Adaptive FEC/ARQ Schemes for Stream Media Multicast over the Internet

Dong Hui Chen

A Thesis
in
The Department
of
Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science at
Concordia University
Montréal, Québec, Canada

June 2002

Abstract

Adaptive FEC/ARQ Schemes for Stream Media Multicast over the Internet

By

Dong Hui Chen

Thesis supervisor: Dr. Ahmed Elhakeem

A growing number of network applications require the use of a reliable multicast protocol to send stream media from a source to a potentially large number of receivers. This work introduces a new technique that integrates word Interleaving, Forward Error Correction (FEC) and Automatic Repeat reQuest (ARQ) to increase reliability. Our laboratory work showed the practical feasibility of a software implementation of the hybrid FEC/ARQ and interleaving scheme. We also present a comparative performance analysis of the hybrid FEC/ARQ scheme and the ARQ only scheme and show the influence of varying sending rates, group sizes, and packet loss probabilities. We evaluate the performance of our hybrid FEC/ARQ scheme and adapt redundancy factor to cope with a high loss rate and large-scale receivers. We demonstrate how appropriate amount of redundancy combined with coding decreases the bandwidth overhead, eliminates NAK implosion, reduces the residual error rate, and helps to meet real-time guarantees.

Acknowledgments

Many Thanks

For Critique, Advice, Comments and Support:

My Supervisor Dr. A.K. Elhakeem

For Support and Encouragement:

Dr. Michel Kadoch

Special Thanks to:

Nasser Shayan, Feng Yan, Shaopeng Zhu,

Maria Bennani, and Bernard Tremblay

Table of Contents

List of Figures

List of Tables

Abbreviations

AAP	Multicast Address Allocation Protocol
ACK	Acknowledgment
ARQ	Automatic Repeat reQuest
BGMP	Border Gateway Multicast Protocol
CRC	Cyclic Redundancy Check
DR	Domain Receiver
DVMRP	Distance Vector Multicast Routing Protocol
EIGRP	Enhanced Interior Gateway Routing Protocol
FEC	Forward Error Correction
IGMP	Internet Group Management Protocol
MADCAP	Multicast Address Dynamic Client Allocation Protocol
MASC	Multicast Address-Set Claim Protocol
MBGP	Multiprotocol BGP (BGP4+)
MBone	Multicast Backbone
MOSPF	Multicast Extensions to OSPF
MSDP	Multicast Source Discovery Protocol
NAK	Negative Acknowledgment
OSPF	Open Shortest Path First
PIM	Protocol Independent Multicast
RIP	Routing Information Protocol
RTT	Round Trip Time
RS	Reed-Solomon codes
RTCP	Real-Time Control Protocol
RTP	Real-Time Transport Protocol
SAP	Session Announcement Protocol
SDP	Session Description Protocol
TPDU	Transport Protocol Data Unit
UDP	User Datagram Protocol

Chapter 1

Introduction

In recent years, Internet Service Providers have started to offer Multicast solutions to their IP customers. Currently, digital techniques in audio/video processing are widely used and stream media one-to-one (unicast) IP applications are getting popular. Some new services, like the Internet conference, video on demand and the Internet TV, can be offered to a large number of customers with an acceptable performance only by combining with IP Multicast. However, current best-effort multicast IP service does not match well with reliable stream media multicast transport. The challenge is how to transmit multicast packets to a large number of receivers within a bounded amount of time. Our research introduces a new technique that integrates word Interleaving, Forward Error Correction (FEC) and Automatic Repeat reQuest (ARQ) to reduce the latency of reliable delivery of data to multiple receivers and to reduce feedback implosion.

1.1 Scope and objectives

The Aims of our project are as follow:

- Allow a sender to transmit a long stream of data to a large number of users over the Internet.

- Assure reliability by FEC and ARQ.
- Provide time-bounded delivery with low delay, low delay jitter and ordering guarantee
- Dynamic fault detection and recovery
- Limit the sender's complexity and the load placed on it, as it could be a bottleneck.
- Do not require changes to the underlying Internet.
- Scale to thousands of receivers over a wide area.
- Simplicity and wide applicability

Due to the time-bound and the scalability requirement, our goals do not include full data reliability.

We need to implement some ideas in order to perform real experiments on the network. And, we also need to analysis the performance on multicast scalability, which cannot be tested in our lab yet. An important aspect of our analysis is to be of practical relevance.

1.2 Thesis organizations

In chapter 2, we start by describing the problem of reliable multicast for large groups of receivers. We continue by giving some detail on the principle of basic schemes to achieve reliability.

In chapter 3, we propose a simple hybrid FEC/ARQ mechanism to allow trade-off between reliability with performance and scalability in a reliable stream media multicast environment. Some performance test results are shown in this chapter.

In chapter 4, we present a comparative performance analysis of the hybrid FEC/ARQ scheme and an ARQ only scheme. We evaluate the effect achieved by our schemes in terms of the average number of transmissions, average number of NAKs generated and expected residual errors. We also show the influence of varying sending rates, group sizes, and packet loss rate.

In chapter 5, we discuss an adaptive media stream multicast policy that extends scalability. When the group size is large and packet loss rates are high, we try to hybrid the two-copy scheme with our FEC/ARQ scheme in order to meet a fixed deadline for reliable delivery. We determine the bandwidth overhead that results from ARQ retransmissions, and demonstrate that having the sender send two-copy FEC encoded packets proactively can decrease the bandwidth overhead, eliminate NAK implosion, reduce the residual error rate, and save one Round Trip Time (RTT) at most receivers. The way to choose schemes and the way to suppress NAKs are also discussed in this chapter.

We give conclusions, and suggest implementation-oriented details for a possible future work in the last chapter.

Chapter 2

Stream Media IP Multicast

A growing number of network applications require the use of IP multicast to deliver audio/video streams from a source to a potentially large number of receivers. Examples for applications are stored video dissemination (like the transmission of TV programs) and interactive multimedia applications.

A brief introduction of issues in stream media IP multicast is presented in this chapter.

2.1 What is IP multicast

Multicast is much like radio or TV in the sense that only those receivers who have chosen a particular channel receive the information.

IP multicast [1, 2] is an internetwork service that allows IP datagrams sent from a source to be delivered to multiple interested recipients. That is, a given source sends a packet to the network with a multicast destination address, and the network transports this packet to all receivers that have registered their interest in receiving these packets. The sender does not need to maintain a list of receivers. Only one copy of a multicast message will pass over any link in the network, and copies of the message will be made

only where paths diverge at a router. Thus IP Multicast yields many performance improvements and conserves bandwidth end-to-end.

2.1.1 Unicast's disadvantage

When the communication is only between two entities, usually one sender and one recipient, such communication is called unicast. Sometimes, information is of interest to multiple recipients. Unicasting a copy of the data to each recipient may be very inefficient. For instance, when the sender connects to the Internet over a single path. Unicasting multiple copies over that single path creates duplication, wasting bandwidth. If each unicast connection across this shared path requires a fixed amount of bandwidth, there is a hard limit to the number of receivers the network can support. Establishing hundreds, even thousands of these connections sending audio/video, both the sending computer and the network would collapse.

2.1.2 Multicast VS. broadcast

Broadcast seems to be a solution, but sometimes it's not. With broadcast, all hosts on the network receive a copy of the message. A multicast message is sent to some subset of all the hosts on the network.

The decision of using broadcast or multicast in an application depends on several issues, including the portion of network hosts interested in receiving the data and the knowledge of the communicating parties. Broadcast works well if a large percentage of the network hosts wish to receive the message; however, if there are many more hosts than receivers, broadcast is very inefficient. In the Internet, broadcasting

would be very expensive even if the communication had a large number of interested receivers because the data would have to be duplicated to every host on the Internet while the hosts' number would be well over the number of interested receivers.

Unlike broadcast, network multicast duplicates the message only to a specific set of receivers. A multicast group address identifies this set of receivers, called a multicast group. These receivers need some mechanism to notify the network of the interest in receiving data sent to a particular multicast address. Once notified, the network can begin forwarding the multicast messages to the receiver. This notification request is called "joining a group."

Because of the negative consequences of Internet-wide broadcast, most routers do not forward broadcast packets; thus, broadcast applications are generally limited to LAN broadcasts only.

2.1.3 The MBone

The main restrict of multicast is that hundreds of hosts and routers don't support it yet. Multicast islands are difficult to communicate without the MBone. The MBone is a virtual multicast network on the top of the Internet. That is: sites with multicast routers between them could communicate directly. But sites joined across unicast routers would send their island's multicast traffic encapsulated in unicast packets to other multicast islands. In the receiving site, traffic would be de-encapsulated, and sent to the island in the original multicast way [3].

The MBone provides an infrastructure for multicast communication on the Internet.

2.1.4 Multicast addresses.

The range of IP addresses is divided into "classes" based on the high order bits of a 32 bits IP address. Multicast Address is the "Class D Address" which starts with "1110" (range 224.0.0.0 - 239.255.255.255). The remaining 28 bits identify the multicast "group" the datagram is sent to.

2.1.5 Sending multicast datagrams.

Multicast traffic is handled at the transport layer with UDP, In principle, an application just needs to open a UDP socket and fill with a class D multicast address the destination address where it wants to send data to.

2.1.6 Receiving multicast datagrams.

A process needs to join a multicast group to advise the network which multicast groups it is interested in. When a process is no longer interested in a multicast group, it informs the network that it wants to leave that group.

There may be many processes on the host joining the same multicast group. The process joining a multicast group only tells the IP and data link layer to accept multicast datagrams destined to that group. It is not a per-process membership, but a per-host membership.

2.1.7 Multicast protocols

Today's IP multicast protocol components is shown in figure 2.1. Multicast protocols are advancing fast [4, 5].

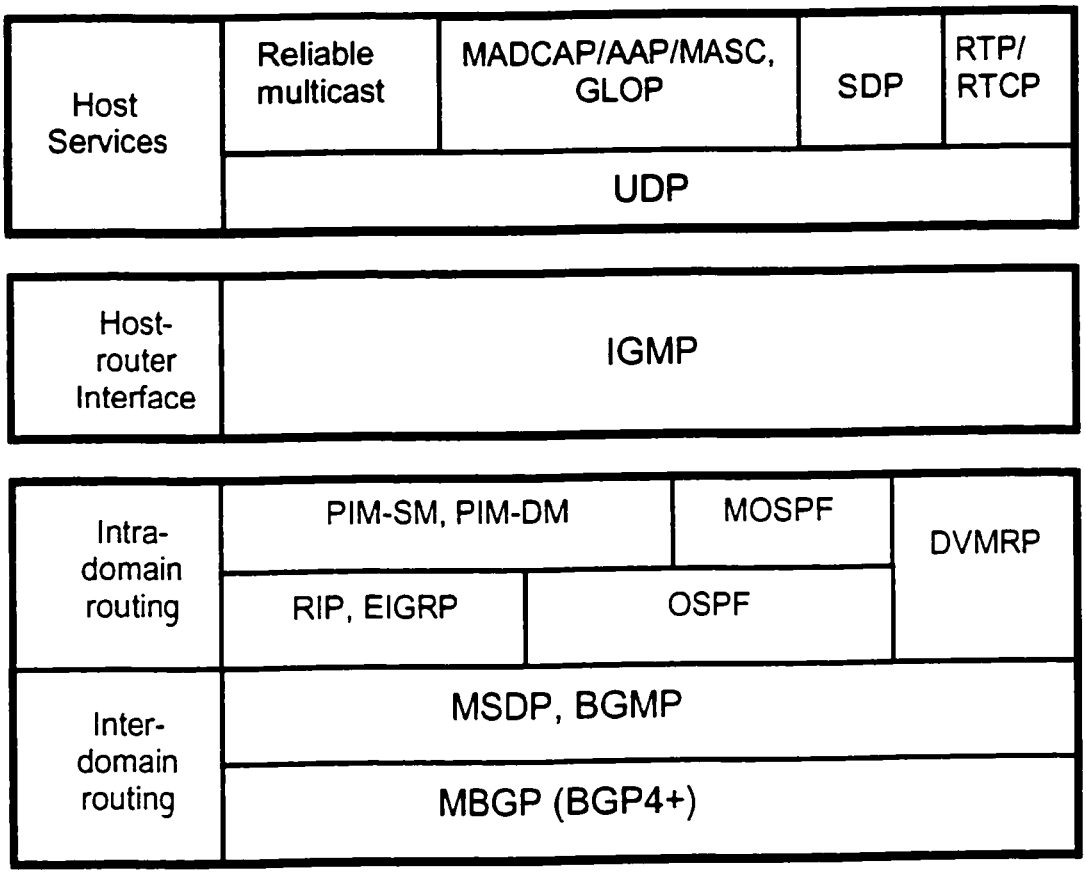

Figure 2.1 Protocol components for IP multicast

2.2 Reliable stream media IP multicast

For IP, only UDP sockets are allowed to multicast. Because data transmitted by UDP sockets are normally subject to delay, delay jitter, out of order, and loss of packets, reliable IP multicast protocols need to be built on top of the UDP layer. Several of these protocols have been implemented and are being tested. But most of these protocols are limited to fully reliable services and ignore real-time aspects. Reliable IP multicast

delivering stream media is still problematic. Mechanisms for IP multicast real-time error control are discussed in this section.

2.2.1 Stream media requirements

Before discussing the stream media IP multicast, Let's go over the general feature of stream media like video and audio:

- Strict timing: If the data packets are delivered later then a certain deadline they are useless.
- Tolerance to small loss: The amount of loss that can be tolerated depends on the application.

When transmitting stream media across IP network, due to the delay, delay jitter, out of order, and loss of packets enrolled, mechanisms are needed to correct the impairments. Because of the hard limit of timing, fully reliable multicast transmission mechanism seems impossible.

A reliable real-time multicast transport service reduces the remaining error probability within a given delay budget. Examples using a reliable real-time multicast transport service with a single source are audio-visual conferencing, as well as dissemination of audio and video streams. Real-time applications mostly have specific requirements, which, for a specific application scenario, frequently make certain protocol mechanisms more suitable than others.

Additionally, application requirements also differ in maximum end-to-end delay. Delay requirements usually vary over less than two orders of magnitude. Depending on whether the application is interactive in nature or not, the delay requirements differ.

Interactive applications, such as audio or videoconferences, cannot tolerate a delay of more than a few hundred milliseconds [6]. Non-interactive applications can tolerate much higher delays in the order of a second or more, since this delay is only noticeable as start-up delay, but is transparent after start of the play-out.

Typical delay requirements for audio-visual applications are:

- Tight (up to 200 ms);
- Medium (200 to 500 ms);
- Loose (larger than 500 ms).

Due to this high variety of required application-specific data unit size and requirements for delay and reliability, it is difficult to design error recovery protocols that are well suited for general applications. Instead, solutions are typically targeted for specific applications.

The main reason for the packets loss, delay and jitter encountered in the Internet is due to congested routers. Applications, especially real-time multicast applications, should take good care of this feature of the Internet to achieve good performance. We should not overload the network. Networks can provide low latency when the requested bandwidth is well below that which can be delivered. Figure 2.2 shows a typical network congestion behavior.

In this work, our target applications basically are video streams that are compressed using the compression standards H.261 or MPEG. The application data unit sizes are medium to long; data rates are about 1.5 Mbps and a delay budget of 500 ms to 5 s.

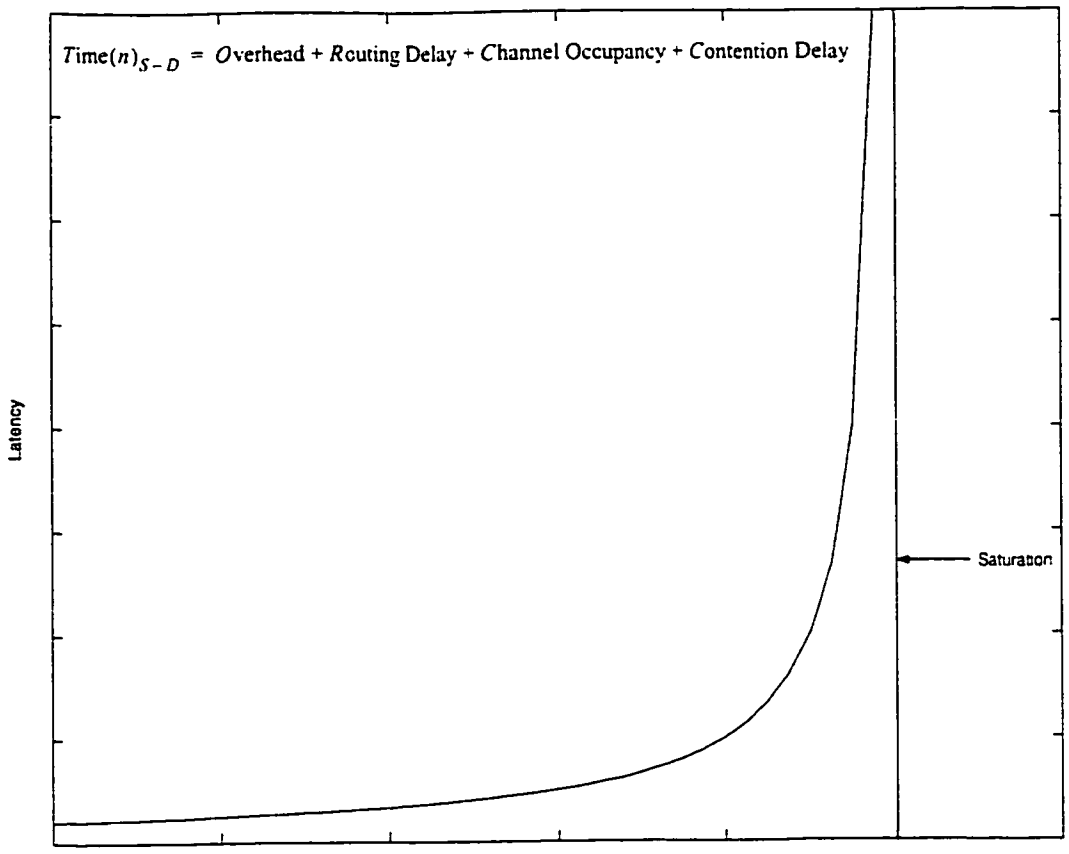

Figure 2.2 Typical network congestion behavior

2.2.2 Mechanisms for error recovery

The basic mechanisms available to recover from the loss or corruption of data packets are ARQ and FEC.

2.2.2.1. Automatic Repeat reQuest (ARQ)

Using ARQ, a lost TPDU will be retransmitted by the sender. ARQ schemes consist of three parts:

- Lost data detection: Loss can be detected by the receiver (gap-based loss detection or timeout) or by the sender (timeout).
- Acknowledgment: The receiver sends either positive ACK or negative ACKs referred to as NAKs.

- Retransmission: The two well-known retransmission schemes are Go-Back N and selective retransmission, which trade off simplicity of the receiver implementation and transmission efficiency.

ARQ based multicast protocols can be classified into sender-initiated and receiver-initiated protocols. Sender-initiated reliable multicast protocols based on the use of ACKs can suffer performance degradation as the number of receivers increases. This degradation is due to the fact that the sender must maintain state information and timers for each of the receivers and respond to receivers' ACKs. A potential solution to this problem is to shift the burden of providing reliable data transfer to the receivers - thus resulting in receiver-initiated multicast error control protocols based on the use of NAKs. In addition, the number of NAKs needed is generally much less then the number of ACKs. Researches have shown that receiver-initiated error control protocols provides higher throughput than sender-initiated protocols [7, 8].

2.2.2.2 Forward Error Control (FEC)

The idea of FEC is to transmit original data together with some "parities" to allow reconstruction of lost packets at the receiver. The redundant data is derived from the original data using techniques from coding theory [9-13]. The data stream is transformed in such a way that reconstruction of a data object does not depend on the reception of specific data packets, but only on the number of different packets received. There are multiple benefits to using the parity for loss recovery instead of retransmitting the lost packets:

Improved transmission efficiency: A single parity packet can be used to repair the loss of any one of the data packets. This means that a single parity packet can repair the loss of different data packets at different receivers.

Improved scalability in terms of group size: When a lost packet is retransmitted via multicast, the packet will be received more than once by all of the receivers that have already successfully received the packet. Such duplicate packets waste transmission bandwidth and processing capacity. Using FEC can significantly reduce the necessity for retransmission requests or make them totally unnecessary, which is also important in delay-sensitive multimedia applications [14, 15].

Another point on FEC error control scheme is its wide availability: FEC implemented in software is reasonably fast on today's desktop PCs. More details on a soft Reed-Solomon encoder/decoder are presented in the following chapter.

2.2.2.3 Hybrid error control (ARQ/FEC)

FEC and ARQ apply to different application scenarios (Table 2.1). Studies show that integrating FEC and ARQ can improve multicast transmission efficiency and scalability [16-20]. A major difficulty when using FEC is to choose the right amount of redundancy in face of changing network conditions. Also, sending redundant data consumes additional bandwidth. In order to overcome this problem, ARQ and FEC can be used in combination. In many cases, proactive repair can reduce feedback implosion and the expected delay of reliable delivery without increasing overall bandwidth usage. A hybrid FEC/ARQ scheme is given in detail in chapter 3, and an adaptive proactive factor scheme is given in chapter 5.

Table 2.1 Applied Cases for FEC and ARQ

FEC	ARQ
- Large groups	- Small groups
- Large RTTs	- Heterogeneous loss probability
- Feedback impossible or undesired	- Loss on shared links dominant
- Individual loss dominant	- Less-interactive applications
- Homogeneous loss probability	
- Limited buffer	
- Interactive applications	

2.2.2.4 Hybrid FEC/ARQ system model

Figure 2.3 shows the hybrid FEC/ARQ system model.

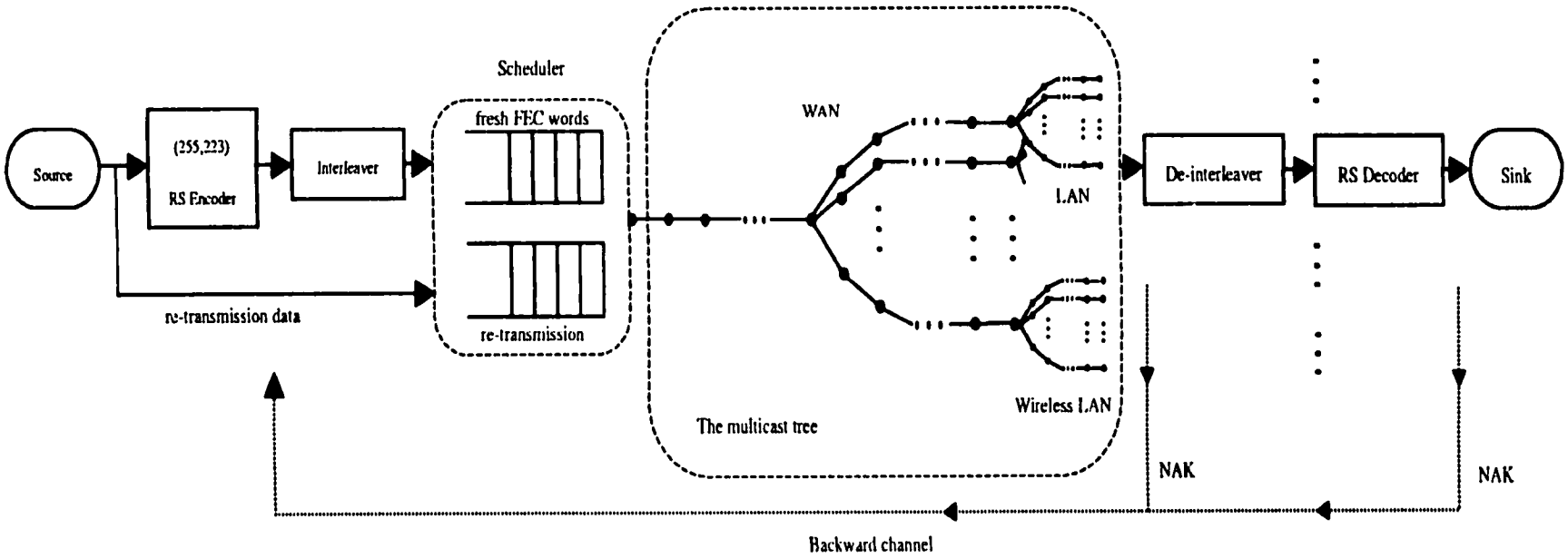

Figure 2.3 The stream media multicast system model

Chapter 3

Hybrid FEC/ARQ

This chapter presents the detailed technical information concerning our on-going hybrid FEC/ARQ multicast implementation issues.

3.1 Reed-Solomon codes

The need for reliable high-speed transmission of video, audio and data causes applications of erasure and error control coding to become omnipresent. Reed-Solomon (RS) codes are robust symbol oriented erasure and error correction codes. RS codes provide good burst loss recovering capability while maintaining well ability to correct random errors. RS are codes of practical interest in the error-control coding process [21, 22]. For instance RS is used in just about every proposal for digital TV transmission in Europe and the US. Here after is an explanation of the concepts of generating this code, and using it for encoding data in a way data loss and errors can be recovered.

3.1.1 RS codes theory

The theory of Reed-Solomon code is based on finite field theory. In particular, the fields used are of the form $\mathbf{GF(q^m)}$, where $\mathbf{q}$ is any prime number and $\mathbf{m}$ is any

positive integer. Our concern is the RS code on $GF(2^m)$. The elements of $GF(2^m)$ are defined by a power series format, i.e. α^0, α, α^2,.... The message length **k** can be any positive integer smaller than **n**, where **n** is the codeword length.

The basic parameters of RS code are:

- Codeword length: $n = 2^m - 1$
- Number of check symbols: $n-k = 2 \cdot t$
- Error-correction capability: $t = \text{floor}((n-k)/2)$
- For an efficient RS code, **n-k** should be an even number.
- The generator polynomial for RS code is a degree **2t** polynomial with its coefficients in $GF(2^m)$.

