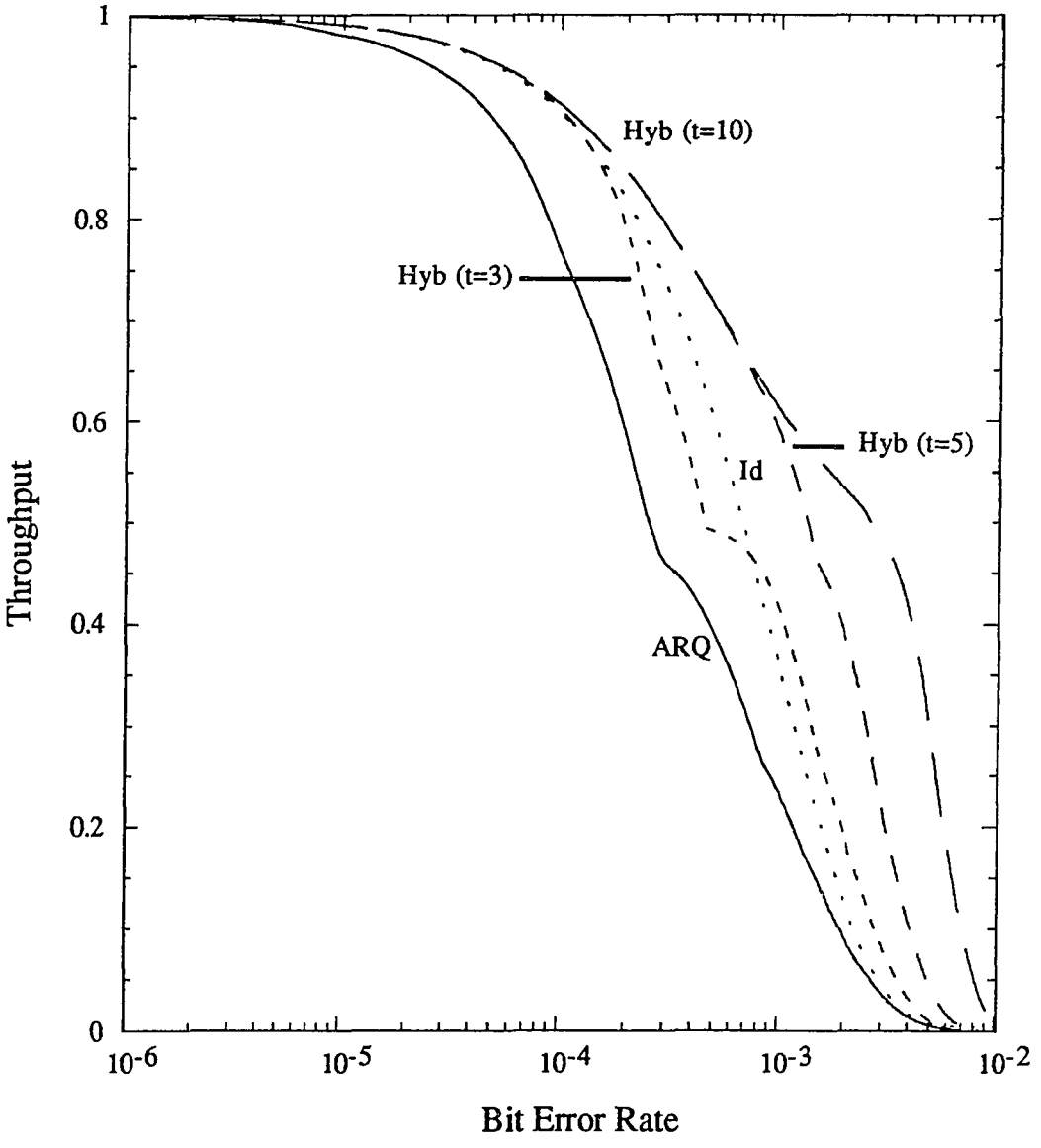

Figure 3.4: Throughput versus Bit-Error-Rate for a point-to-point channel.

Chapter 4

A Robust Error Control System Using Cascaded Coding

In this chapter we present a robust error-control scheme for homogeneous broadcast channels. The scheme is suitable for data communication in an environment that is always noisy. The coding scheme used here is obtained by cascading two error-correcting codes: the inner code and the outer code. The inner code is a binary code designed for simultaneous error correction and detection. The outer code is obtained by interleaving a non-binary code with symbols from Galois Field $GF(2^l)$. This code is designed for correcting symbol errors and erasures. The errors handled by this code are either caused by the channel or the inner code decoder, whereas the erasures are introduced by the inner code decoder only. The interleaving facilitates burst-error correction. In addition, we also have a parity retransmission feature for the recovery of incorrectly received messages. The scheme is a combination of *type-1* and *type-2* hybrid ARQ schemes. The goal of this scheme is to facilitate high speed file transfer over very noisy channels.

Parity blocks for retransmissions are formed based on the original data block and a half rate *invertible* code. This half-rate code is obtained by shortening the inner code. The data and parity transmissions both use the same format hence their decoding procedures are very similar. The original message can be recovered from the parity block also, by inversion, if the decoding is successful. Otherwise, the parity and data blocks are combined for further error correction using the half-rate code. If the parity

block fails to recover the original message, another retransmission is requested. The next retransmission is a data block. The retransmissions are alternate repetitions of data and parity blocks.

In section 4.1, we describe the cascaded coding scheme. A hybrid ARQ scheme based on the cascaded coding is described in section 4.2. In section 4.3, we obtain performance analysis results for the coding scheme described in section 4.2. We obtain the throughput efficiency results for the ARQ scheme for the general case of multiple receivers in section 4.4, where each receiver is assumed to have a finite buffer capacity. We discuss our results in section 4.5.

4.1 The Cascaded Coding

The cascaded coding scheme presented here is similar to the work of Kasami et. al. [Kasami 88]. We have made some modifications to their cascaded scheme and also added a parity retransmission feature to their scheme. The inner code C_1 of this cascaded scheme [Kasami 88] is a binary (n_1, k_1) cyclic code or shortened cyclic code with minimum distance d_1. C_1 is designed to correct t_1 or fewer errors and detect λ_1 $(\lambda_1 \geq t_1)$ or fewer errors, where $t_1 + \lambda_1 + 1 \leq d_1$. However, for channels having bandwidth limitation, a bandwidth efficient modulation code can be used in place of C_1 to achieve a certain coding gain while conserving bandwidth also.

The code C_2 is a maximum distance separable (MDS) (n_2, k_2) code over $GF(2^l)$ with minimum distance d_2. We note that for MDS codes, $d_2 = (n_2 - k_2 + 1)$. This code is chosen for correcting symbol errors and erasures. The outer code of the proposed cascaded scheme is obtained by interleaving this code to a depth m_1. Thus the outer code is a $(m_1 n_2, m_1 k_2)$ code over $GF(2^l)$.

There is yet another code $C_{\frac{1}{2}}$, which is used for parity retransmissions. This code is obtained by shortening C_1 by $(2k_1 - n_1)$ bits (code C_1 is chosen such that

$k_1 > (n_1 - k_1)$). The resultant $(2(n_1 - k_1), (n_1 - k_1))$ code has the same minimum distance as C_1, but will be used on sub-frames which are only $(n_1 - k_1)$-bits long. Since $C_{\frac{1}{2}}$ is obtained by shortening C_1, both C_1 and $C_{\frac{1}{2}}$ can be encoded and decoded using the same circuits. The code $C_{\frac{1}{2}}$ is invertible [Lin-Cos 83], hence $(n_1 - k_1)$ parity bits can be inverted to uniquely determine the $(n_1 - k_1)$ information bits. We assume that the code parameters satisfy the following conditions:

$$k_1 = m_1 l. \tag{4.1}$$

$$(n_1 - k_1) = \rho l. \tag{4.2}$$

$$m_1 = (m_2 - 1)\rho. \tag{4.3}$$

Therefore we get

$$\begin{aligned} n_1 &= m_2(n_1 - k_1), \\ &= m_2 \rho l. \end{aligned} \tag{4.4}$$

$$\tag{4.5}$$

Let $\bar{v}_i$ be a codeword in C_1. Since $n_1 = m_2(n_1 - k_1)$, we can divide $\bar{v}_i$ into m_2 sub-frames $\bar{v}_{i1}$, $\bar{v}_{i2}$, $\cdots \bar{v}_{im_2}$, each of them consisting of $(n_1 - k_1)$ bits. Let $Q(\bar{v}_{ij})$ be the $(n_1 - k_1)$ parity bits formed by $C_{\frac{1}{2}}$ encoder based on $\bar{v}_{ij}$. Then for $1 \leq j \leq m_2$, $(\bar{v}_{ij}, Q(\bar{v}_{ij}))$ is a codeword in $C_{\frac{1}{2}}$. Let

$$Q(\bar{v}_i) = (Q(\bar{v}_{i1}),\ Q(\bar{v}_{i2}),\ \cdots Q(\bar{v}_{im_2})). \tag{4.6}$$

Then it can be easily shown that $Q(\bar{v}_i)$ is also a codeword in C_1. This property is very useful in the decoding process. Henceforth, $Q(\bar{v}_i)$ will be referred to as the parity frame of the frame $\bar{v}_i$.

4.1.1 The Encoder

An incoming message consists of $k_1 \times k_2$ information bits. This can be viewed as an array of k_2 rows and k_1 columns (see Fig. 4.1). Each row of k_1 bits can also be viewed as m_1 bytes of l bits each, each of which is a symbol in $GF(2^l)$. Each of the $k_2 - byte$ column is first encoded into an $n_2 - byte$ codeword by C_2 encoder resulting in a $n_2 \times k_1$ array. Then the C_1 encoder encodes each row into an n_1 bit codeword. This gives us the $n_1 n_2$ bit codeword in the cascaded code C corresponding to the $k_1 k_2 - bit$ input message. This array of n_2 rows and n_1 columns is called a data block. It is transmitted by rows. The first k_1 $(m_1 l)$ columns of this array can be viewed as m_1 $l - bit$ byte columns. Each of these byte columns is a codeword in C_2 and is called a data-section. Each of the n_2 rows of the code array is a codeword in C_1 and is called a frame.

There is another part of the encoder. This is the $C_{\frac{1}{2}}$ encoder used for parity retransmission. Let $\bar{v}_i$ be the i^{th} frame in the code array. Then $\bar{v}_i$ consists of m_2 subframes $\bar{v}_{i1}$, $\bar{v}_{i2}$, $\cdots \bar{v}_{im_2}$. The $C_{\frac{1}{2}}$ encoder forms m_2 corresponding parity subframes $Q(\bar{v}_{i1})$, $Q(\bar{v}_{i2})$, $\cdots Q(\bar{v}_{im_2})$. We again note that if C_1 is cyclic then

$$Q(\bar{v}_i) = (Q(\bar{v}_{i1}),\ Q(\bar{v}_{i2}),\ \cdots Q(\bar{v}_{im_2})),\quad 1 \leq i \leq n_2. \tag{4.7}$$

is a codeword in C_1. These n_2 frames are stored in the buffer at the encoder for possible retransmission. The array with these frames as rows constitute the parity block. Each row of a parity block, thus obtained, is called a parity frame and each of the first m_1 $l - bit$ columns is called a parity section. The parity block is also transmitted frame by frame.

In both the data block as well as the parity block, the set of m_1 sections can be further partitioned into $(m_2 - 1)$ parts, each of which consists of ρ $l - bit$ sections (columns). These are called super-sections (see Fig. 4.1). Each row in a super-section

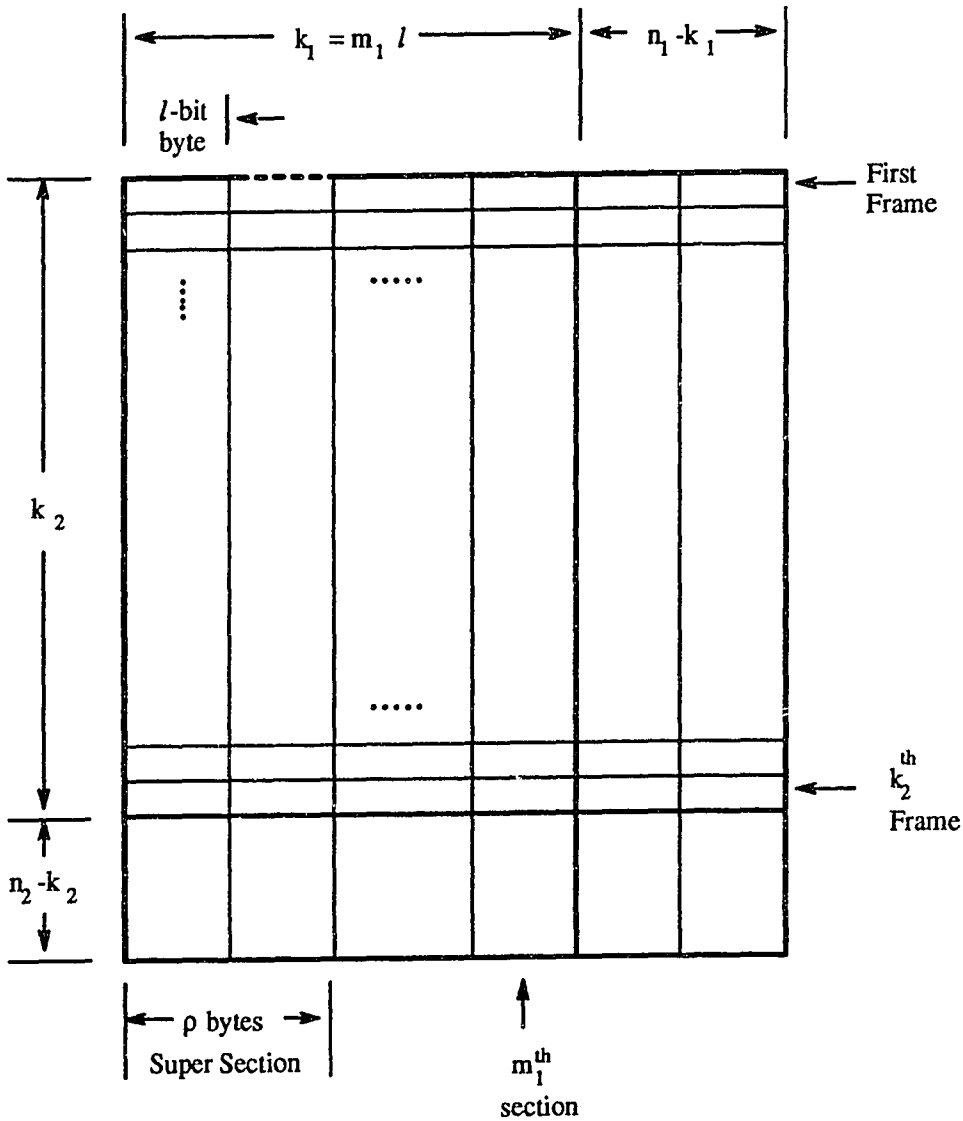

Figure 4.1: Block Format

consists of $(n_1 - k_1)$ bits and is a data or parity sub-frame in $C_{\frac{1}{2}}$ code depending upon whether the block is a data or parity block.

4.1.2 Decoding of a Data Block

The decoding of a data block is similar the one described in [Kasami 88]. It consists of inner code decoding followed by outer code decoding. The inner decoding consists of one of the following three operations: *error correction, erasure* and *leave-it-alone* (LIA). When a data block is received, the syndrome of each frame in a block is computed based on C_1. If the syndrome corresponds to an error pattern of t_1 or fewer errors, error correction is performed on the received frame. A successfully decoded frame will be referred to simply as a *decoded frame.* We note that the *decoded frame* is error-free if the number of errors in the received frame is t_1 or less. If the number of errors is greater than λ_1, the error pattern may result in a syndrome which corresponds to a correctable error pattern of t_1 or fewer errors. In this case, the decoding will be successful but the decoded frame will contain undetected errors. If an uncorrectable error pattern is detected in a received frame, the inner decoder performs one of the following operations,

- *Erasure Operation*: The erroneous frame is regarded as being erased. Such a frame is called an *erased frame.* This frame is not physically erased but is marked as erased for subsequent outer decoding. The frame is still stored in the buffer for later use.

- *Leave-it-alone (LIA) Operation*: The erroneous frame is left alone in the buffer. Such a frame will be referred to as *LIA-decoded* and is marked as such. The LIA-decoded frame may contain error-free symbols.

Whether the decoder performs erasure or LIA operations depends upon the number of errors in a received frame [Kasami 88]. Both of these two operations are done to facilitate a successful decoding of a block by the outer decoder. Our goal is to adopt a decoding policy that will make optimal use of the outer code minimum-distance d_2. The correction of a symbol erasure requires one unit of outer code distance whereas a symbol error-correction requires two units of minimum-distance. When a frame is erased, m_1 symbol erasures are created. We require m_1 units of minimum-distance to correct these erasures. We emphasize here that all of these erasures are on different codewords (sections) of the outer code. Not all of these symbols were erroneous to begin with. However, the erasure operation forces the outer decoder to expend one unit of minimum distance on each section due to this operation alone. Clearly, this operation is not optimal if the number of errors in the frame is less than or equal to $\lfloor m_1/2 \rfloor$. A frame that is LIA decoded has at least one and at most m_1 symbol errors. Therefore, 2 to $2m_1$ units of minimum distance of C_2 will be required to correct them. Therefore, the LIA operation is optimal if the number of symbol errors in a frame is less than $\lfloor m_1/2 \rfloor$. Based on the above discussion, it would be sensible to perform the LIA operation whenever the number of symbol errors is less than or equal to $\lfloor m_1/2 \rfloor$ and an erasure operation whenever the number of symbol errors is greater than $\lfloor m_1/2 \rfloor$.

However, we would like to give a probabilistic interpretation to this idea, since the C_1 decoder cannot determine the number of errors in an uncorrectable error pattern (much less find the number of symbol errors). Hence we will choose the following strategy: if the probability that an erroneous frame contains less than or equal to $\lfloor m_1/2 \rfloor$ symbol errors is greater than the probability that an erroneous frame contains more than $\lfloor m_1/2 \rfloor$ symbol errors, then LIA operation is the right choice or else we choose erasure operation. If the channel is stationary and we have

an estimate of channel BER, we can choose our decoding policy apriori. However, if the channel is non-stationary, then we need channel measurement information to update our decoding policy from time to time.

There is another decoding policy that we could adopt which does not depend on channel measurement. We could configure our decoder so that it performs only the error correction and erasure operations. Such a decoding will be called *erasure-only* inner decoding. Alternately, the decoder may perform error correction and LIA operations only. Such a decoding will be called *LIA-only* inner decoding.

The outer decoding begins on a received block as soon as all the frames are decoded. The C_2 decoder corrects symbol errors and erasures on each section/column of the code array. Let i and h be the number of erased and LIA-decoded frames respectively. The receiver stops the decoding process and requests a retransmission for the data block if the number i is greater than a certain predetermined erasure threshold T_{es}, where $T_{es} \leq (d_2 - 1)$. For a given value of i, the decoder then compares h to yet another threshold $T_{el}(i)$. Decoding is stopped and a retransmission requested if h is greater than $T_{el}(i)$, where $T_{el}(i) \leq \lfloor (d_2 - i - 1)/2 \rfloor$. The outer decoder starts the error correction operation on each section, if none of the two thresholds are exceeded. Each section has i erased symbols and some erroneous symbols. For a given i, the C_2 decoder has an error-correction threshold $t_2(i)$, where $t_2(i) \leq \lfloor (d_2 - i - 1)/2 \rfloor$. The erroneous symbols in a section can be from either a decoded frame or a LIA-decoded frame. A section can be decoded successfully if it has $t_2(i)$ or fewer errors for a given i erasures, otherwise a section is said to contain an uncorrectable error pattern. If none of the sections contain an uncorrectable error pattern, the block is decoded successfully and the $k_1 k_2$ bits of the original message are delivered to the user. If one or more sections contain uncorrectable error patterns, the decoding process is stopped

and a retransmission is requested. The erroneously received data block (prior to any decoding) is stored in the buffer for possible future use.

4.1.3 Decoding of a Parity Block

Let $\bar{\mathbf{v}}$ be the transmitted data block and $\tilde{\mathbf{v}}$ be the received block. Upon request of a retransmission, a parity-block $P(\bar{\mathbf{v}})$ based on $\bar{\mathbf{v}}$ is transmitted. Each row in a parity-block is a parity frame. We recall that the transmitter had already constructed parity frames corresponding to each of the original data frames.

Let $\tilde{P}(\bar{\mathbf{v}})$ be the received parity block. The decoding procedure for $\tilde{P}(\bar{\mathbf{v}})$ is similar to that of a data block $\bar{\mathbf{v}}$. Again the decoding consists of inner decoding followed by outer decoding. Each parity frame is a codeword in C_1. The only difference is that the decoder will not be performing LIA operation here. If an uncorrectable error pattern is encountered, the C_1 decoder performs an erasure operation, otherwise error correction is performed on the frame. In other words, the C_1 decoding for the parity block is an erasure-only decoding. After a frame is successfully decoded, each of it's sub-frames is inverted to get the original data sub-frame back. After decoding n_2 such frames in this manner, the outer decoding is started. The outer decoding of a parity block is exactly identical to that of a data block. If the decoding is successful, the message is delivered to the user and the erroneous data block, stored in the buffer, is discarded.

However, if the outer decoding fails to decode $\tilde{P}(\bar{\mathbf{v}})$, the erroneously received (unaltered) parity block and the stored data block $\tilde{\mathbf{v}}$ are combined for error correction using $C_{\frac{1}{2}}$ decoding. For $1 \leq i \leq n_2$ and $1 \leq j \leq m_2$, let $\tilde{v}_{ij}$ and $\tilde{Q}(\bar{v}_{ij})$ be the j^{th} sub-frames of the i^{th} frame in the received data and parity blocks respectively. Clearly, $\tilde{v}_{ij}$ and $\tilde{Q}(\bar{v}_{ij})$ form a data- parity pair in the $C_{\frac{1}{2}}$ code. The $C_{\frac{1}{2}}$ decoder decodes this pair and obtains an estimate $\bar{v}_{ij}^*$ for $\bar{v}_{ij}$. This decoding process is similar to the

C_1 decoding. Here also, the decoding consists of error correction, erasure and LIA decoding. If the frame is LIA decoded, then $\bar{v}_{ij}{}^* = \tilde{v}_{ij}$. An erasure decoding causes the sub-frame to be marked as erased. We again note that the code $C_{\frac{1}{2}}$ has the same error-correction capability as C_1. Hence it is very powerful when used on a sub-frame,

After $n_2 \times m_2$ such decodings, the receiver has an estimate of the original code array,

$$\bar{\mathbf{v}}^* = [\bar{v}_{ij}{}^*], \quad 1 \leq i \leq n_2,\ 1 \leq j \leq m_2. \tag{4.8}$$

All the rows of this array where every sub-frame decoding was successful, should be a codeword in C_1. The decoder then performs further error-detection on these successfully decoded rows to identify any erroneous frames. For this error-detection process, C_1 is used as a pure error-detection code. Upon encountering an erroneous frame, the decoder erases the frame. This process improves the reliability. Next, the outer decoding is carried out in this block. The C_2 decoding process is identical to the one described for the case of a data block. If the C_2 decoding is successful, the receiver removes the parity bits from the block and delivers the information bits to the user. If the outer decoding is not successful, the receiver stores the parity block, discards the erroneous data block and requests a retransmission. The second retransmission is the data block $\bar{\mathbf{v}}$ itself. Let $\tilde{\mathbf{v}}'$ be the received block. The decoding of $\tilde{\mathbf{v}}'$ is same as that of the first data block. If the combined C_1 and C_2 decoding is successful, then the message is delivered to the user and the stored parity block discarded. If the combined C_1 -C_2 decoding is unsuccessful, then $\tilde{\mathbf{v}}'$ and stored parity block $\tilde{P}(\bar{\mathbf{v}})$ are combined for $C_{\frac{1}{2}}$ -C_2 decoding. If this decoding is unsuccessful, the receiver discards $\tilde{P}(\bar{\mathbf{v}})$ and stores $\tilde{\mathbf{v}}'$ and requests a retransmission. The next retransmission is a parity block $P(\bar{\mathbf{v}})$. The retransmissions continue in this manner until the message is recovered by the receiver.