Suppose we have a set of **k** data packets $\{ M_{k-1}, M_{k-2}, \cdots, M_0 \}$ each of which is **m** bits long. The RS encoder takes $\{ M_{k-1}, M_{k-2}, \cdots, M_0\}$ and produces a set $\{C_{2t-1}, C_{2t-2}, \cdots, C_0 \}$ of packets each **m** bits long called parities. We also use the parameter **r** to denote

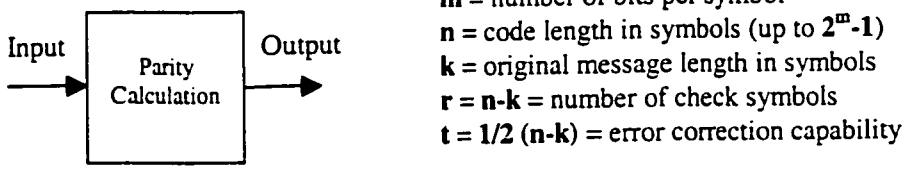

Figure 3.1 Reed-Solomon encoder

the number **n-k** of parities. For the purpose of coding, we consider the data packets $\{M_{k-1}, M_{k-2}, \cdots, M_0\}$ as elements of the Galois field $GF(2^m)$ [11] and define the polynomial **M(x)** as $M_{k-1}x^{k-1} + M_{k-2}x^{k-2} + \cdots + M_1x^1 + M_0$.

If $G(x) = (x-\alpha^0)(x-\alpha^1)\cdots(x-\alpha^{2t-1})$ is the field generator polynomial, where α is a root of the primitive polynomial, the RS encoder computes the check polynomial:

$C(x) = x^{2t}\, M(x) \bmod G(x) = C_{2t-1}x^{2t-1} + C_{2t-2}x^{2t-2} + \cdots + C_1x^1 + C_0$. Then, the encoded codes is $\{ M_{k-1}, M_{k-2}, \cdots, M_0, C_{2t-1}, C_{2t-2}, \cdots, C_0 \}$.

The codes generated this way which include the source symbols in clear is called systematic codes. Systematic codes count much less processing overhead to decode when only a few erasures are expected; besides, they might allow partial reconstruction of data even when fewer than **k** packets are available. The receivers without RS decoder might receive the data.

In case of symble loss, the RS decoder at the receiver side can reconstruct the data packets $\{ M_{k-1}, M_{k-2}, \cdots, M_0\}$ whenever it has received any **k** out of the **n** packets $\{ M_{k-1}, M_{k-2}, \cdots, M_0, C_{2t-1}, C_{2t-2}, \cdots, C_0 \}$. The **k** data packets will also be referred to as a transmission group. The **n** packets $\{ M_{k-1}, M_{k-2}, \cdots, M_0, C_{2t-1}, C_{2t-2}, \cdots, C_0 \}$ will be referred to as an FEC block. Sending the original data as the first **k** packets of the FEC block simplifies decoding:

- If all first **k** data packets are received, no decoding at all is required at the receiver.
- If $\mathbf{l < n-k}$ out of the **k** data packets are lost, the decoding overhead is proportional to $\mathbf{l}$.

data[x]	Received vector	**deg λ**	Number of errors+erasures
n_eras	Number of erasures	**ω [x]**	Error-erasure evaluator polynomial
eras_pos [i]	Erasures vector	**loc[i]**	Error locations
s[i]	Syndrome polynomial	**Y[i]**	Error magnitudes
λ [x]	Error+erasure locator polynomial	**data'[x]**	Recovered code word

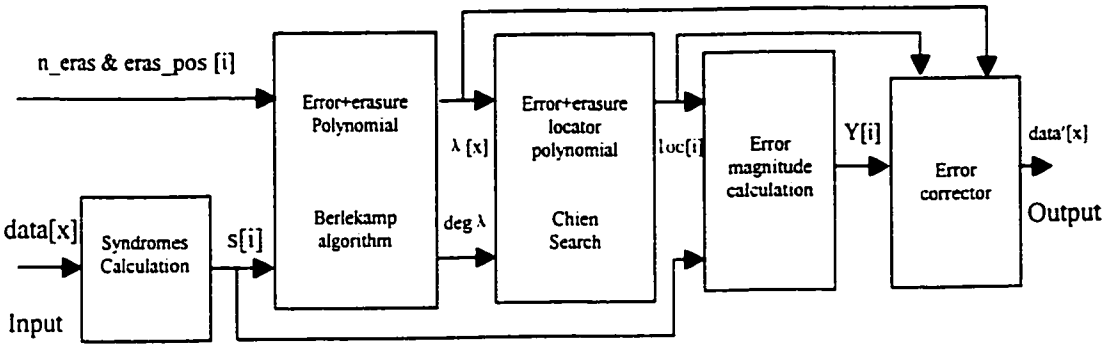

Figure 3.2 RS decoder architecture

Figure 3.2 illustrates the main stages of the decoder. It starts with the received codeword **data(x)** and goes on to output the recovered codeword **data'(x)** assuming there were **n_eras + 2·n_errors ≤ 2t** erasures and errors. The first step in decoding a received message **data(x)** is to compute its syndromes. Next, it will solve for the error locator and error evaluator polynomials using Berlekamp-Massey Algorithm. After that, it want to compute the error locations **loc[i]**. The Chien's algorithm will help it to search in an efficient way. Then, it just needs to compute the error magnitudes. Finally it can compute the codeword **data'(x)** using the formula **data'(x) = data(x) + E(x)** and the decoding is complete.

3.1.2 RS codes PC software

We use an open source general purpose Reed-Solomon encoding and decoding software in our project. The software may be used under the terms of the GNU public license. The authors in turn are Simon Rockliff, Robert Morelos-Zaragoza, Hari Thirumoorthy, and Phil Karn.

The code is optimized to run on PC by means of using Pentium CPU hardware pipelines, using lookup table to do matrix multiplying and dividing, and using a feedback shift register with appropriate connections specified by the elements of **G(x)** to encoding the source.

Table 3.1 RS Software Specification

Code type	RS block code
Symbol size	8 Bits
Codeword size (n)	255 Bytes
Data bytes per codeword (k)	223 Bytes
Parity bytes per codeword(r)	32 Bytes
Maximum burst error correction capability	128 Bits (r/2)
Maximum erasure error recover capability	32 Bytes (r)

To minimize the IP network overhead, we need a larger sized block code. But while the code size grows, the computation complexity increases very fast. To overcome large block code speed problem, a interleaving technique is adopted. This technique is introduced in next section.

3.1.3 Codes Interleaving

Interleaving is a technique that allows constructing a big block code out of a smaller one. i.e., given a (n, k) block code, we can construct a (L*n, L*k) block code. Interleaving schemes includes matrix, random, algebraic, and helical scan interleaving. A matrix interleaver block is used here. The interleaver allocates a block of memory as a 2-dimensional matrix. The encoded symbols of each block are then written into memory in rows. Next, they are read out and transmitted in columns. Finally, a corresponding de-interleaver uses the inverse mapping to restore the original sequence of symbols at receivers.

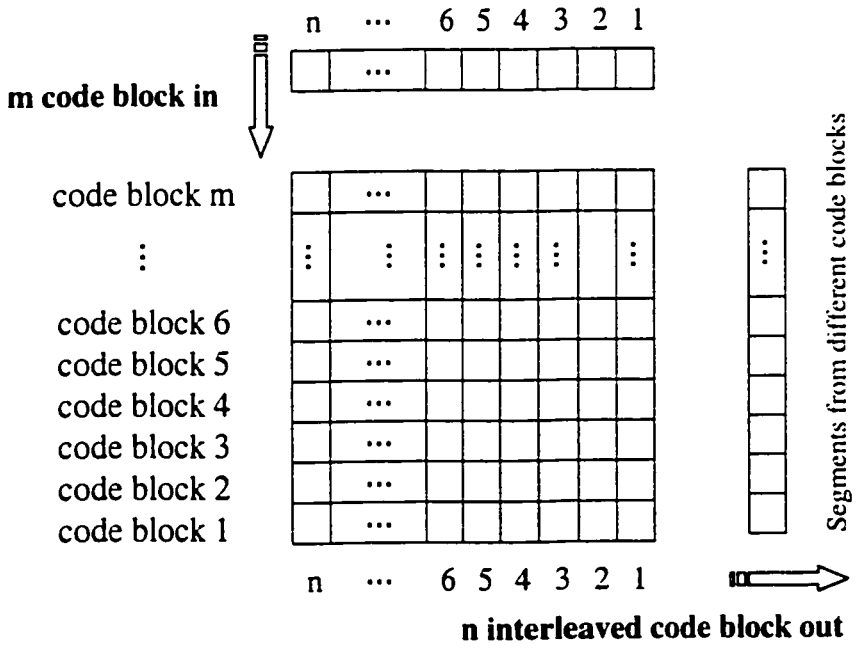

Figure 3.3 Interleaving process

Interleaving and de-interleaving can be useful for reducing errors caused by burst errors or losses in a communication system. A burst now must be more than n symbols long to affect more than one symbol in each code block.

3.2 Description of the loss and error recovery techniques

The main entities of the new techniques are the server and client algorithms. These are applications that reside on top of the Internet UDP layer. The server routine typically exists at the sender of the video or audio multicast session or one of the intermediate routers (called domain receiver DR in multicast terminology). The client routine exists within the receiver part of DR or the end user [23].

The frames transmitted from the sender (first sender or intermediate DR) are shown in Figure 3.4. The figure shows the application multicast data encapsulated within the frames transmitted from the sender. Each block of data, 217 bytes is given a sequence number in the range 1 to 65536, The sequence number field in the corresponding word is preceded by a type filed, so far we have only two designations where 11111111 and 10101010 denotes a fresh or retransmitted words respectively. A one byte data length DL field is also needed to indicate to the receiving end possible padding of data less than 217 bytes, in which case the remainder of the 217 bytes is filled with padding. A two bytes Cyclic Redundancy Check (CRC) field is also formed at the end of the word. This CRC operates over the whole word.

The resulting 223 bytes are fed to the (255,223) Reed Solomon encoder routine, which outputs a corresponding 255 bytes that we call an encoded word. The RS code is assumed to be systematic in which case the 32 parity bytes are added by the encoder at the end of the word. A one byte of dummy data is added to obtain a 256 bytes encoded word.

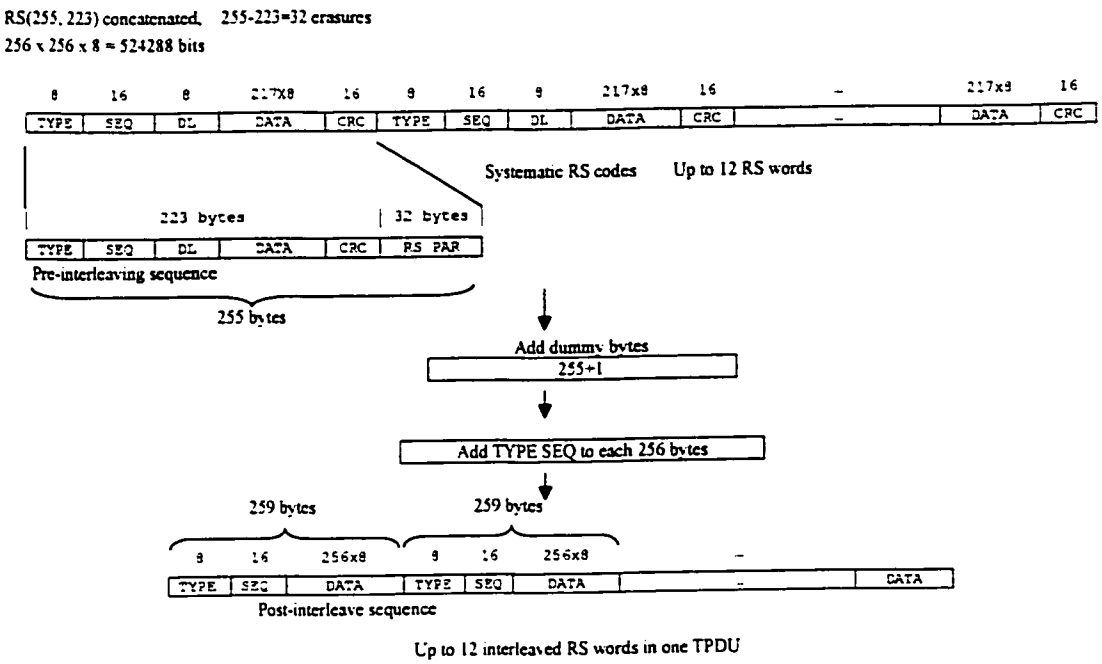

Figure 3.4 TPDU format

Interleaving is next applied to a record of 256 such encoded RS words, to yield a corresponding interleaved record, The resulting first interleaved word of such record consists of the first byte of each of the pre-interleaving words. The 256th interleaved word is the similarly the concatenation of bytes number 256 of each pre-interleaving word.

It is also possible to have a smaller depth interleaving e.g. 128 in which case the first interleaved word will encompass the 128 first bytes of the first 128 pre-interleaving words followed by the128 bytes of the second bytes of each of the 128 pre-interleaving words. The first 128 bytes of the 64th interleaved word are bytes numbered 255 and the second 128 bytes are bytes numbered 256 in the original pre-interleaved words.

If the interleaving depth is 256, one can easily see that up to 32 consecutive interleaved words can be lost, yet the receiver will not ask for any corresponding retransmissions.

Because of interleaving, if these 32 words were lost, only 32 out of the 255 bytes of each of the corresponding pre-interleaved words would be lost. Because RS decoders can recover Up to **255 – 223 = 32** lost bytes, which would be considered as erasures. Such loss is not detrimental and will not result in retransmission requests. For error recovery of multicast traffic over the Internet, minimizing the need for retransmissions is an essential asset, which subsequently leads to minimizing NAK and repair packets implosions [10, 16, 24, 25].

Returning back to Figure 3.4, the server routine groups up to 12 interleaved words in one UDP data unit, i.e. TPDU. A 2 bytes interleaved sequence number field is added to each interleaved word, this is preceded by a one byte field denoting an interleaved word, i.e. 11111111. Recall that each original RS encoded word carries a **type** and **sequence** fields, but these are scrambled following interleaving. So the need arises for a new type and sequence fields as above. Occasionally and upon request of the client, retransmitted words are also included in some TPDUs , a 4 bytes dummy filed is added to each such retransmission so as to make the total (255+4=259 bytes) equal to that of the fresh (i.e. interleaved) data.

The constant length of 259 bytes, the type, and the sequence, fields of each word will enable correct parsing and Processing at the client side.

If an interleaving depth of 256 is used, each TPDU has 8 interleaved words, then 4 consecutive TPDUs can be lost without any retransmissions.

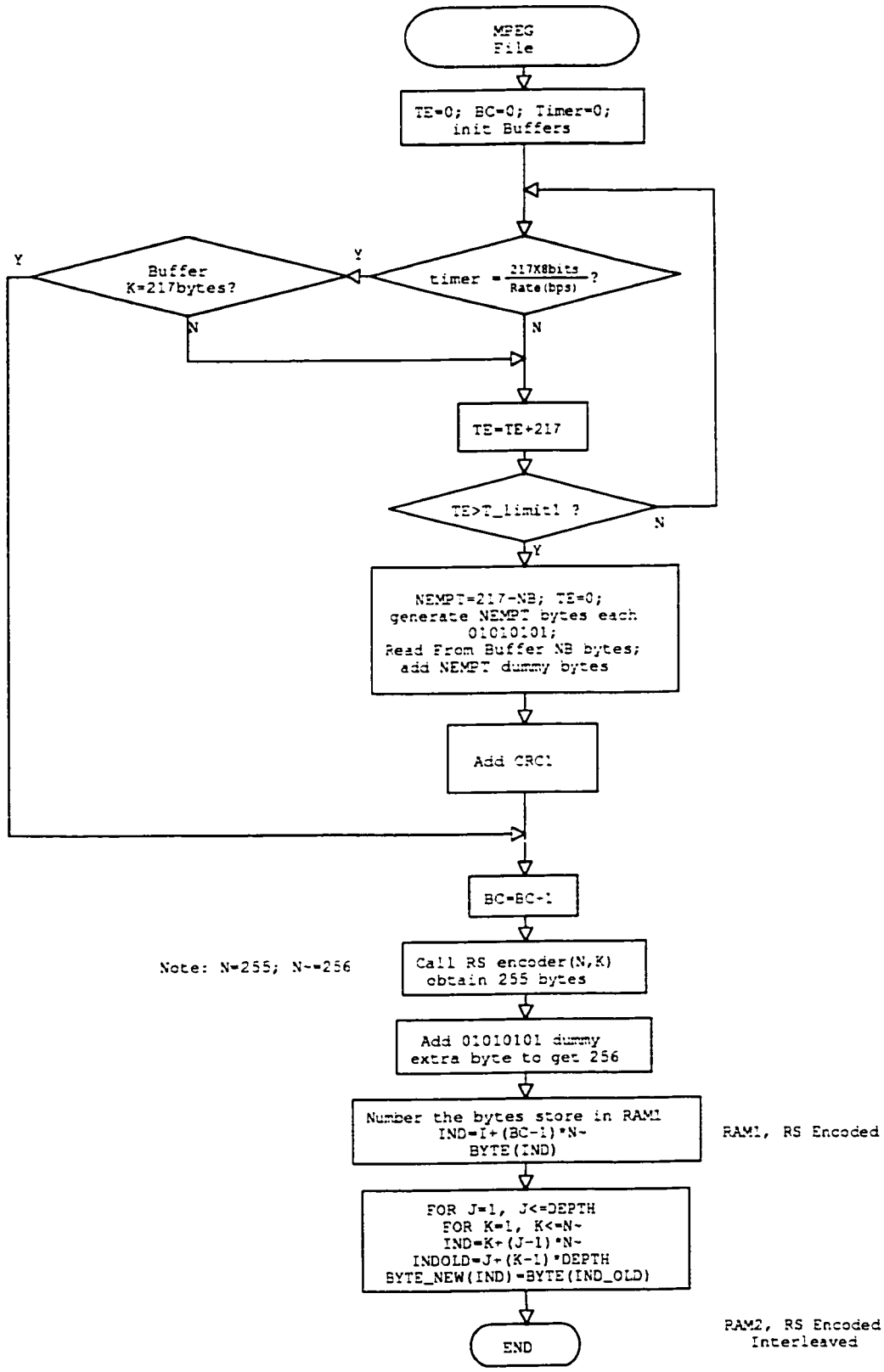

Figure 3.5a Source realtime encoding at server

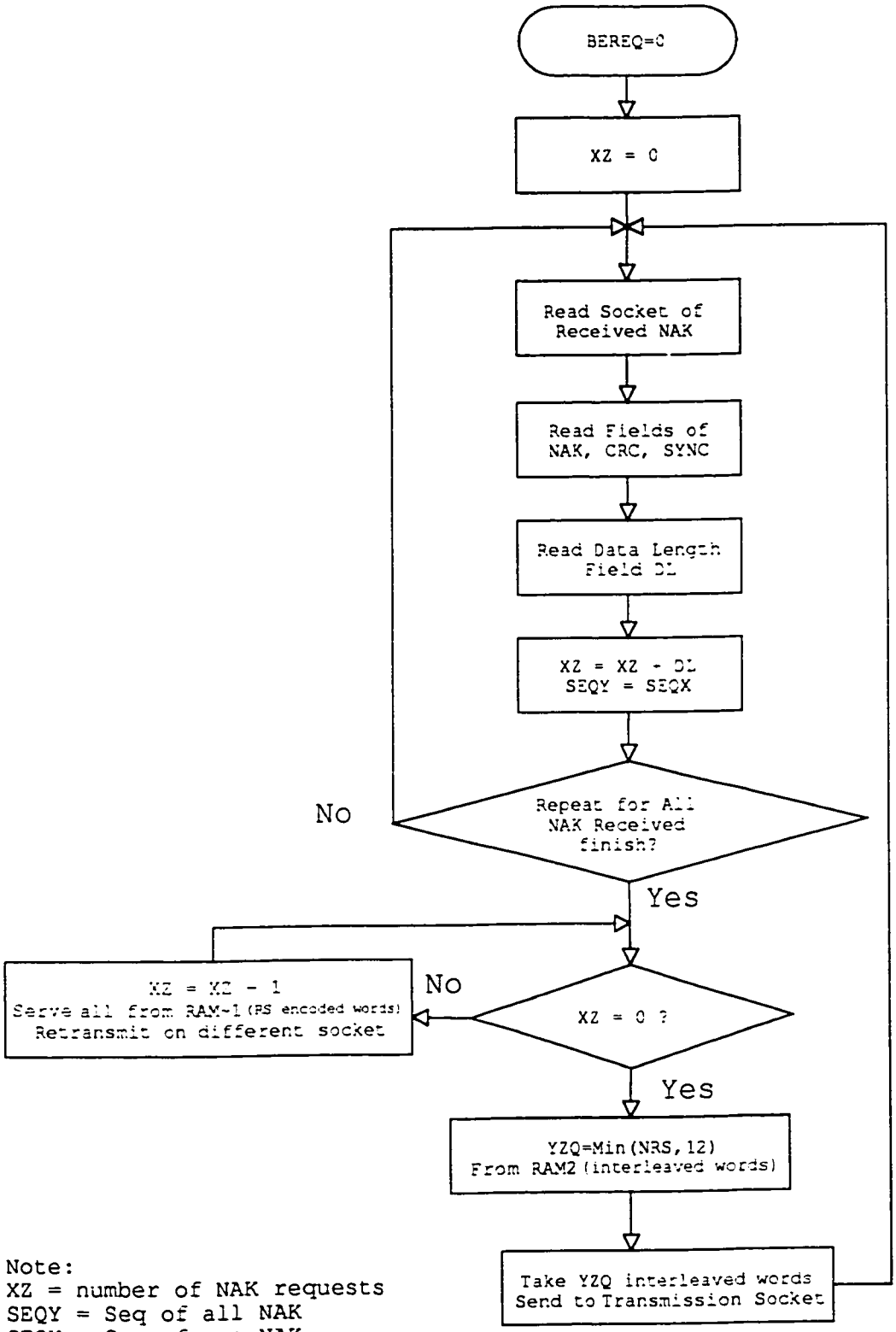

Figure 3.5b Transmission duty cycle at server

Figure 3.5.a shows the details of byte processing of the Multicast file to be transmitted, as outlined above, while Figure 3.5.b underlines the processes of formulating repairs (i.e. retransmission) packets, and how to combine them with new interleaved Words.

Much of the processes above are done offline in multicast applications, i.e. the whole file, etc. is FEC encoded, interleaved as above and stored before starting the multicast in real time. The process of mixing repairs and interleaved words has to be executed in real time though once the multicast session starts.

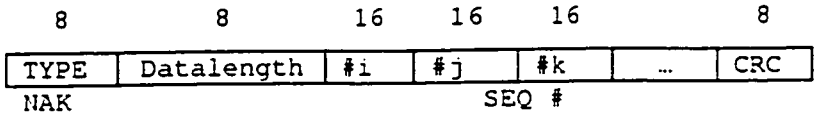

Figure 3.6 NAK packet format

Before describing Figure 3.5.b, the Negative Acknowledgment NAK frame format is shown in Figure 3.6. Recall, for our Multicast application at hand, this is the lonely kind of packet that the client may transmit. However, for video teleconferencing data may also be retransmitted from the client to the server. In Figure 3.6, we include a type field, i.e. 01010101 to denote NAK messages. A one byte field denotes the data length of the NAK message in bytes. The subsequent fields give the RS encoded sequence numbers of the lost words as asked by the client. A final one byte CRC field is used for error checking at the server.

The client routine appears in Figure 3.7 which shows the processing that takes place once a record of data is read from the UDP layer and the corresponding socket.

Blocks of 259 bytes are sampled from such record. If the type field indicates a repair word, and if further CRC indicates correct reception, the first 4 bytes are stripped

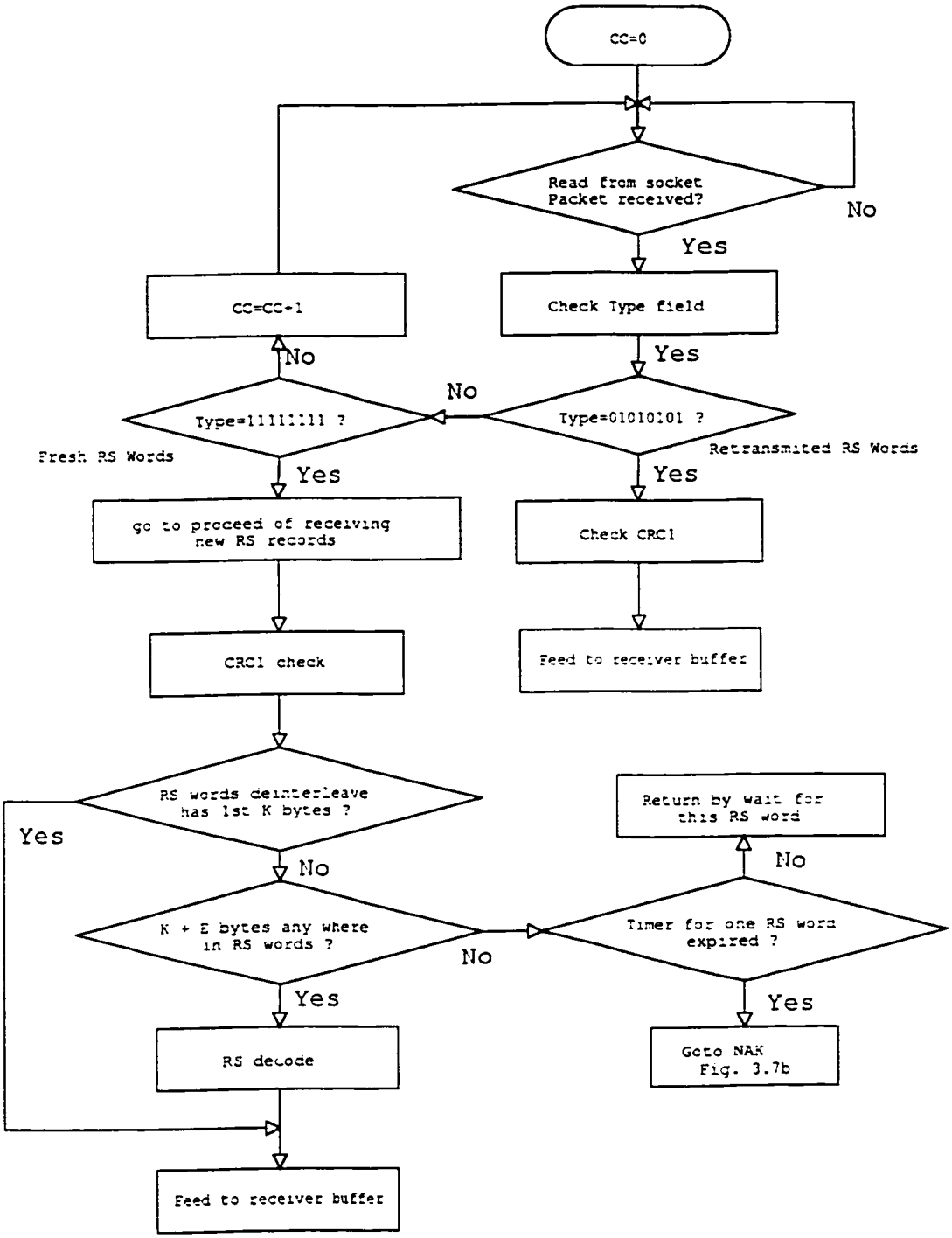

Figure 3.7a Receive at client

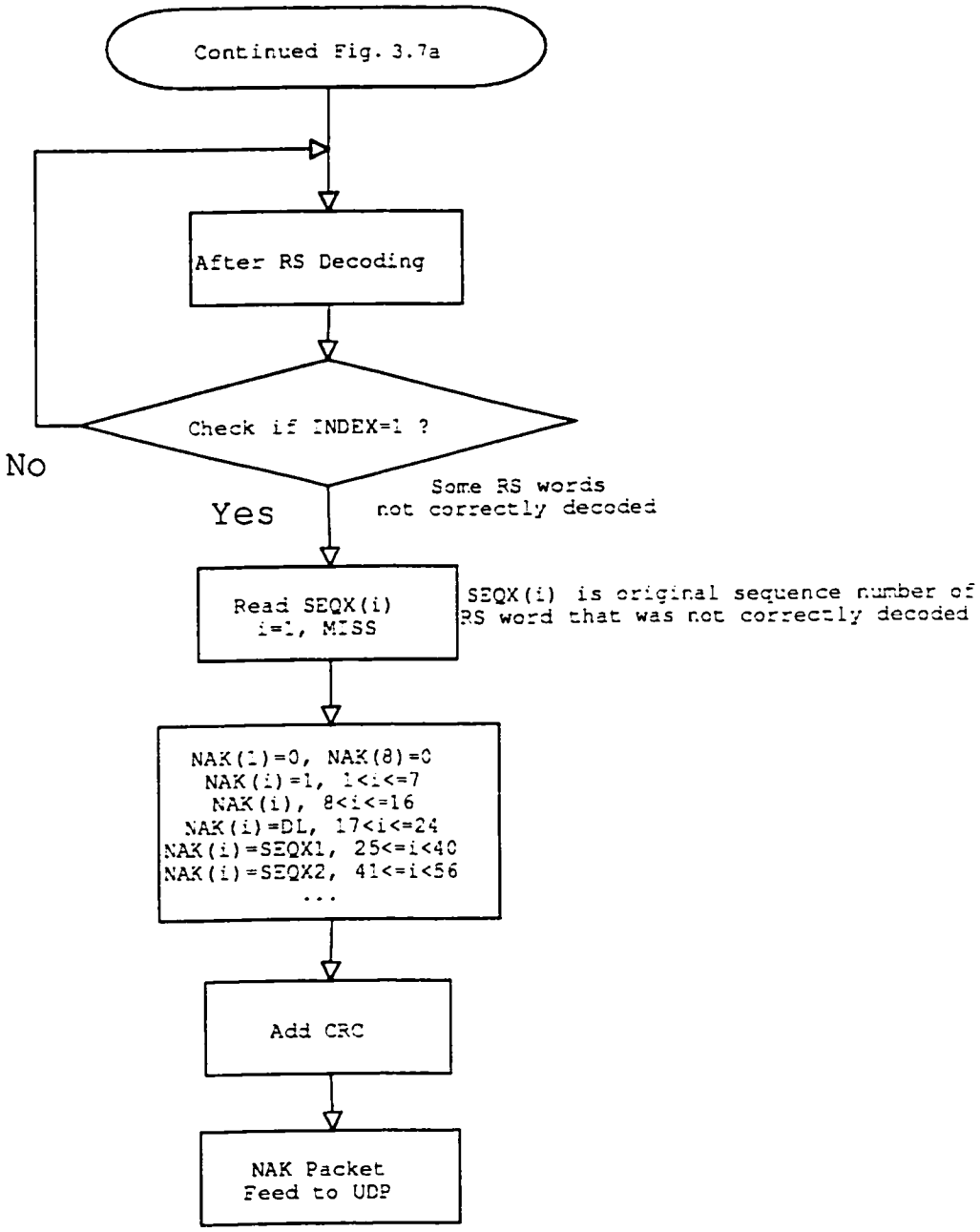

Figure 3.7b Send NAK packet at client

and the following 217 bytes are sent to the receiver buffer. The DL field may be used for separating real data from padding.

If the type field indicates an interleaved word, this interleaved word is sent to the de-interleaving memory. A timer is reset upon the arrival of the first byte to arrive from a certain de-interleaved RS word (may not be the first byte of the word).

Routinely and over shorter periods of time, the de-interleaved RS words are checked one after another. If in a certain RS word Figure3.7a, the first K=223 bytes have been received (before the expiry of a certain time out TH1 on the timer above), and CRC indicates correct reception, the first 4 bytes are stripped and the following 217 data bytes delivered to the receiver's buffer. If CRC checking fails then erasure based RS decoding will be tried followed by CRC checking again. If CRC works in this second trial after RS decoding the 217 bytes are delivered to the receiver buffer, etc. If CRC fails for the second time in a row, the client waits for more bytes before trying RS and CRC decoding again.

If (K+E) bytes out of N=255 are received (not necessarily the first K bytes), before the expiry of another time out TH2 (TH2 TH1), erasure based RS decoding, then CRC checking and 217 data bytes delivery to the receiver buffer takes place.

If CRC checking fails in the latter case, or time out TH2 is exceeded without receiving K+E bytes, then a NAK (repair request) is formulated and sent to the server. E is a suitably selected number in the range 1 to (N-K).

TH1 is a period of time corresponding to a number of bits in the range ((K+E)*256*8) To (N*256*8), while TH2 corresponds to the range (N*256*8) to ((N+E)*256*8).

Needless to say, one has to pay for the loss resilience properties of interleaving, i.e. by having to wait longer time before a meaningful number of bytes (at least K+E) of the same RS word are received. Successive bytes arrive separated by a time that corresponds to one RS word i.e., 256 bytes.

Only 2 NAKs, Repair requests are allowed for the same RS word. The processing of The received repairs (RS encoded but de-interleaved words) will be similar to the above for both repairs, with the lonely difference that no RS decoding takes place for the second repair word. Finally, the previous 217 bytes are repeated at the receiver buffer (for video and voice files) if two repair trials for the same RS word fail.

It is also possible that too much data will arrive at the client, while being busy with de-interleaving and RS decoding takes place, which may lead to eventual data loss. To protect against this loss of data due to processing constraints, flow control may be adopted, this is not very feasible in view of the unguaranteed service of the underlying UDP protocol, but will be investigated in the near future.

To give some time for processing of the miscellaneous error recovery mechanisms in this work, all successive 217 bytes words delivered to the receiver buffer at the client, are not played immediately, rather they will be stored in a buffer. This buffer has a bit length that corresponds to 2 interleaving windows, i.e. (2x256x217x8) bits. Once the buffer is filled, it is discharged to the receiver player. Figure 3.7b. shows the step by step generation of the NAK request at the client. Returning back to Figure 3.5b, the processing steps for such a NAK message as it arrives at the server.

This Figure Also shows the multiplexing of the server transmission times between fresh RS (interleaved) and repair (retransmission) words. Priority is given to repair words, and Interleaved words are transmitted when there is no repair to be sent (XZ=0). Following the sending of all available repair words, a number of encoded and interleaved RS words equal to the minimum of the number of encoded RS words in server memory, and 8 is served. This is followed by serving all available repair packets and the process repeats as in Figure 3.5b.

The timing diagram, Figure 3.8, shows the sequence of events taking place under the control of our new FEC/ARQ control for a typical scenario. Equal length interleaving windows for data records N-1, N, N+1 are shown. The end of the next window sets the time limit for playing received data packets of the current record at the MPEG player (or delivery to receiver buffer in the file transfer case). This guarantees delay jitter free play at the receiver or player since packets not lost of each record are delayed (to allow time for FEC decoding, ARQ ...of those unlucky packets that suffered loss etc.) and then played in sequence. The figure also shows for record N the NAK and retransmissions that took place following the packets losses. For record N+1 the RS FEC decoder was able to correct the loss and errors following error detection over the k RS data symbols without asking for any retransmissions, so no NAK were transmitted. For record N+2 no loss no error were encountered for the first k RS symbols (bytes), so all went well, the RS decoder was not called, and no NAK generated.

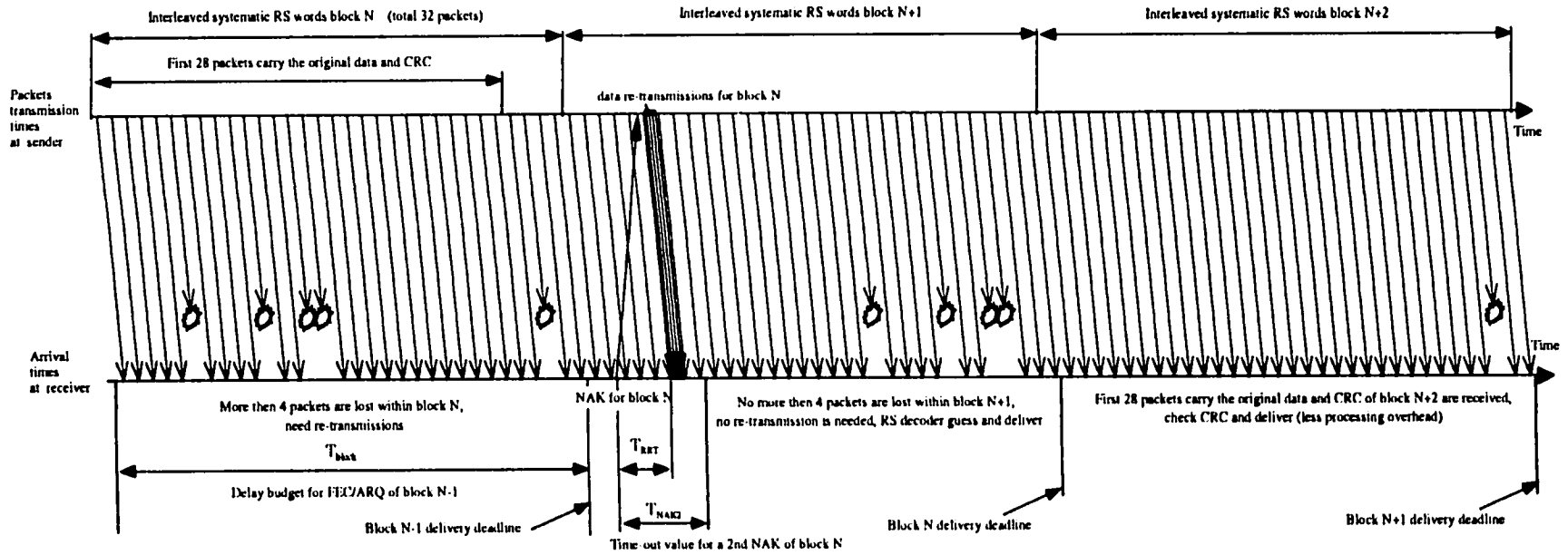

Figure 3.8 The hybrid FEC/ARQ system timing

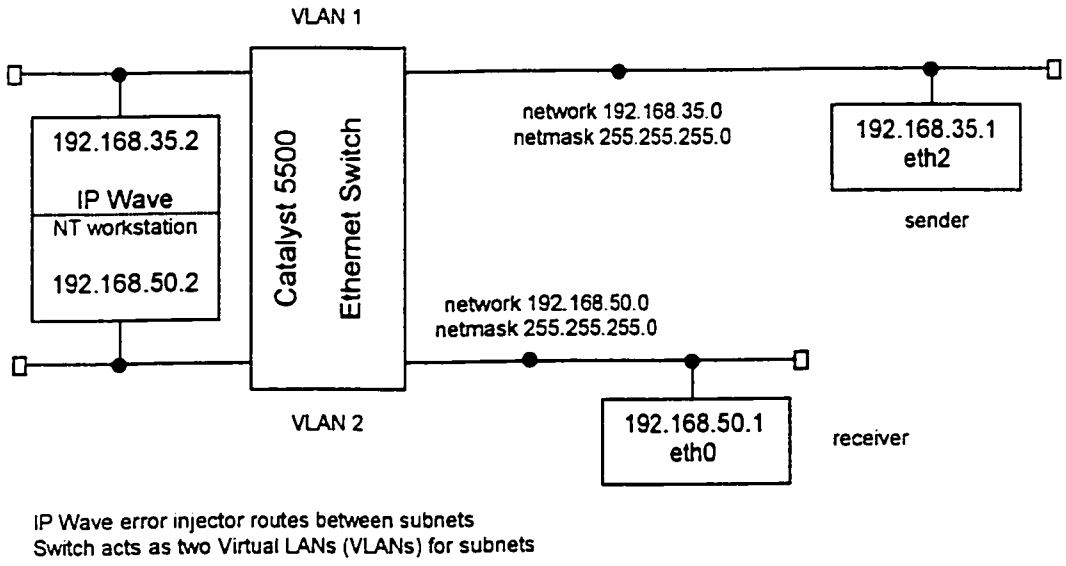

Figure 3.9 Block diagram of the experimental testbed

3.3 Description of the experimental testbed

The techniques described in the previous section have been implemented on our experimental testbed. A simple configuration of two PCs (Pentium II 600Mhz with 256MB memory) connected to 100 BaseT Cisco Catalyst 5500 Ethernet switch. Catalyst is configured acting as two virtual LANs and IP Wave Network Impairment Emulator is configured as the default router between these two virtual LAN. Figure 3.9 illustrates the testbed. IP Wave running on Windows NT platform was used as an error and loss-injecting machine to resemble the channel and network effects between the file-sending server (one PC) and the client (the other PC). Linux Red Hat 7.1 distribution with kernel 2.4.2.2. was used for evaluating performance of the proposed error recovery techniques.

3.4 Implementation

The error recovery techniques described in section 3.2 belong to our FEC multicast project group. Based on the sequential models, both the server and the client are converted to event driven models to make the implementation easier.

We choose Linux Red Hat distribution for its rich available source codes like routing, Mbone multicast routing support - Mrouted, good MPEG player MPEGPlay, etc. But its kernel is not a real-time kernel. The timers can be called at user space have a granularity limit to 10 milliseconds and the context switch time is unpredictable. Therefore we use a scheduler run in user space, it spin all the times to wait for the event to happen.

A shorten pseudo C code for the client and server processes is given in Appendix A.

3.5 Experimental results

Many experiments were conducted, for different amounts of packet loss, and byte errors at different transmission rate. The results are presented in this section.

3.5.1 Recover from loss

The Internet is a best-effort packet network, so it may lose packets arbitrarily. With error protection to packets, Packet error rate is several orders less of the magnitude than loss rate. Therefore, packet loss is our main concern.

Figures 3.10-3.15 show a sample of the obtained results while recover from loss only. Figure 3.10 shows the percentage of words for which the RS FEC decoder was

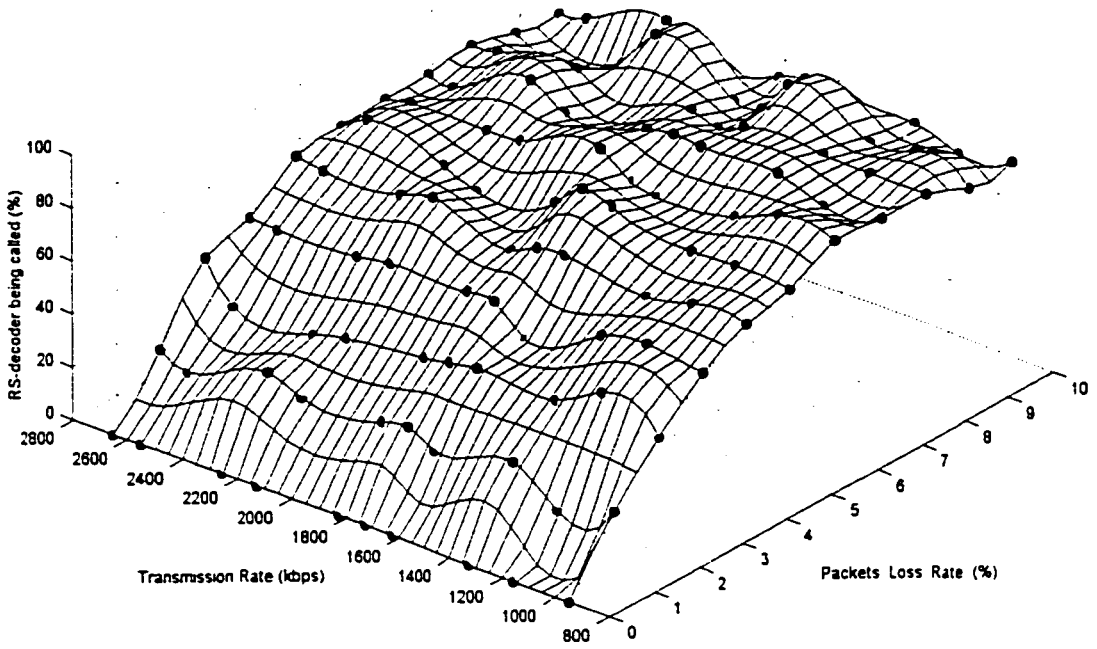

Figure 3.10 Percent of the RS decoder being called vs. rate and loss for the hybrid FEC/ARQ scheme

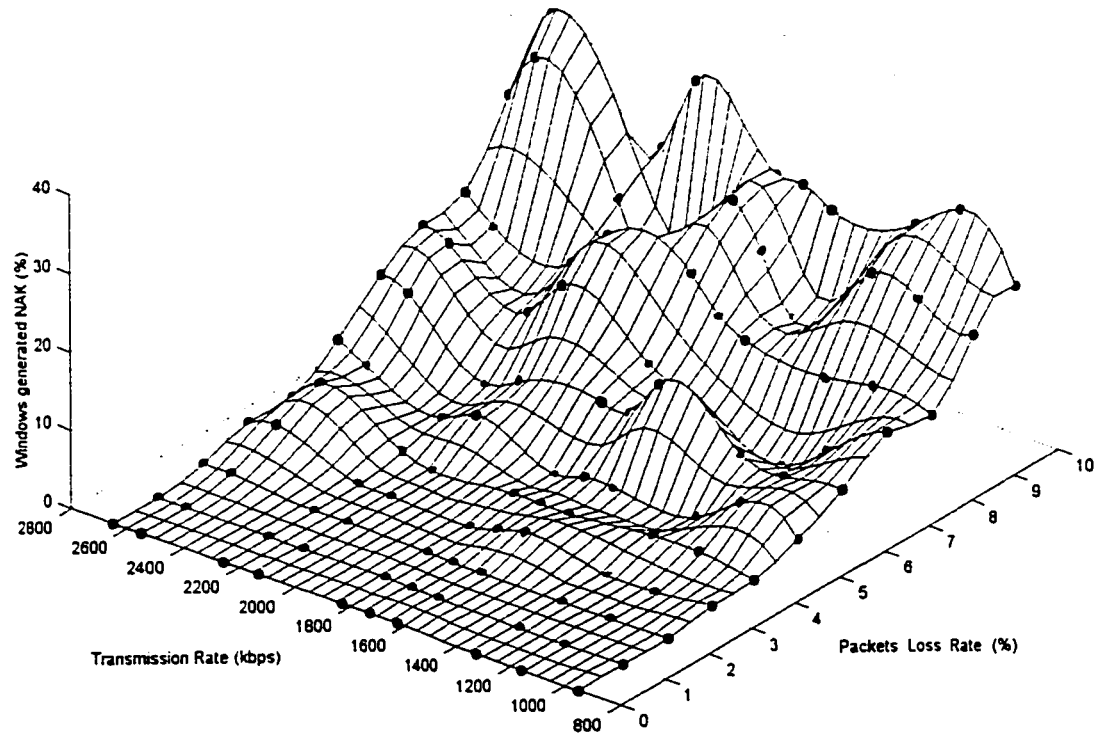

Figure 3.11 Percent of windows generated NAK vs. rate and loss (32 packet per window)

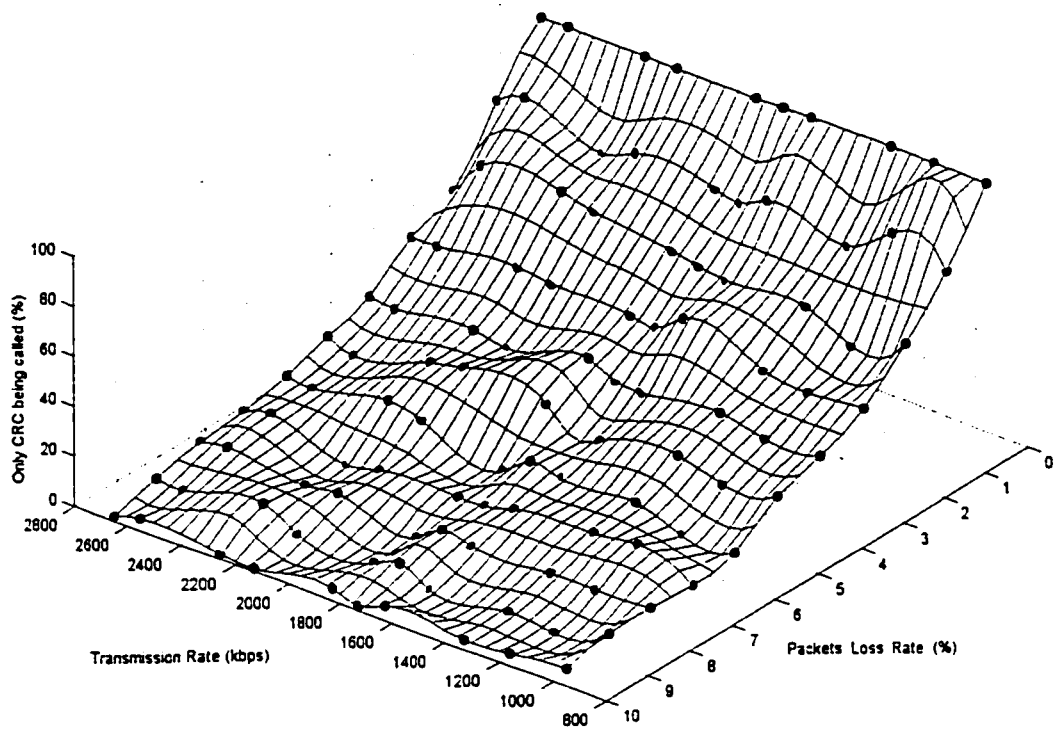

Figure 3.12 Percent of only CRC being called vs. rate and loss for the hybrid FEC/ARQ scheme

called. For low data rates and/or low random loss, this percentage is low but steadily rises as the induced loss increase. Figure 3.11 shows the percentage of windows (interleaved blocks) generating a NAK did not increase much as Figure 3.10 due to the fact that many RS words were recovered by the erasure decoding capability of RS codes.

Figure 3.12 reflects the contribution of systematic codes used to minimize processing overhead.

Figure 3.13 shows that very few retransmissions are possible even at high data rate, and high loss probabilities.