The proposed error-control scheme has the advantage that the extra parity bits are transmitted only when they are needed. The parity block has the same number of bits as the data block and carries the same information, so there is no bandwidth expansion. If the code $C_{\frac{1}{2}}$ is powerful, the probability of recovering the data after $C_{\frac{1}{2}}$ -C_2 decoding can be made very close to 1. When the channel is not very noisy, the cascaded code C alone is sufficient for data recovery. So only one copy of the message is needed at the receiver. If R_1 and R_2 are the rates of C_1 and C_2 respectively, then the system throughput is R_1R_2 in this case. When the channel becomes noisy, a second copy of the message (parity block) enables the recovery of the message with high probability. However, this reduces the throughput to $R_1R_2/2$. Because of the powerful coding used here, this scheme provides the maximum possible throughout (R_1R_2) most of the time.

Since $C_{\frac{1}{2}}$ is a shortened version of C_1, the C_1 decoder can be used for $C_{\frac{1}{2}}$ decoding. Hence, the receivers require only the C_1 and the C_2 decoders.

4.2 A Selective Repeat ARQ Scheme using Cascaded Coding

In this section, we briefly describe a selective repeat ARQ system for a homogeneous broadcast channel using the cascaded coding scheme proposed above. The scheme we propose is similar to the one in chapter 3. The protocol used here is same as the one described in chapter 2.

A message is said to be in $state-0$ if it has never been transmitted before. The message is said to have reached $state-j$ $(j \geq 1)$, if a request for it's j^{th} retransmission is received by the transmitter. In this ARQ scheme, multiple copies of the message (where data and parity blocks are arranged alternately) are sent out at every state of the message. The number of copies of a message at any given state is

chosen to optimize the throughput, using dynamic programming optimization. This optimization process takes into account the state of a message as well as the number of receivers that are yet to acknowledge the message. Each receiver is assumed to carry out a *single memory* decoding of the received blocks, i.e., the receivers store only one previous block for combined decoding purposes. Each receiver has a buffer of βN, where N is the number of blocks that can be transmitted in one round-trip propagation delay and β is a positive real number.

In section 4.4, we have obtained the throughput efficiency of such a scheme based on the assumption that there is always a message waiting to be transmitted and that the transfer of messages from receivers to users is instantaneous.

4.3 Performance Analysis

In this section, we analyze the performance of the cascaded coding scheme proposed in section 4.1. We obtain upper bounds on the probabilities of error and decoding failures associated with the C_1, C_2 and $C_{\frac{1}{2}}$ decodings. We assume that the channel is a binary symmetric channel with bit-error-rate (BER) ϵ.

4.3.1 First Transmission

Let P_{er} and P_{es} denote the probabilities of a decoding error and decoding failure for a data block by the combined $C_1 - C_2$ decodings. Also, let $\tilde{P}_{er}(u)$ be the probability of decoding error of the u^{th} section. Upper bounds on P_{er}, $\tilde{P}_{er}(u)$ and $P_{er} + P_{es}$ have been obtained in [Kasami 88]. Here, we obtain an upper bound on P_{es}.

In order for the combined $C_1 - C_2$ decoding to have an erroneous decoding, at least one of the data sections has to be incorrectly decoded. In particular,

$$P_{er} \geq \min_{u} \tilde{P}_{er}(u). \tag{4.9}$$

Therefore,

$$P_{es} \leq \overline{P_{er} + P_{es}} - \min_{u} \tilde{P}_{er}(u) \stackrel{\text{def}}{=} \overline{P_{es}}, \tag{4.10}$$

where $\overline{P_{er} + P_{es}}$ represents the bound obtained in [Kasami 88].

4.3.2 First Retransmission

When there is a decoding failure at the receiver for the first received data block, a retransmission is requested. The decoding of the first retransmission (parity block) is similar to that of the data block. As mentioned earlier, the C_1 decoder performs only the *erasure only* decoding. Therefore, the probabilities of error and decoding failure are same as that for the data block. However, if the receiver fails to recover the message by combined C_1 - C_2 decoding, then this parity block is combined with the message block that is stored in the buffer for $C_{\frac{1}{2}}$ decoding.

Let $\tilde{\mathbf{v}}$ and $\tilde{P}(\bar{\mathbf{v}})$ be a received block and it's corresponding received parity block respectively. Each of the n_2 frames in these blocks have m_2 sub-frames. For $1 \leq i \leq n_2$ and $1 \leq j \leq m_2$, let $\tilde{v}_{ij}$ and $\tilde{Q}(\bar{v}_{ij})$ be the j^{th} sub-frame of the i^{th} frame and it's corresponding parity frame respectively. Just as in the case of C_1 decoding, the decoder performs either the error-correction, the erasure or the leave-it-alone operation on each sub-frame pair $(\tilde{v}_{ij}, \tilde{Q}(\bar{v}_{ij}))$. Let e and e' be the number of errors in $\tilde{v}_{ij}$ and $\tilde{Q}(\bar{v}_{ij})$ respectively. A successful decoding results if $e + e' \leq t_1$. For every successful decoding, the $C_{\frac{1}{2}}$ decoder produces an estimate $\tilde{v}_{ij}^*$. If every sub-frame pair is successfully decoded, we get an estimate $\tilde{v}_i^* = (\tilde{v}_{i1}^*, \tilde{v}_{i2}^*, \cdots \tilde{v}_{im_2}^*)$ which should be a codeword in C_1. The C_1 decoder then performs error detection on this estimate using it's entire minimum distance for error detection purposes. This estimate is erased if errors are detected by the decoder.

Let $P_c^{\frac{1}{2}}(f)$ and $P_{ic}^{\frac{1}{2}}(f)$ be the probabilities of correct and incorrect decoding of a sub-frame by the $C_{\frac{1}{2}}$ decoder, then

$$\begin{aligned} P_c^{\frac{1}{2}}(f) &= \sum_{i=0}^{t_1} \binom{r_1}{i} \epsilon^i (1-\epsilon)^{r_1-i} \sum_{j=0}^{t_1-i} \binom{r_1}{j} \epsilon^j (1-\epsilon)^{r_1-j}, \quad i+j \leq t_1, \\ &= \sum_{i=0}^{t_1} \binom{2r_1}{i} \epsilon^i \, (1-\epsilon)^{2r_1-i}, \end{aligned} \tag{4.11}$$

where $r_1 = (n_1 - k_1)$.

Let $P_c^{\frac{1}{2}}$ and $P_{ic}^{\frac{1}{2}}$ be the probabilities of correct and incorrect decodings of a frame after $C_{\frac{1}{2}}$ decoding. Then, $P_c^{\frac{1}{2}}$ is given by

$$P_c^{\frac{1}{2}} = [P_c^{\frac{1}{2}}(f)]^{m_2-1}. \tag{4.12}$$

Let $A_i^{\frac{1}{2}}$ and $B_i^{\frac{1}{2}}$ be the number of codewords of weight i in the code $C_{\frac{1}{2}}$ and it's dual code $C_{\frac{1}{2}}{}^{\perp}$ respectively. Let $W_{j,s}^{(i)}(2r_1)$ be the number of binary $2r_1$-tuples of weight j which are at a Hamming distance s from a given binary $2r_1$-tuple of weight i. The generating function of $W_{j,s}^{(i)}(2r_1)$ has been obtained by MacWilliams [Mac 63] and is given by

$$\sum_{j=0}^{2r_1} \sum_{s=0}^{2r_1} W_{j,s}^{(i)}(2r_1) X^j Y^s = (1+XY)^{2r_1-i}(X+Y)^i. \tag{4.13}$$

Whenever the $C_{\frac{1}{2}}$ decoder encounters an error pattern of $(\lambda_1 + 1)$ or more errors whose syndrome corresponds to an error pattern of t_1 or fewer errors, we have an incorrect decoding. Using MacWilliams results [Mac 63], we can write

$$\begin{aligned} P_c^{\frac{1}{2}}(f) + P_{ic}^{\frac{1}{2}}(f) &= \sum_{i=0}^{2r_1} A_i^{\frac{1}{2}} \sum_{j=0}^{2r_1} \sum_{s=0}^{t_1} W_{j,s}^{(i)}(2r_1) \, \epsilon^i (1-\epsilon)^{2r_1-i}, \\ &= 2^{-r_1} \sum_{i=0}^{2r_1} B_i^{\frac{1}{2}} \, (1-2\epsilon)^i \, P_{t_1}(i-1, 2r_1-1), \end{aligned} \tag{4.14}$$

where $P_{t_1}(\cdot,\cdot)$ is a Krawtchouk polynomial [Mac-Slo 77, page 129], whose generating function is given by

$$\sum_{k=0}^{2r_1} P_k(i, 2r_1) Y^k = (1+Y)^{2r_1-i}(1-Y)^i. \tag{4.15}$$

We note that $P_c^{\frac{1}{2}}(f)+P_{ic}^{\frac{1}{2}}(f)$ gives the probability of successful decoding (correct or incorrect) of a sub-frame by the $C_{\frac{1}{2}}$ decoder. Therefore, the probability of successful decoding of a frame by the $C_{\frac{1}{2}}$ decoder is given by

$$P_c^{\frac{1}{2}} + P_{ic}^{\frac{1}{2}} = \prod_{f=1}^{m_2-1} \{P_c^{\frac{1}{2}}(f) + P_{ic}^{\frac{1}{2}}(f)\}. \tag{4.16}$$

The last four equations can be used to compute $P_c^{\frac{1}{2}} + P_{ic}^{\frac{1}{2}}$ if $A_i^{\frac{1}{2}}$ or $B_i^{\frac{1}{2}}$ is known or if r_1 is small enough so that $A_i^{\frac{1}{2}}$ and $B_i^{\frac{1}{2}}$ can be computed by generating all the codewords in $C_{\frac{1}{2}}$ or $C_{\frac{1}{2}}^{\perp}$. Note that $C_{\frac{1}{2}}$ is a rate-$\frac{1}{2}$ code, hence the amount of computation is the same whether $C_{\frac{1}{2}}$ or $C_{\frac{1}{2}}^{\perp}$ is used to obtain the weight distribution information.

Let $P_{es}^{\frac{1}{2}}(f)$ be the probability that a sub-frame is erased and let $P_{el}^{\frac{1}{2}}(f)$ be the probability that an LIA-operation is performed on a sub-frame. Then for a particular sub-frame

$$P_c^{\frac{1}{2}}(f) + P_{ic}^{\frac{1}{2}}(f) + P_{es}^{\frac{1}{2}}(f) + P_{el}^{\frac{1}{2}}(f) = 1. \tag{4.17}$$

Note that $P_{es}^{\frac{1}{2}}(f) + P_{el}^{\frac{1}{2}}(f)$ represent the probability of a decoding failure of a sub-frame. For LIA-only $C_{\frac{1}{2}}$ decoding $P_{es}^{\frac{1}{2}}(f) = 0$. In this case, the decoder performs a LIA operation whenever an uncorrectable error pattern is detected. Similarly, for erasure-only $C_{\frac{1}{2}}$ decoding; an erasure operation is performed whenever an uncorrectable error pattern is detected. In this case, $P_{el}^{\frac{1}{2}}(f) = 0$. If $P_{es}^{\frac{1}{2}}(f)$ (or $P_{el}^{\frac{1}{2}}(f)$) is known, then $P_{el}^{\frac{1}{2}}(f)$ (or $P_{es}^{\frac{1}{2}}(f)$) can be computed using equations (4.11)- (4.18).

In what follows, we consider the erasure-only and the LIA-only $C_{\frac{1}{2}}$ decoding and obtain probabilities of decoding failure for these two cases in terms of the detailed weight distribution of $C_{\frac{1}{2}}$.

4.3.3 Byte Error Probabilities in $C_{\frac{1}{2}}$ Decoding

There is yet another way of obtaining the probabilities of correct and incorrect decodings by the $C_{\frac{1}{2}}$ decoder, $P_c^{\frac{1}{2}}(f)$ and $P_{ic}^{\frac{1}{2}}(f)$, in terms of the detailed weight distribution

of the code $C_{\frac{1}{2}}$. Each sub-frame can be viewed as a string of ρ $l-bit$ bytes. Let $P_{e,w}^{\frac{1}{2}}$, $0 \leq w \leq \rho$ be the joint probability that a sub-frame is successfully decoded and the number of symbol (byte) errors in the decoded sub-frame is w. Then, we can write

$$P_c^{\frac{1}{2}}(f) = P_{e,0}^{\frac{1}{2}} , \tag{4.18}$$

and

$$P_{ic}^{\frac{1}{2}}(f) = \sum_{w=1}^{\rho} P_{e,w}^{\frac{1}{2}} . \tag{4.19}$$

The probability $P_{e,w}^{\frac{1}{2}}$ will be used to obtain results for outer code decoding. We obtain $P_{e,w}^{\frac{1}{2}}$ as follows.

Let $\bar{v}_{ij}$ and $Q(\bar{v}_{ij})$ be a data-parity sub-frame pair. For $1 \leq b \leq \rho$, let i_b be the binary weight (Hamming) of the b^{th} $l-$ bit byte of $\bar{v}_{ij}$. Let $i_{\rho+1}$ be the weight of $Q(\bar{v}_{ij})$. Then the $(\rho+1)-$tuple $(i_1, i_2, \cdots i_{\rho+1})$ is called the weight structure of $(\bar{v}_{ij}, Q(\bar{v}_{ij}))$ [Kasami 88].

Suppose that $\bar{v} = (\bar{v}_{ij}, Q(\bar{v}_{ij}))$ is the transmitted sub-frame pair and $\bar{r} = (\tilde{v}_{ij}, \tilde{Q}(\bar{v}_{ij}))$ is the received pair. Let the error pattern $\bar{e}$ in $\bar{r}$ have the weight-structure $(j_1, j_2, \cdots j_{\rho+1})$. The probability of occurrence of $\bar{e}$ is

$$\begin{aligned} P(\bar{e}) &= \prod_{b=1}^{\rho} \epsilon^{j_b}(1-\epsilon)^{l-j_b}\epsilon^{j_{\rho+1}}(1-\epsilon)^{r_1-j_{\rho+1}}, \\ &= (1-\epsilon)^{2r_1}\prod_{b=1}^{\rho+1}(\epsilon/(1-\epsilon))^{j_b}. \end{aligned} \tag{4.20}$$

Suppose that $\bar{u} \in C_{\frac{1}{2}}$ is a codeword at a distance t_1 or less from $\bar{e}$. Since the minimum distance of $C_{\frac{1}{2}}$ is assumed to be greater than $2t_1$, $\bar{u} \in C_{\frac{1}{2}}$ is unique, if it exists. The $C_{\frac{1}{2}}$ decoder then assumes that $\bar{v} + \bar{u}$ is the transmitted codeword and $\bar{e} + \bar{u}$ is the error pattern. If $\bar{u}$ is a non-zero codeword then we have an incorrect decoding and the first r_1 bits of $\bar{u}$ represent the errors introduced by $C_{\frac{1}{2}}$ decoding process. If there is no such codeword $\bar{u} \in C_{\frac{1}{2}}$ then there is a decoding failure and

the decoder performs either an erasure or LIA operation on the sub-frame. For a codeword $\bar{u} \in C_{\frac{1}{2}}$ with weight structure $(i_1, i_2, \cdots i_{\rho+1})$, the number of error patterns $\bar{e}$ with weight structure $(j_1, j_2, \cdots j_{\rho+1})$, such that the weight structure of $(\bar{u} + \bar{e})$ is $(s_1, s_2, \cdots s_{\rho+1})$ is given by

$$\left[\prod_{h=1}^{\rho} W_{j_h, s_h}^{(i_h)}(l)\right] W_{j_{\rho+1}, s_{\rho+1}}^{(i_{\rho+1})}(r_1). \tag{4.21}$$

Let $A_{i_1, i_2, \cdots i_{\rho+1}}^{(\frac{1}{2})}$ be the number of codewords in $C_{\frac{1}{2}}$ with weight structure $(i_1, i_2, \cdots i_{\rho+1})$. For $0 \leq w \leq \rho$, let

$$\begin{aligned} I_w = \; \{ \; & (i_1, i_2, \cdots i_{\rho+1}) : 0 \leq i_b \leq l \text{ for } 1 \leq b \leq \rho, \; 0 \leq i_{\rho+1} \leq r_1 \\ & \text{and exactly } w \text{ components of } (i_1, i_2, \cdots i_\rho) \text{ are non-zero}\} \end{aligned} \tag{4.22}$$

Then $P_{e,w}^{\frac{1}{2}}$ is given by

$$\begin{aligned} P_{e,w}^{\frac{1}{2}} = & \sum_{(i_1, i_2, \cdots i_{\rho+1}) \in I_w} A_{i_1, i_2, \cdots i_{\rho+1}}^{\frac{1}{2}} \sum_{j_1=0}^{l} \sum_{j2=0}^{l} \cdots \sum_{j_{\rho+1}}^{r_1} \\ & \cdot \sum_{(s_1, s_2, \cdots s_{\rho+1}) \in S_{t_1}} \left[\prod_{b=1}^{\rho} W_{j_b, s_b}^{(i_b)}(l)\right] W_{j_{\rho+1}, s_{\rho+1}}^{(i_{\rho+1})}(r_1)(1-\epsilon)^{2r_1} \\ & \cdot \left[\prod_{b=1}^{\rho+1} \left(\frac{\epsilon}{(1-\epsilon)}\right)^{j_b}\right] \end{aligned} \tag{4.23}$$

where

$$\begin{aligned} S_{t_1} = \; & \{(s_1, s_2, \cdots s_{\rho+1}) : 0 \leq s_b \leq l \text{ for, } 1 \leq b \leq \rho, \; 0 \leq s_{\rho+1} \leq r_1 \\ & \text{and } \sum_{b=1}^{\rho+1} s_b \leq t_1\} \end{aligned} \tag{4.24}$$

The expressions above can be used to compute $P_{e,w}^{\frac{1}{2}}$ if r_1, the dimension of $C_{\frac{1}{2}}$, is small enough so that it is possible to compute the detailed weight distribution $\{A_{i_1, i_2, \cdots i_{\rho+1}}^{\frac{1}{2}}\}$ by generating all codewords in $C_{\frac{1}{2}}$. We again note that $C_{\frac{1}{2}}$ is a

rate-$\frac{1}{2}$ code and hence the amount of computation involved in obtaining the weight distribution information from it's dual code $C_{\frac{1}{2}}{}^{\perp}$ will be the same. For the case of erasure only decoding, the value of $P_{es}^{\frac{1}{2}}(f)$ can now be obtained from (4.11) to (4.17) or (4.18)-(4.24) and (4.18).

4.3.4 Byte Error Probabilities in LIA Decoding

In this section, we will find an expression for the probability of a byte error in a LIA-decoded sub-frame. Let $P_{el,w}^{\frac{1}{2}}$ be the joint probability that a sub-frame is LIA-decoded (left alone) and the number of byte errors in the sub-frame is w. Then

$$P_{el}^{\frac{1}{2}}(f) = \sum_{w=1}^{\rho} P_{el,w}^{\frac{1}{2}} \tag{4.25}$$

Let us consider the case of LIA-only $C_{\frac{1}{2}}$ decoding. Let

$$\begin{aligned} J_w \;=\; & \{(j_1, j_2, \cdots j_{\rho+1}) : 0 \le j_b \le l \text{ for } 1 \le b \le \rho,\ 0 \le j_{\rho+1} \le r_1 \\ & \text{and exactly } w \text{ components of } (j_1, j_2, \cdots j_\rho) \text{ are non-zero}\} \end{aligned} \tag{4.26}$$

Then we can write

$$\begin{aligned} P_{el,w}^{\frac{1}{2}} \;=\; & \binom{\rho}{w}[1-(1-\epsilon)^l]^w(1-\epsilon)^{\rho l - wl} - \sum_{i_1=0}^{l}\sum_{i_2=0}^{l}\cdots\sum_{i_{\rho+1}=0}^{r_1} A_{i_1, i_2, \cdots i_{\rho+1}}^{\frac{1}{2}} \\ & \cdot \sum_{J_w}\sum_{S_{t_1}}\left[\prod_{b=1}^{\rho} W_{j_b, s_b}^{(i_b)}(l)\right] W_{j_{\rho+1}, s_{\rho+1}}^{(i_{\rho+1})}(r_1)(1-\epsilon)^{2r_1}\left[\prod_{b=1}^{\rho+1}\left(\frac{\epsilon}{1-\epsilon}\right)^{j_b}\right] \end{aligned} \tag{4.27}$$

where S_{t_1} is same as defined earlier. The first term in the above equation is the probability that there are exactly w erroneous bytes in the ρ bytes of a received sub-frame, and the second term is the probability that the syndrome of these byte errors correspond to an error pattern of t_1 or fewer errors. The value of $P_{el}^{\frac{1}{2}}(f)$ can now be obtained using (4.25)-(4.27).

The question of deciding between erasure-only and LIA-only $C_{\frac{1}{2}}$ decoding can be settled in a manner similar to that discussed in section 4.1.2, for the C_1 decoding.

Every sub-frame erasure causes ρ bytes of erasure requiring ρ units of outer code distance to correct them. A LIA-decoded sub-frame may require anywhere from 2 to 2ρ units of distance. Hence, if the probability that an erroneous sub-frame contains greater than $\lfloor \rho/2 \rfloor$ byte errors is smaller than the probability that it contains less than or equal to $\lfloor \rho/2 \rfloor$ byte errors then LIA operation is the right choice, or else we choose erasure operation. In quantitative terms, LIA decoding is the sensible choice if

$$\sum_{w=\lfloor \rho/2 \rfloor+1}^{\rho} P_{el,w}^{\frac{1}{2}} < \sum_{w=1}^{\lfloor \rho/2 \rfloor} P_{el,w}^{\frac{1}{2}} \tag{4.28}$$

where $P_{el,w}^{\frac{1}{2}}$ is computed under the assumption that the $C_{\frac{1}{2}}$ decoding is LIA-only decoding.

4.3.5 Outer Code Decoding

After decoding each of the n_2 frames, the receiver begins the C_2 decoding on columns of the $C_{\frac{1}{2}}$ decoded array. Each of the m_1 symbol columns ($n_2 \times l$ $sub-array$) in this decoded array is a section. The set of ρ consecutive sections in the $C_{\frac{1}{2}}$ decoded array constitutes a super-section (see Fig. 4.1). Each section consists of symbols from sub-frames that are either decoded, erased or LIA-decoded. Each sub-frame erasure creates one symbol erasure in ρ sections. Hence, each section may consist of symbol erasures and errors. Since the outer code is interleaved to a degree m_1, the u^{th} section consists of u^{th} symbol of every frame. For $1 \leq u \leq \rho$, let $p_c^{\frac{1}{2}}(\{u\})$ be the probability that u^{th} symbol of a decoded sub-frame is error-free. An expression for $p_c^{\frac{1}{2}}(\{u\})$ is derived in the Appendix. Clearly, this expression will hold for every $u \ mod \ \rho$.

Let $p_e^{\frac{1}{2}}(u)$ be the probability that the u^{th} symbol of a decoded sub-frame is erroneous. If the code $C_{\frac{1}{2}}$ is quasi-cyclic for every $s-bit$ shift where s divides l, then

$p_e^{\frac{1}{2}}(u)$ is independent of u. We have

$$p_e^{\frac{1}{2}}(u) = P_c^{\frac{1}{2}}(f) + P_{ic}^{\frac{1}{2}}(f) - p_c^{\frac{1}{2}}(\{u\}) \tag{4.29}$$

This again is true for every $u \ mod \ \rho$.

Let $p_{el}^{\frac{1}{2}}(u)$ be the probability that the u^{th} symbol of a LIA-decoded sub-frame is erroneous. Then, $p_{el}^{\frac{1}{2}}(u)$ can be found using equation (4.28), as follows,

$$\begin{aligned} p_{el}^{\frac{1}{2}}(u) &= 1-(1-\epsilon)^l - \sum_{i_1=0}^{l}\sum_{i_2=0}^{l}\cdots\sum_{i_{\rho+1}}^{r_1} A_{i_1,i_2,\cdots i_{\rho+1}}^{\frac{1}{2}} \sum_{J(u)}\sum_{S_{t_1}} \\ &\quad \cdot \left[\prod_{b=1}^{\rho} W_{j_b,s_b}^{(i_b)}(l)\right] W_{j_{\rho+1},s_{\rho+1}}^{(i_{\rho+1})}(r_1)(1-\epsilon)^{2r_1}\left[\prod_{b=1}^{\rho+1}\left(\frac{\epsilon}{1-\epsilon}\right)^{j_b}\right] \end{aligned} \tag{4.30}$$

where

$$\begin{aligned} J(u) &= \{(j_1,j_2,\cdots j_{\rho+1}) : 1 \le j_b \le l \text{ for } 1 \le b \le \rho,\ j_u \ne 0, \\ &\quad \text{and } 1 \le j_{\rho+1} \le r_1\} \end{aligned} \tag{4.31}$$

Let $P_c^{(2)}(u)$, $P_{es}^{(2)}(u)$ and $P_{er}^{(2)}(u)$ denote the probabilities of a correct decoding, a decoding failure and an incorrect decoding for the u^{th} section, respectively, after first retransmission. Then

$$\begin{aligned} P_c^{(2)}(u) &= \sum_{i=0}^{T_{es}} \binom{n_2}{i} \left[P_{es}^{\frac{1}{2}}(f)\right]^i \sum_{h=0}^{T_{el}(i)} \binom{n_2-i}{h} \left[P_{el}^{\frac{1}{2}}(f)\right]^h \sum_{j=0}^{t_2(i)}\sum_{s=0}^{t_2(i)-j} \\ &\quad \cdot \binom{n_2-i-h}{s}\binom{h}{j-s}\left[1-P_{es}^{\frac{1}{2}}(f)-P_{el}^{\frac{1}{2}}(f)-p_e^{\frac{1}{2}}(u)\right]^{n_2-i-h-s} \\ &\quad \cdot \left[p_e^{\frac{1}{2}}(u)\right]^s \left[p_{el}^{\frac{1}{2}}(u)\right]^{j-s} \left[P_{el}^{\frac{1}{2}}(f)-p_{el}^{\frac{1}{2}}(u)\right]^{h-(j-s)} \end{aligned} \tag{4.32}$$

where $T_{es} \le (d_2-1)$, $T_{el}(i) \le \lfloor (d_2-i-1)/2 \rfloor$, $i \ge 0$ and $t_2(i) \le T_{el}(i)$, $i \ge 0$.

For erasure-only $C_{\frac{1}{2}}$ decoding, we have

$$P_c^{(2)}(u) = \sum_{i=0}^{T_{es}} \binom{n_2}{i} \left[P_{es}^{\frac{1}{2}}(f)\right]^i \sum_{j=0}^{t_2(i)} \binom{n_2-i}{j} \left[p_e^{\frac{1}{2}}(u)\right]^j$$

$$\cdot \left[1 - P_{es}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{n_2-i-j} \tag{4.33}$$

and for the case of LIA-only $C_{\frac{1}{2}}$ decoding, we have

$$P_c^{(2)}(u) = \sum_{h=0}^{T_{el}(0)} \binom{n_2}{h} \left[P_{el}^{\frac{1}{2}}(f)\right]^h \sum_{j=0}^{t_2(0)} \sum_{s=0}^{t_2(0)-j} \binom{n_2-h}{s} \binom{h}{j-s}$$

$$\cdot \left[p_e^{\frac{1}{2}}(u)\right]^s \left[1 - P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{n_2-h-s} \left[p_{el}^{\frac{1}{2}}(u)\right]^{j-s}$$

$$\cdot \left[P_{el}^{\frac{1}{2}}(f) - p_{el}^{\frac{1}{2}}(u)\right]^{h-(j-s)} \tag{4.34}$$

Let $P_{er\rho}^{(2)}$ and $P_{es\rho}^{(2)}$ be the probabilities of an incorrect decoding and a decoding failure of a super-section respectively, after first retransmission. Also, let $P_c^{(2)}$, $P_{er}^{(2)}$ and $P_{es}^{(2)}$ be the probabilities of a correct decoding, incorrect decoding and a decoding failure of a block respectively, after first retransmission. Then these probabilities can bounded as follows.

The probability $P_{er}^{(2)}(u)$ is given by

$$P_{er}^{(2)}(u) \leq \sum_{i=0}^{T_{es}} \binom{n_2}{i} \left[P_{es}^{\frac{1}{2}}(f)\right]^i \sum_{h=0}^{T_{el}(i)} \left[P_{el}^{\frac{1}{2}}(f)\right]^h \binom{n_2-i}{h} \sum_{j=d_2-i-t_2(i)}^{n_2-i}$$

$$\cdot \sum_{s=0}^{n_2-i-h} \binom{n_2-i-h}{s} \binom{h}{j-s} \left[p_e^{\frac{1}{2}}(u)\right]^s \left[p_{el}^{\frac{1}{2}}(u)\right]^{j-s}$$

$$\cdot \left[1 - P_{es}^{\frac{1}{2}}(f) - P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{n_2-i-h-s} \left[P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{h-(j-s)} \tag{4.35}$$

For the case of erasure only $C_{\frac{1}{2}}$ decoding, we get

$$P_{er}^{(2)}(u) \leq \sum_{i=0}^{T_{es}} \binom{n_2}{i} \left[P_{es}^{\frac{1}{2}}(f)\right]^i \sum_{j=d_2-i-t_2(i)}^{n_2-i} \binom{n_2-i}{j} \left[p_e^{\frac{1}{2}}(u)\right]^j$$

$$\cdot \left[1 - P_{es}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{n_2-i-j} \tag{4.36}$$

For the case of LIA-only $C_{\frac{1}{2}}$ decoding

$$P_{er}^{(2)}(u) \leq \sum_{h=0}^{T_{el}(0)} \binom{n_2}{h} \left[P_{el}^{\frac{1}{2}}(f)\right]^h \sum_{j=d_2-t_2(0)}^{n_2} \sum_{s=0}^{n_2-h} \binom{n_2-h}{s} \binom{h}{j-s}$$

$$\cdot \left[p_e^{\frac{1}{2}}(u)\right]^s \left[1 - P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{n_2-h-s} \left[p_{el}^{\frac{1}{2}}(u)\right]^{j-s}$$

$$\cdot \left[P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{h-(j-s)} \tag{4.37}$$

We then obtain

$$P_{er\rho}^{(2)} \leq \sum_{u=1}^{\rho} P_{er}^{(2)}(u) \tag{4.38}$$

and hence

$$P_{er}^{(2)} \leq 1 - (1 - P_{er\rho}^{(2)})^{m_2-1} \tag{4.39}$$

Now we obtain upper bounds on the sum $P_{er}^{(2)} + P_{es}^{(2)}$. The sum $P_{er}^{(2)}(u) + P_{es}^{(2)}(u)$ is upper bounded as

$$P_{er}^{(2)}(u) + P_{es}^{(2)}(u) \leq \sum_{i=0}^{T_{es}} \binom{n_2}{i} \left[P_{es}^{\frac{1}{2}}(f)\right]^i \left\{ \sum_{h=0}^{T_{el}(i)} \binom{n_2-i}{h} \left[P_{el}^{\frac{1}{2}}(f)\right]^h \right.$$

$$\cdot \left. \sum_{j=t_2(i)+1}^{n_2-i} \sum_{s=0}^{n_2-i-h} \binom{n_2-i-h}{s} \binom{h}{j-s} \right.$$

$$\cdot \left[1 - P_{es}^{\frac{1}{2}}(f) - P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{n_2-i-h-s}$$

$$\cdot \quad \left[p_e^{\frac{1}{2}}(u)\right]^s \left[p_{el}^{\frac{1}{2}}(u)\right]^{j-s} \left[P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{h-(j-s)}$$

$$+ \sum_{h=T_{el}(i)+1}^{n_2-i} \binom{n_2-i}{h} \left[P_{el}^{\frac{1}{2}}(f)\right]^h \left[P_c^{\frac{1}{2}}(f) + P_{ic}^{\frac{1}{2}}(f)\right]^{n_2-i-h} \}$$

$$+ \sum_{i=T_{es}+1}^{n_2} \binom{n_2}{i} \left[P_{es}^{\frac{1}{2}}(f)\right]^i \left[1 - P_{es}^{\frac{1}{2}}(f)\right]^{n_2-i} \tag{4.40}$$

For the case of erasure-only $C_{\frac{1}{2}}$ decoding, we have

$$P_{er}^{(2)}(u) + P_{es}^{(2)}(u) \leq \sum_{i=0}^{T_{es}} \binom{n_2}{i} \left[P_{es}^{\frac{1}{2}}(f)\right]^i \sum_{j=t_2(i)+1}^{n_2-i} \binom{n_2-i}{j} \left[p_e^{\frac{1}{2}}(u)\right]^j$$

$$\cdot \quad \left[1 - P_{es}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{n_2-i-j}$$

$$+ \sum_{i=T_{es}+1}^{n_2} \binom{n_2}{i} \left[P_{es}^{\frac{1}{2}}(f)\right]^i \left[1 - P_{es}^{\frac{1}{2}}(f)\right]^{n_2-i} \tag{4.41}$$

For the case of LIA-only $C_{\frac{1}{2}}$ decoding, we get

$$P_{er}^{(2)}(u) + P_{es}^{(2)}(u) \leq \sum_{h=0}^{T_{el}} \binom{n_2}{h} \left[P_{el}^{\frac{1}{2}}(f)\right]^h \sum_{j=t_2(0)+1}^{n_2} \sum_{s=0}^{n_2-h} \binom{n_2-h}{s} \binom{h}{j-s}$$

$$\cdot \quad \left[p_e^{\frac{1}{2}}(u)\right]^s \left[1 - P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{n_2-h-s}$$

$$\cdot \quad \left[p_{el}^{\frac{1}{2}}(u)\right]^{j-s} \left[P_{el}^{\frac{1}{2}}(f) - p_e^{\frac{1}{2}}(u)\right]^{h-(j-s)}$$

$$+ \sum_{h=T_{el}+1}^{n_2} \binom{n_2}{h} \left[P_{el}^{\frac{1}{2}}(f)\right]^h \left[P_c^{\frac{1}{2}}(f) + P_{ic}^{\frac{1}{2}}(f)\right]^{n_2-h} \tag{4.42}$$

We then have

$$P_{er\rho}^{(2)} + P_{es\rho}^{(2)} \leq \sum_{u=1}^{\rho} \{P_{er}^{(2)}(u) + P_{es}^{(2)}(u)\} \stackrel{\text{def}}{=} z \tag{4.43}$$

Therefore, the sum $P_{er}^{(2)} + P_{es}^{(2)}$ is upper bounded as

$$\begin{aligned} P_{er}^{(2)} + P_{es}^{(2)} &\leq \sum_{j=1}^{m_2-1} \binom{m_2-1}{j} z^j (1-z)^{m_2-1-j} \\ &\leq 1-(1-z)^{m_2-1} \stackrel{\text{def}}{=} \overline{P_{er}^{(2)} + P_{es}^{(2)}} \end{aligned} \tag{4.44}$$

The probability of correct decoding of a block $P_c^{(2)}$ can now be lower bounded as follows.

$$P_c^{(2)} \leq 1 - \overline{P_{er}^{(2)} + P_{es}^{(2)}} \tag{4.45}$$

For the case of erasure-only $C_{\frac{1}{2}}$ decoding, we can obtain a tighter bound on $P_{er}^{(2)}$ in a manner similar to the one in [Kasami 88]. For $1 \leq u \leq \rho$, let $p_e^{\frac{1}{2}}(u,\alpha)$ be the probability that u^{th} error symbol in a sub-frame that is successfully decoded by the $C_{\frac{1}{2}}$ decoder is $\alpha \in GF(2^l)$. Then $p_e^{\frac{1}{2}}(u,\alpha)$ can be obtained using the procedure similar to that in [Kasami 88]. We can write

$$\begin{aligned} P_{er}^{(2)}(u) \quad \leq & \sum_{i=0}^{T_{es}} \binom{n_2}{i} \sum_{w=d_2-i}^{n_2-i} \binom{n_2-i}{w} \sum_{h=0}^{\min\{t_2, n_2-i-w\}} \binom{n_2-i-w}{h} \\ & \cdot \sum_{j=w+h-t_2}^{w} \binom{w}{j} P^{\frac{1}{2}}(u,i,w,h,j) \end{aligned} \tag{4.46}$$

where

$$\begin{aligned} P^{\frac{1}{2}}(u,i,w,h,j) = \quad & \left[P_{es}^{\frac{1}{2}}(f)\right]^i \left[p_e^{\frac{1}{2}}(u)\right]^h \left[p_e^{\frac{1}{2}}(u,0)\right]^{n_2-i-w-h} \\ & \cdot \left[1-P_{es}^{\frac{1}{2}}(f)\right]^{w-j} \max_{\alpha \neq 0} \left[p_e^{\frac{1}{2}}(u,\alpha)\right]^j, \quad \alpha \in GF(2^l) \end{aligned} \tag{4.47}$$

Then we obtain

$$P_{er\rho}^{(2)} \leq \sum_{u=1}^{\rho} P_{er}^{(2)}(u) \tag{4.48}$$

Therefore,

$$P_{er}^{(2)} \leq 1 - (1 - P_{er\rho}^{(2)})^{m_2 - 1} \tag{4.49}$$

Next, we will obtain an upper bound on the probability of a decoding failure $P_{es}^{(2)}$ for the first retransmission.

It is easy to see that

$$P_{er}^{(2)} \geq \min_{\substack{u \\ 1 \leq u \leq \rho}} P_{er}^{(2)}(u) \stackrel{\text{def}}{=} \underline{P_{er}^{(2)}} \tag{4.50}$$

Then we can obtain

$$P_{es}^{(2)} \leq \overline{P_{er}^{(2)} + P_{es}^{(2)}} - \underline{P_{er}^{(2)}} \stackrel{\text{def}}{=} \overline{P_{es}^{(2)}} \tag{4.51}$$

where $\overline{P_{er}^{(2)} + P_{es}^{(2)}}$ represents the upper-bound on the sum $P_{er}^{(2)} + P_{es}^{(2)}$ obtained above.

4.4 Throughput Efficiency Results

In this section, we obtain the throughput efficiency results for the optimum Hybrid ARQ scheme, described in section 4.2. The analysis procedure is same as the one used in Chapter 3.

Let r_0 be the probability of a decoding failure of a message after the first copy of the message has been received. Let r_j $(j \geq 1)$ be the probability of a decoding failure of a message after the j^{th} retransmission has been processed, given that the message was not received successfully from the $(j-1)^{th}$ copy. Then

$$r_0 \leq \overline{P_{es}} \tag{4.52}$$

and

$$r_j \leq (\overline{P_{es}}\,\overline{P_{es}^{(2)}})^j \qquad j \geq 1 \tag{4.53}$$

The quantities $\overline{P_{es}}$ and $\overline{P_{es}^{(2)}}$ have been obtained in equations 4.10 and 4.53 respectively. Since the receiver uses only one previous copy of the message for combined

Scheme	*Inner Code* C_1	*Outer Code* C_2	*Parity Retransmission* $C_{1/2}$
1	None	R-S (255,223) code over $GF(2^8)$	None
2	(56,48) distance-4 shortened Hamming Code	R-S (255,223) code over $GF(2^8)$	None
3	(56,48) distance-4 shortened Hamming Code	R-S (255,223) code over $GF(2^8)$	(16,8) distance-4 shortened Hamming Code

Table 4.1: The Example Schemes

decoding, by *single memory* argument, we get

$$r_1 = r_2 = r_3 \cdots = r_j \qquad j \geq 1 \tag{4.54}$$

Therefore, the probabilities $q^{(j)}$ (defined in chapter 3) can be lower bounded as follows:

$$q^{(j)} \geq 1 - r_j \qquad j \geq 0 \tag{4.55}$$

From here on, the optimum throughput efficiency of this hybrid ARQ scheme can be obtained using the procedure outlined in Chapter 3.

4.5 Discussion of Results

In this section we have considered three specific example schemes and compared the various performance measures for those schemes. The three example schemes (in order of increasing complexity) are listed in Table 4.1.

We note that the rate-1/2 code in scheme-3 is obtained by shortening the (56,48) code. In the plots, the schemes have been referred to by numbers 1,2 and 3. Symbol E (or L) is used to indicate erasure-only (or LIA-only) inner decoding. For simplicity, we have not considered a combined E-L decoding.

The probability of incorrect decoding P_{er} versus the bit-error rate (BER) have been plotted in figure 4.2 for the three schemes. Each point in the curves has been chosen to obtain the minimum possible value of P_{es} (probability of decoding failure) under the condition that $P_{er} < 10^{-10}$. In case of scheme-1, this is achieved by varying the number of errors corrected ($\leq d_{min}$). In the erasure-only decoding (2E and 3E) we vary the two parameters T_{es} and $t_2(i)$ and in the case of LIA-only decoding (2L and 3L) we vary T_{el} and t_2 to achieve the desired performance. The curve for the scheme 3L is not shown because the probabilities are too close to zero ($\leq 10^{-200}$) and hence couldn't be accommodated in the graph. The corresponding value of P_{es} versus the bit-error rate for the various schemes is shown in fig. 4.3. Note that both P_{er} and P_{es} increase as BER increases. However, at a certain BER ϵ_c, the decoding error probability P_{er} peaks and then decreases as BER increases. This is due to the fact that above that value of BER, the probability of a decoding failure for a block becomes 1. The decoder rejects every received block without decoding. Since there is no decoding, there is no decoding error.

From Fig. 4.2, it might appear that the two LIA-decoded schemes (one of them not shown in the figure) are better but, this lower P_{er} is achieved by not decoding often enough. The two erasure-only decoded schemes reach decoding failure probabilities of 1 at much higher values of BER and therefore they are better. There is a substantial improvement in performance due to the parity retransmission feature. In Fig. 4.4, we show the probabilities of correct block decoding for the various schemes.

The throughput performance of the hybrid ARQ schemes using these three schemes are shown in Figs 4.5-4.8 for various BERs. The chosen BERs are the highest values of BERs at which schemes 1, 2(E and L), 3L and 3E can still achieve reliable communication ($P_{er} < 10^{-10}$) respectively. The schemes 2 and 3 achieve the maximum possible throughput at $\epsilon = 3\text{x}10^{-3}$. The scheme 3E achieves almost the maximum

possible throughput for $\epsilon \leq 6\text{x}10^{-3}$ and provides a 20% channel utilization with a hundred receivers at $\epsilon = 10^{-2}$.

The superior performance of the erasure-only schemes can be explained as follows. If the inner decoder detects the presence of errors, then by doing nothing (as in LIA), it is wasting not only its capability but that of the outer code. In contrast, the erasure operation helps the outer code in decoding successfully hence the better performance. We have thus shown that these schemes can provide us reliable communication with acceptable channel utilization levels at bit-error rates $\leq 10^{-2}$.

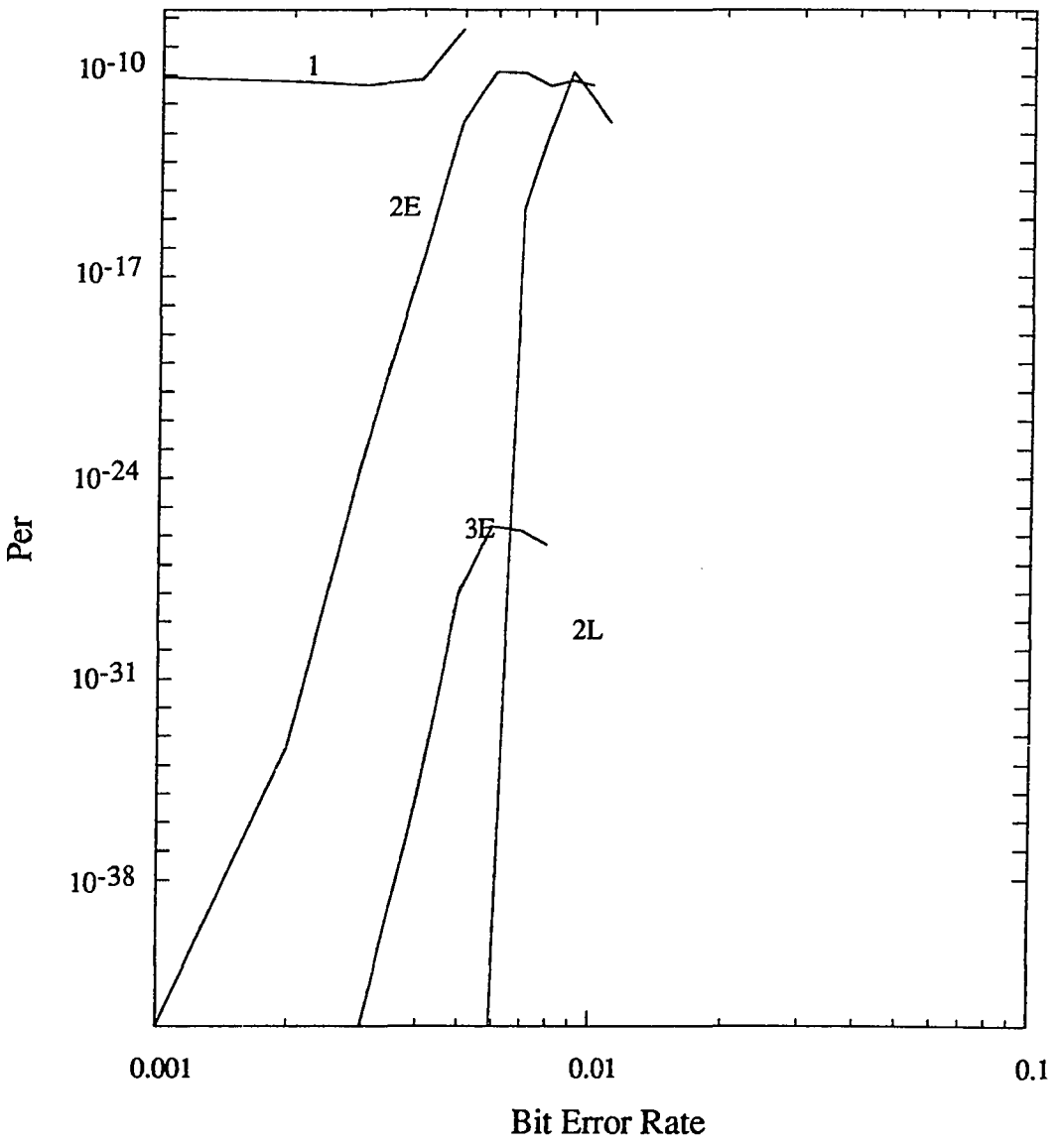

Figure 4.2: Upper Bounds on Probabilities of Decoding Error.

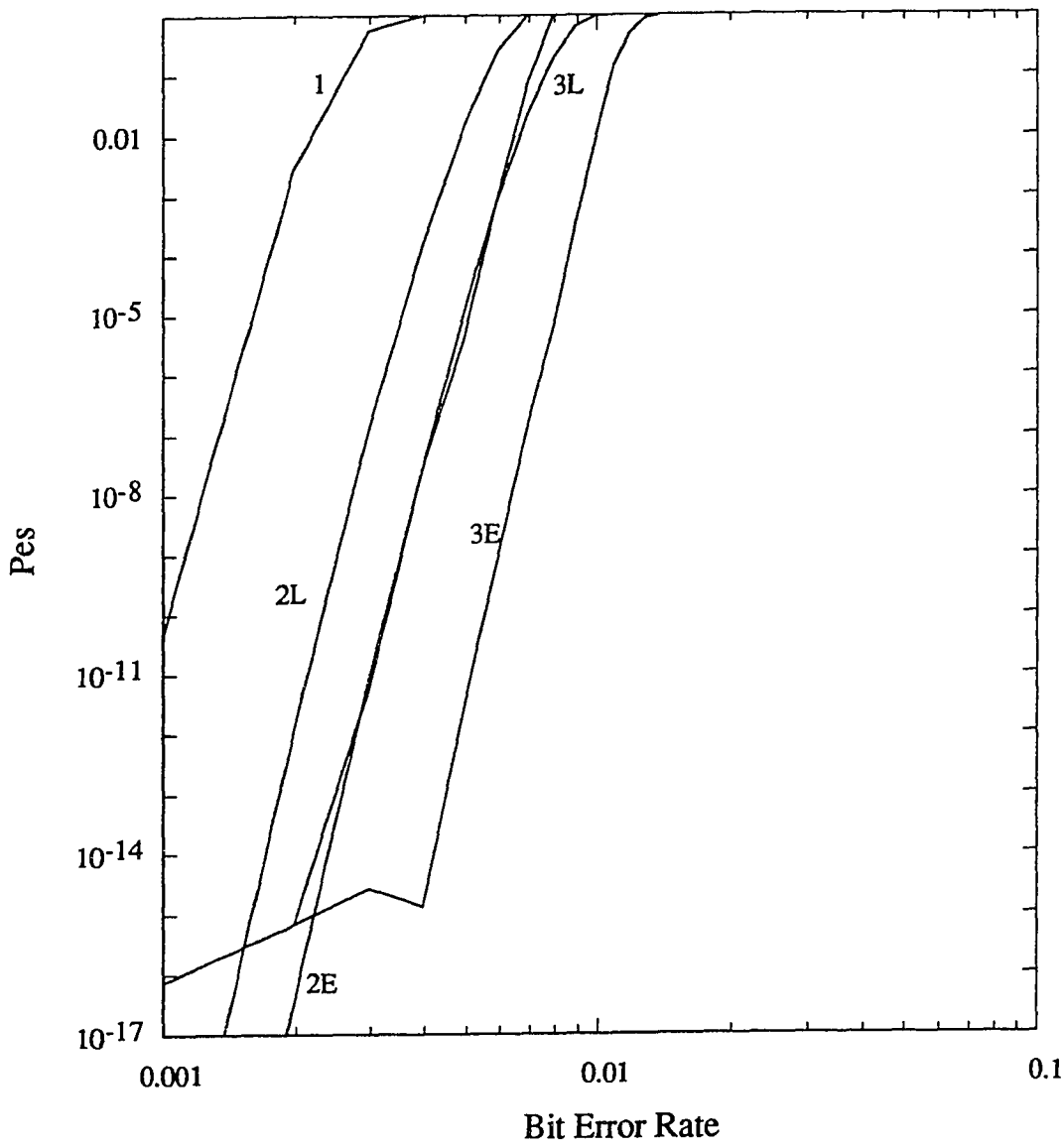

Figure 4.3: Upper Bounds on Probabilities of Decoding Failure.