Figure 3.14 yields the overall efficiency and improvement of the proposed FEC/ARQ algorithms, i.e. the percent of those RS words that were delivered to the receiver buffer before the final time out. This reflect all the powers of techniques used, i.e. the FEC erasure decoding capability of RS codes, breaking of word losses by

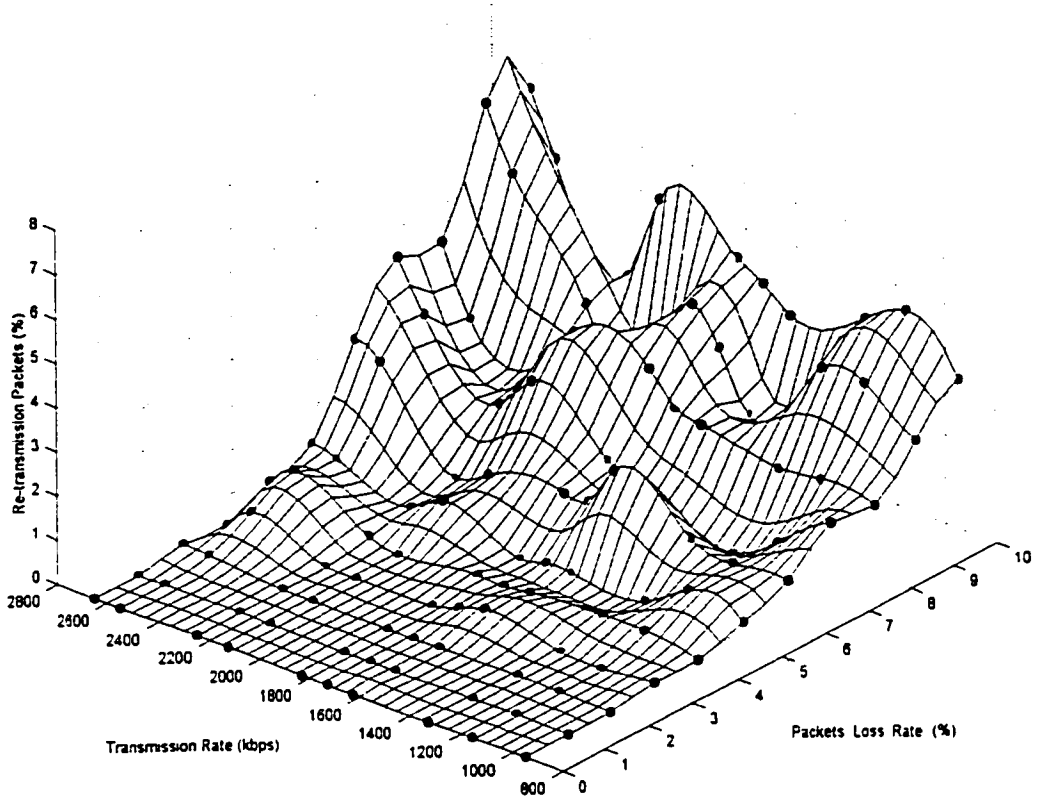

Figure 3.13 Percent of retransmissions vs. rate and loss for the hybrid FEC/ARQ scheme

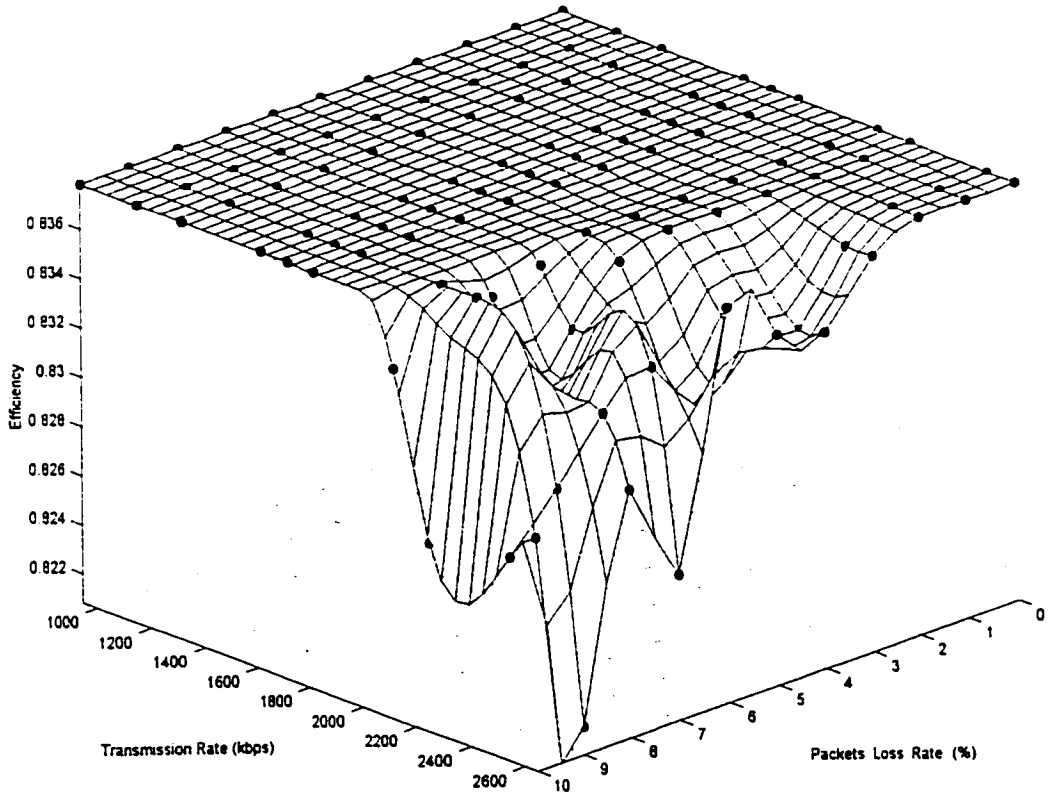

Figure 3.14 Efficiency vs. rate and loss for the hybrid FEC/ARQ scheme

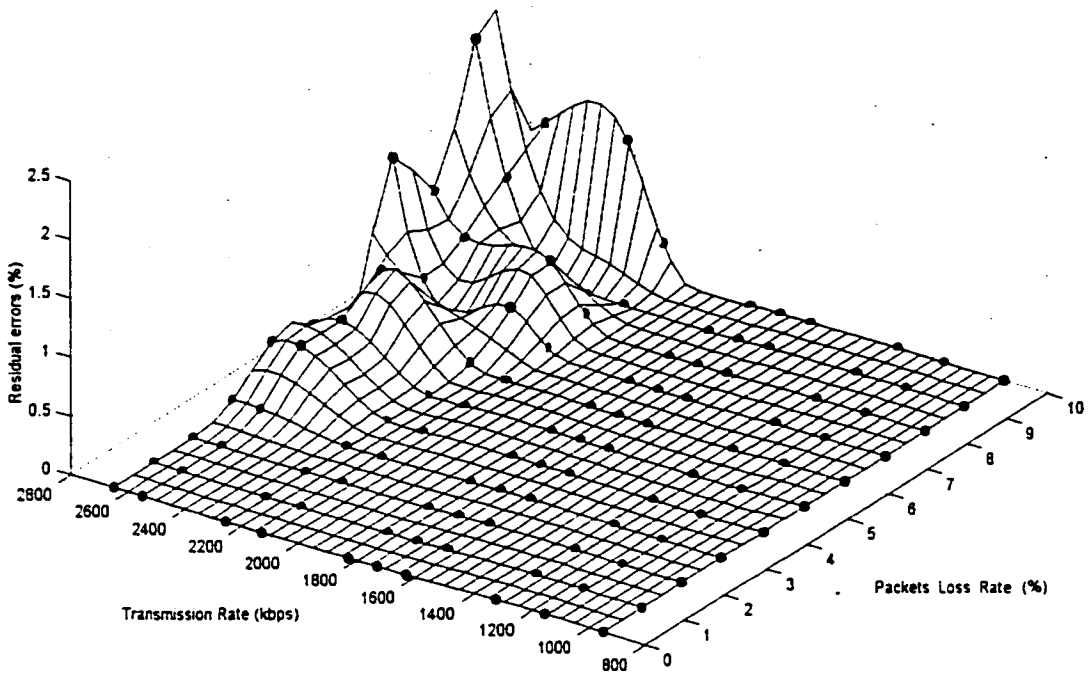

Figure 3.15 Residual error vs. rate and loss for the hybrid FEC/ARQ scheme

interleaving, and recovery obtained from ARQ. Subtracting this percent of Figure 3.14 from 1 yields the residual error of Figure 3.15 which reflect the percentage of those words for which none of the techniques used worked.

Figures 3.13-3.15 also show that the processing constrain appears after the rate exceeding 2.2 Mbps.

3.5.2 Recover from loss and error

RS codes can recover from both erasures and errors. This is a very important feature while some of the multicast group members use wireless links with potential high bit error rate. These wireless links may adopt a slightly modified IP layer that makes use of corrupted packets. Figures 3.16-3.25 show a sample of the obtained results while recover from both erasures and errors. Figure 3.16-3.17 show the percentage of

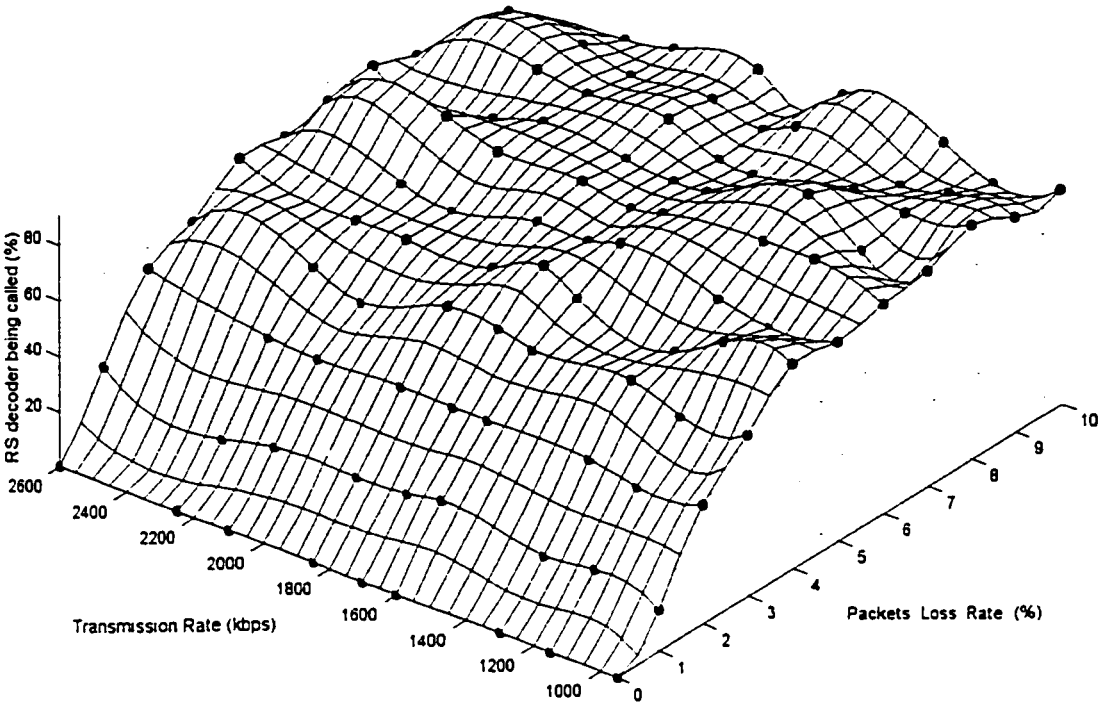

Figure 3.16 Percent of RS decoder being called vs. rate and loss for the RS/ARQ hybrid scheme (1% RS words with byte errors)

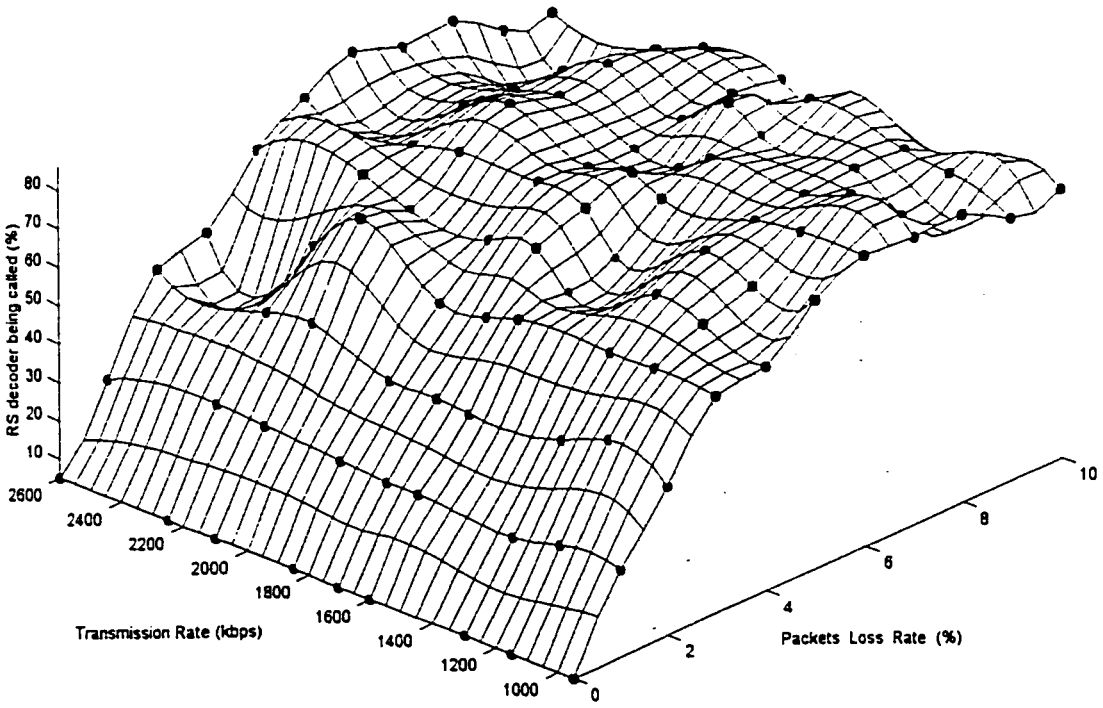

Figure 3.17 Percent of RS decoder being called vs. rate and loss for the RS/ARQ hybrid scheme (6% RS words with byte errors)

words for which the RS FEC decoder was called following the error detection by CRC. For low data rates and /or low random loss, and/or low byte errors, this percentage is low but steadily rises as the induced loss and error increase. Figure 3.18-3.19 show the percentage of generating a NAK did not increase much as Figure 3.16-3.17 due to the fact that many RS words were corrected by the erasure decoding capability of RS codes.

Figure 3.20-3.21 show that very few retransmissions are possible even at high data rate, and high error and loss probabilities.

Figure 3.22-3.23 yield the overall efficiency and improvement of the proposed FEC/ARQ algorithms, i.e. the percent of those RS words that were delivered to the receiver buffer before the final time out. This reflect all the powers of techniques used, i.e. the FEC erasure decoding capability of RS codes, breaking of word losses by interleaving, and recovery obtained from ARQ. Making Figure 3.22-3.23 up side down yields the residual error of Figure 3.24-3.25, which reflect the percentage of those words for which none of the techniques used worked. Processing constrain is evident in these figures at higher transmission rate.

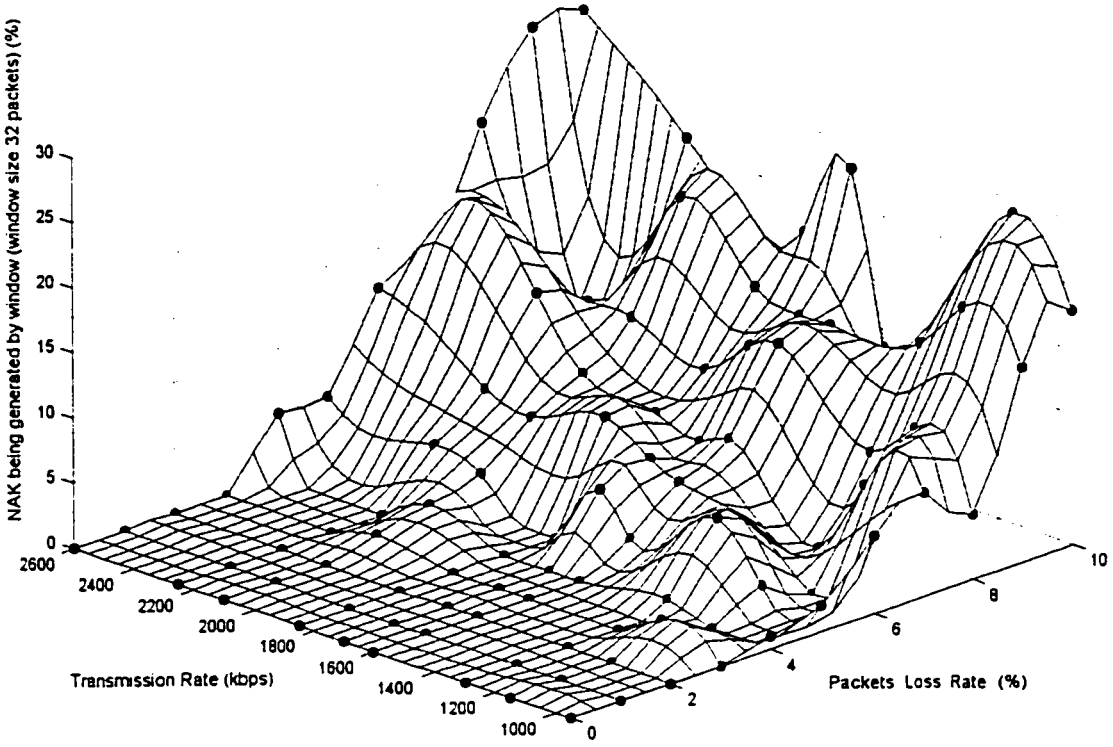

Figure 3.18 Percent of NAKs being generated vs. rate and loss for the RS/ARQ hybrid scheme (1% RS words with byte errors)

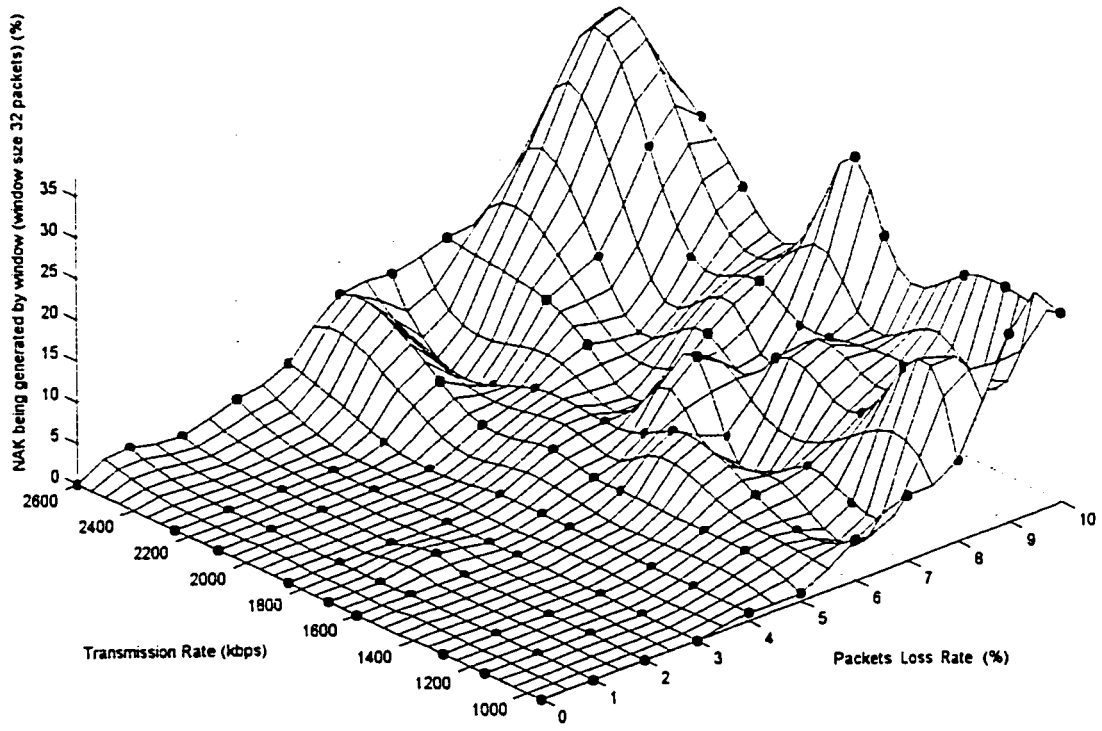

Figure 3.19 Percent of NAKs being generated vs. rate and loss for the RS/ARQ hybrid scheme (6% RS words with byte errors)

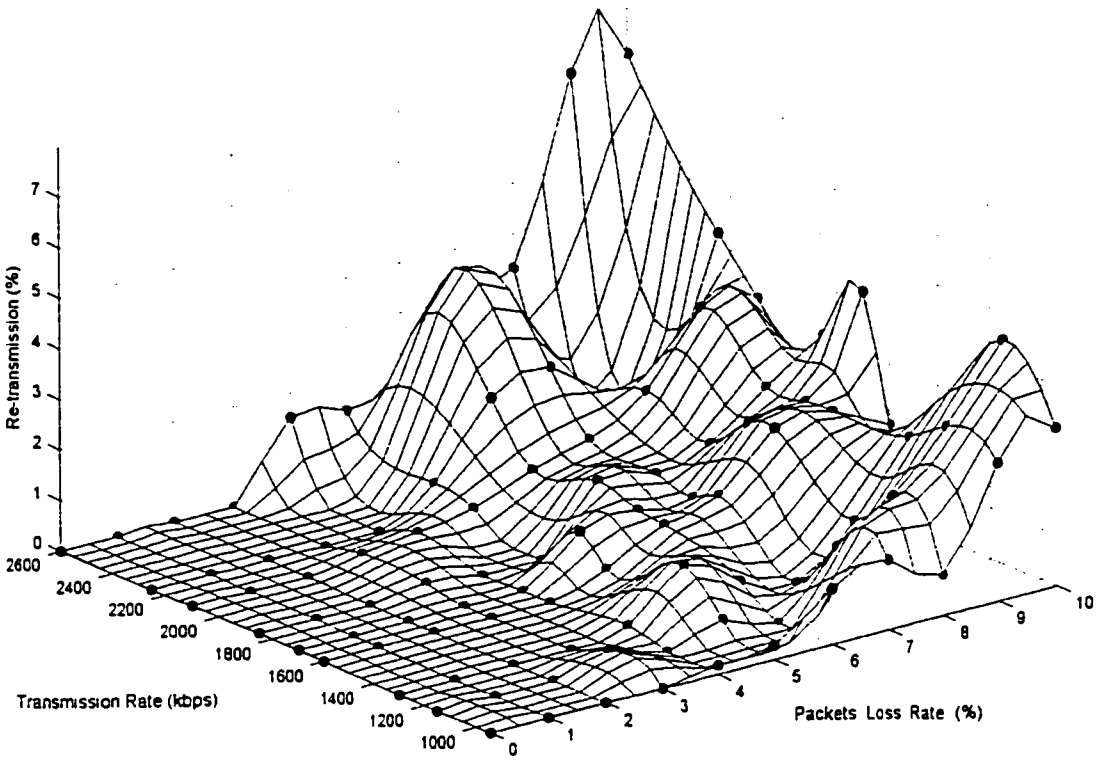

Figure 3.20 Percent of re-transmissions vs. rate and loss for the RS/ARQ hybrid scheme (while 1% RS words with byte errors)

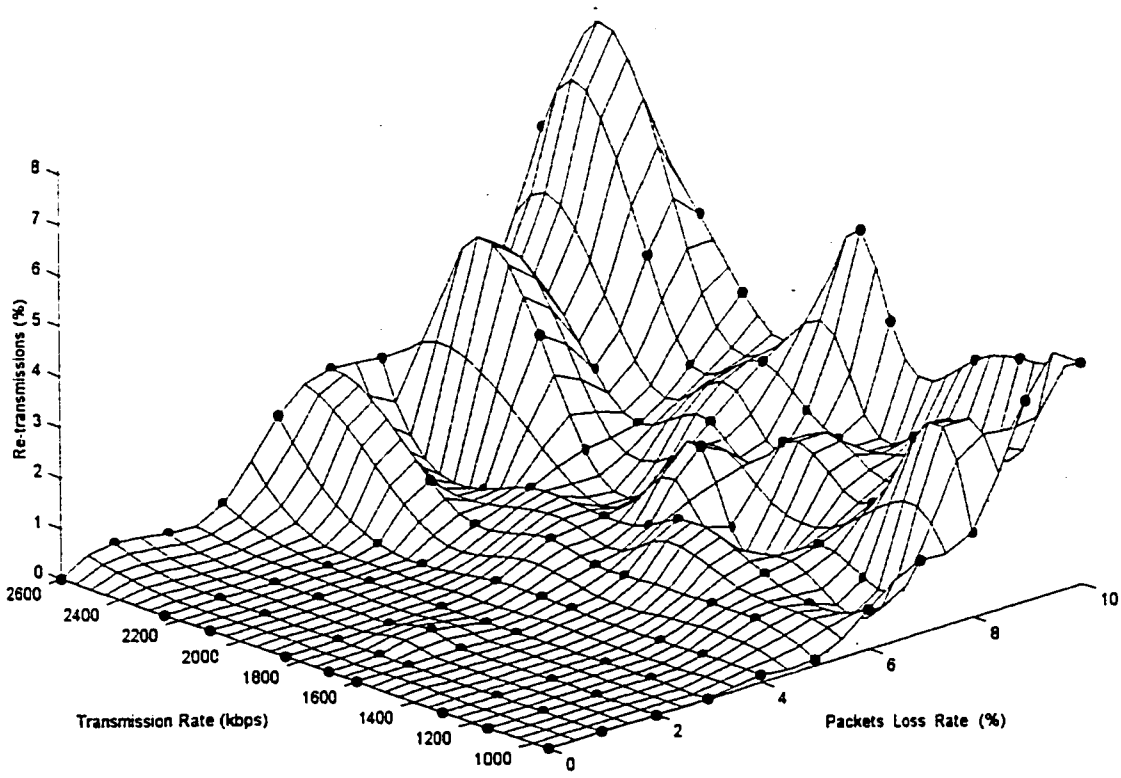

Figure 3.21 Percent of re-transmissions vs. rate and loss for the RS/ARQ hybrid scheme (while 6% RS words with byte errors)

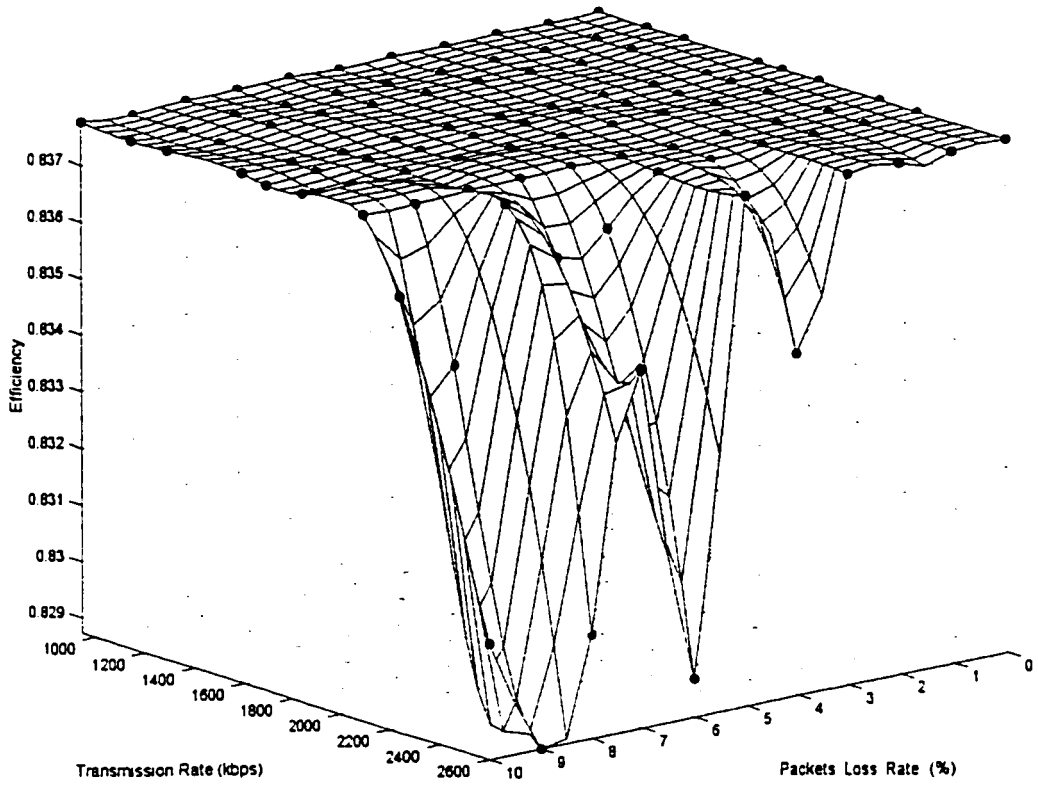

Figure 3.22 Efficiency vs rate and loss for the RS-ARQ hybrid scheme (while 1% RS words with byte errors)