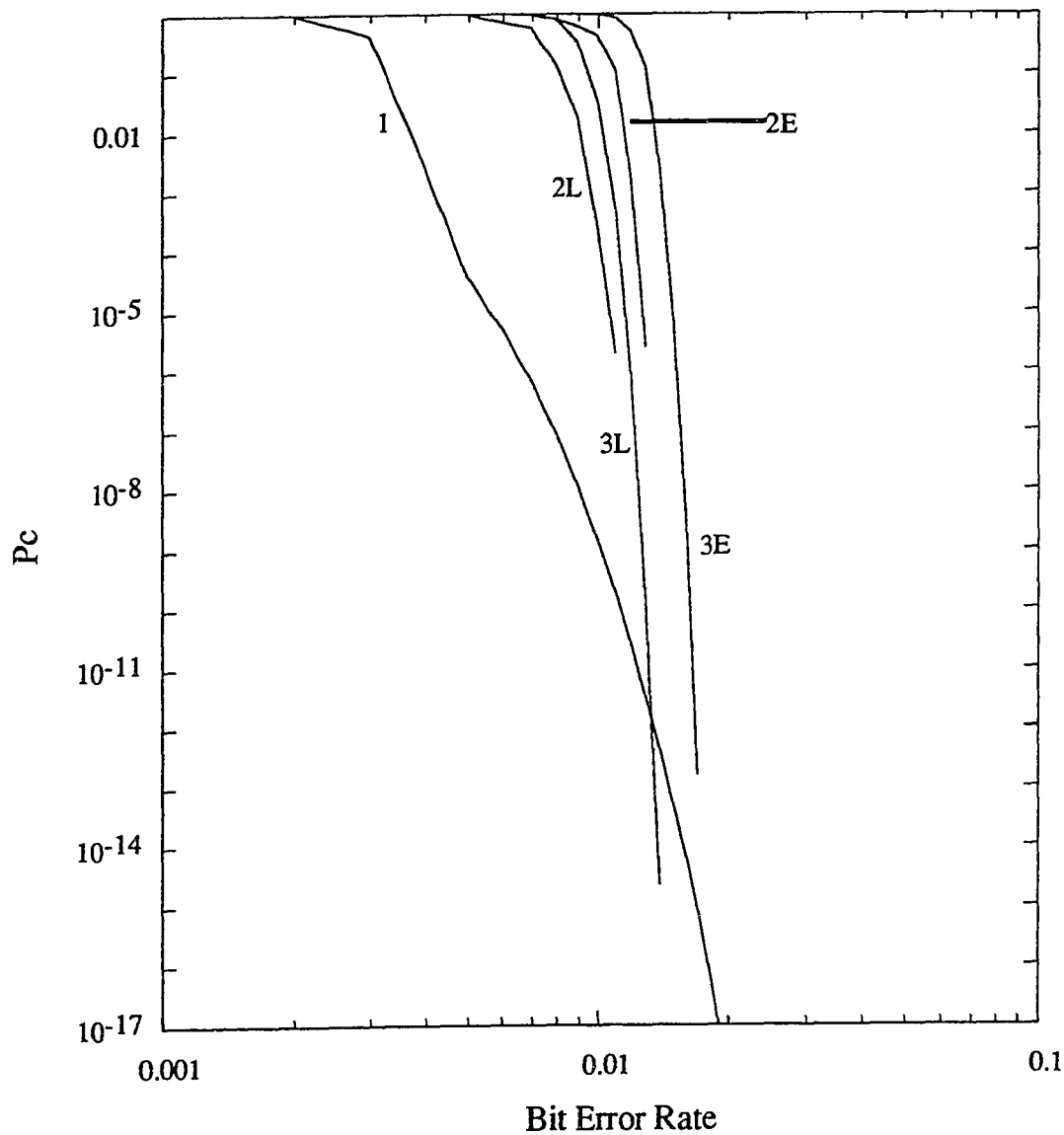

Figure 4.4: Lower Bounds on Probabilities of Correct Decoding.

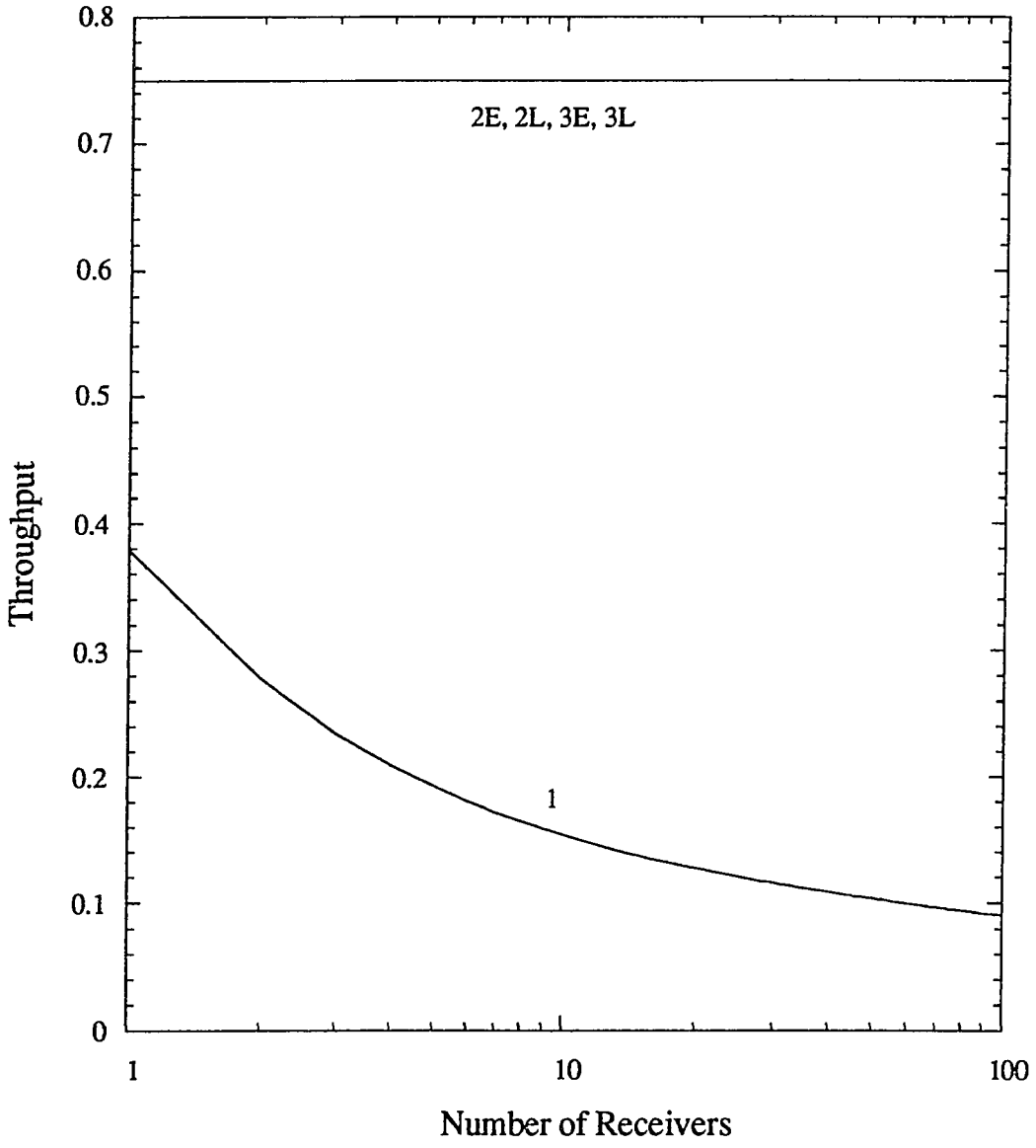

Figure 4.5: Throughput versus number of receivers ($\epsilon = 2.95\text{x}10^{-3}$).

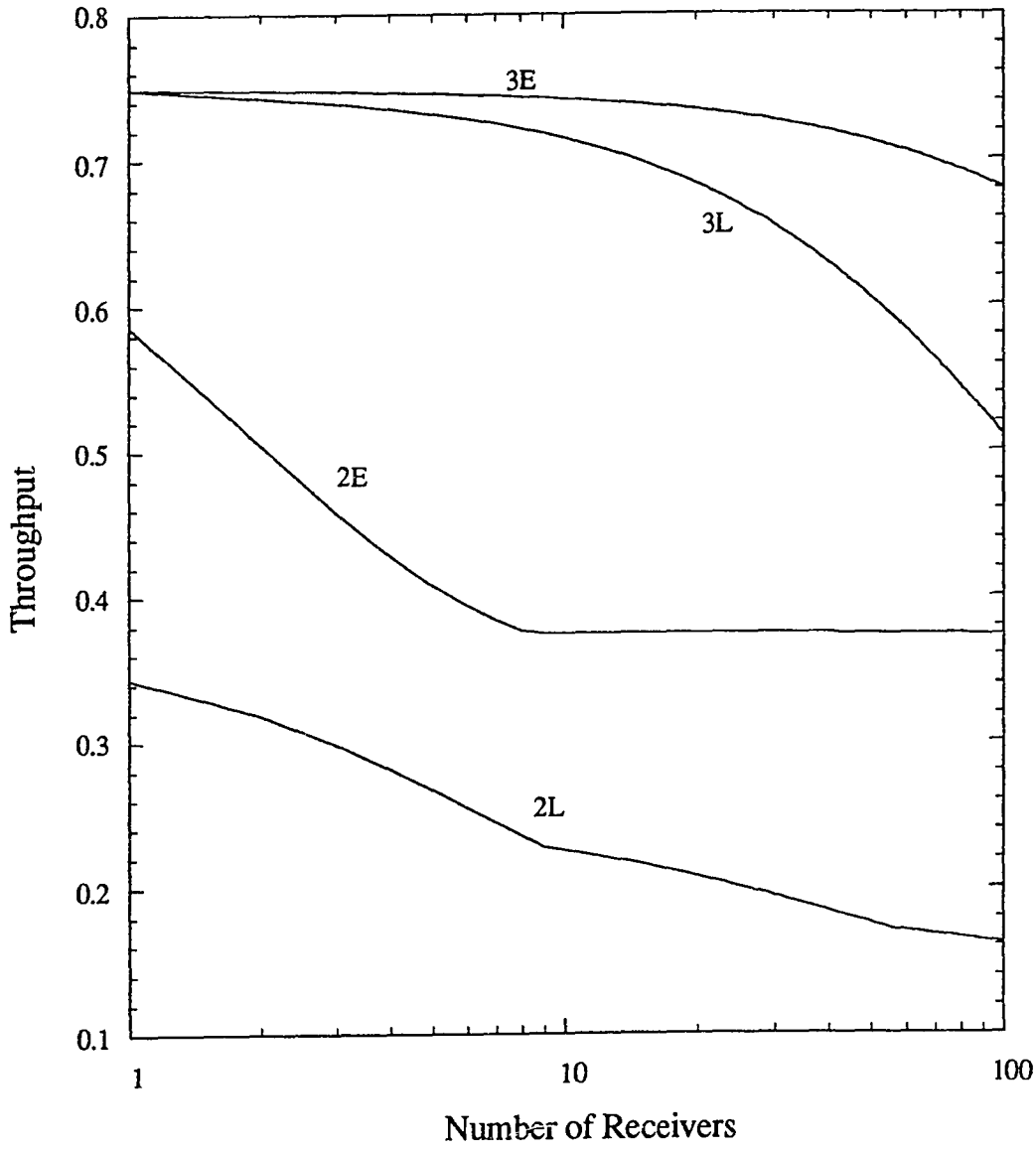

Figure 4.6: Throughput versus number of receivers ($\epsilon = 5.95\text{x}10^{-3}$).

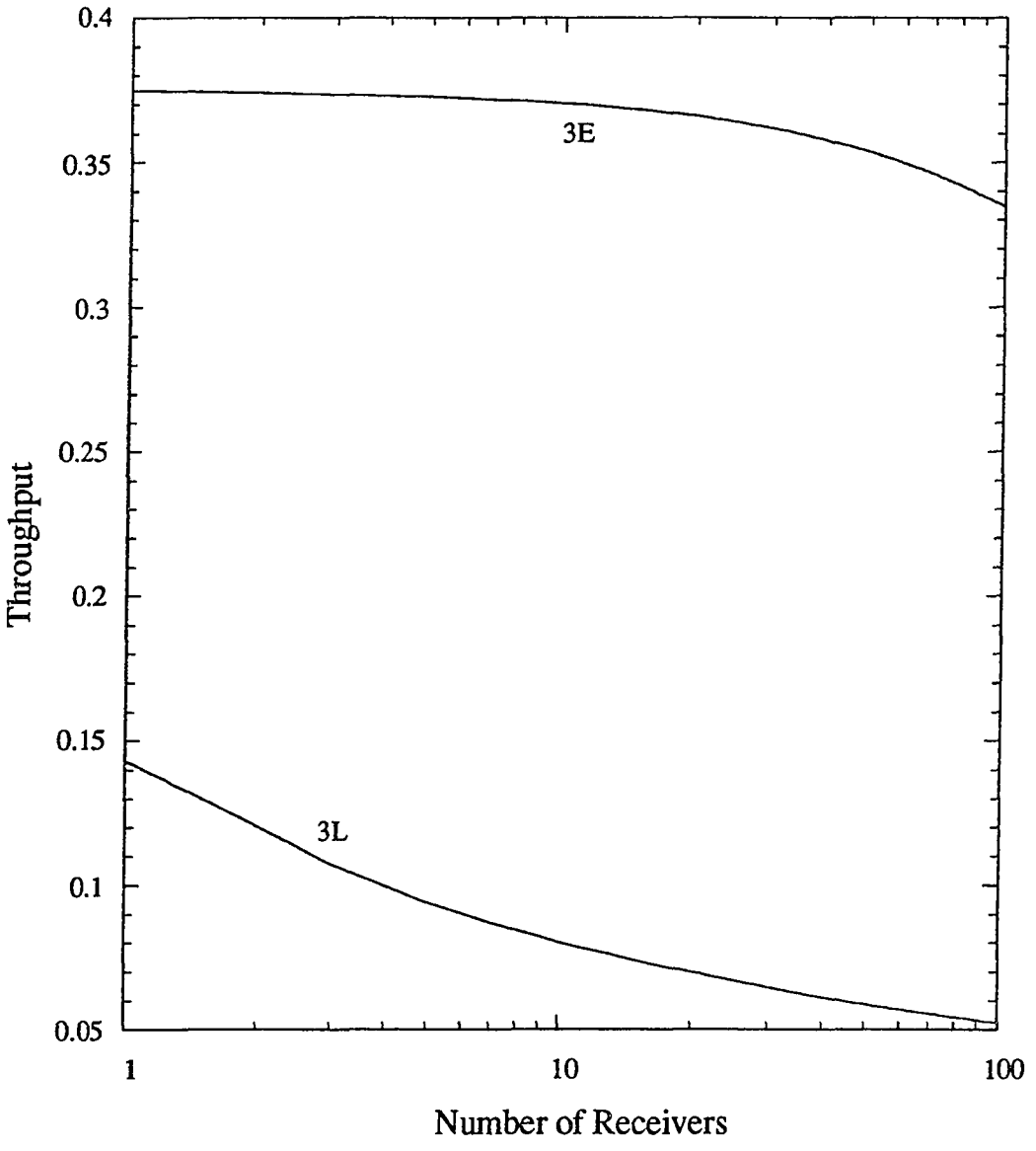

Figure 4.7: Throughput versus number of receivers ($\epsilon = 8.92\text{x}10^{-3}$).

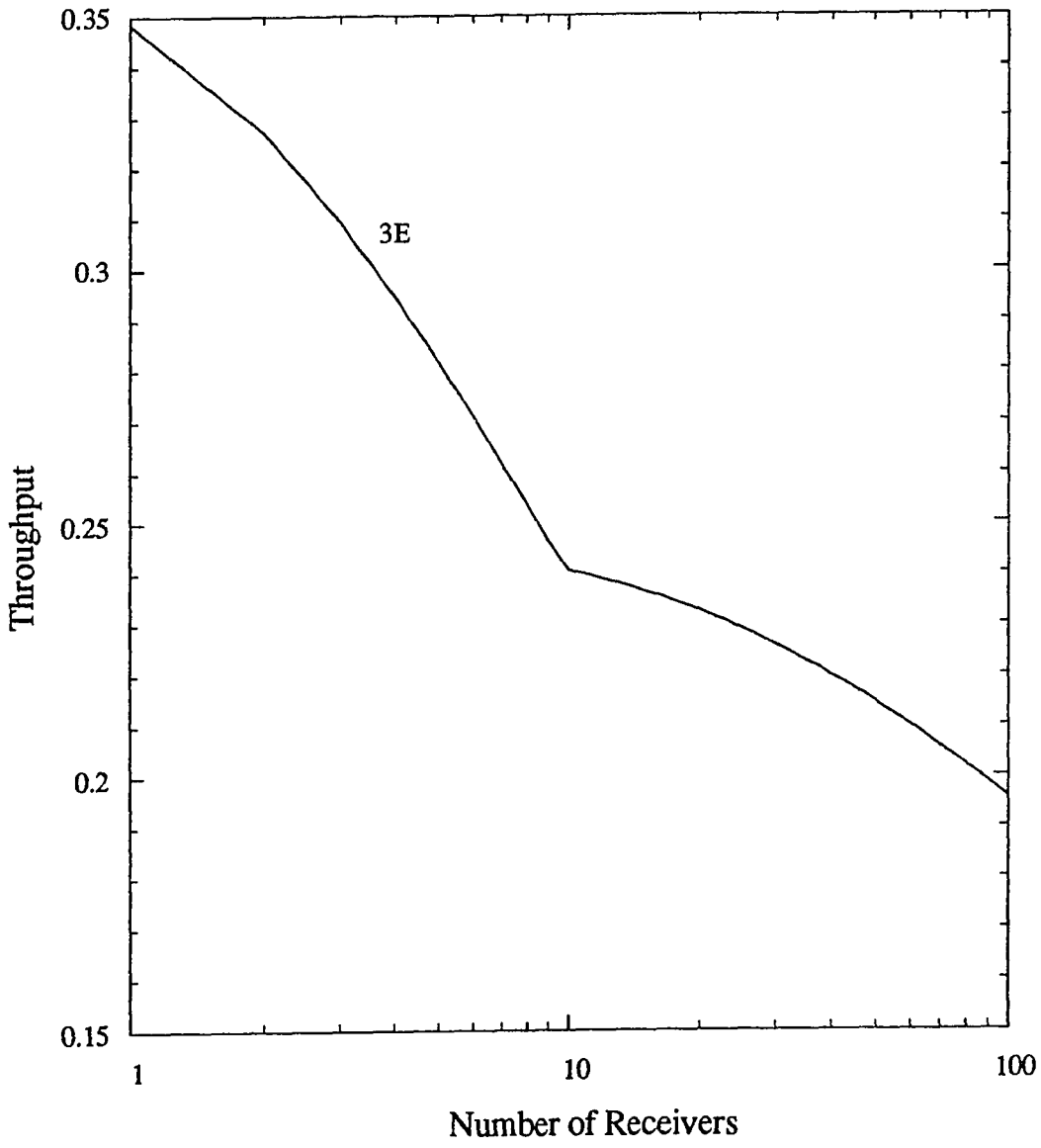

Figure 4.8: Throughput versus number of receivers ($\epsilon = 10^{-2}$).

Chapter 5

Selective Repeat ARQ Schemes for Non-Homogeneous Broadcast Channels

5.1 Introduction

In the last three chapters, we considered error-control schemes for homogeneous broadcast channels where the transmitter communicates with the entire broadcast population. The HB channels could also be considered as the case of transmitters that always transmit to the same set of receivers. A natural generalization of HB channels would be to consider a broadcast scenario where a transmitter sends out messages intended for only a subset of the entire broadcast population even though the messages could have been potentially received by the entire population (e.g., computer networks, packet radio networks etc.). Successive messages from the transmitter may be intended for different subsets of the broadcast population and these subsets may not be disjoint. We will hereafter refer to this channel as the Non-Homogeneous Broadcast (NHB) channel. These channels have also been referred to as multicast channels in literature [Go-Rom 90].

In this chapter, we investigate retransmission error-control schemes for the NHB channel. Recently, Gopal and Rom [Go-Rom 90] have proposed some go-back-N and some selective repeat ARQ schemes for the NHB channel. Their selective repeat ARQ schemes are based on time-sharing ideas and hence do not exploit the full potential of the broadcast channel. Also, they only consider the ideal case where the receivers are

assumed to have infinite buffer capacity. In this chapter however, we consider some error-control schemes that are not time-shared. We obtain the time-shared schemes as special cases of the proposed schemes. We have examined both the ideal as well as the non-ideal cases. We show that significant gains can be achieved in channel utilization if we exploit the broadcast nature of the channel, especially when the messages are being sent out to overlapping sets of receivers. In the next section, we briefly identify the issues that make the NHB channel different from the HB channel.

5.2 Motivation

In case of the HB channel, the messages carried a single sequence number space because the target audience for the messages was always fixed. As pointed out in [Go-Rom 90], a straightforward extension of the results obtained for the HB channel will create confusion for the receivers in the NHB channel, rendering them unable to differentiate between a lost message and a message not addressed to them. Let us consider an example to illustrate this point. Consider an NHB environment consisting of transmitter Tr and receivers $Rec_1, Rec_2, \cdots, Rec_5$. Let us assume that all the transmitted messages use the same sequence number (SN) field. Say the first message with SN=1 is sent to Rec_1, Rec_2 and Rec_3, the second message (SN=2) is sent to Rec_1, Rec_2 and Rec_4 and the third (SN=3) is sent to Rec_2, Rec_3 and Rec_5. Let us assume that Rec_3 receives the messages 1 and 3 successfully, as expected, but does not receive message 2. Since, Rec_3 has no way of knowing that message 2 was not intended for it, it cannot release message 3 to the user because it is out of sequence. This simple example illustrates the fact that for the NHB case, we will have to handle the sequence numbering differently.

One way to solve this sequence number problem would be to require every receiver in the broadcast population to acknowledge every message (regardless of whether the

receiver is in the intended target audience or not). The transmitter on it's part is required to keep records of every receivers' ACKs/NAKs for each message. These requirements are same as those in the case of the HB channel and hence equivalent to the problem solved earlier. If the number of receivers in the target audience of the message is small compared to the total broadcast population, then this scheme will be extremely inefficient as far as throughput efficiency is concerned because we require every receiver to correctly receive and acknowledge the message. This scheme is also wasteful of buffer resources at the transmitter because of it's need to maintain information on all the receivers in the broadcast population. In addition, we need to provide additional information to the receivers to let them know whether a particular message is intended for them or not. We will be doing all this just to have the ease of maintaining a single sequence number field. Clearly, this approach is too simple-minded and too wasteful.

Another solution is to set up a separate sequence number field for each of the receivers in the broadcast population. Every message will then have to include the identification and the associated sequence number for each of the receivers to which the message is transmitted. We note that the identification field grows as log of the total population. However, the transmitter will have to maintain a separate sequence number field for each receiver (probably thousands of them), even though only a handful of them may actually be communicating with the transmitter at that time. We could even have a separate sequence number field for each destination set but that approach is even more complex and wasteful of memory because the number of subsets of a set with P elements grows exponentially in P. In fact we will be dealing with the number of subsets of each subsets (2^{2^P}) in this case, which grows even faster.

Typically for data networks, only a small fraction of the total broadcast population is active at any time. The transmitter would then be communicating with a

subset of this active group. It is then possible to have a separate sequence number field for each group of destinations in communication with the transmitter. This is the approach we intend to pursue. This is perhaps less complex and more efficient than maintaining separate sequence numbering for each of the active destinations as suggested in [Go-Rom 90].

Since the messages are being sent out to different subsets of the broadcast population, one way to accomplish the information transfer would be to time-share the channel. This is the approach suggested in [Go-Rom 90]. Time-sharing is simple and effective but it is not the most efficient way of communicating in a broadcast channel [Cover 72]. It is possible to improve channel utilization by combining messages intended for various groups of destinations since each transmission can potentially be received by the entire broadcast population. This is precisely the goal of the schemes proposed here. The time-shared (TS) schemes can then be realised as special cases of the proposed schemes.

In the next section, we briefly describe the schemes proposed here. We then analyze the performance of these schemes for various sets of parameters. We consider the cases of infinite and finite broadcast populations. By comparing the time-shared and non-time-shared schemes, we obtain the trade-offs between complexity and gains in channel utilization.

5.3 The Protocol

In the proposed scheme the transmitter maintains q sequence number counters, where q is the number of groups of receivers in communication with the transmitter. This situation can be viewed as the case of q files being transmitted to q different groups. The files may all be of different length. As soon as a file transmission is over, that particular counter is reset and restarted for the next transmission.

The messages intended for various groups are combined. Each transmitted message, therefore, consists of q message segments. The message segments carry the addresses of each of the destinations in the group. The successful receipt of a message enables the receiver to recover all the message segments (in the message) addressed to it. We will use the idea of states (defined in chapter 2) to describe the progress of message segments. A message segment in state$-j$ will be transmitted $m_j(r)$ $(j \geq 0)$ times, where r is the number of receivers that are yet to receive the message segment. Multiple copies of the message segments are sent out in different messages (we don't want to put all our eggs in one basket). Before each transmission, the transmitter collects q message segments from the input queue, combines them and sends them out. The first and last message segment to be sent to a group should clearly indicate the beginning and end of the file. The choice of the parameters $m_j(r)$ $(j \geq 0)$ is done based on an optimization process which will be described during the following analysis.

If we let $q = 1$ in the scheme described above, we obtain a scheme where each message has only one message segment. This scheme is the time-shared case.

5.4 Performance Analysis

In this section, we obtain throughput efficiency results for the the NHB ARQ scheme proposed in the previous section. For the purpose of analysis, we classify the problem into two broad categories based on the size of the total broadcast population: the infinite population case and the finite population case.

Since each message segment is being sent to a different subset of the broadcast population, it is only sensible to randomize the size of the group to which the message segment is being sent. This set will be referred to as the destination group. The size of the i^{th} destination group will be denoted by the random variable R_i. We assume that

the individual destination groups are picked independently from message segment to message segment and they are identically distributed. In other words, each message segment has the same probability of having a certain number of receivers in it's destination group. Therefore, the random variables R_i, $(1 \leq i \leq q)$ are independent and identically distributed. Let $p(r)$ be the probability that a message segment has r destinations.

We assume that the messages are transmitted in fixed length frames of n bits each and that the transmitter has an unlimited supply of message segments intended for the q destination groups. Over one such long transmission session, we obtain the throughput efficiency of the various schemes. In both the infinite as well as the finite population cases, we first analyze the ideal ARQ schemes. We then restrict the buffer sizes to be finite and look at their non-ideal counterparts. In the next section, we obtain the throughput efficiency results for the infinite population case.

5.5 The Infinite Population Case

5.5.1 The Ideal ARQ Scheme

Each message in this case consists of q message segments, each intended for a different destination group. We now show that the probability of having overlapping receivers between two independently chosen destination groups goes to zero as the total broadcast population goes to infinity. Let P be the total broadcast population. Let us assume that we independently pick two groups of sizes i and j, respectively, from this population. The probability, $C(i,j)$, that there is at least one receiver in common between these two destination groups is given by,

$$C(i,j) = 1 - \frac{\binom{P-i}{j}}{\binom{P}{j}}, \tag{5.1}$$

It is not very difficult to show that this probability goes to zero as P goes to infinity. Moreover, it goes to zero faster if i and j are relatively small compared to P. That is precisely the problem we described earlier.

In view of the result obtained above, we can ignore any overlap between the destination groups in the infinite population case. We note that in the ideal ARQ scheme, the optimum number of copies of a message to be transmitted is one ($m_j = 1,\ j \geq 0$), since there is no possibility of buffer overflow (see chapter 2 for discussion). The throughput can be obtained by counting the total number of time-slots used to deliver an entire message to it's constituent destination groups. This process is simplified by the fact that the destination groups can be considered non-overlapping. Let $T(r_1, r_2, \cdots, r_q)$ be the total average number of time-slots used in successfully delivering all the message segments of the message to their respective destination groups, given that the group sizes are $r_1,\ r_2, \cdots r_q$ respectively. Let $T_i(r_i)$ $(1 \leq i \leq q)$, be the total average number of time slots used to deliver the i^{th} message segment to it's destination group of r_i receivers. Since the destination groups are disjoint, it follows that

$$T(r_1,\ r_2, \cdots, r_q) = \sum_{i=1}^{q} (n_i/n)\ T_i(r_i), \tag{5.2}$$

where n_i is the length of the i^{th} message segment such that $\sum n_i = n$. We note that for a message segment to reach it's destination group successfully, the entire message has to arrive there error-free. Therefore the probability of successful reception of a message segment by a particular receiver is same as the probability of successful reception of the entire message by that receiver. Then it is not very difficult to see that $T_i(r_i)$ is the total average number of time-slots required in an ideal ARQ scheme for a HB channel with r_i receivers. In a manner similar to that in chapter 2, we can write

$$T_i(r_i) = \sum_{j=0}^{\infty} [1 - \{1 - (1-p)^j\}^{r_i}], \tag{5.3}$$

where $p = (1-\epsilon)^n$ and ϵ is the BER of the channel.

Let T_{I-Id} be the total average number of time-slots used to successfully deliver all the message segments of a message to their respective destination groups and let η_{I-Id} be the throughput efficiency of the infinite population ideal scheme, then

$$\begin{aligned} T_{I-Id} &= \mathbf{E}_{R_1, R_2, \cdots, R_q}[T(r_1, r_2, \cdots, r_q)], \\ &= \sum_{i=1}^{q} (n_i/n)\, \mathbf{E}_{R_i}\, [T_i(r_i)]. \end{aligned} \tag{5.4}$$

Using the fact that the random variables R_i are $i.i.d$, we obtain

$$T_{I-Id} = \sum_{i=1}^{q} (n_i/n)\, \mathbf{E}_R\, [T(r)] = \mathbf{E}_R\, [T(r)], = \sum_{r=0}^{\infty} T(r)p(r). \tag{5.5}$$

Therefore,

$$\eta_{I-Id} = 1/T_{I-Id}. \tag{5.6}$$

The Time-Shared Case

As a special case, we can obtain the ideal time-shared scheme by letting $q = 1$ in the expressions above. This scheme is somewhat similar to that proposed by [Go-Rom 90]. However, the throughput expression obtained in the last subsection is independent of q. Therefore the throughput efficiency of the infinite population ideal scheme is same whether the channel is time-shared or not, except for minor differences in overhead and implementation. However, there are some advantages of the non-time-shared scheme ($q > 1$) which we will discuss later in this chapter.

Computational Results

Figure 5.1 shows the throughput of the infinite population ideal schemes versus bit-error rate for Poisson distributed rv's R_i $(i = 1, 2)$ with means $\lambda_i = 5, 10$. For comparison we have also obtained the throughput for some cases where the destination group size is fixed. We have also shown some curves for mismatched parameters in the destination groups. The curves show (as expected) that the throughput goes down as the average destination group size goes up. Also, we notice that better channel utilization is obtained if more bits are sent to the smaller destination group.

5.5.2 The Non-Ideal ARQ Scheme

In the non-ideal case, the receivers have only a finite buffer to store message segments. An immediate consequence is that the optimum number of copies of a message segment to be transmitted at any state is not one. Therefore the throughput efficiency expressions, in this case, will depend upon round-trip delay (N) and receivers' buffer size. A lower bound on the throughput efficiency can be obtained by counting the total average number of time-slots used to deliver a random message to all the intended destinations in the worst case.

Since we can ignore the overlap between the destination groups of the different message segments, it is sufficient to consider the statistics of one message segment. The worst case scenario is obtained when all all the message segments transmitted to that destination group after the random message in question are successfully received by all the receivers in the destination group. This will cause the receivers' buffer to overflow sooner than in any other case and result in maximum penalty for retransmission of the random message.

In the analysis, we need to find the optimum values of the parameters $m_j(r_i)$ $(j \geq 0)$, using a dynamic programming optimization technique.

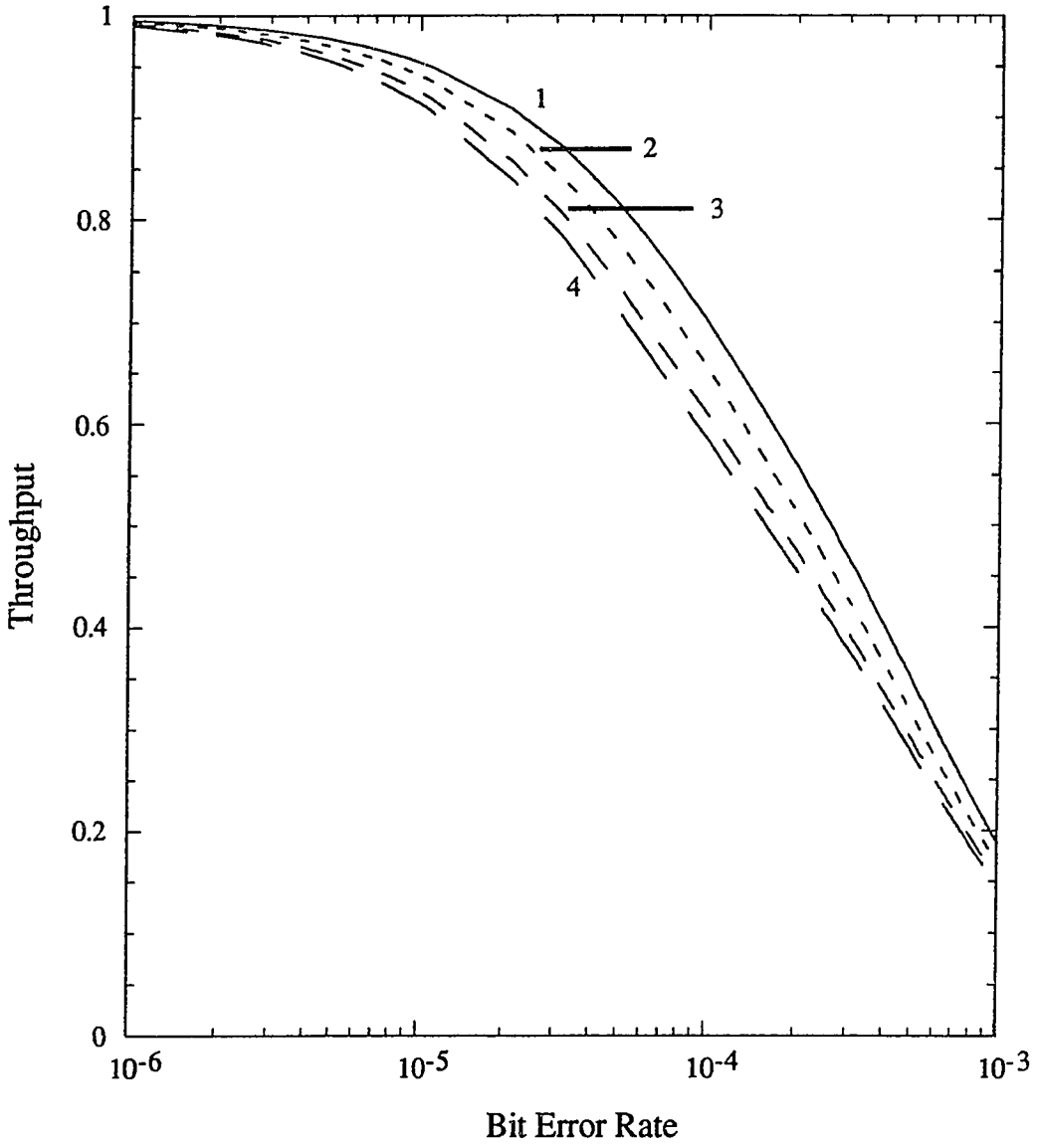

Figure 5.1: Performance of Infinite Population Ideal Schemes: (1) $\lambda_i = 5$ and $R_i = 5$. (2) $\lambda_1 = 5, n_1 = 700, \lambda_2 = 10, n_2 = 300$. (3)$\lambda_1 = 5, n_1 = 300, \lambda_2 = 10, n_2 = 700$. (4)$\lambda_i = 10$ and $R_i = 10$.

We first make the following definitions:

- βN — The number of $n-$bit frames each receiver can store. Here β is assumed to be a positive real number.

- B_i $(1 \leq i \leq q)$ — The buffer capacity of a receiver in the i^{th} destination group, in terms of the number of message segments it can store. We have

$$B_i = \frac{\beta N n}{n_i}, \tag{5.7}$$

  where n_i is the length of the i^{th} message segment such that $\sum n_i = n$.

- γ_i — The "instant" in terms of the number of segment slots when the number of distinct message segments equals the buffer capacity of a receiver in the i^{th} destination group. Then we can write,

$$\gamma_i = \frac{B_i - 1}{\lceil \frac{N-1}{m_0(r_i)} \rceil}. \tag{5.8}$$

- δ_i — The state of the i^{th} message segment when the buffer overflows. This quantity is given,

$$\delta_i = \lfloor \gamma_i \rfloor + 1. \tag{5.9}$$

Once the two parameters associated with buffer overflow γ_i and δ_i are known, the number of message segments lost in different states due to buffer overflow can easily be calculated in a manner similar to that in chapter 2 (equations 2.5-2.9). The optimization process to obtain $m_s^*(r_i)$ $(s \geq 0, 1 \leq r_i \leq R_i)$ is very similar to that described in chapter 2. The only difference is that the parameter R_i is a random

variable here. We can use the following algorithm to obtain the total average number of segment slots required to deliver a message segment in the i^{th} destination group to all the receivers in the group. This quantity will be denoted by T_i^* $(1 \leq i \leq q)$. In the following algorithm we will drop the subscript i, recognizing that we are talking about the i^{th} destination group.

1. Set $R = 1$.
2. Set $m_0(R) = 1$.
3. Compute δ and let $J = \delta + 1$.
4. Compute $m_J^*(r)$ $(1 \leq r \leq R)$ and corresponding $T^*(m_0(R), J, r)$, $1 \leq r \leq R$.
5. For $0 \leq s < J$, compute $m_s^*(r)$ and corresponding $T^*(m_0(R), s, r)$ for $0 \leq r \leq R$. Obtain $T^*(m_0(R), 0, R)$.
6. $m_0(R) = m_0(R) + 1$. Go to step 3 and repeat steps 3 through 6 till $m_0(R)$ reaches some predetermined limit.
7. $T^*(0, R) = \min_{m_0(R)} [T^*(m_0(R), 0, R)]$.
8. $R = R + 1$. Go to step 2 and repeat steps 2 through 8 till R reaches some predetermined limit $\mathcal{R}$, such that $Pr\{R = \mathcal{R}\}$ is negligibly small.
9. $T^* = \mathbf{E}_R[T^*(0, R)]$.

The total average number of time-slots required to deliver a message to all the intended destinations is then given by,

$$T^* = \sum_{i=1}^{q} (n_i/n) T_i^*. \tag{5.10}$$

The throughput efficiency of the infinite population, non-ideal scheme η_{I-NID} is given by

$$\eta_{I-NID} = 1/T^*. \tag{5.11}$$

The Time-Shared Case

By letting $q = 1$ in the expressions above, we can obtain the time-shared case. We notice from equation (5.7) that $B_i = \beta N$ (because $n_i = n$). The value of γ_i and δ_i will be proportionately smaller. Therefore, for the same buffer size at the receivers, the buffer overflows sooner in the time-shared case. If n_i's are all equal (i.e., $n_i = n/q$), then there will be a $q-$fold increase in the available buffer at each receiver in the general case ($q > 1$), delaying the buffer overflow proportionately. Due to this fact, the throughput in the time-shared case will be less.

Computational Results

The throughput performance for the infinite population non-ideal schemes are shown in Figs 5.2 - 5.5. The throughput when both the destination groups are Poisson distributed with mean 5 is shown in Fig. 5.2. The corresponding curves for $\lambda_i = 10$ is shown in Fig. 5.3. These curves also show the performance for the time-shared cases. We notice that the best throughput is achieved when both the destination groups have equal length message segments. The curves also show that the time-shared cases yield lower throughput (as explained earlier) in spite of the fact that there is no overlap between destination groups.

Fig. 5.4 shows the performance when there is mismatch both in the destination sizes as well as the segment length. The comparison between the best of ideal and non-ideal schemes is shown in Fig. 5.5. We see that the non-ideal schemes achieve throughput close to that of ideal schemes for $BER < 10^{-4}$.

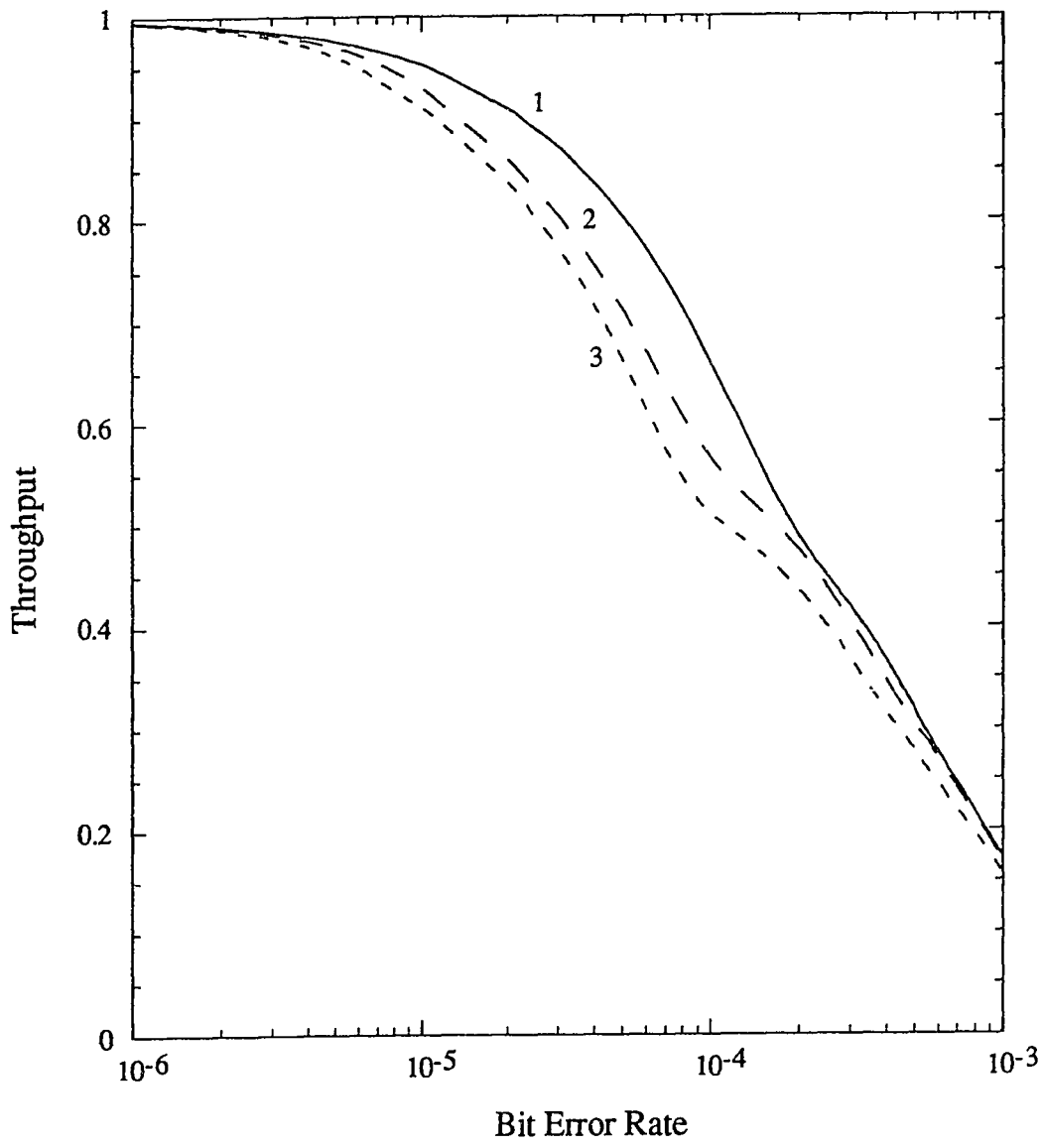

Figure 5.2: Performance of Infinite Population Non-ideal Schemes for $\lambda_i = 5$: (1) $n_1 = 500, n_2 = 500$ and $R_i = 5$, $n_1 = 500, n_2 = 500$. (2) $n_1 = 700, n_2 = 300$ and $R_i = 5$, $n_1 = 300, n_2 = 700$. (3)$q_1 = 1, n_1 = 1000$.

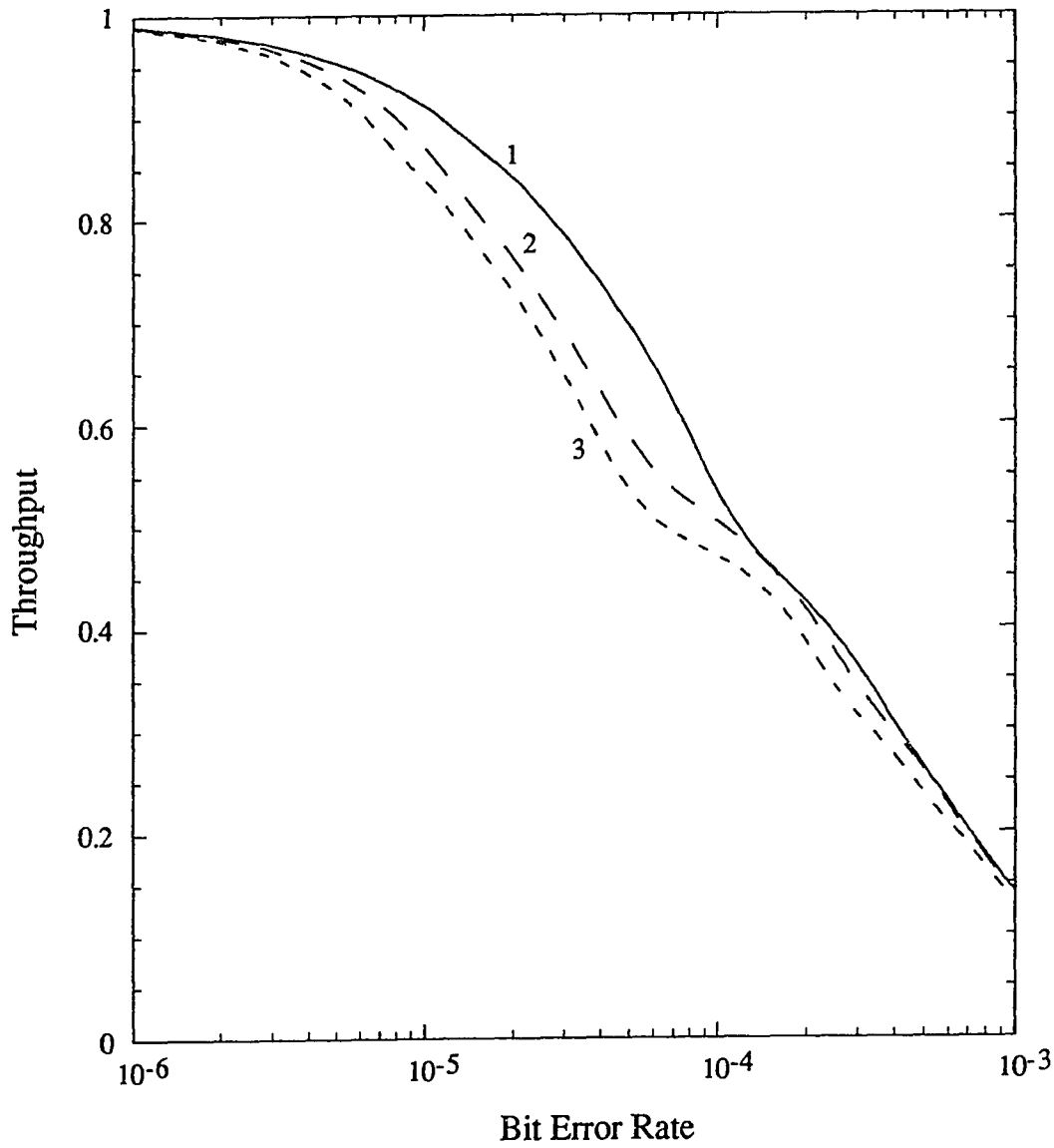

Figure 5.3: Performance of Infinite Population Non-ideal Schemes for $\lambda_i = 10$: (1) $n_1 = 500, n_2 = 500$ and $R_i = 10$, $n_1 = 500, n_2 = 500$. (2) $n_1 = 700, n_2 = 300$ and $R_i = 10$, $n_1 = 300, n_2 = 700$. (3)$q_1 = 1, n_1 = 1000$.