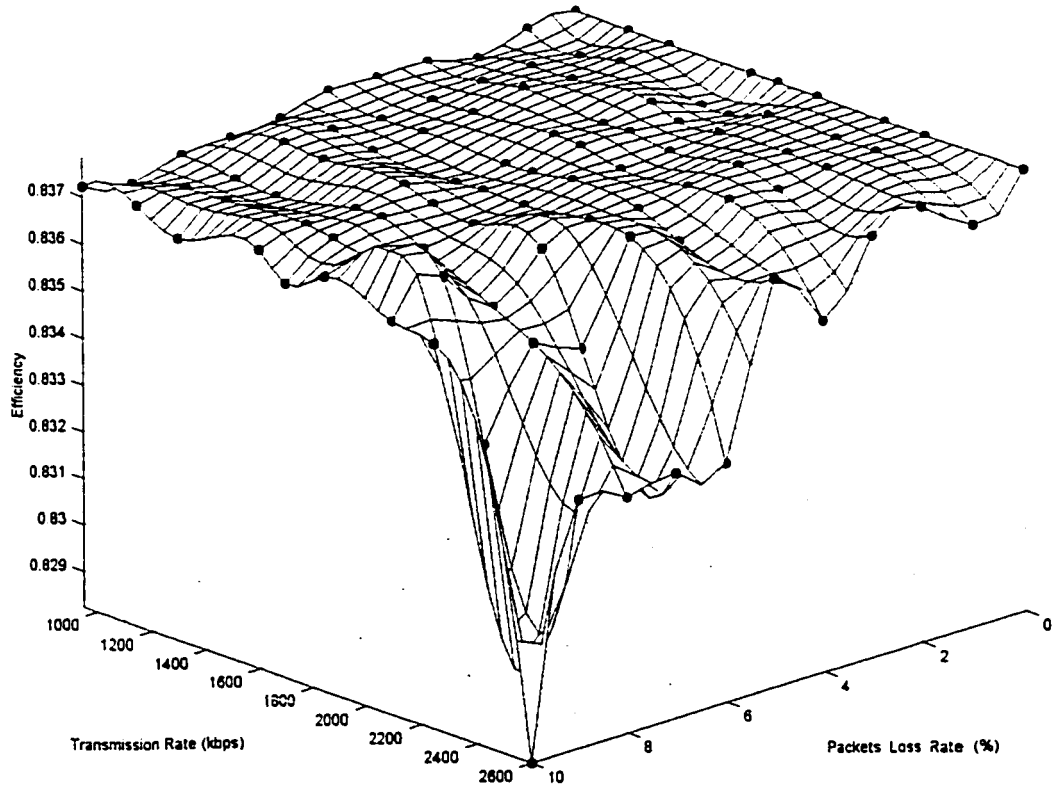

Figure 3.23 Efficiency vs. rate and loss for the RS-ARQ hybrid scheme (while 6% RS words with byte errors)

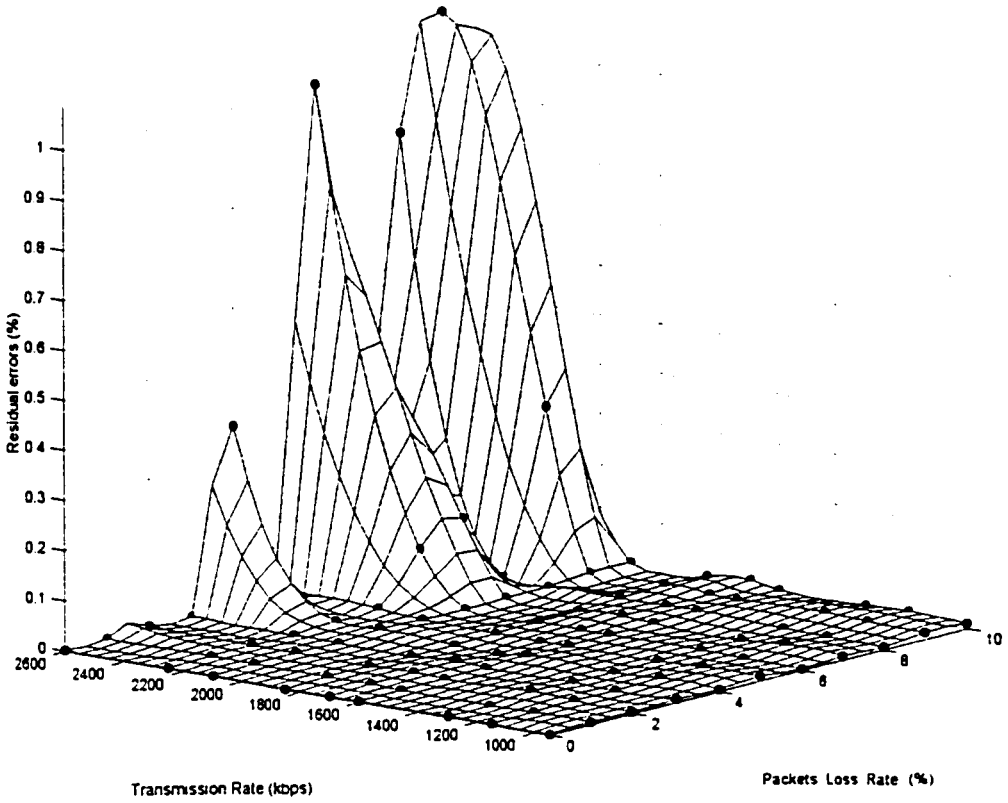

Figure 3.24 Percent of Residual errors vs rate and loss for the RS/ARQ hybrid scheme (while 1% RS words with byte errors)

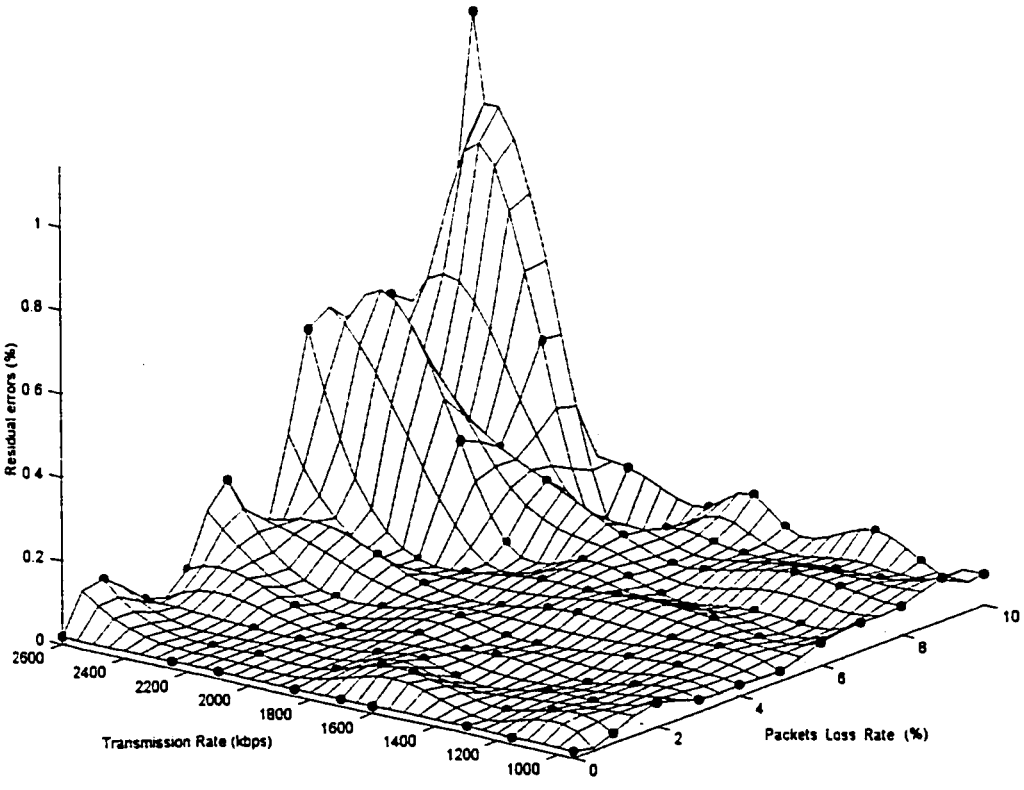

Figure 3.25 Percent of Residual errors vs rate and loss for the RS/ARQ hybrid scheme (while 6% RS words with byte errors)

Chapter 4

Performance Evaluation

The main difficulty of multimedia reliable multicast is the scalability it could offer. Firstly, we analyze the basic characteristic of our hybrid FEC/ARQ techniques. Then we evaluate its performance in multicast environment. We present a comparative performance analysis of the hybrid FEC/ARQ scheme and an ARQ only scheme. We evaluate the effect achieved by our schemes in terms of the average number of transmissions, average number of NAKs generated and expected residual errors. We also show the influence of varying sending rates, group sizes, and packet loss rate. Only packet loss, no random error, is considered in this chapter.

4.1 Basic characteristic of hybrid FEC/ARQ

Some important performance measures of the scheme are average transmissions per packet generated (bandwidth usage), number of NAKs generated per block (feedback), and residual error.

Starting with our test results, let's go over some figures from last chapter. Taking a look at Figs 3.11, 3.18, and 3.19, one notices that the NAK generated by interleaved RS block (window) decrease to near zero while the packet lost rate is less

than 3%. Similar results can be observed from Figs 3.13, 3.20, 3.21: There are fewer retransmissions when the packet lost rate is less than 3%. These regions show the real power of FEC. This inherent FEC feature (recover of lost bytes by RS erasure decoding) is great for scalable real-time multicast because it minimize feedback and retransmission.

The performance can be computed analytically in some simple cases, i.e. with independent, yet having same loss rate group members.

The notation used in this section is described as follow:

P	Probability of packet loss
i	Number of packets lost
NN	Number of interleaved RS words per packet, NN=8
N	Total number of packets in an interleaved RS words block (32)
K	Number of packets containing original data in an interleaved RS words block (28)
P'	Probability of a packet being RS decoded incorrectly
$P_{no\ NAK}$	Probability there is no NAK generated for a certain block at receiver
P_{NAK}	Probability a block generates NAK at receiver (FEC/ARQ)
$P_{error\ RS/ARQ}$	Expected residual packet error for the FEC/ARQ scheme
$P_{error\ ARQ}$	Expected residual packet error for the ARQ only scheme

Each packet contains 8 interleaved RS words, so one packet loss implies only 8 symbols lost in each of 256 RS encoded words. Since (255, 223) Reed-Solomon codes can correct 32 erasures per RS word, then loss of 4 packets will lead to 32 erasures in each de-interleaved RS words, which is correctable. When the number of packets lost is more than four, NAK will be generated. The sender will re-transmit only interleaved words containing original data (no parity bytes). The following equation gives the probability that a packet cannot be RS decoded correctly and hence has to be

retransmitted. Note that as above if 0 or 1 or 2, 3, 4 (less then N-K) packets are lost, then 0 or 8 or 16, 24, 32 bytes will be lost in the de-interleaved RS words.

$$P' = \frac{1}{N} \sum_{i=N-K+1}^{N} i \binom{N}{i} (1-P)^{N-i} P^i \tag{4.1}$$

Figure 4.1 shows the packet retransmission performance results (P') for different cases, where we borrow the FEC lab results of Figure 3.13, which represents the average of the test results between 1Mbps and 2.25Mbps. When the transmission rate is too high, the RS processing exceeds its limitation, and the lab result don't match the simple analytical result in equation 4.1.

Figure 4.1 also shows that the retransmission percentage (P') is much higher without FEC, especially when the loss rate is small. Figure 4.1 also shows that equation 4.1 and lab results are in good agreement.

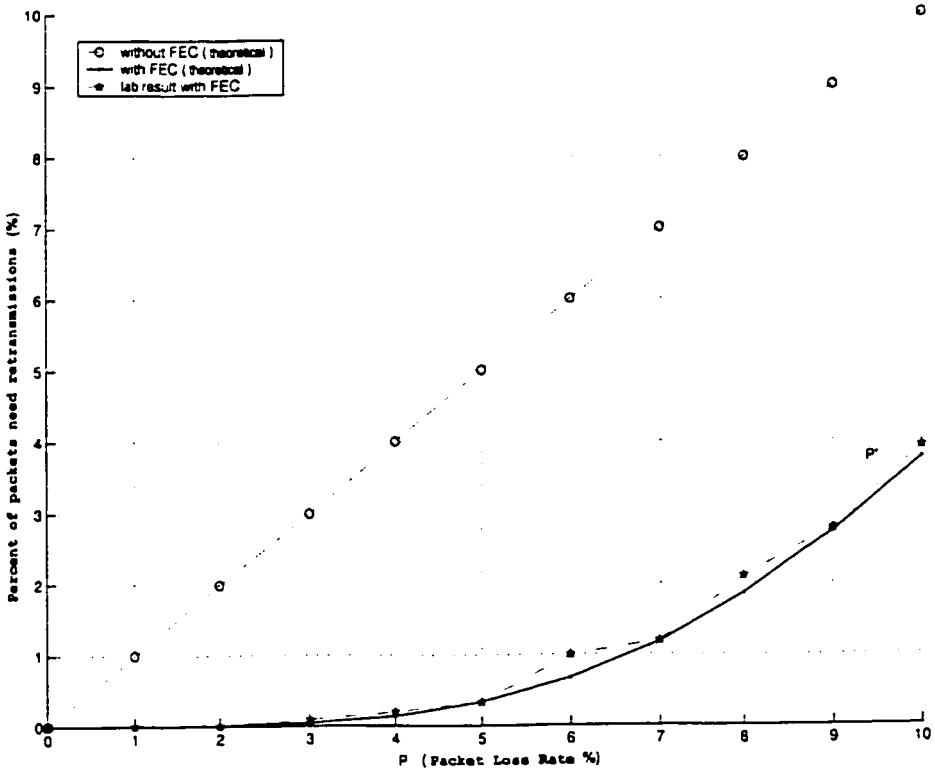

Figure 4.1 Percent of packets need retransmissions

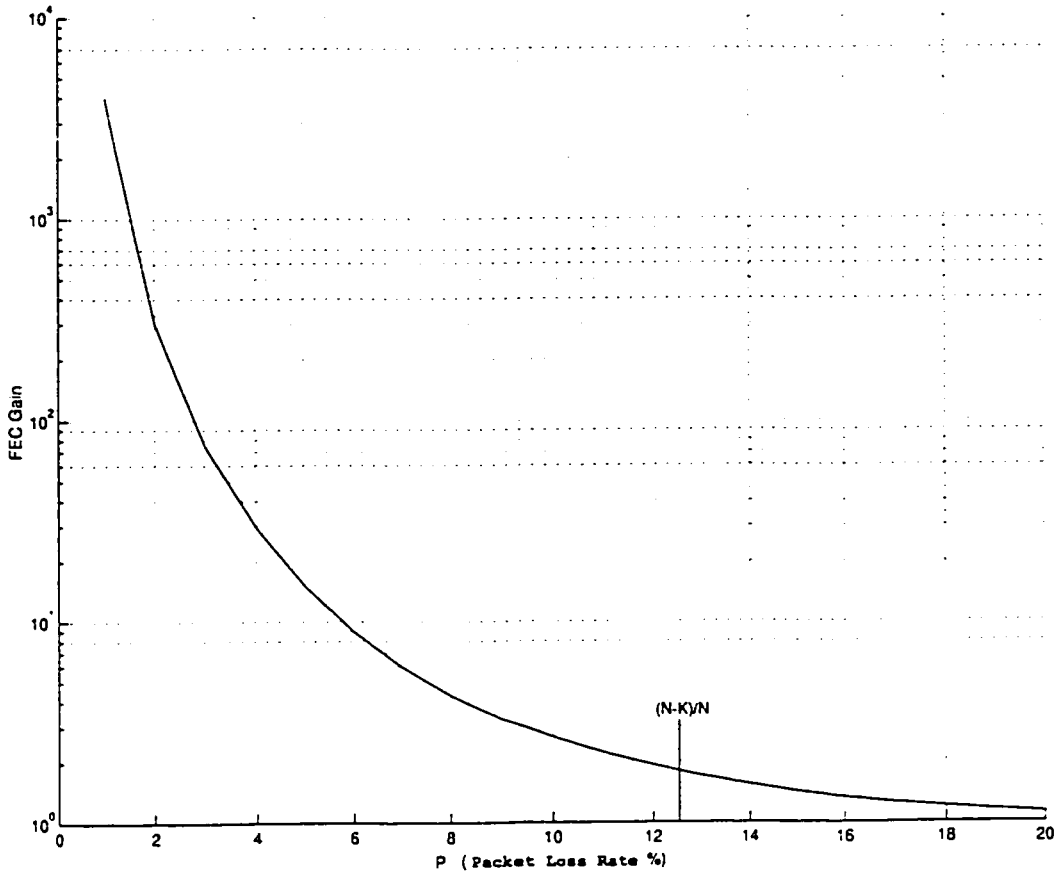

Figure 4.2 The FEC gain in term of number of retransmissions

Figure 4.2 gives the ratio of retransmission probability without and with RS codes (obtained from Figure 4.1). The FEC gain becomes less while the packet loss rate increases. RS codes are strong when the loss is much less than the maximum number of correctable erasures per block (2t = N-K).

For a systematic RS words block, there is no NAK being generated when the number of packets lost is less than *N-K*. Therefore, the probability that there is no NAK being generated for a block is:

$$P_{no\ NAK} = \sum_{i=0}^{N-K} \binom{N}{i} (1-P)^{N-i} P^i \qquad (4.2.a)$$

The probability that a block generates NAK to one or more packets at one receiver is:

$$P_{NAK} = 1 - P_{no\ NAK} \qquad (4.2.b)$$

Figure 4.3 shows a comparison of the probability of NAK generated while transferring a fixed block while using RS coding and without coding. Figure 4.3 shows that RS codes reduce the chance of NAK being sent for a certain block, actually reducing this chance to near zero as the loss rate being less then 3%. The test results in Figure 4.3 are actually the average of those results in Figure 3.11 in the range 1Mbps and 2.25Mbps. The test results are almost identical with our analytical results, which show that the analytical model is adequate.

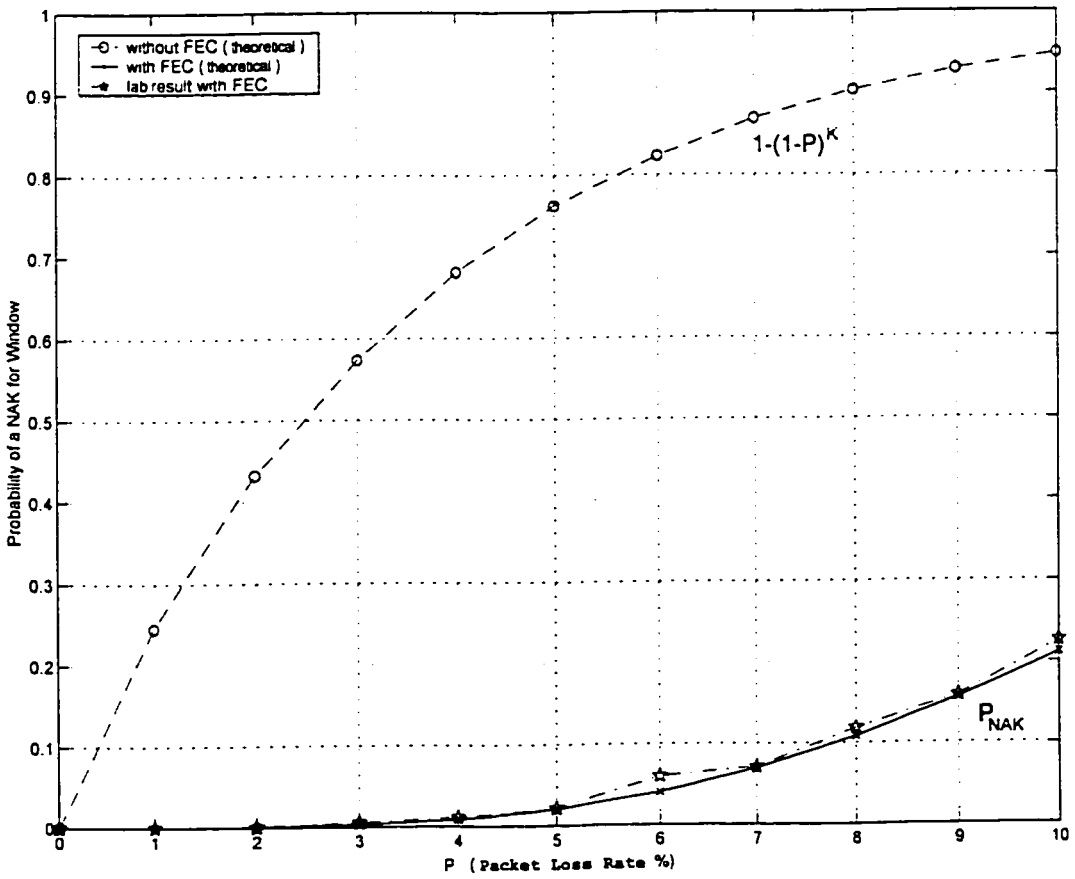

Figure 4.3 A comparison of the Probability of Generating a NAK during a certain window

Limiting the number of NAKs to 2, the expected residual error for one packet in the FEC/ARQ scheme is:

$$P_{error\ RS/ARQ} = P' \cdot P^2 \tag{4.3}$$

This means one RS decoding and two retransmissions trials did not work.

Without FEC, the residual error rate simply is:

$$P_{error\ ARQ} = P^3 \tag{4.4}$$

This means one transmission and two retransmissions did not work.

Figure 4.4a and 4.4b compare residual errors for the error control schemes with FEC and without FEC (linear and log respectively).

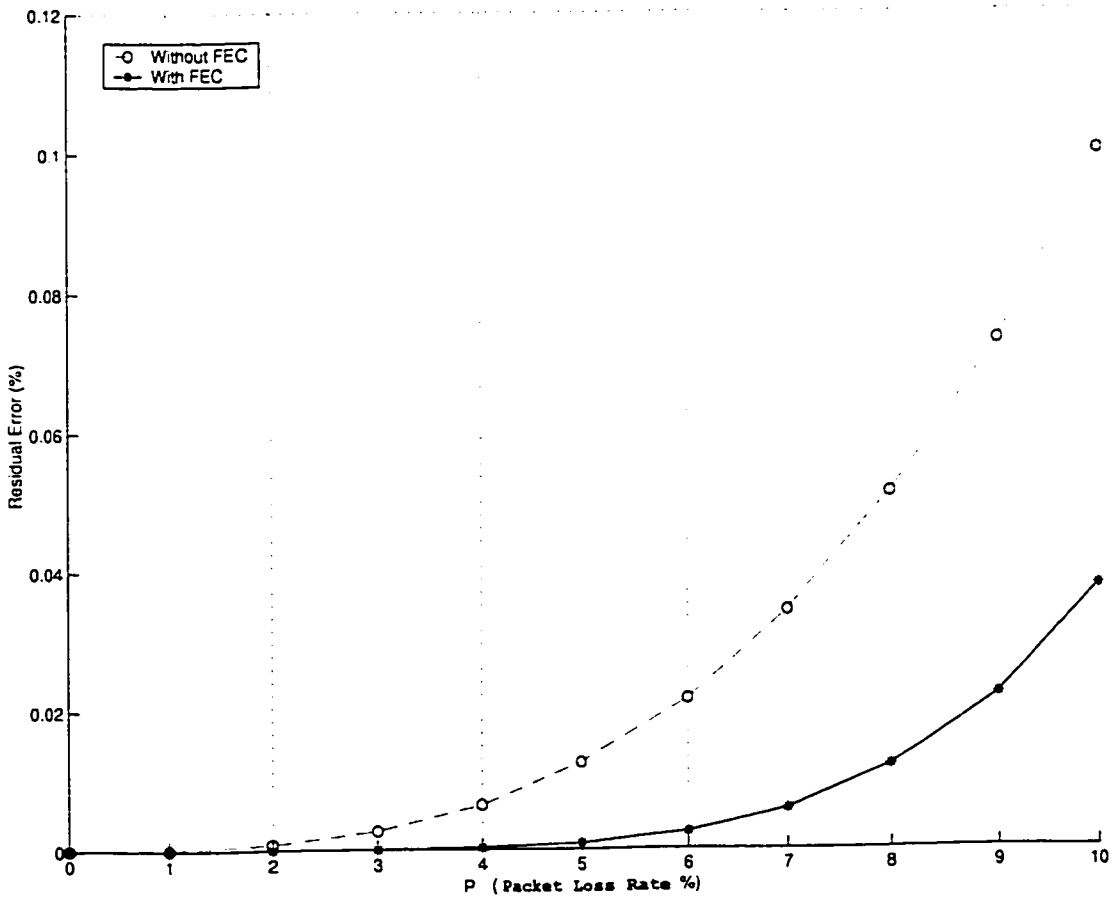

Figure 4.4a Residual error comparison (with Max. two retrasmissions) (linear)

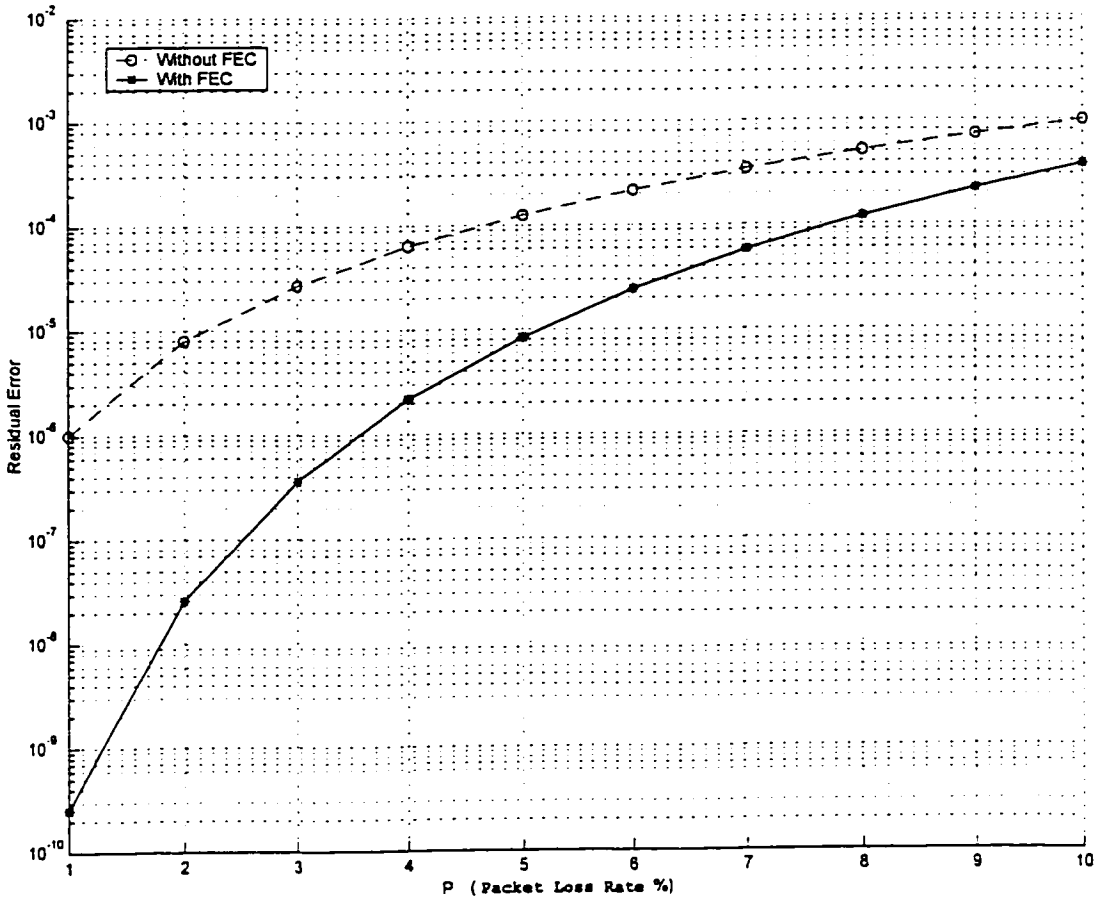

Figure 4.4b Residual error comparison (with Max. two retrasmissions) (log)

Constrained by the delivery deadline, NAK will be generated no more than twice for every interleaved block. For the same reason, retransmission contains original data only, i.e. after receiving retransmitted packets, no RS decoder is called. These assumptions apply in whole this work.