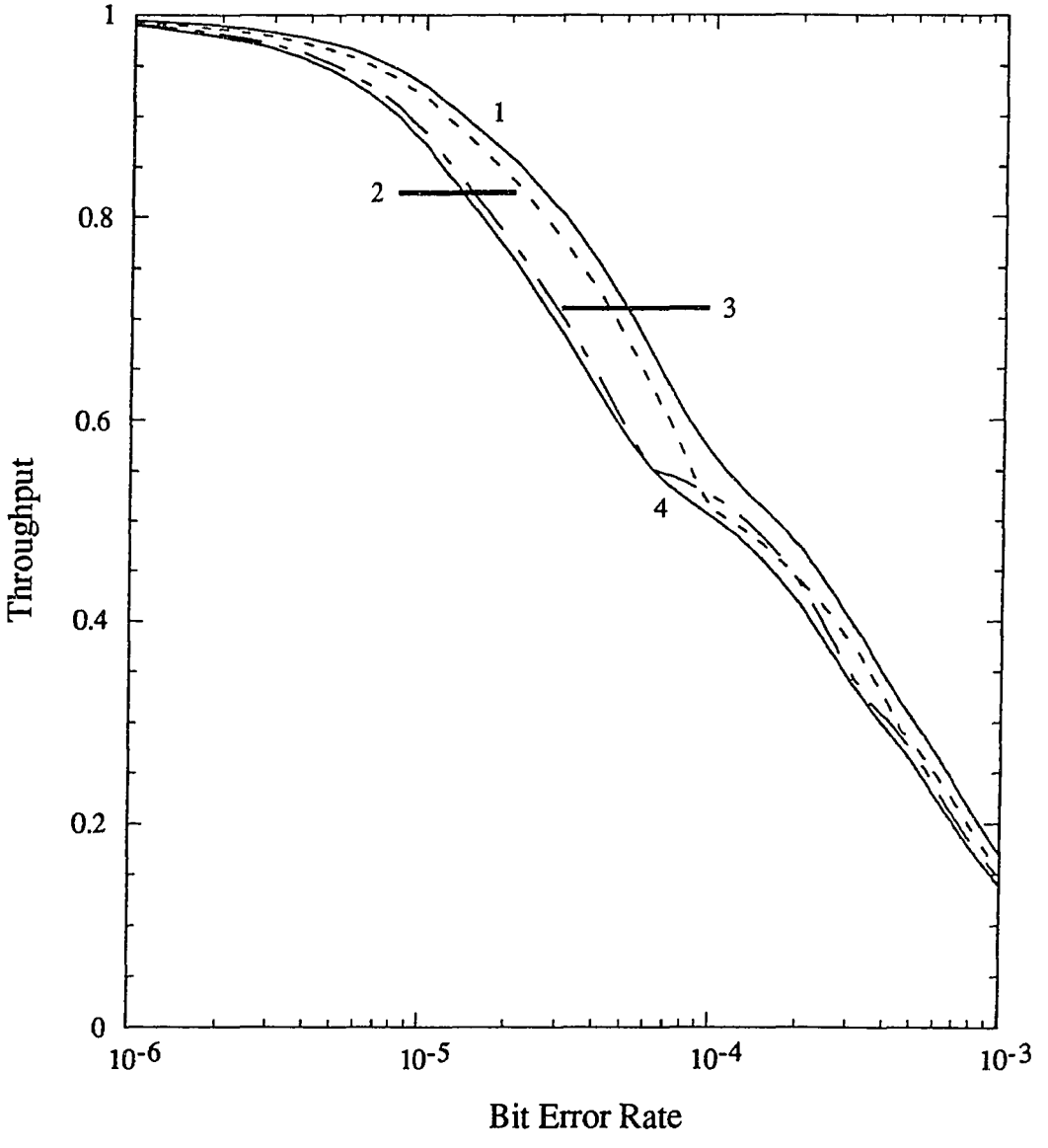

Figure 5.4: Performance of Infinite Population Non-ideal Schemes with mismatched parameters: (1) $\lambda_i = 5, n_1 = 300, n_2 = 700$. (2) $\lambda_1 = 5, n_1 = 700, \lambda_2 = 10, n_2 = 300$. (3) $\lambda_1 = 5, n_1 = 300, \lambda_2 = 10, n_2 = 700$. (4) $\lambda_i = 10, n_1 = 300, n_2 = 700$.

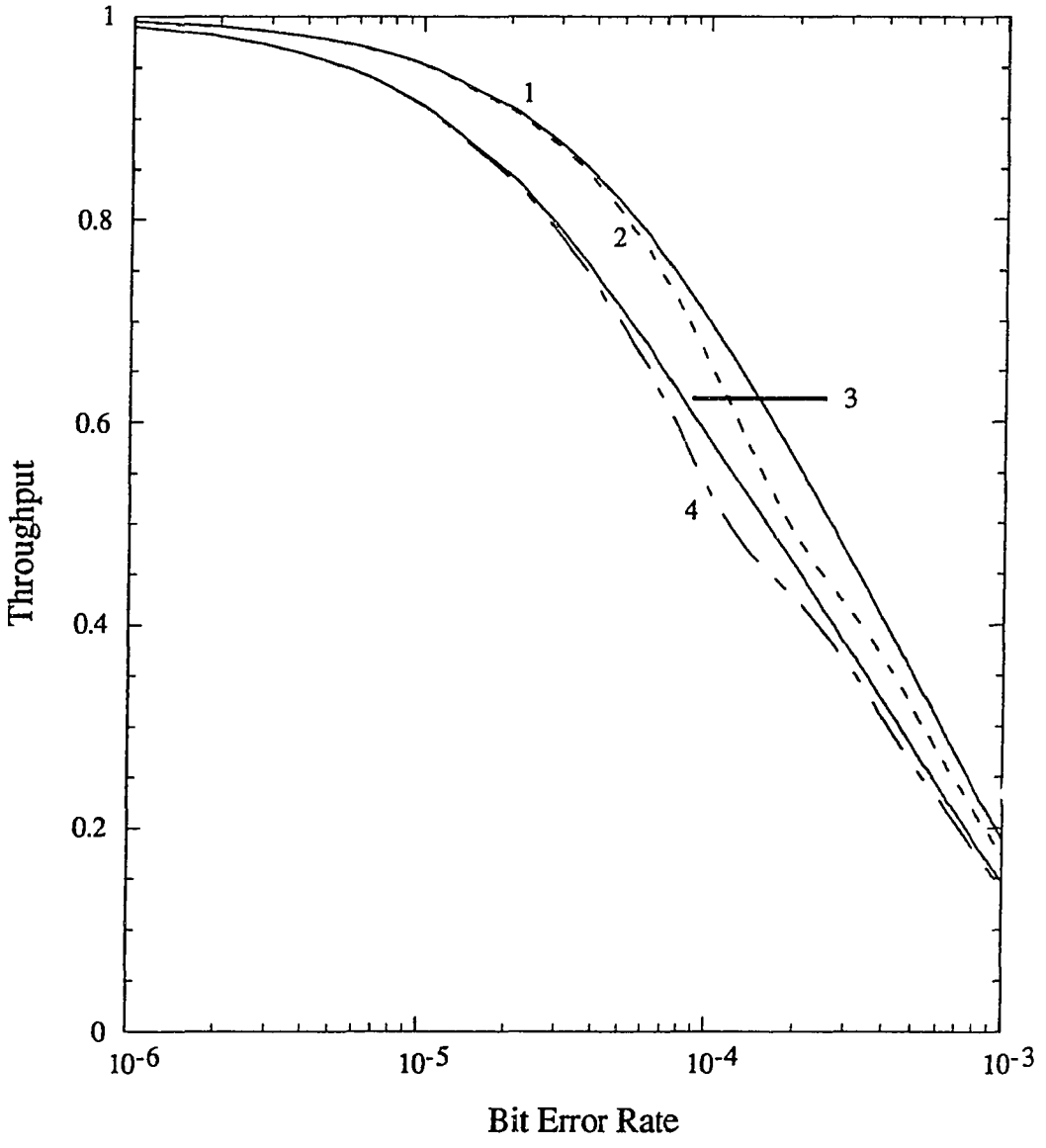

Figure 5.5: Performance Comparison of Infinite Population Ideal and Non-ideal Schemes: (1) Ideal, $\lambda_i = 5$. (2) Non-ideal, $\lambda_i = 5, n_i = 500$. (3) Ideal, $\lambda_i = 10$. (4) Non-ideal, $\lambda_i = 10, n_i = 500$.

5.6 The Finite Population Case

Even though the number of potential receivers in a network may be very large, the number of active receivers at any time is likely to be small, therefore the assumption of finite population is a more realistic model. Each message segment would then be sent to a subset of this active (finite) group. In this case the analysis becomes more difficult because we have to consider the overlapping receivers between successive destination groups. It is not very difficult to see that the results for the time-shared case will remain the same as in the infinite population case. The overlap among the destination sets has no bearing on the throughput efficiency in the time-shared case because the messages are never combined (see earlier discussion). However, for the general case ($q > 1$), we will have to account for this overlap. As before, we will analyze both the ideal and non-ideal ARQ schemes in the finite population environment. We begin by analyzing the simpler case, namely the ideal ARQ scheme, in the next section.

5.6.1 The Ideal Scheme

For simplicity, we carry out the analysis for $q = 2$. For $q > 2$, the analysis is similar but tedious. We make the following definitions:

- ξ_j — The probability that a frame is successfully received by a particular receiver within j retransmissions. Then

$$\xi_j = 1 - (1 - p)^{j+1}. \tag{5.12}$$

- P — Total Broadcast Population.

- A_i — The destination set of i^{th} message segment. Let $|A_i| = R_i$. Clearly, R_i's are $i.i.d$ random variables if the sets A_i are independently chosen.

- R_{ij} — The random variable denoting the number of overlapping receivers between the destination sets A_i and A_j. Then the conditional mass function of R_{12} can be obtained as follows.

$$Pr\{R_{12} = r_{12}/R_1 = r_1, R_2 = r_2\} = \frac{\binom{r_1}{r_{12}}\binom{P-r_1}{r_2-r_{12}}}{\binom{P}{r_2}} \stackrel{\text{def}}{=} \phi(r_1, r_2, r_{12}), \tag{5.13}$$

where $1 \leq r_i \leq P$ and $0 \leq r_{12} \leq r_1$.

Let T be the total average number of time-slots required to successfully deliver the message to all the intended destinations, then T can be expressed as the sum of following three quantities:

- T_{12} — Average number of total time-slots required to successfully deliver the combined message to R_{12} receivers.
- T_i, $i = 1, 2$ — Average number of additional time-slots required to successfully deliver the i^{th} $(i = 1, 2)$ message segment to $(R_i - R_{12})$ receivers.

For given values of random variables R_1, R_2 and R_{12}, the quantity T_{12} can be obtained as follows.

$$T_{12}(r_1, r_2, r_{12}) = \sum_{i=0}^{\infty}[1 - \{1 - (1-p)^i\}^{r_{12}}]. \tag{5.14}$$

Therefore,

$$T_{12} = \mathbf{E}_{R_1R_2}[\mathbf{E}_{R_{12}}[T_{12}(r_1, r_2, r_{12})]]. \tag{5.15}$$

The quantities T_i $(i = 1, 2)$ can be obtained in a similar manner as follows.

$$T_i(r_1, r_2, r_{12}) = \frac{n_i}{n} \{1.(\zeta_{T_{12}} - \zeta_{T_{12}-1}) + 2.(\zeta_{T_{12}+1} - \zeta_{T_{12}}) + \cdots \}, \qquad (5.16)$$

where $\zeta_j = \xi_j^{r_i - r_{12}}$ $(j \geq 0)$, is the probability that a message is successfully received by all $(r_i - r_{12})$ receivers within j retransmissions. After some simplification, we obtain

$$T_i(r_1, r_2, r_{12}) = \frac{n_i}{n} \sum_{j=0}^{\infty} [1 - \{1 - (1-p)^{T_{12}+j}\}^{r_i - r_{12}}, \quad i = 1, 2. \qquad (5.17)$$

And

$$T_j = \mathbf{E}_{R_1 R_2}[\mathbf{E}_{R_{12}}[T_j(r_1, r_2, r_{12})]], \quad j = 1, 2. \qquad (5.18)$$

Therefore,

$$T = T_{12} + T_1 + T_2. \qquad (5.19)$$

Let η_{F-Id} be the throughput efficiency of the finite population, ideal scheme then

$$\eta_{F-Id} = 1/T. \qquad (5.20)$$

Computational Results

Figures 5.6 and 5.7 show the throughput performance of the finite population ideal schemes for binomially distributed destination sizes with means 5 and 10 respectively. We have also shown a few examples of fixed and mixed population cases. We see that the throughput deteriorates as the the overlap between the two destination groups decreases.

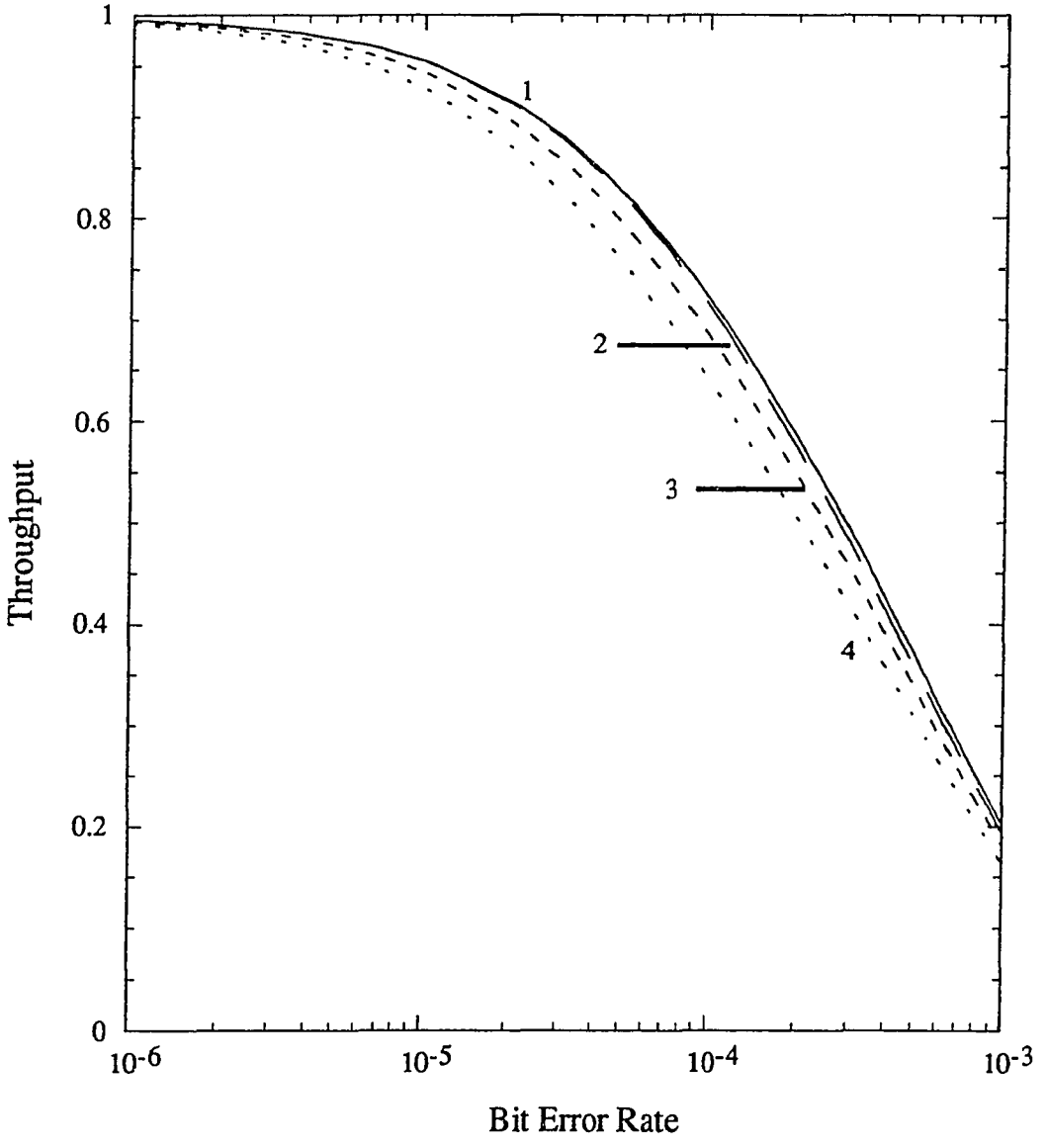

Figure 5.6: Performance Comparison of Finite Population Ideal Schemes (with $P = 20$, $\lambda = 5$): (1) $\lambda_i = 5$ and $R_i = 5, R_{12} = 3$. (2) $R_i = 5$, $R_{12} = 1$. (3) $R_1 = 5$, $n_1 = 700$, $R_2 = 10$, $n_2 = 300$, $R_{12} = 2$. (4) $R_1 = 5$, $n_1 = 300$, $R_2 = 10$, $n_2 = 700$, $R_{12} = 2$.

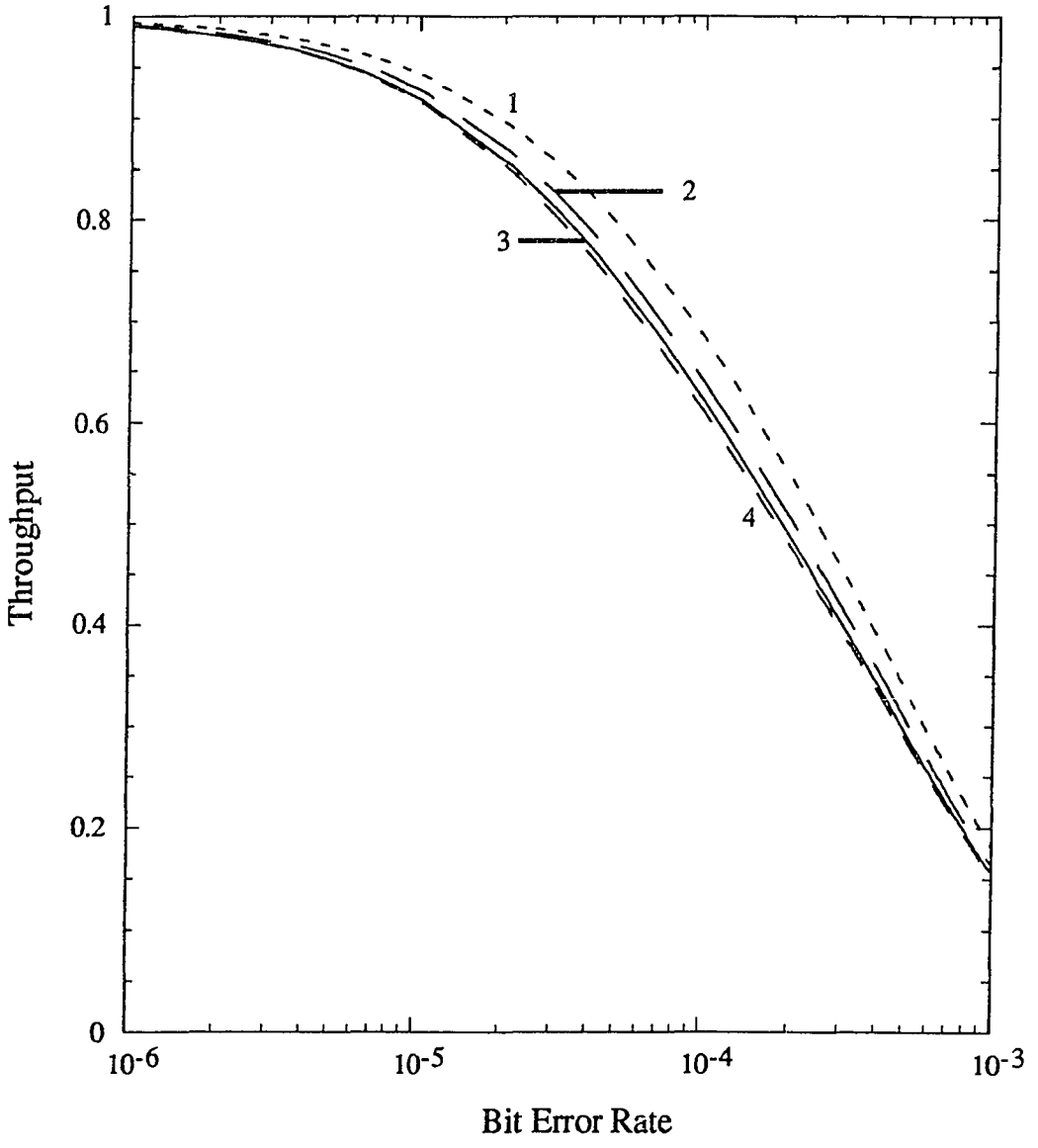

Figure 5.7: Performance Comparison of Finite Population Ideal Schemes (with $P = 20$, $\lambda = 10$): (1) $R_1 = 5$, $n_1 = 700$, $R_2 = 10$, $n_2 = 300$, $R_{12} = 2$. (2) $R_1 = 5$, $n_1 = 300$, $R_2 = 10$, $n_2 = 700$, $R_{12} = 2$. (3) $\lambda_i = 10$ and $R_i = 10$, $R_{12} = 5$. (4) $R_i = 10$, $R_{12} = 2$.

5.6.2 The Non-Ideal Scheme

The case of finite population non-ideal scheme represents the problem of NHB ARQ at it's maximum generality. In that sense, the results in this section represent the culmination of all the ideas developed so far. The receivers in this case have finite buffer and the destination sets are not disjoint. The throughput efficiency results can be obtained by a dynamic programming optimization technique as before. However, the optimization process is complicated in this case by the fact that we will have to consider all the message segments and all the destination sets simultaneously. Therefore, the parameters s and r, used previously in the optimization processes, are both vectors here.

Once again, we carry out the analysis for $q = 2$, for simplicity. The results for $q > 2$ can be obtained by using a similar but more tedious procedure. We introduce some terminology now.

- $\underline{s} = (s_1, s_2)$ — Here s_i $(i = 1, 2)$ is the state of the i^{th} message segment.
- $\underline{R} = (R'_1, R'_2, R_{12})$, where the random variables R'_1 and R'_2 are given by

$$R'_1 \stackrel{\text{def}}{=} |A'_1| = |A_1 - A_1 \cap A_2| = R_1 - R_{12}, \tag{5.21}$$

and

$$R'_2 \stackrel{\text{def}}{=} |A'_2| = |A_2 - A_1 \cap A_2| = R_2 - R_{12}. \tag{5.22}$$

The introduction of random variables R'_1 and R'_2 allows us to deal with disjoint groups of the destinations.

- $\underline{r} = (r'_1, r'_2, r_{12})$ — This vector represents a particular value of the random vector $\underline{R}$ where $R'_1 = r'_1$, $R'_2 = r'_2$ and $R_{12} = r_{12}$.

- $T(\underline{s},\ \underline{r})$ — The residual average number of time-slots needed to deliver the message to all the receivers in all the destination groups, when the combined message is in state-$\underline{s}$ and $\underline{r}$ receivers are yet to acknowledge the message.

As before, the throughput efficiency is obtained by following the worst-case progress of a random message from the transmitter's point of view. The worst case for a random message is obtained when all the subsequently transmitted message segments to either of the two destination groups are successfully received. The transmitter transmits $m_{s_1}(r_1)$ copies of the first message segment and $m_{s_2}(r_2)$ copies of the second message segment when the segments are in states s_1 and s_2 and r_1 and r_2 receivers are yet to acknowledge the message segments in their respective destination groups.

To obtain the throughput efficiency results, we first need to understand the circumstances under which the buffers at the receivers will overflow. We have already partitioned the set of receivers that are addressed by the random message into three disjoint sets, namely, A_1', A_2' and A_{12}. Let the buffer overflow parameters associated with the set of receivers A_1' and A_2' be γ_i $(i = 1, 2)$ and δ_i $(i = 1, 2)$ respectively. These parameters can be obtained from equations (5.8) and (5.9).

However, we need to find similar parameters for the set A_{12} of receivers. In the worst case, described earlier, these receivers are going to have a buffer overflow sooner than the receivers in sets A_1' and A_2' because they are in the destination sets of both the message segments of each transmission after the random message. Let γ_{12} be the number of message transmissions after the transmission of the random message when the buffer of a receiver in set A_{12} is full. If $m_0(r_1) > m_0(r_2)$ then,

$$\gamma_{12} \leq \frac{\beta N - 1}{\lceil \frac{N-1}{m_0(r_2)} \rceil}. \tag{5.23}$$

Let us define a quantity δ_{12} as follows,

$$\delta_{12} \stackrel{\text{def}}{=} \lfloor \gamma_{12} \rfloor + 1. \tag{5.24}$$

It is not very difficult to see that in the worst case, the receivers in A_{12} will have buffer overflow whenever the minimum of s_1 or s_2 equals δ_{12}. We note here that δ_{12} is a lower bound on the state of either of the message segments when receivers in A_{12} have buffer overflow.

In state $\underline{s} = (0,0) \stackrel{\text{def}}{=} \underline{0}$, the transmitter sends $m_0(r_1)$ copies of first message segment of the first destination group and $m_0(r_2)$ copies of the first message segment of the second destination group. Without loss of generality, we can assume that $m_0(r_1) \geq m_0(r_2)$ and that the sequence numbers of the two message segments in the random message are both one. Then in the first $m_0(r_2)$ time-slots, both the first segments will be transmitted together. In the next $(m_0(r_1) - m_0(r_2))$ time-slots, the first message segment of the first destination group will be transmitted with other message segments of the second destination group. As far as the recovery of the first message segments is concerned, the receivers in the set A_1' will be listening to all the $m_0(r_1)$ frames whereas the receivers in the set A_2' will be interested in only the first $m_0(r_2)$ frames. The receivers in the set A_{12} will try to recover the combined message from the first $m_0(r_2)$ time-slots and only the first message segment from the next $(m_0(r_1) - m_0(r_2))$. Those receivers in A_{12} that successfully receive the first message segment during the last $(m_0(r_1) - m_0(r_2)$ slots will then join the set A_2'. Then for all $(0 \leq r_1' \leq r_1,\ 0 \leq r_2' \leq r_2$ and $0 \leq r_{12} \leq r_1)$, we obtain

$$T^*(\underline{0},\ \underline{r}) = \min_{m_0(r_1)\ m_0(r_2)} \left[\sum_{i=0}^{r_1'} \sum_{j=0}^{r_2'} \sum_{k=0}^{r_{12}} \binom{r_1'}{i} w_0{}^i (1-w_0)^{r_1'-i} \right.$$

$$\left. \cdot \binom{r_2'}{j} x_0{}^j (1-x_0)^{r_2'-j} \binom{r_{12}}{k} x_0{}^k (1-x_0)^{r_{12}-k} \right.$$

$$\cdot \quad \sum_{l=0}^{r_{12}-k} \binom{r_{12}-k}{l} y_0{}^l (1-y_0)^{r_{12}-k-l}$$

$$\cdot \quad \{m_0(r_1).n_1/n + m_0(r_2).n_2/n + T^*(\underline{1},\, \underline{v})\}\,], \tag{5.25}$$

where

$$\begin{aligned} w_0 &= 1-(1-p)^{m_0(r_1)}, \\ x_0 &= 1-(1-p)^{m_0(r_2)}, \\ y_0 &= 1-(1-p)^{(m_0(r_1)-m_0(r_2)}, \\ \underline{1} &= (1,1), \\ \underline{v} &= (r_1'-i,\, r_2'-j+l,\, r_{12}-k-l). \end{aligned} \tag{5.26}$$

In state-$\underline{1}$, we would like to find the number of slots in which the two first message segments are transmitted together because those are the slots in which the broadcast nature of the channel is being exploited.

Let us assume that the first message segment of the first destination group has a destination set of r_{01} receivers of which r_{11} are yet to receive the message segment in state-1. We can define similar parameters r_{02} and r_{12} for the first message segment of the second destination group. If we sent out $m_0(r_{01})$ copies of the first message segment of the first destination group in state-0 then in the worst case, the first retransmission of this segment will begin $\lceil (N-1)/m_0(r_{01}) \rceil m_0(r_{01})$ slots after the first transmission has ended. Similarly the first retransmission of the first message segment of the second destination group will begin $\lceil (N-1)/m_0(r_{02}) \rceil m_0(r_{02})$ slots after the first transmission of $m_0(r_{02})$ copies has ended. The transmitter will then send $m_1(r_{11})$ and $m_1(r_{12})$ copies of these two segments respectively. Using all this

information, we can figure out the number of slots in which the two message segments will be sent out together.

However, there is an unfortunate consequence of this procedure. If the i^{th} $(i = 1, 2)$ message segment in state$-s$ has not been acknowledged by r_{si} receivers then the optimum number of copies for this state depends on the number of unsuccessful receivers in each of the previous states. For example, let a message segment started off with a destination set of 20 receivers. Say four receivers received the message successfully in the first transmission, three more in second, five more in third. Then we can write a sequence of the number of receivers that are yet to acknowledge the message in successive states. In state-3, we have a sequence $(20, 16, 13, 8)$. From the discussion above, we see that for each such sequence, there is a different optimal value of the number copies to be sent. In all the optimization processes done so far, the optimum number of copies in any state depended only on the number of receivers that had not received the message yet. In that sense, they were all memoryless. Due to this reason, this optimization process is complex both in terms of analysis and, more importantly, in terms of implementation. We would like to somehow make it insensitive to how we arrived at a given state.

One way to solve this dilemma would be to say that as soon as a message reaches state-1, we would use the channel as though it is time-shared. This might seem like a crude approach (because it is), but it is not as bad as it seems because of the following reason. For all the error rates over which ARQ schemes are a viable approach, the probability of a message reaching it's destination in it's first transmission is very high. For example, for $n = 1024$ and $\epsilon = 10^{-4}$, the message reaches it's destination after one copy with probability 0.9027. After two and three copies the probability is 0.99 and 0.999 respectively. Therefore, most of the advantages of the broadcast channel can be exploited within the first transmission. If we time-share the channel after the

each message segment reaches state−1 then we can write

$$T^*(\underline{s},\ \underline{r}) = \sum_i n_i/nT_i^*(s_i,\ r_i), \quad s_i \geq 1. \tag{5.27}$$

It appears, at this point, as though we have forgotten about states $(0,1)$ and $(1,0)$. But that is not true. As soon as either of the two message segments reach state−1, their progress is covered by the time-shared part of the analysis. Everything else is covered by the equation (5.25) above.

The throughput efficiency can be obtained by recursively solving for $T_i^*(m_0(R_i), s_i, r_i)$ for $s_i \geq 1$ and $0 \leq r_i \leq R_i$, using the procedure in section 5.5.2. Then we could use equation (5.25) to obtain $T_i^*(m_0(R_i), 0, r_i)$. The minimum over all values of $m_0(R_i)$ could be averaged over the random variables R_1, R_2 and R_{12} to obtain the throughput.

Computational Results

In figures 5.8 and 5.9 we show the throughput performance of non-ideal ARQ schemes for binomially distributed destination groups with means 5 and 10 respectively. For comparison purposes, we have also shown the corresponding ideal schemes, the schemes with equal length message segments and several cases of fixed destination sizes. We notice that the throughput of the case when the message segments are of same length is closer to the ideal case. The throughput of the non-ideal scheme can however be made arbitrarily close to that of the ideal case by increasing the buffer size.

In figures 5.10 and 5.11, we compare the throughput of finite-population non-ideal schemes with the time-shared cases for the two mean destination sizes. We see that the time-shared cases are handily outperformed by the non-time-shared cases, especially when the message segments are of same length.

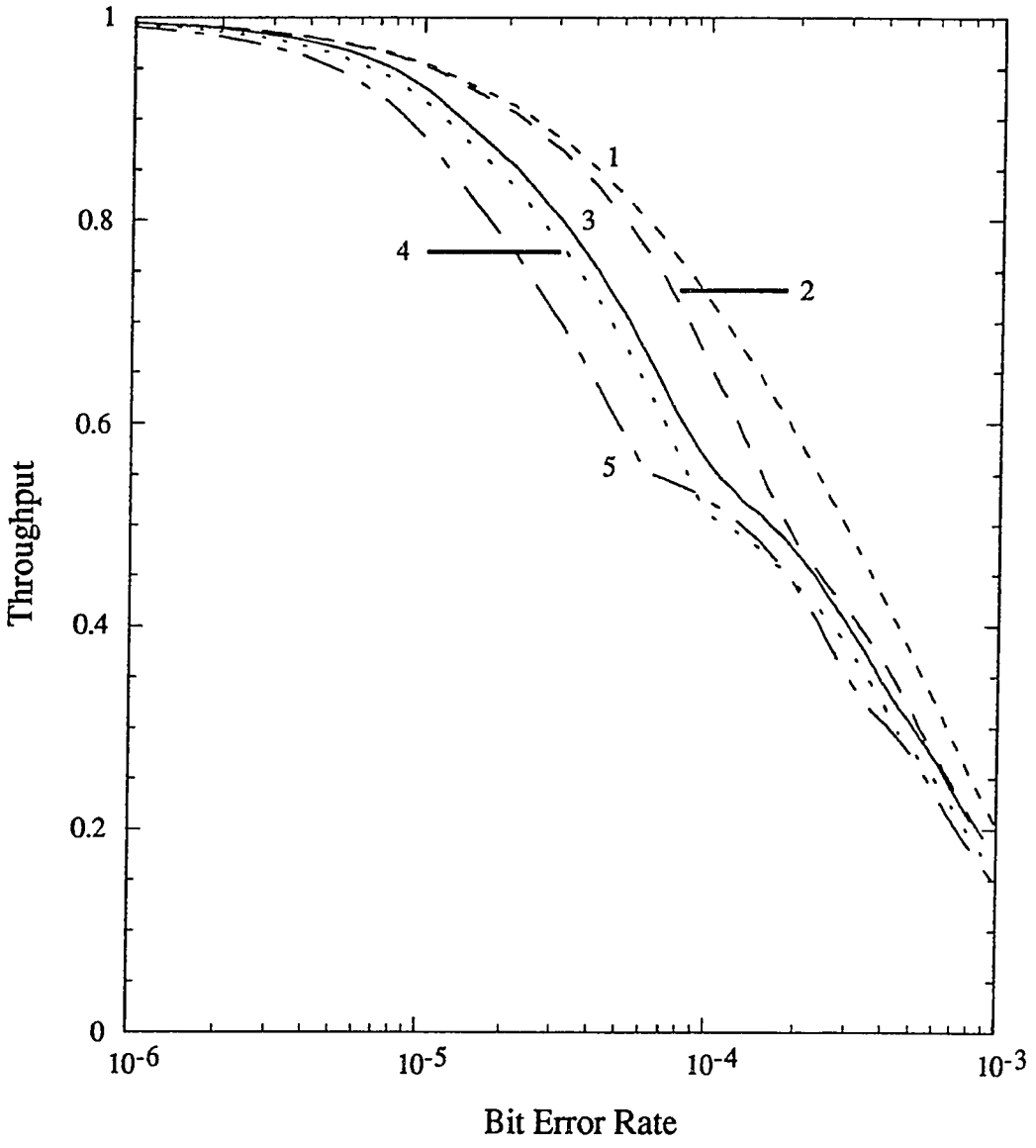

Figure 5.8: Performance of Finite Population Non-ideal Schemes (with $\lambda = 5$): (1)$P = 20$, Ideal. (2) $P = 20$, $n_1 = 500$, $n_2 = 500$.(3)$P = 20$, $n_1 = 300$, $n_2 = 700$ and $R_1 = 5$, $R_2 = 5$, $R_{12} = 2$. (4)$P = 20$, $R_1 = 5$, $n_1 = 700$, $R_2 = 10$, $n_2 = 300$, $R_{12} = 2$. (5) $P = 20$, $R_1 = 5$, $n_1 = 300$, $R_2 = 10$ $n_2 = 700$, $R_{12} = 2$.

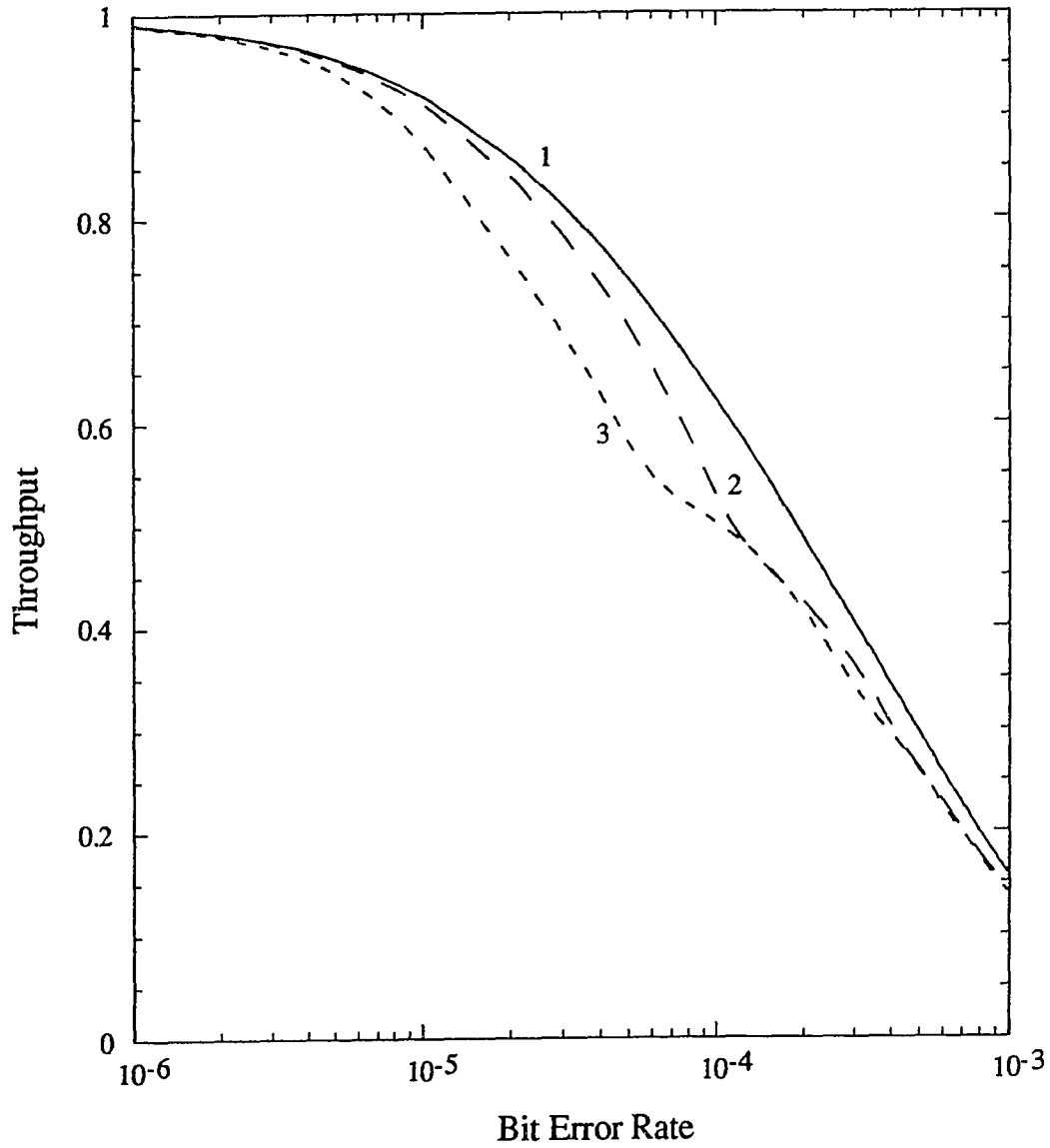

Figure 5.9: Performance of Finite Population Non-ideal Schemes (with $\lambda = 10$): (1)$P = 20$, Ideal. (2) $P = 20$, $n_1 = 500$, $n_2 = 500$ and $R_1 = 10$, $R_2 = 10$, $R_{12} = 5$. (3) $P = 20$, $n_1 = 300$, $n_2 = 700$.

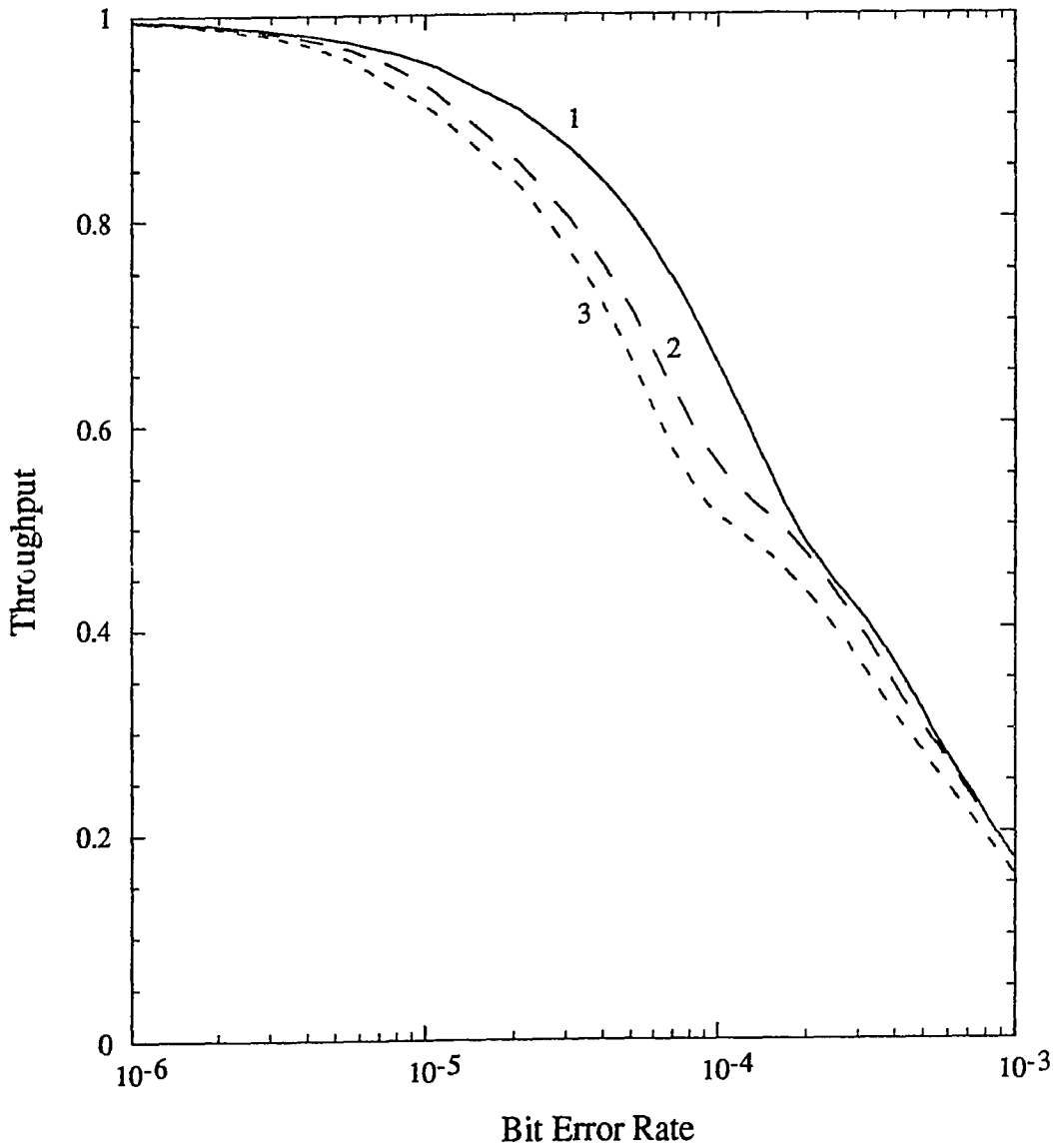

Figure 5.10: Performance Comparison of Time-shared and Non-time-shared Non-ideal Schemes (with $\lambda = 5$): (1) $P = 20$, $n_1 = 500$, $n_2 = 500$. (2) $P = 20$, $n_1 = 300$, $n_2 = 700$. (3) $q = 1$, $n_1 = 1000$.

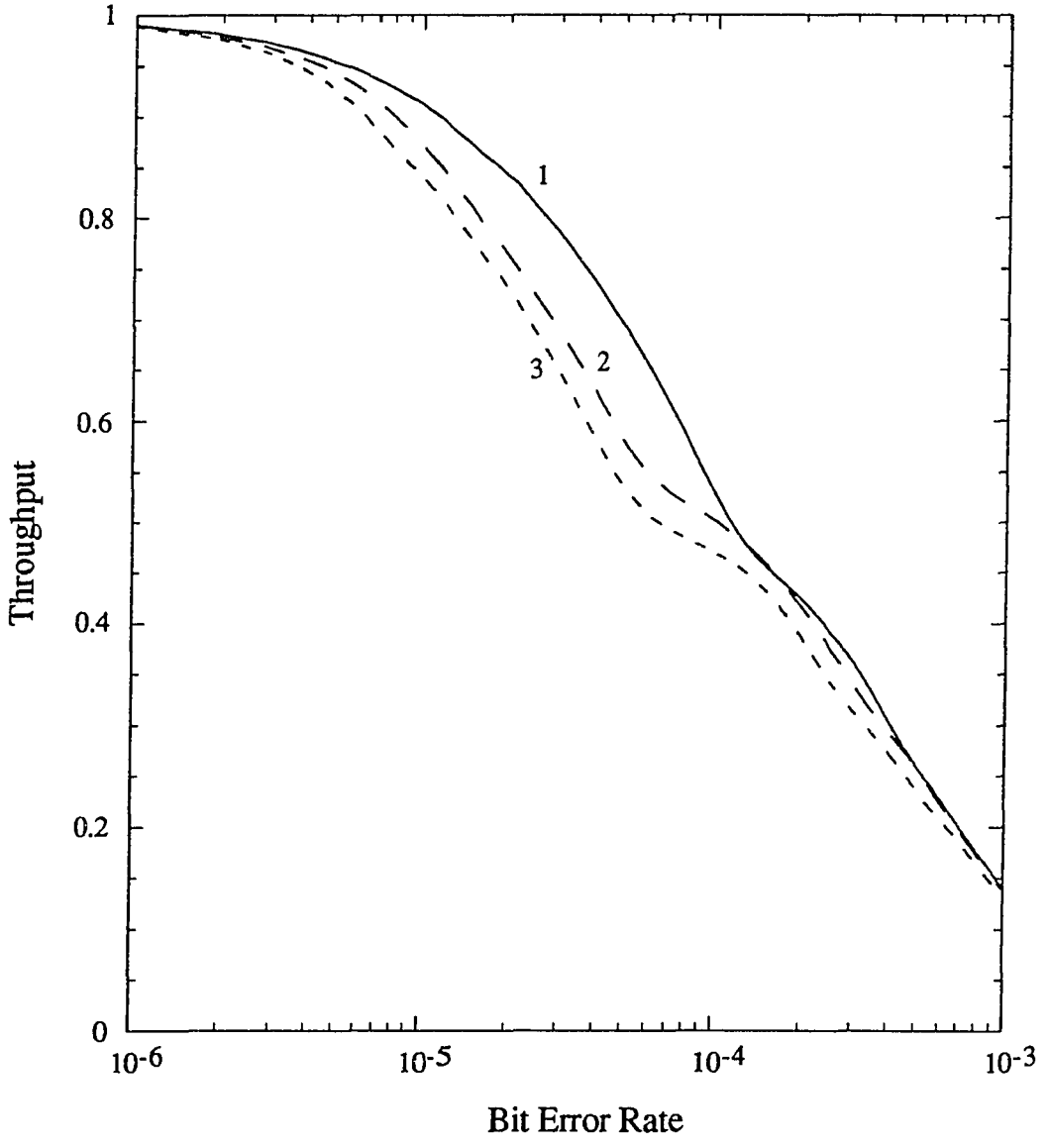

Figure 5.11: Performance Comparison of Time-shared and Non-time-shared Non-ideal Schemes (with $\lambda = 10$): (1) $P = 20$, $n_1 = 500$, $n_2 = 500$. (2) $P = 20$, $n_1 = 300$, $n_2 = 700$. (3) $q = 1$, $n_1 = 1000$.