4.2 Performance evaluation for group communication

Now we extend the point-to-point lab and theoretical FEC/ARQ results outlined previously to our main application, i.e. media stream multicast. Group communication

performance is evaluated for hybrid FEC/ARQ scheme and ARQ only scheme in this section.

Assumptions:

In order to focus on the performance implications of hybrid FEC/ARQ, we make several simplifying assumptions about the network model. We assume that link loss rates are not affected by the rate at which the sender transmits. This is reasonable in a scenario where the congested links utilized by the protocol are also utilized by many other sessions. We assume only the sender transmits repairs, and these repairs are always multicast to receivers. We consider the case that all the receivers have the same loss rate.

NAKs are sent using unicast ports only (Figure 2.3). NAKs feed back from the various users are aggregated. When some sub-trees share common losses, and no local loss exist in these sub-trees, they are treated as one member when counting the equivalent group size with independent loss. These assumptions make the system model simpler to evaluate. The way of NAKs aggregation and equivalent group size counting is discussed in the last chapter.

The notation used in this section is described as follow:

P	Probability of packet loss
NN	Number of interleaved RS words per packet, NN=8
N	Total number of packets in an interleaved RS words block (32)
K	Number of packets containing original data in an interleaved RS words block (28)
U	Independent loss equivalent group size
P'	Probability of a packet being RS decoded incorrectly

P_{NAK}	Probability a block generates NAK at receiver (FEC/ARQ)
P'_{NAK}	Probability a block generates NAK 2nd time at receiver (FEC/ARQ)
α	Probability of sender needing to retransmit a packet to any group member (FEC/ARQ)
β	Probability of sender needing to retransmit a packet to any group member (ARQ only)
$E(TRAN_{RS/ARQ\_group})$	Average total transmission times per packet in the group (FEC/ARQ)
$E(TRAN_{ARQ\_group})$	Average total transmission times per packet in the group (ARQ only)
$E(NAK_{RS\ ARQ})$	Average total NAKs generated per block in the group (FEC/ARQ)
$E(NAK_{ARQ})$	Average total NAKs generated per block in the group (ARQ only)

While $(1-P')^U$ means all members of the group do not need retransmission of a packet, the probability of the sender needing to retransmit a packet to any group member for the FEC/ARQ scheme is:

$$\alpha = 1-(1-P')^U \tag{4.5}$$

The average number of receivers who need retransmission is $P' \cdot U$. The probability of the packet loss in retransmission is P. Therefore, we can make a useful approximation for the probability of sender needing to retransmit a packet second time to any group member who may need it for the FEC/ARQ scheme:

$$\alpha' \approx 1-(1-P)^{P' \cdot U} \tag{4.6}$$

Similarly, the probability of sender needing to retransmit a packet to any group member for the ARQ only scheme is:

$$\beta = 1-(1-P)^{U} \tag{4.7}$$

An approximation for the probability of sender needing to retransmit a packet second time to any group member who may need it for the ARQ only scheme is:

$$\beta' \approx 1-(1-P)^{P\cdot U} \tag{4.8}$$

Considering the FEC/ARQ scheme, the first transmission of an interleaved RS block will take N packets. The retransmissions will only have the first K original data packets. In the second retransmission, the packet is accepted as is whether it's right or wrong. Therefore, the average total transmission times per packet in the group for the FEC/ARQ scheme is:

$$\mathrm{E}(\mathrm{TRAN}_{RS/ARQ\_group}) = \frac{1}{K}\left((1-\alpha)N + \alpha(1-\alpha')\cdot(N+K) + \alpha\cdot\alpha'(N+2K)\right)$$

$$\therefore \quad \mathrm{E}(\mathrm{TRAN}_{RS/ARQ\_group}) = \frac{N}{K} + \alpha + \alpha\cdot\alpha' \tag{4.9}$$

Where first term corresponds to success of first trial, second term corresponds to first retransmission trial, or one has to go to last trial in third term.

Similarly, the average total transmission times per packet in the group for the ARQ only scheme is:

$$\mathrm{E}(\mathrm{TRAN}_{ARQ\_group}) = (1-\beta)\cdot 1 + \beta(1-\beta')\cdot 2 + \beta\cdot\beta'\cdot 3$$

$$\therefore \quad \mathrm{E}(\mathrm{TRAN}_{ARQ\_group}) = 1 + \beta + \beta\cdot\beta' \tag{4.10}$$

Because there is no parity packets transmitted, the N/K part in (4.9) corresponding to "1" in (4.10), other terms in (4.10) carry similar interprets to that in (4.9).

An approximation for the probability of a block generating NAK the 2nd time at one receiver (FEC/ARQ scheme) is: $P'_{NAK} \approx 1-(1-P)^{K \cdot P'}$. Combining this P'_{NAK} and the P_{NAK} in equation (4.2), we derive the average total NAKs generated per block in the whole group for the FEC/ARQ scheme:

$$\mathrm{E}(\mathrm{NAK}_{\mathrm{RS\,ARQ}}) = \left(P_{NAK} \cdot (1-P'_{NAK}) \cdot 1 + P_{NAK} \cdot P'_{NAK} \cdot 2\right) \cdot U$$

Where the first and second term respectively account for probability of having one and two NAKs transmitted in the two trials.

$$\therefore \quad \mathrm{E}(\mathrm{NAK}_{\mathrm{RS\,ARQ}}) = U \cdot P_{NAK}(1+P'_{NAK}) \tag{4.11}$$

Similarly, using $P'_{NAK} = 1-(1-P)^{PK}$, we can approximate the average total NAKs generated per block in the group for the ARQ only scheme:

$$\mathrm{E}(\mathrm{NAK}_{\mathrm{ARQ}}) \approx U \cdot \left(1-(1-P)^{K}\right) \cdot \left(2-(1-P)^{PK}\right) \tag{4.12}$$

The following figures are some of the multicast FEC/ARQ performance results obtained. We assume a multicast group size ranging from 1 to 10000 with independent loss; the packet loss rate ranges from 0 to 10 percent, which is the very common case for the Internet. We use our hybrid FEC/ARQ scheme that limits the number of NAKs to 2. We compare the results with ARQ only schemes to check the improvement gained by using FEC. We choose the window size as 28 packets for the ARQ only scheme, which is equal to the size of interleaved RS block's original data part. Now, based on these equations, we present following results.

Figure 4.5 shows that the average number of first trial re-transmissions increase while packet loss rate and group size increase for the hybrid multicast FEC/ARQ

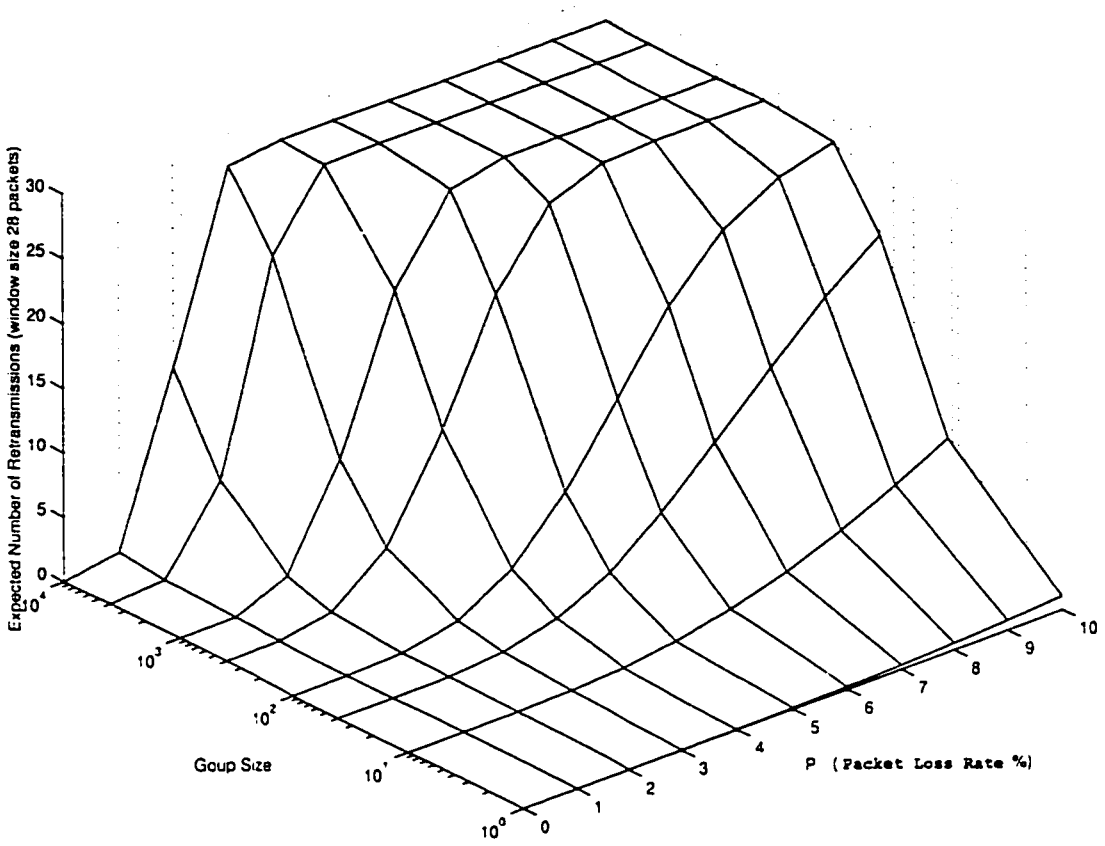

Figure 4.5 Expected number of packet retransmissions per block for 1st NAK (FEC/ARQ)

scheme (equation 4.5). At a loss rate less than 1%, there are few re-transmissions even the group size increases to 10000. There is a flat part for the low packet loss part of Figure 4.8, which indicates few re-transmissions needed when the group size is smaller than 100 and loss rate less than 4%. At the peak of this figure, there is another flat part. This part indicates that when the group size and the packet loss rate exceed certain threshold, all 28 data packets need to be re-transmitted.

Figure 4.6 shows that number of first trial re-transmissions increase very fast with packet loss rate and group size for the ARQ only scheme (equation 4.7). There is no flat part in this figure for low packet loss, which means that re-transmissions are needed even for low packet loss. At a loss rate 1% and group size of 400, the re-

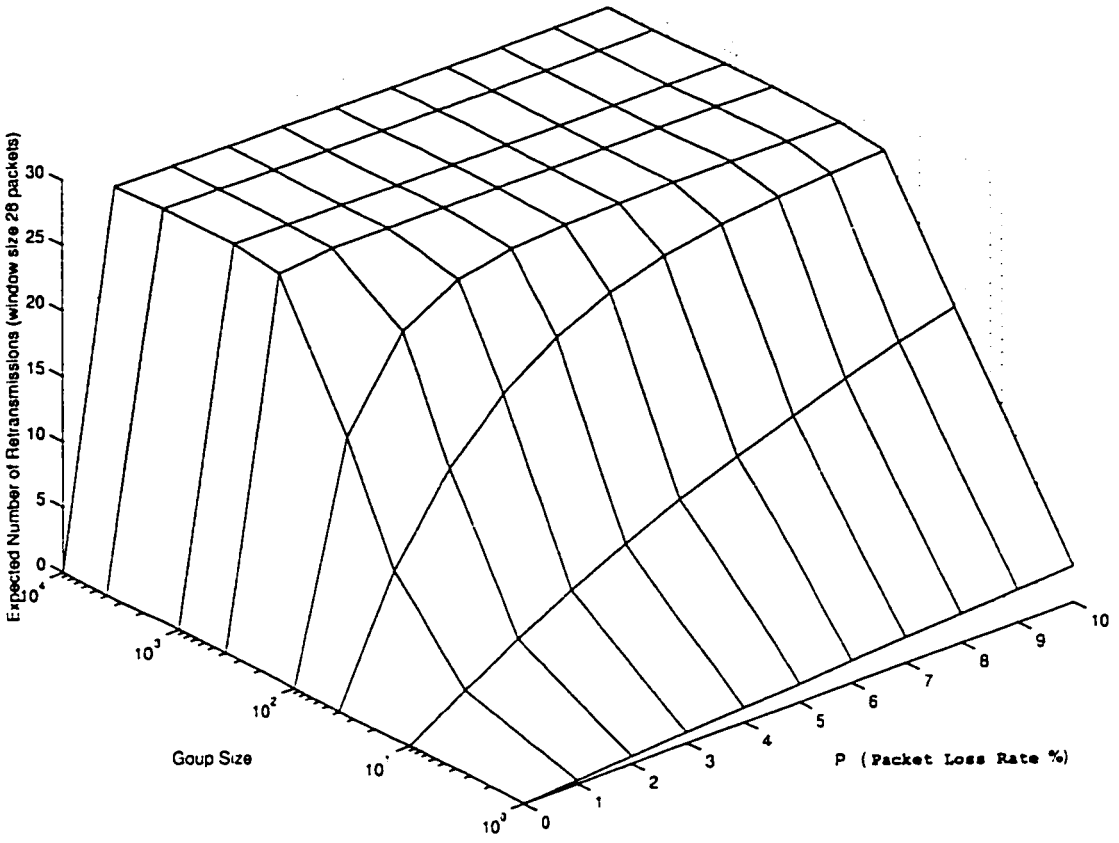

Figure 4.6 Expected number of packet retransmissions per block for 1st NAK (ARQ only)

transmissions have reached the ceiling. There is a large area at the top of this figure where all 28 data packets need to be re-transmitted for high loss and high group size.

This figure reflects the scalability problem of the ARQ only scheme without FEC even at low loss rate. Comparing figures 4.5 and 4.6, the FEC/ARQ hybrid scheme is superior in terms of number of re-transmissions needed.

Figure 4.7 shows the average total time of transmissions per packet at sender in the group for the FEC/ARQ scheme (equation 4.9). The expected number of transmissions starts more than 1 time unit in this figure even at zero packet loss because the use of parities. Transmission times also increase while packet loss rate and/or group size increases. When the loss rates are high and group size is very large, all original data

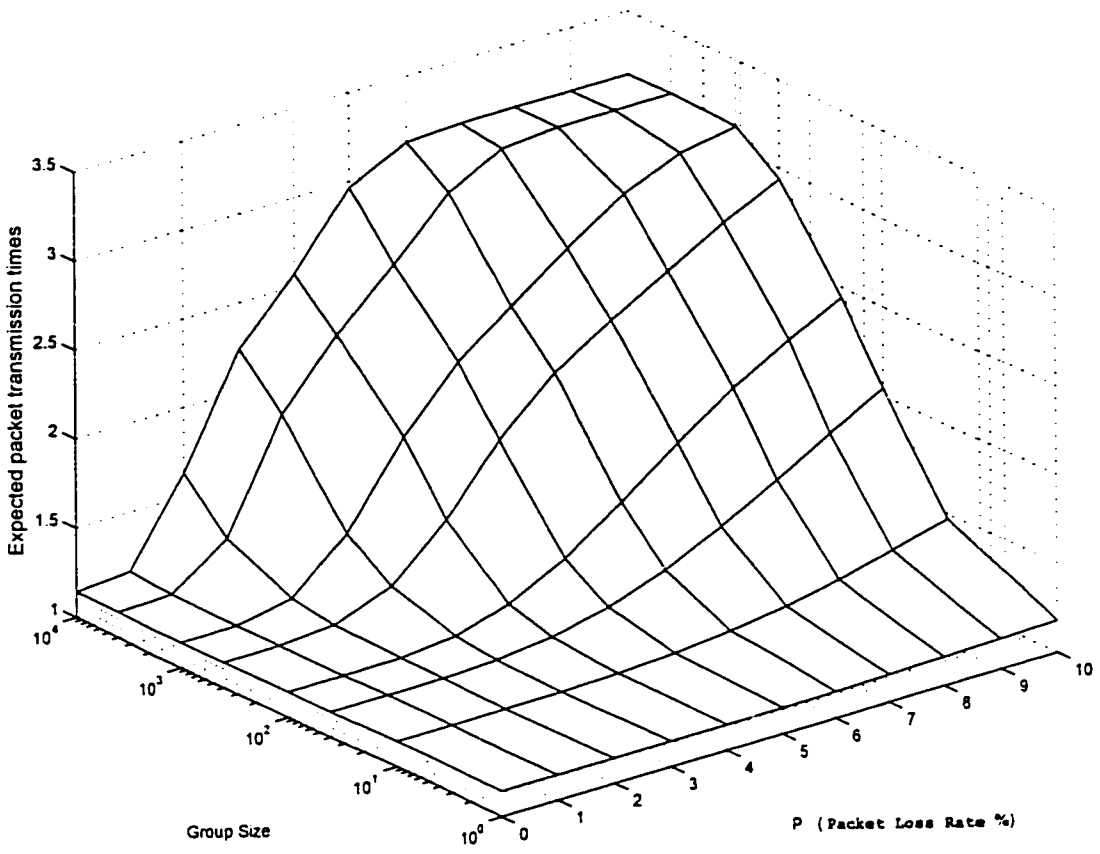

Figure 4.7 Expected packet transmission times (RS and 2 NAK)

are transmitted three times. When the loss rate is low, the flat part of the figure indicates most of the transmissions succeed at the first time. Our hybrid FEC/FEC scheme shows its merit in this loss rate range.

Figure 4.8 shows the average total time of transmissions per packet in the group for the ARQ only scheme (equation 4.10). The expected number of transmissions starts at 1 time unit in this figure because there is no parity for this case. Transmission times increase while packet loss rate and/or group size increases. Transmission times increase directly starting from the packet loss rate is more than zero. When the loss rates are high and group size is large, all original data are transmitted three times because we limit the re-transmission to twice.

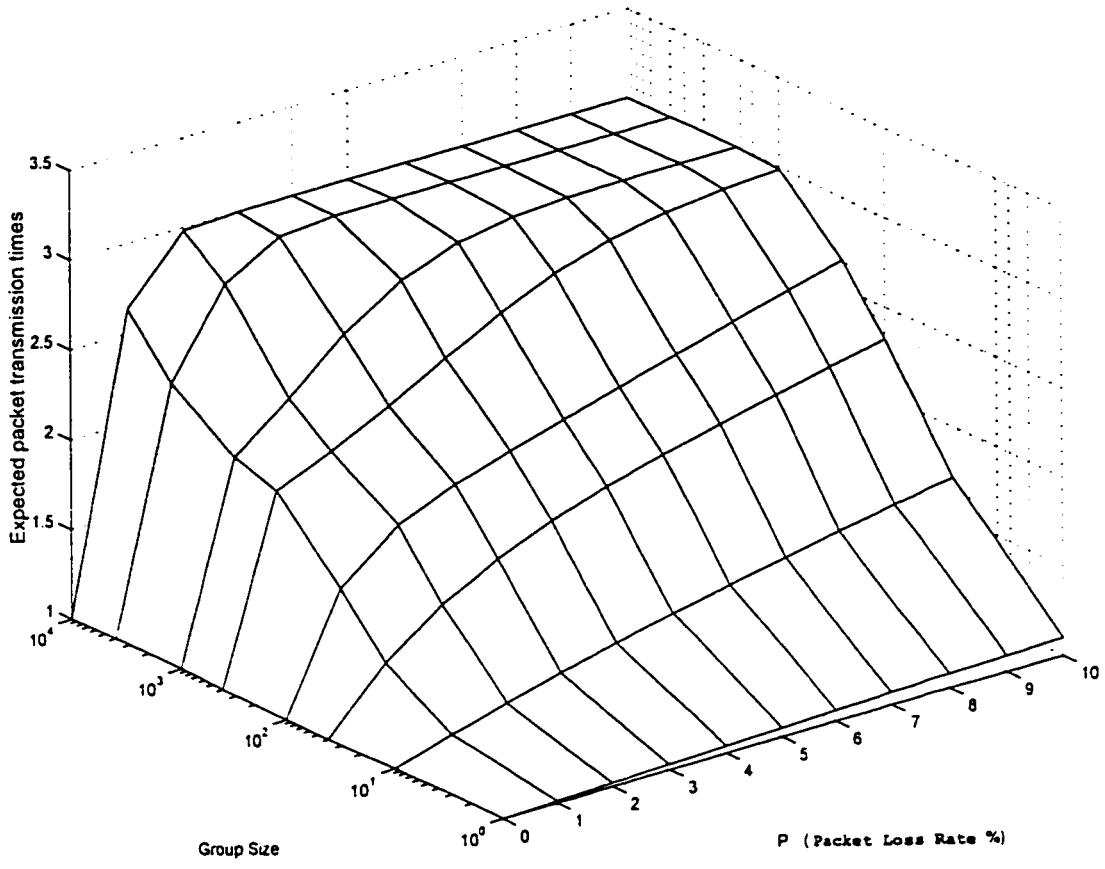

Figure 4.8 Expected packet transmission times (ARQ only 2 NAKs)

Figure 4.9 shows the comparison of the transmission times needed for the FEC/ARQ scheme and ARQ only scheme (equation 4.9, 4.10). There is a large space between transmissions of the two schemes as indicated, which shows the coding power of FEC, especially when the loss rate is less than 4% and the group size grows. When packet loss rates are high and group size is large, the FEC/ARQ scheme needs more transmissions because of the parities sent at the first time. But this does not necessarily mean the ARQ only scheme is better. Because this small amount of parities will contribute to less NAKs generated and less residual error rate, which are shown later on.

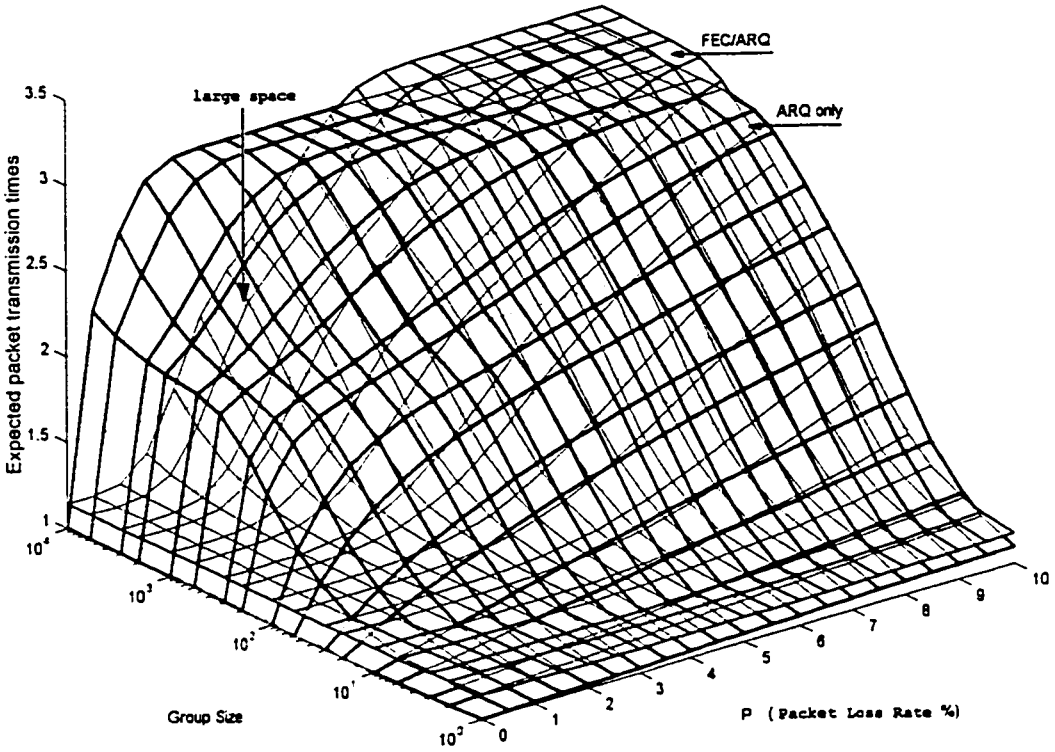

Figure 4.9 Comparison of expected packet transmission times (hybrid FEC/ARQ vs. ARQ only)

Figure 4.10a, b show the average total number of NAKs generated per packet in the multicast group for the FEC/ARQ scheme (equation 4.11).

On one hand, when the group size is small than 400 and the loss rate is less than 10%, or when the group size is small than 10000 and the loss rate is less than 4%, the number of NAKs is less than 100 per interleaved block (28 original data packets). This mean our hybrid scheme provides good performance.