The Single Copy Case

What if we send only a single copy of each message segment in each transmission or retransmission regardless of the number of receivers ? This case bears looking into because of two reasons. For almost all the channels where ARQ schemes are feasible, single copy is perhaps the optimum choice. Also, for the single copy case, we can obtain bounds on throughput that are much tighter than the one in the last section.

The difficulty we had in analyzing the non-ideal case (in the last section) was due to the mismatch between the number of times the two message segments were being sent out. That difficulty has been eliminated in this case. The analysis, therefore, is similar to that of the finite population ideal scheme (section 5.6.1) except that here we will have to account for buffer overflow. Let T_{12} and T_i $(i = 1, 2)$ be the same as that defined in section 5.6.1. Using a procedure similar to that for equation (2.16), we obtain

$$T_{12}(r_1, r_2, r_{12}) = \sum_{i=0}^{\infty}[1 - \{1 - (1-p)^i\}^{r_{12}}] + \sum_{i=0}^{\infty}\lambda_i[1 - \{1 - (1-p)^i\}^{r_{12}}], \qquad (5.28)$$

where

$$\begin{aligned} \lambda_i &= 0, && i < \delta_{12}. \\ &= (N-1), && i \geq \delta_{12}. \qquad (5.29) \\ \delta_{12} &= (\beta + 1). && \qquad (5.30) \end{aligned}$$

The value of λ_i $(i = \delta_{12})$ can be found from equations (2.3 - 2.6). Therefore,

$$T_{12} = \mathbf{E}_{R_1 R_2}[\mathbf{E}_{R_{12}}[T_{12}(r_1, r_2, r_{12})]]. \qquad (5.31)$$

The quantities T_i $(i = 1, 2)$ can be obtained in a similar manner.

$$T_i(r_1, r_2, r_{12}) = \frac{n_i}{n}\left\{\sum_{j=0}^{\infty}[1 - \{1 - (1-p)^{T_{12}+j}\}^{r_i - r_{12}}\right.$$

$$\left. + \sum_{j=0}^{\infty} \nu_{ij}[1 - \{1 - (1-p)^{T_{12}+j}\}^{r_i - r_{12}}\}\right], \qquad (5.32)$$

where ν_{ij} is the number of message segments lost due buffer overflow for the i^{th} destination group when the segment is in state–j. This quantity can be obtained as follows.

$$\begin{aligned} \nu_{ij} &= 0, && j < \delta_i. \\ &= \lceil(\lceil\gamma\rceil - \gamma)(N-1)\rceil, && j = \delta_i. \\ &= (N-1), && j > \delta_i. \end{aligned} \qquad (5.33)$$

where the quantities B_i, γ_i and δ_i are the same as in section 5.5.2.

After averaging the above quantity, the total average number of time slots required to successfully deliver a message to all the receivers, T, can be obtained from equation (5.19). The throughput will then be the inverse of that quantity.

Computational Results

Figures 5.12 and 5.13 show the throughput of single-copy case for the two mean destination sizes. We have also compared the results with the ideal case and the case when the buffer size $\beta = 3$. The curves show that the single copy case gives good throughput only for $\epsilon \leq 10^{-5}$. However, with a larger buffer size, the throughput performance of this scheme is almost identical to that of the ideal scheme for $\epsilon \leq 2\text{x}10^{-4}$. This scheme is an important special case because almost all the selective

retransmission schemes in practice (e.g., TCP/IP based protocols) send out only single copies. This scheme gives the performance we can expect if we use single copies in a broadcast environment.

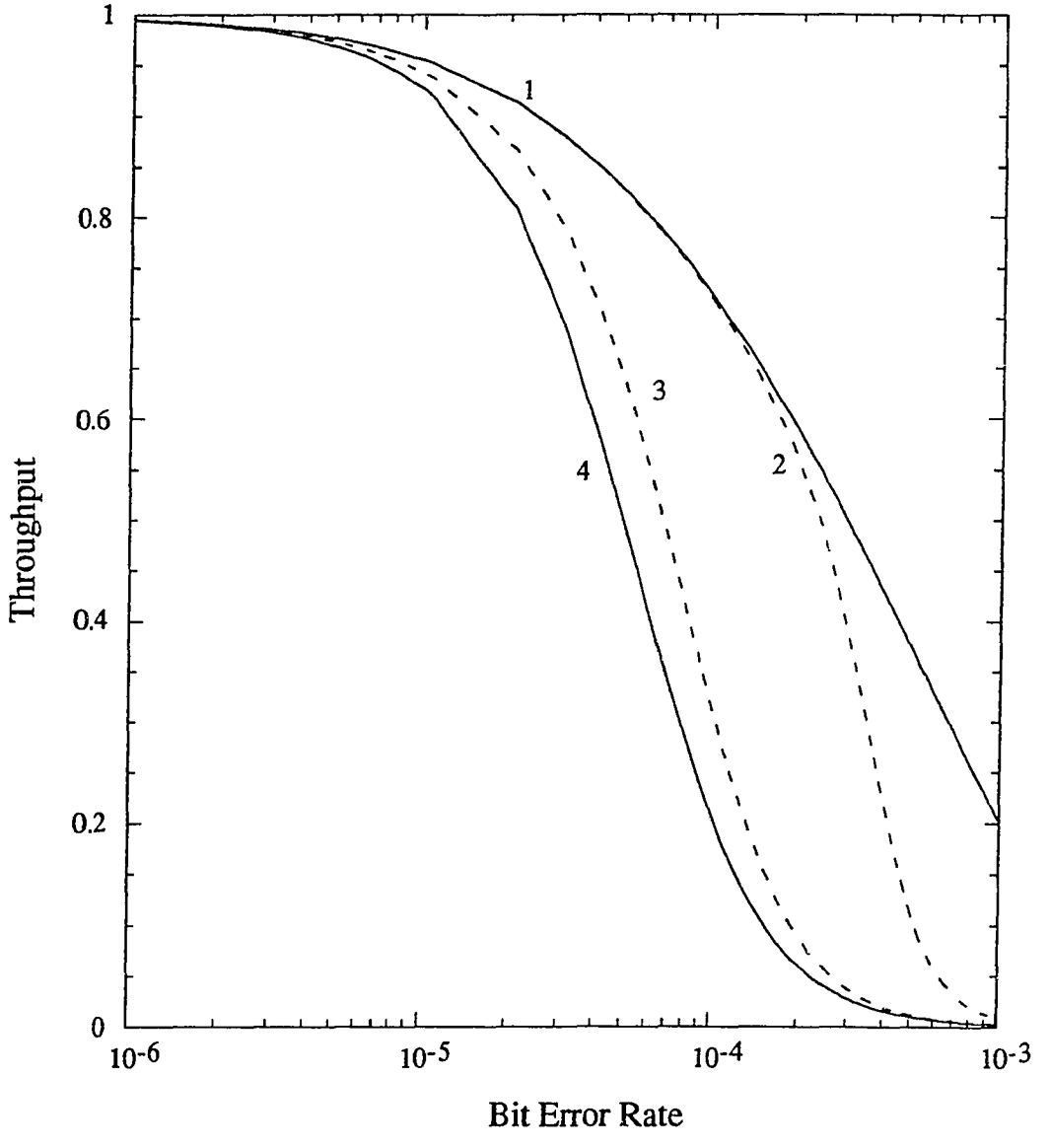

Figure 5.12: Performance Single-copy Non-time-shared Schemes (with $\lambda = 5$): (1) Ideal, $P = 20$. (2)$P = 20$, $n_1 = 500$, $n_2 = 500$, $\beta = 3$. (3)$P = 20$, $n_1 = 500$, $n_2 = 500$. (4)$P = 20$, $n_1 = 700$, $n_2 = 300$.

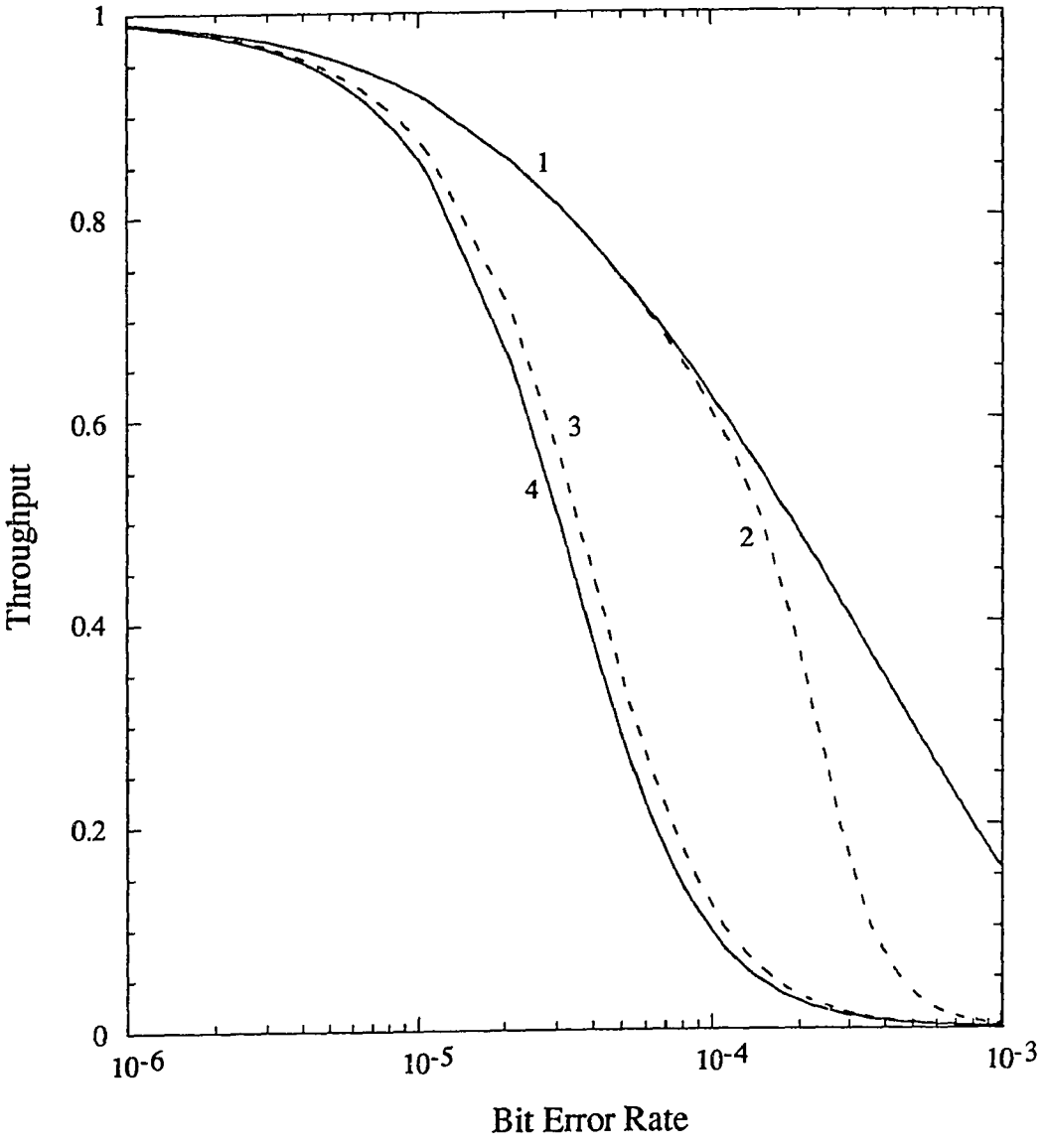

Figure 5.13: Performance Single-copy Non-time-shared Schemes (with $\lambda = 10$): (1) Ideal, $P = 20$. (2)$P = 20$, $n_1 = 500$, $n_2 = 500$, $\beta = 3$. (3)$P = 20$, $n_1 = 500$, $n_2 = 500$. (4)$P = 20$, $n_1 = 700$, $n_2 = 300$.

Chapter 6

Conclusions

In this dissertation, we have investigated several error-control schemes for reliable transmission of data over broadcast channels. The schemes proposed here make use of a feedback channel to improve reliability. We have considered two types of broadcast channels: Homogeneous (HB) and Non-homogeneous (NHB). Recent work in this area has been almost exclusively for the HB channel. So far, almost no work has been done to investigate the error-control problem for the NHB channel.

We have proposed a selective repeat ARQ scheme for the HB channel where each receiver has a finite buffer to store messages. The protocol was motivated by a desire to simplify the transmitter operation. Our analysis shows that the proposed scheme achieves that goal and also improves throughput. We have used this protocol together with the Lin-Yu coding scheme to obtain a type-2 hybrid ARQ scheme which gives significant improvement in throughput performance when the channel noise becomes high.

In Chapter 4, we have proposed a very robust error-control system using cascaded coding. Our analysis shows that this scheme can provide arbitrarily high reliability even when the channel is extremely noisy ($10^{-3} \leq \epsilon \leq 10^{-2}$). The coding scheme has the added feature of parity retransmission. We have proposed a hybrid ARQ scheme for the HB channel based on this scheme. This ARQ scheme is a combination of type-1 and type-2 hybrid ARQ. The ARQ scheme provides very high throughput

over channels with high bit-error-rates. This scheme can be used for high speed file transfer over very noisy channels.

The NHB channels constitute a more realistic model of the modern day data networks. We have investigated the error control problem for this channel in chapter 5. The main conclusion we draw from our study is that the channel utilization can be improved significantly if we exploit the broadcast nature of the channel. In case when the broadcast population is finite, the successive destination groups are mutually overlapping therefore the throughput efficiency can be improved by combining messages intended for various destination groups as opposed to the time-sharing of the channel. However, in the limiting case when the broadcast population is infinite, this advantage is lost because of the low overlap among various destination groups. It would appear that time-sharing is a better alternative for the infinite population case because it is less complex to implement. If the channel is noisy, we will have to use forward-error-correction to achieve the required reliability. The use of hybrid schemes in the NHB channel has not been investigated here. The schemes proposed here can be used in conjunction with UEP/multi-level codes to improve the reliability of communication and also achieve bandwidth compression. If we use coding then non-time-shared schemes will have an added advantage over the time-shared schemes because better rates can be obtained from the use of unequal-error-protection codes. This added advantage can be realized even if the destination groups are non-overlapping. Moreover, the use of UEP codes give us the flexibility to use different levels of protection to different messages. This is the added benefit of non-time-shared schemes that was hinted at earlier in chapter 5.

This work has laid down a systematic procedure for the characterization and analysis of the error-control issues for broadcast channels. We hope that the methodology

developed here will be helpful for future work in this area. We conclude this dissertation with a brief discussion of some problems for future work.

6.1 Future Work

As mentioned in the last section, the use of forward-error-correction and hybrid schemes for the NHB channel is a candidate for future research. There has been considerable progress in the areas of UEP codes and multi- level coded-modulation schemes. The use of these codes will give good reliability and yield higher throughput in the NHB channel.

All the work (including this work) in the area of error-control for broadcast channels till now has been done by considering the broadcast channel as a separate entity. In view of the fact that the broadcast channel almost always occurs in tandem with multiple-access channel (see chapter 1), there is a need for a more unified approach in this area. Our understanding of both these forms of communication has improved enormously over the past two decades therefore we are in a position to combine these two problems to see if there is any advantage to be gained.

Appendix

Derivation of $p_{\overline{C}}^{\frac{1}{2}}(u)$

In this appendix, we derive a more general result. Let H be a subset of $\{1,2,3,\cdots,\rho\}$. Let $p_{\overline{C}}^{\frac{1}{2}}(H)$ be the probability that for $h \in H$, the h^{th} l-bit byte of a $C_{\frac{1}{2}}$ decoded subframe is error-free. We now obtain an expression for $p_{\overline{C}}^{\frac{1}{2}}(H)$. Let $\overline{H}$ be the complement of H in $\{1,2,3,\cdots,2\rho\}$. Let

$$C(H) = \{(i_1, i_2, \cdots, i_{2\rho}) : i_h = 0 \text{ for } h \in H, 0 \leq i_h \leq l \text{ for } h \in \overline{H}\}. \tag{.1}$$

Then we have

$$p_{\overline{C}}^{\frac{1}{2}}(H) = \sum_{(i_1, i_2, \cdots, i_{2\rho}) \in C(H)} A_{i_1, i_2, \cdots, i_{2\rho}}^{\frac{1}{2}} \sum_{j_1=0}^{l} \sum_{j_2=0}^{l} \cdots \sum_{j_{2\rho}=0}^{l} \tag{.2}$$

$$\cdot \sum_{(s_1, s_2, \cdots s_{2\rho}) \in S_{t_1}} \left[\prod_{h=1}^{2\rho} W_{j_h, s_h}^{(i_h)}(l)\right] (1-\epsilon)^{2r_1} \left[\prod_{h=1}^{2\rho} \left(\frac{\epsilon}{(1-\epsilon)}\right)^{j_h}\right], \tag{.3}$$

where

$$S_{t_1} = \{(s_1, s_2, \cdots s_{2\rho}) : 0 \leq s_h \leq l \text{ for } 1 \leq h \leq 2\rho, \text{ and } \sum_{h=1}^{2\rho} s_h \leq t_1\}. \tag{.4}$$

Bibliography

[Cover 72] Thomas Cover, "Broadcast Channels," IEEE Trans. on Information Theory, vol. IT-18, No. 1, pp. 2-14, January 1972.

[Stal 87] William Stallings, "Handbook of Computer Communicaitons Standards," vol. 1, Macmillan 1987.

[Lin-Yu 82] S. Lin and P. S. Yu, "A Hybrid ARQ Scheme with Parity Retransmission for Error Control of Satellite Channels," IEEE Trans. Comm., vol. COM-30, No. 7, pp. 1701-1719, July 1982.

[Mase 83] K. Mase, T. Takenaka, H. Yamamoto and M. Shinohara, " Go-back-N ARQ Schemes for Point-to-Multipoint Satellite Communications," IEEE Trans. Comm., vol. COM-31, No. 4, pp. 583-589, April 1983.

[Gopal 84] Inder S. Gopal and Jeffrey M. Jaffe, "Point-to-Multipoint Communication Over Broadcast Links," IEEE Trans. Comm., vol. COM-32, No. 9, pp. 1034-1044, Sept. 1984.

[Tows 85] D. Towsley, "An Analysis of Point-to-Multipoint Channel Using a Go-back-N Error Control Protocol," IEEE Trans. Comm., vol. COM-33, No. 3, pp. 282-285, Mar. 1985.

[Sabnani 82] K. K. Sabnani, "Multidestination Protocols for Satellite Broadcast Channels," PH.D. dissertation, Columbia University, New York, 1982.

[Sabnani 85] K. Sabnani and M. Schwartz, "Multidestination Protocols for Satellite Broadcast Channels," IEEE Trans. Comm., vol. COM-33, pp. 232-240, Mar. 1985.

[Chandran-Lin 86] S. Ram Chandran and S. Lin, "A Selective Repeat ARQ Scheme for Point-to-Multipoint Communications and it's Throughput Analysis," Computer Communications Review, vol. 16, No. 3, pp. 292-301, Aug. 1986.

[Chandran-Lin 88] S. Ram Chandran and S. Lin, "A Selective Repeat Hybrid ARQ Scheme for Point-to-Multipoint Communications," Porceedings of the IEEE International Symposium on Information Theory, Kobe, Japan, June, 1988.

[Chandran-Lin 90] S. Ram Chandran and S. Lin, " Optimum Selective Repeat ARQ Schemes for Broadcast Links," Proceedings of the IEEE International Symposium on Information Theory, San Diego, CA, January, 1990.

[Chandran-Lin 92] S. Ram Chandran and S. Lin, "Selective Repeat ARQ Schemes for Broadcast Links," To appear in IEEE Transactions on Communicaitons, January, 1992.

[Tow-Mit 87] D. Towsley and S. Mithal, "A Selective Repeat ARQ Protocol for a Point-to-Multipoint Channel," Proc. IEEE Infocom., San Francisco, CA, pp. 521-526, Mar. 1987.

[Wang-Sil 87] J. L. Wang and J. A. Silvester, "Throughput Optimization of the Adaptive ARQ Protocols for Point-to-Multipoint Communication," Tech. Report CSI-87-12-01, Communications Sciences Institute, Dept.

of Electrical Engineering-Systems, University of Southern California, Dec. 1987.

[M-Q-R 88] S. Mohan, J. Qian and N. L. Rao, "Efficient Point-to-Point and Point-to-Multipoint Selective Repeat ARQ Schemes with Multiple Retransmissions: A Throughput Analysis," Proc. ACM SIGCOMM Symposium, pp. 49-57, 1988.

[Yu-Lin 81] P. S. Yu and S. Lin, "An Efficient Selective Repeat ARQ Scheme for Satellite Channels and It's Throughput Analysis," IEEE Trans. Comm., vol. COM-29, No. 3, pp. 353-363, Mar. 1981.

[Go-Rom 90] Inder Gopal and Raphael Rom, "Multicasting to Multiple Groups over Broadcast Channels," Submitted to IEEE Trans. on Comm., 1990.

[Weldon 82] E. J. Weldon, Jr., "An Improved Selective Repeat ARQ Strategy," IEEE Trans. Comm., vol. COM-30, No. 3, pp. 480-486, Mar. 1982.

[Chang 84] Y. Chang and C. Leung, "On Weldon's ARQ Strategy," IEEE Trans. Comm., vol. COM-32, No. 9, pp. 297-300, Mar. 1984.

[Lin-Cos 83] S. Lin and D. J. Costello, Jr., "Error Control Coding: Fundamentals and Applications," Prentice-Hall, 1983.

[Mandel 74] D. M. Mandelbaum, "Adaptive-Feedback Coding Scheme Using Incremental Redundancy," IEEE Trans. Infrom. Theory, vol. IT-20, pp. 388-389, May 1974.

[Metz 79] J. J. Metzner, "Improvements in Block Retransmission Schemes," IEEE Trans. Comm., vol. COM-27, pp. 525-532, Feb. 1979.

[Anch 79] T. C. Ancheta, "Convolutional Parity Check Automatic Repeat Request," presented at IEEE Int. Symposium on Inform. Theory, Grignano, Italy, June 1979.

[Lin-Ma 79] S. Lin and J. S. Ma, "A Hybrid ARQ System with Parity Retransmission for Error Control of Satellite Channels," IBM Res. Rep. 7478(32232), Jan. 11 1979.

[Kasami 88] T. Kasami, T. Fujiwara, T. Takata and S. Lin, "A Cascaded Coding Scheme for Error Control and It's Performance Analysis," IEEE Trans. Information Theory, vol. 34, No. 3, pp. 448-462, May 1988.

[Mac 63] J. MacWilliams, "A Theorem on the distribution of weights in a systematic code," Bell Syst. Tech. J., vol. 42, pp. 79-94, 1963.

[Mac-Slo 77] F. J. MacWilliams and N. J. A. Sloane, "Theory of Error-Correcting Codes," Amsterdam, The Netherlands: North Holland, 1977.