On the other hand, when the group size is larger or loss rate higher, NAK implosions occur. When these happen, routers near the sender are congested, and sender faces scheduling problem in processing all NAKs. There are schemes to aggregate or to suppress NAKs, but all these schemes have their shortcomings. We try to eliminate this

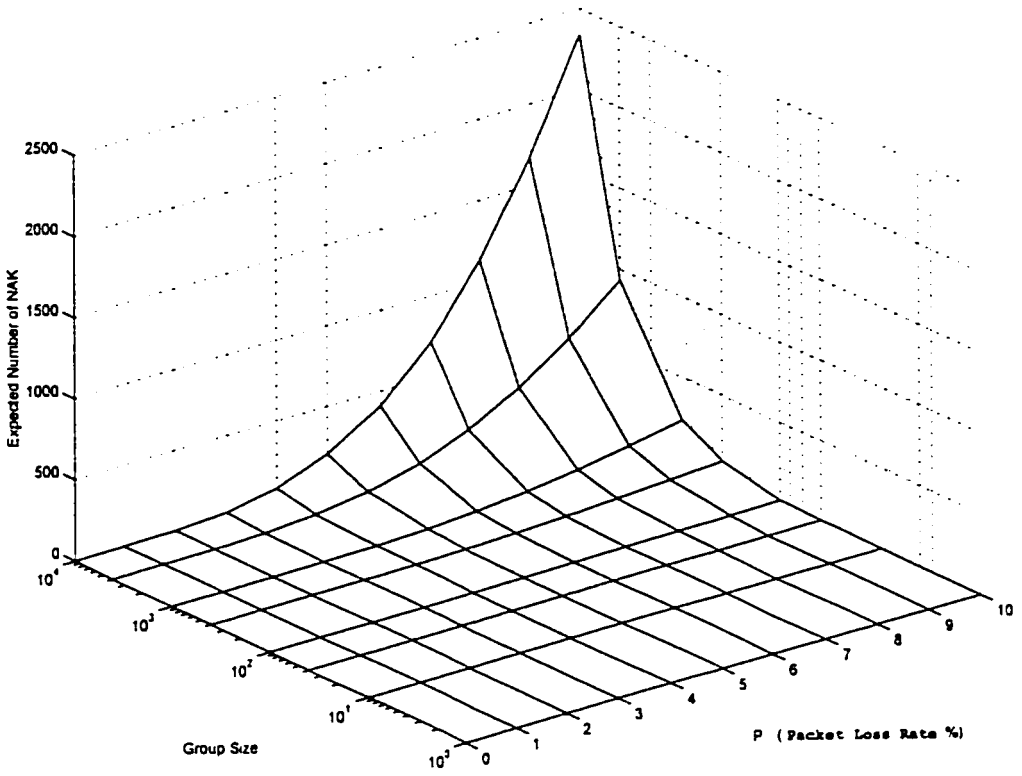

Figure 4.10a Expected number of NAK generated per window (hybrid FEC/ARQ)

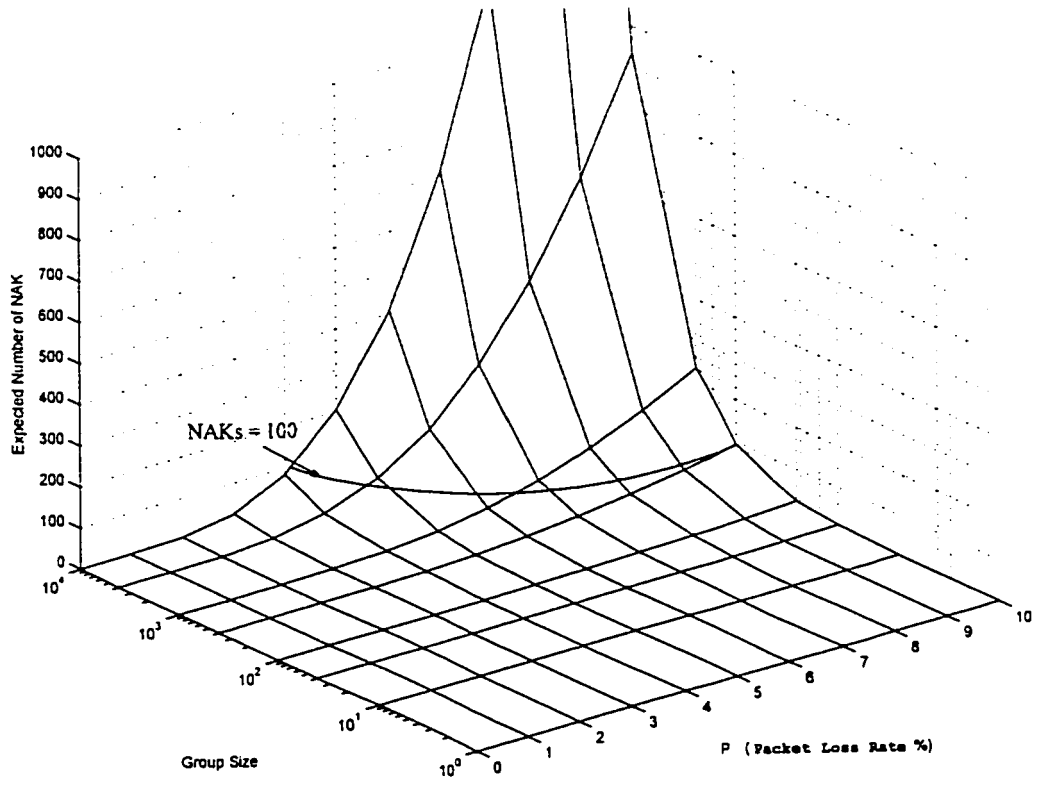

Figure 4.10b Expected number of NAK generated per window (hybrid FEC/ARQ) enlarged

NAK implosion by using more FEC redundancy, such that fewer NAKs would be generated and less NAK implosion would take place.

Figure 4.11a and b show the average total number of NAKs generated per packet in the group for the ARQ only scheme (equation 4.12). When group size is large, NAK explosion occurs even with very few packet losses, which demonstrates that the ARQ only scheme is not scalable.

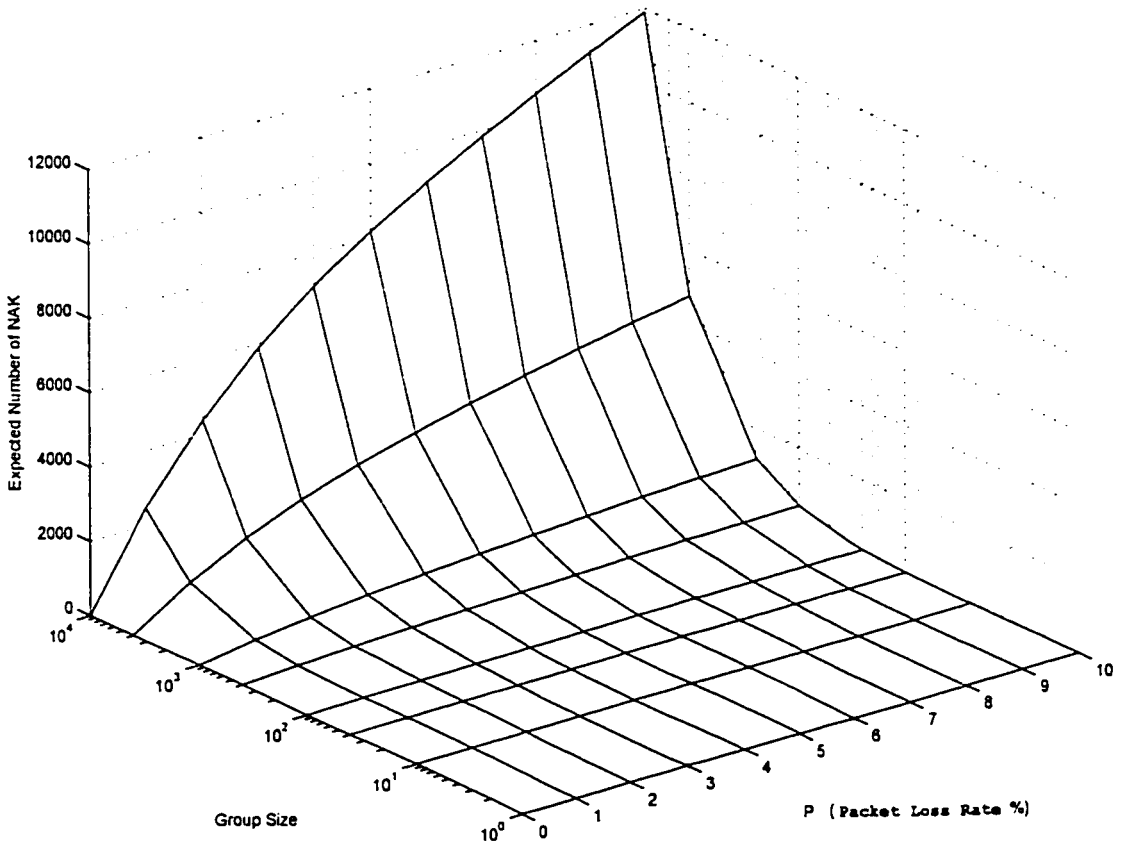

Figure 4.11a Expected number of NAK generated per window (without FEC)

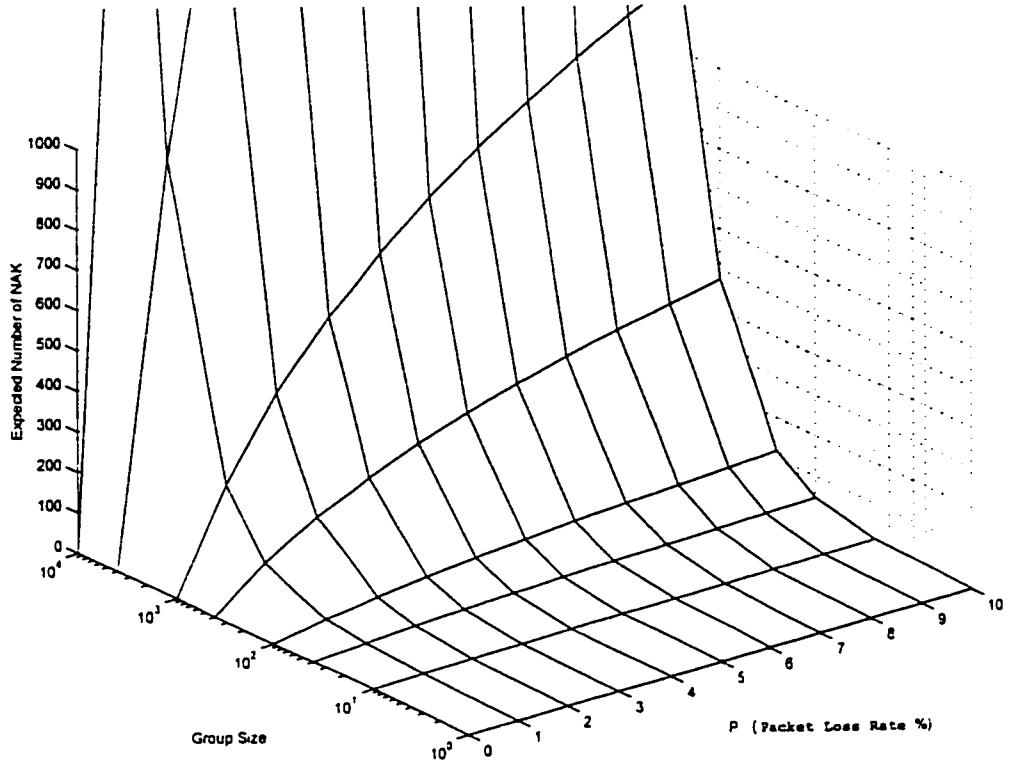

Figure 4.11b Expected number of NAK generated per window (without FEC) enlarged

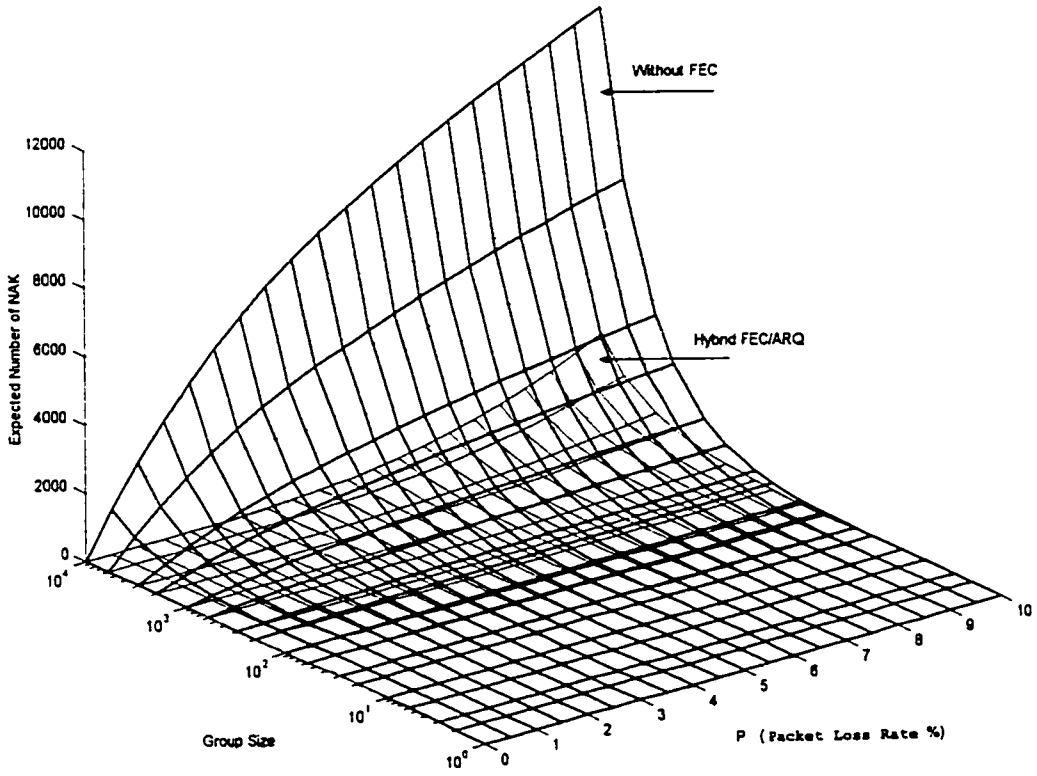

Figure 4.12 Comparison of expected number of NAK generated per window (hybrid FEC/ARQ and ARQ no FEC)

Comparing the two schemes, figure 4.12 shows the strong effect of FEC in terms of minimizing NAKs even though the amount of parities is small (equation 4.11, 4.12, FEC rate = 233/255 not 1/2).

Figure 4.13 and Figure 4.14 compare the residual error rate of the FEC/ARQ scheme and the ARQ only scheme (equation 4.3, 4.4).

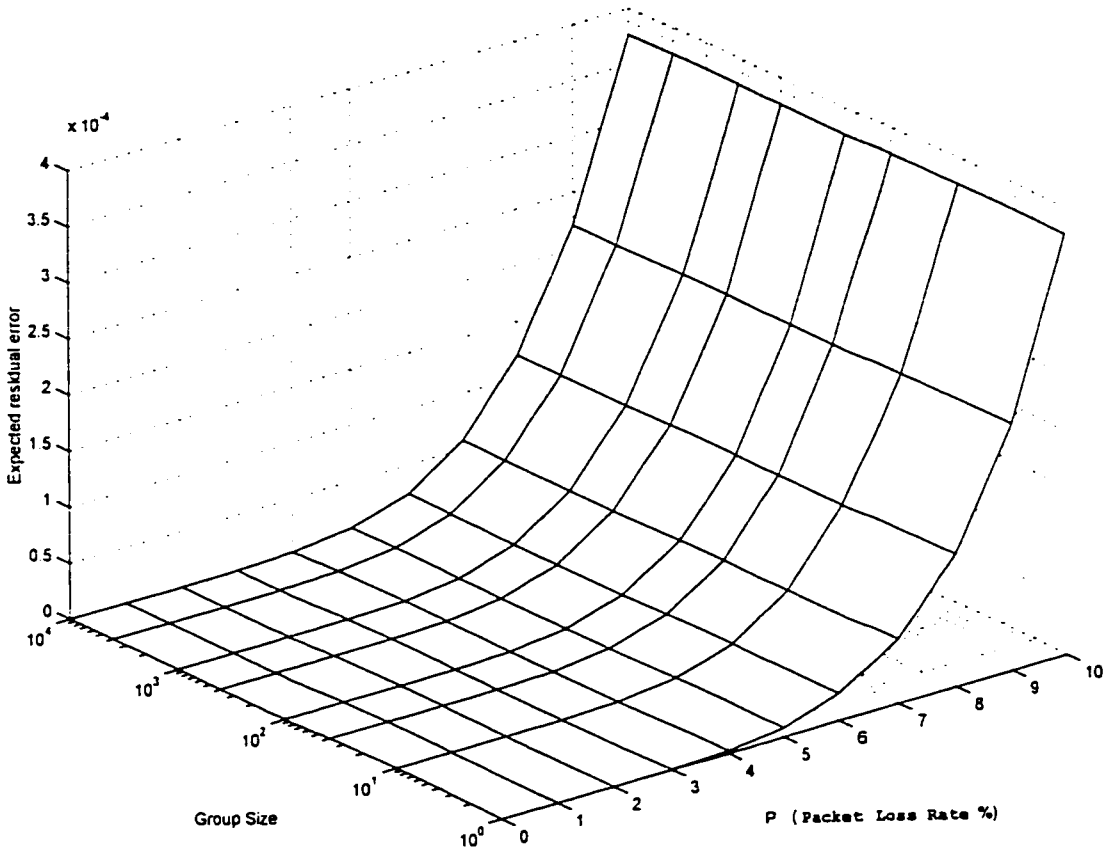

Figure 4.13 Expected residual error (hybrid FEC/ARQ 2 NAKs MAX)

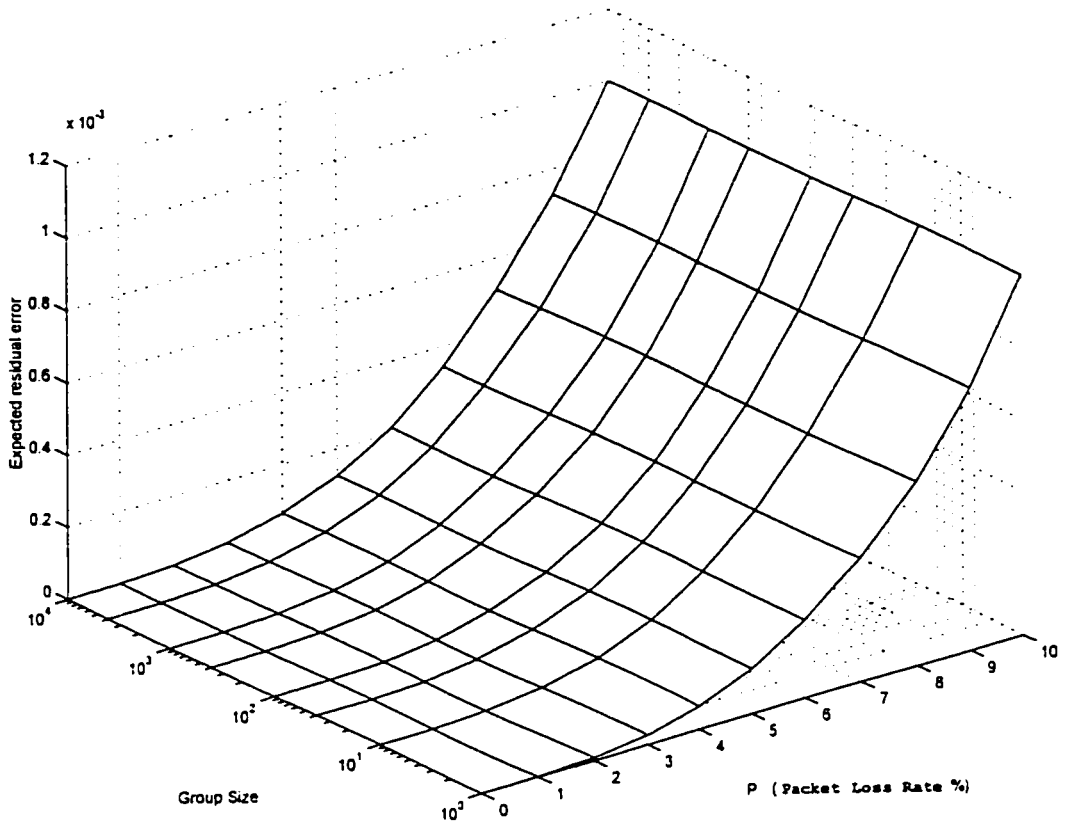

Figure 4.14 Expected residual error (ARQ 2 NAKs MAX)

In summary, Comparing the two schemes, FEC/ARQ performs better not only in terms of less transmission, but also in terms of less NAKs and lower residual error rate. There is a reasonable range of values of packet loss and group size that the hybrid FEC/ARQ scheme is applicable.

To extend the applicability of the hybrid scheme, an adaptive/hybrid scheme, which provides more redundancy in certain circumstances, is needed. In the following chapter, we examine this adaptive/hybrid scheme.

Chapter 5

Proactive FEC to Extend Scalability

In this chapter, we discuss a new adaptive media stream multicast policy that extends scalability. We try to proactively send two copies of FEC encoded words in our FEC/ARQ scheme when group size is large and packet loss rates are high [27]. The way to choose schemes is also discussed because it's a necessary part of this adaptive/hybrid scheme.

5.1 Types of redundancy

In order to minimize NAKs, we need a way to increase the amount of redundancy when the group size is large and links' loss rates are high. Changing N or K (e.g. to have higher rate FEC) in run time at sender and a large number of receivers is hard to implement. But there are other choices; the easiest way to increase redundancy is sending packets twice. For example, this way, all receivers do not need to know dynamic parameter changes (K, N values).

We all knew sending packet twice is costly in unicast, never mention how bad it is in a multicast environment. Here, we compare the following two ARQ only schemes without FEC:

- Scheme 1: Send data 1 copy, and generate maximum 2 NAKs, i.e. 2 retransmissions
- Scheme 2: Send data 2 copies, and accept only 1 NAK, i.e. 1 retransmission

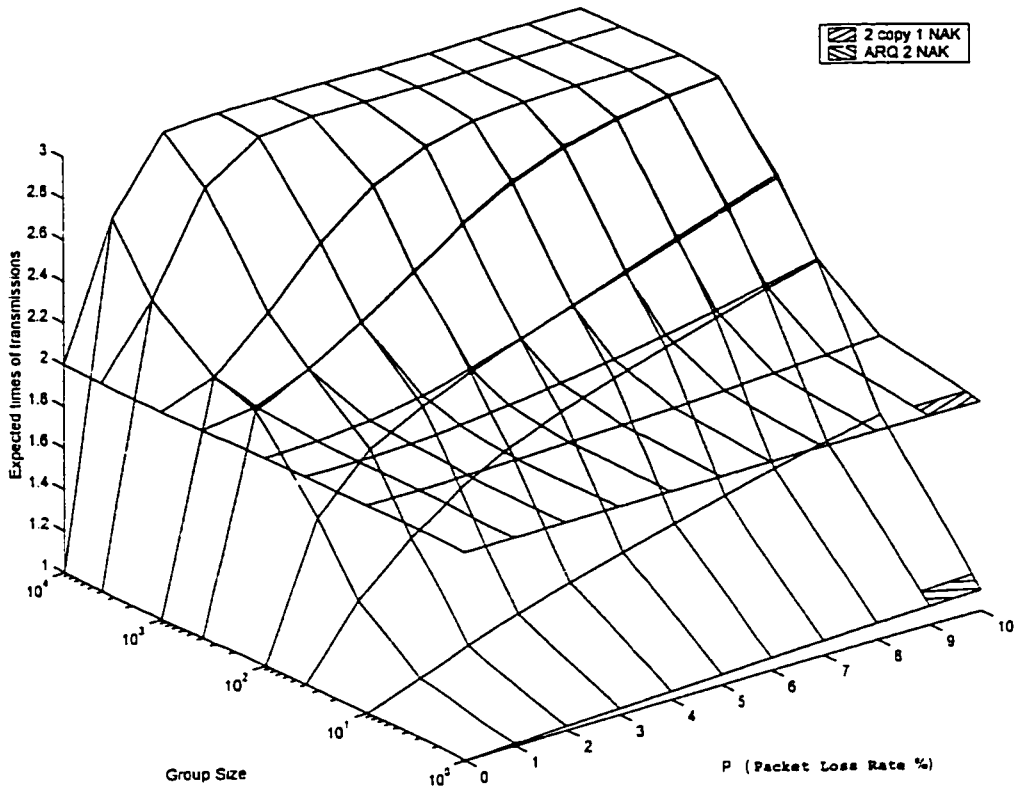

Figure 5.1 Expected transmission times (ARQ only with 2 NAKs VS. 2 copy and 1 NAK)

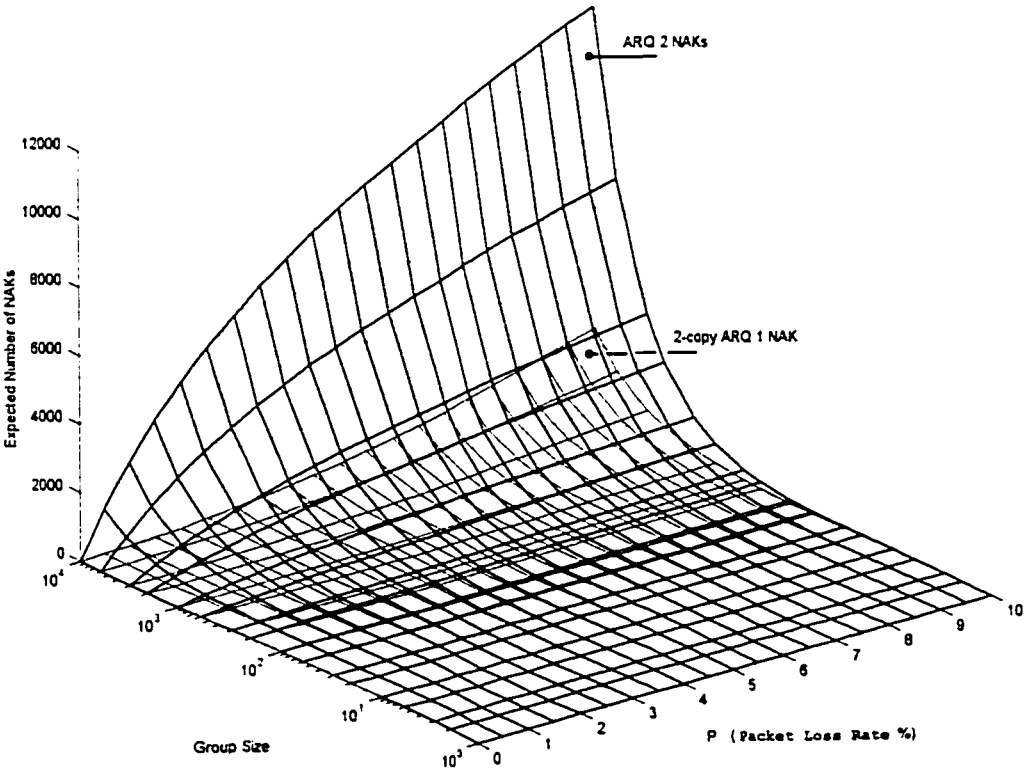

Figure 5.2 Comparison of Expected Number of NAKs Generated Per Window (2-copy ARQ 1 NAK vs. ARQ 2 NAKs)

Considering the fact that the expected residual errors are the same (equal to P^3) for these two schemes, we begin with a comparison in the number of transmissions. Equation 4.10 gives Average total transmission times per packet in the group using 1-copy-2-NAK scheme. Using the 2-copy-1-NAK scheme, the average total transmission times per packet in the group is equal to $2+\beta=3-(1-P')^U$, while $P'=P^2$. (See equation 4.7) Figure 5.1 shows expected transmission times of these two schemes. When the group size and loss rate grows up, the transmission times overlapped. This means the transmission efficiencies of these two schemes are the same in this situation.

Next, we consider number of NAKs being generated, which is a very important factor in IP multicast schemes. Equation 4.12 gives the average total number of NAKs generated per packet in the group using 1-copy-2-NAK scheme. Using the 2-copy-1-NAK scheme, the average total number of NAKs generated per packet in the group is equal to $U\cdot(1-(1-P^2)^K)$. (See equation 4.2). Figure 5.2 compares the number of NAKs generated for each of these two schemes. Because the first round of NAKs of the one-copy-two-NAK scheme is unnecessary in the two-copy-one-NAK case, the number of NAKs generated in two-copy-one-NAK scheme is much lower, which means two-copy-one-NAK scheme is better.

Now, if we can make a good decision as to where to use the two-copy scheme in terms of population U and packet loss rate P, NAKs will be suppressed, i.e. no more transmissions than the one-copy scheme would be needed.

Because the number of NAKs and residual error are still unacceptably high for the two-copy-one-NAK ARQ only scheme, we will try next an FEC/ARQ hybrid

scheme sending interleaved FEC blocks twice proactively in high population and high loss rate environments.

5.2 Adaptive FEC/ARQ/2-COPY

Suppose we apply two hybrid FEC/ARQ schemes:

- Send interleaved FEC block 1 time, and accept 2 NAKs
- Send interleaved FEC block twice, and accept 1 NAK

For the 2-copy scheme, after the block is sent twice, symbols' losses are more randomly distributed even with short burst losses in the transmission channel. The probability of getting less than K symbols is minimized. Therefore, the FEC decoder has a good chance to recover from losses.

First, we check their efficiencies. Equation 4.9 gives the average total transmission times per packet in the group for the 1-copy-2-NAK FEC/ARQ scheme. Using 2-copy-1-NAK scheme, the average total number of transmission times is equal to $\frac{2N}{K}+\alpha$, while $\alpha=1-(1-P')^U$ and $P'=\frac{1}{N}\sum_{i=N-K+1}^{N} i\binom{N}{i}(1-P^2)^{N-i}P^{2i}$, see equation 4.1 and 4.5. Figure 5.3 compares expected transmission times of these two schemes. When the group size and loss rate grows up, the number of transmission times' relation reversed. The expected transmission times for the scheme with 2-copy are less than the 1-copy scheme because fewer retransmissions are needed after 2-copy and FEC decode due to low residual error rate. So, precious bandwidth is saved.

Next, we check how many NAKs are still needed after sending RS encoded packets twice. The average number of NAKs generated by the two-copy-FEC scheme is

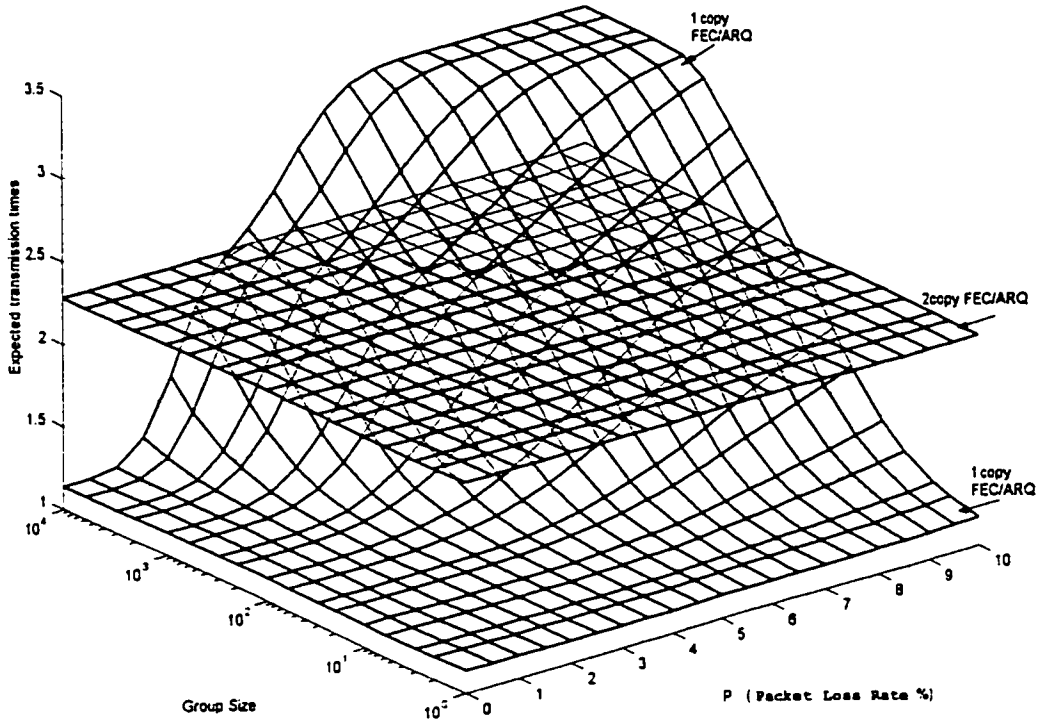

Figure 5.3 Expected transmission times (1-copy or 2-copy RS and ARQ)

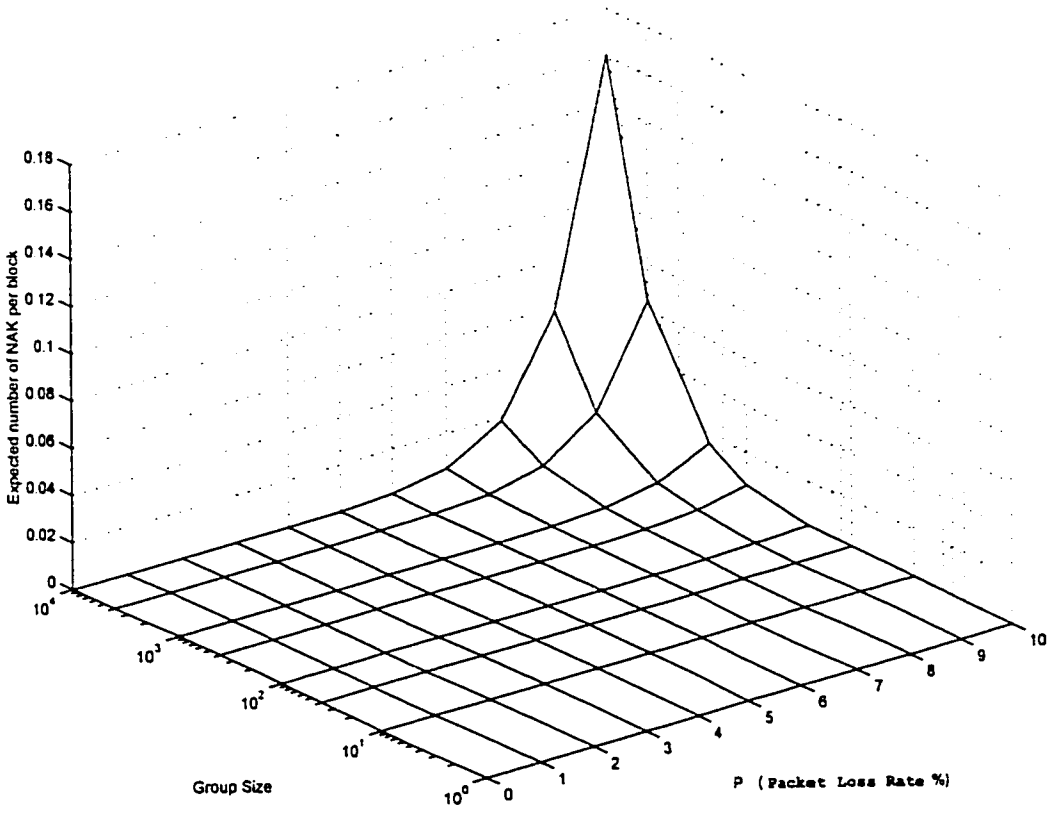

Figure 5.4 Expected number of NAK generated per interleaved block (2 copy RS and 1 NAK)

equal to $U \cdot P_{NAK} = U \cdot \left(1 - \sum_{i=0}^{N-K} \binom{N}{i} (1-P^2)^{N-i} P^{2i} \right)$, see equation 4.2. Figure 5.4 shows the average number of NAKs generated by the two-copy-FEC scheme. Expected numbers of NAKs are extremely low.

Combining enough packet redundancy and coding gain, the two-copy-FEC scheme is attractive in certain conditions for having less number of NAKs. And 1 RTT (Round Trip Time) retransmission time may be saved when the probability of packet loss is high. This implies a lower delay and delay jitter.

Using the best part of both these policies in terms of total transmission times, one can build another adaptive/hybrid scheme. We will next check the performance of this adaptive/hybrid scheme from all points of view, and will compare its results with those of ARQ only scheme.

The new adaptive/hybrid scheme can be summarized as follows:

- When packet loss rate and number of users are low, then one-copy policy is used.
- When packet loss rate and number of users exceed certain limits, then 2-copy policy is used.

Finding threshold on group size and packet loss is important for the implementation of the new adaptive/hybrid scheme. Later on in Figure 5.14, we show the way to find the threshold.

Figure 5.5 shows the expected total number of transmission times of this adaptive/hybrid scheme (derived from Figure 5.3). The front flat part of this figure demonstrates the situation that the RS decoder recovers most of the losses, only one transmission and very few retransmissions take place. The slop in the middle shows the

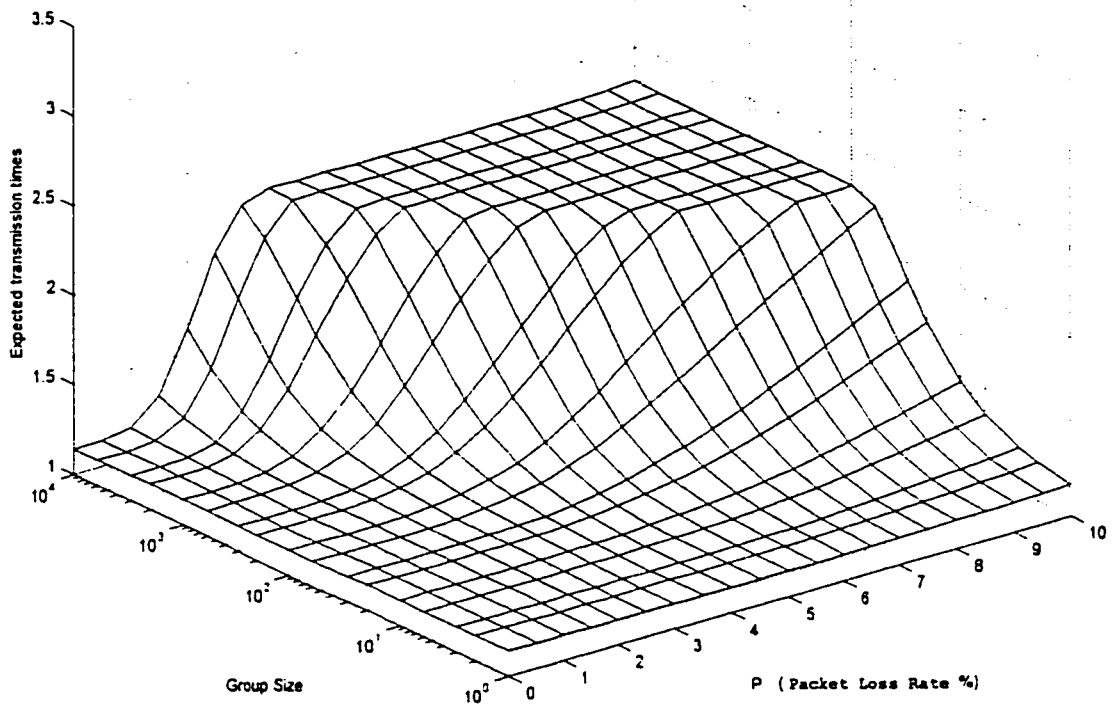

Figure 5.5 Expected transmission times (hybrid/adaptive scheme)

increasing of the number of retransmissions. When number of retransmissions reaches certain threshold, scheme with 2-copy is applied. The top flat part demonstrates this situation, which shows that RS decoder recovers most of the losses, two transmissions and very few retransmissions take place.

Figure 5.6 compares expected transmission times of the new adaptive/hybrid scheme and the ARQ only scheme (see Figure 4.8, 5.5). The results show that bandwidth occupied is less for the new adaptive/hybrid scheme.

Figure 5.7 shows the expected number of NAKs generated per block in the new adaptive/hybrid scheme (see Figure 4.10, 5.4, and 5.5). The maximum is 25 NAKs per block in this figure. Compared to 28 data packets for the block, it means about

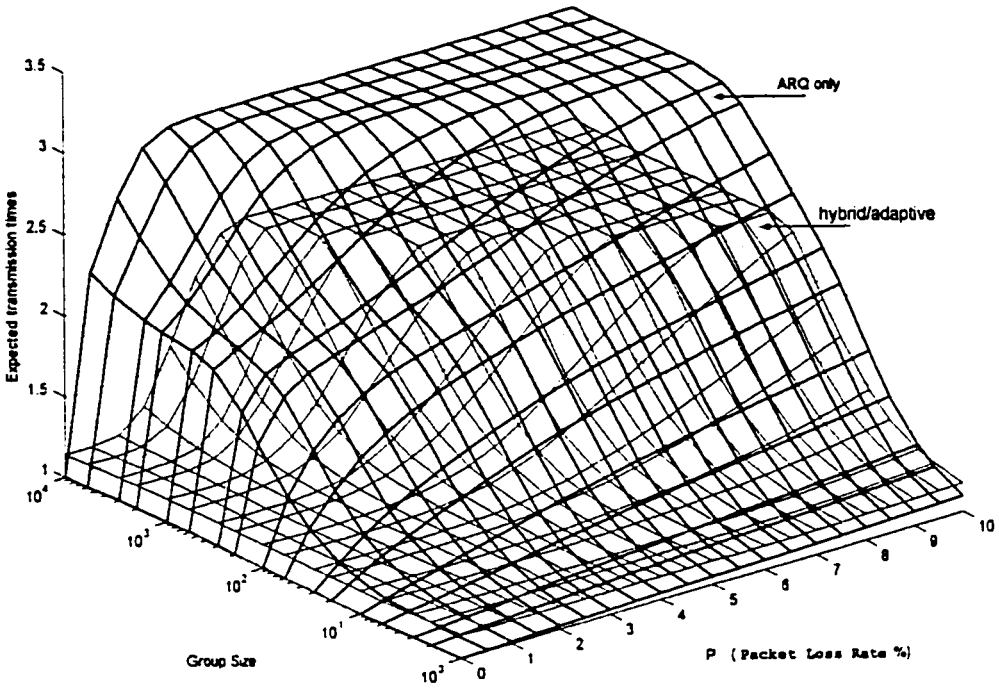

Figure 5.6 Comparison of expected transmission times (hybrid/adaptive vs. ARQ only)

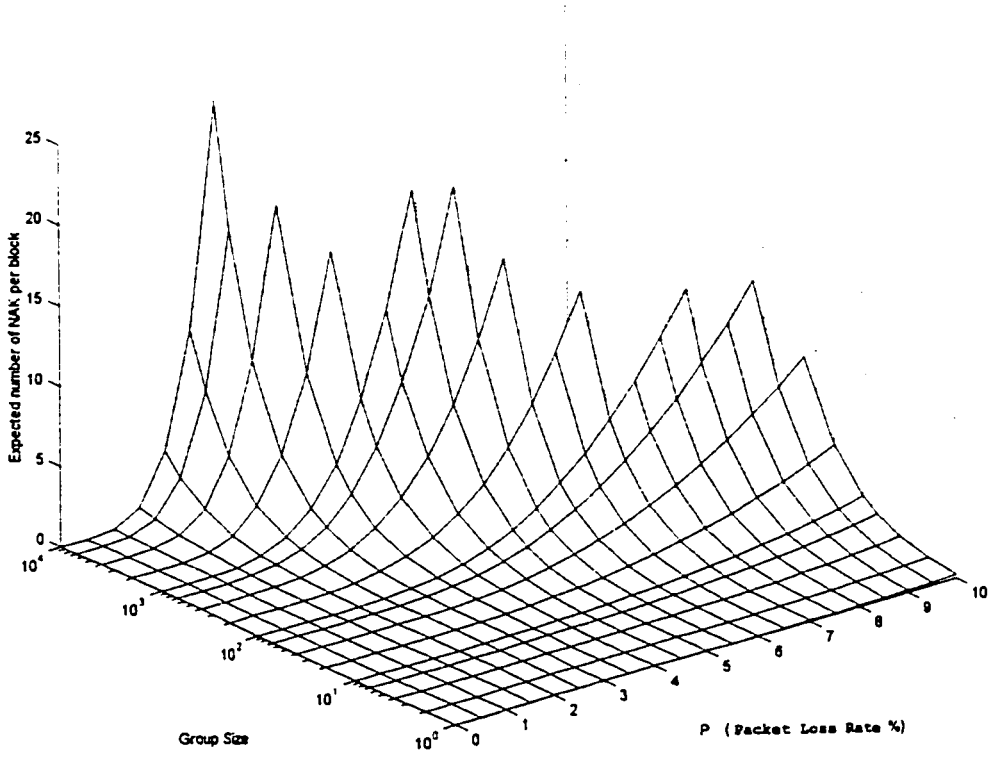

Figure 5.7 Expected number of NAKs generated per interleaved block (hybrid/adaptive scheme)

maximum 1 NAK generated to 1 data packet, which means a reasonable load for the sender to process for very large population with high packet loss rate.

Figure 5.8 compares the expected number of NAKs generated per block in the new adaptive/hybrid scheme and in the ARQ only scheme (see Figure 5.7, 4.11) . From this figure, we can observe that the effect of the new adaptive/hybrid scheme in term of minimizing NAKs is very noticeable. Studying NAK aggregating and suppressing schemes [26], one can easily draw a conclusion from our results: preventing NAKs as in the new adaptive/hybrid scheme should be better than aggregating or suppressing NAKs as in the schemes without using FEC.

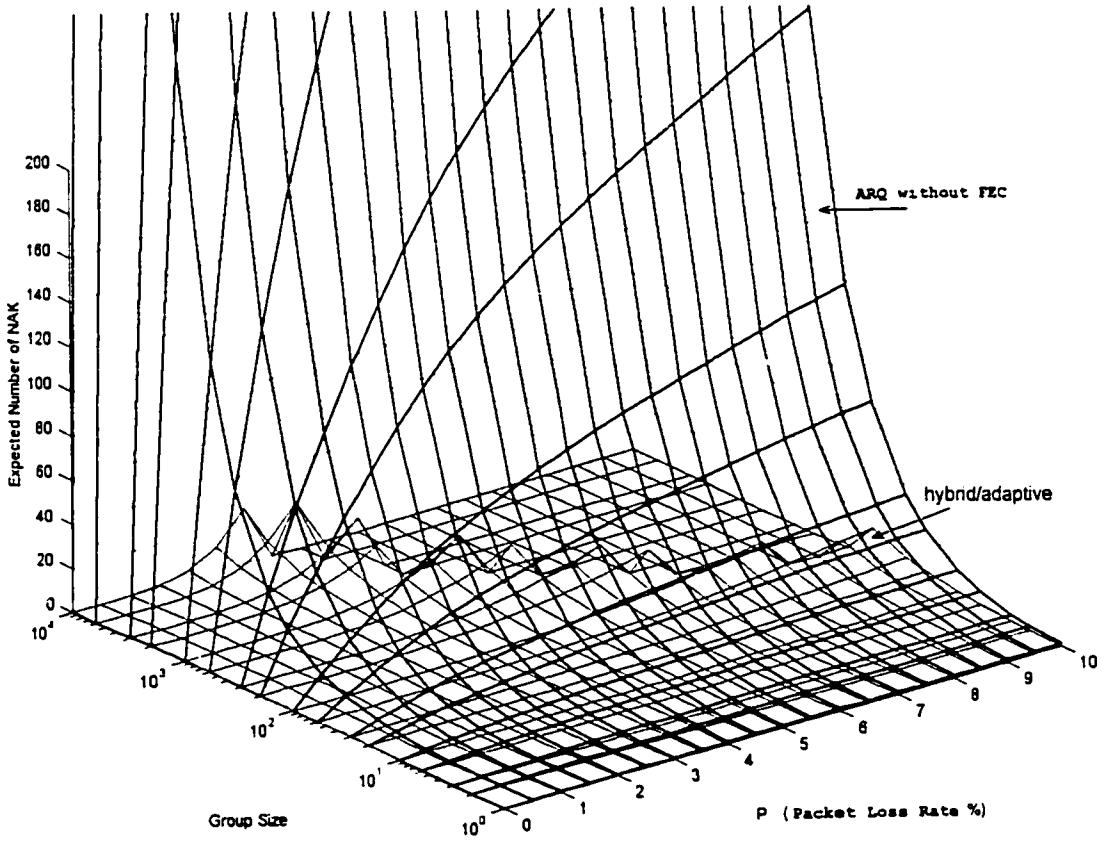

Figure 5.8 Comparison of expected blumber of NAK generated per window (hybrid/adaptive vs. ARQ only)

Equation 4.3 gives the expected residual error in the group for the 1-copy-2-NAK FEC/ARQ scheme. Using 2-copy-1-NAK scheme, the expected residual error is equal to $P \cdot P' = \frac{P}{N} \sum_{i=N-K+1}^{N} i\binom{N}{i}(1-P^2)^{N-i} P^{2i}$, see equation 4.1 and 4.3.

Figure 5.9 shows the expected residual errors of the new adaptive/hybrid scheme.

Figure 5.10 compares the expected residual errors of the new adaptive/hybrid scheme and the ARQ only scheme (see Figure 5.9 and 4.14).

Figure 5.11 shows the average transmission times of the new adaptive/hybrid scheme while experiencing higher loss rates (equations used are same with that of Figure 5.5). The front flat part of this figure demonstrates the situation when the RS

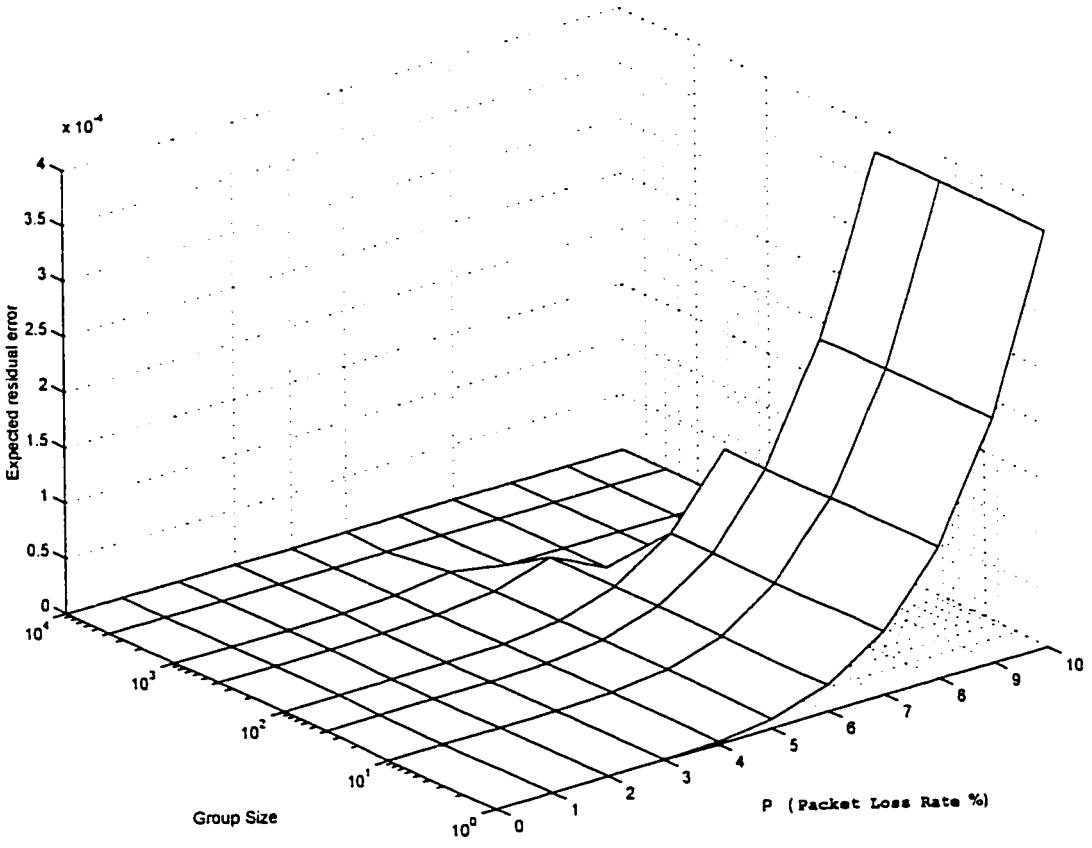

Figure 5.9 Expected residual error (hybrid/adaptive scheme)

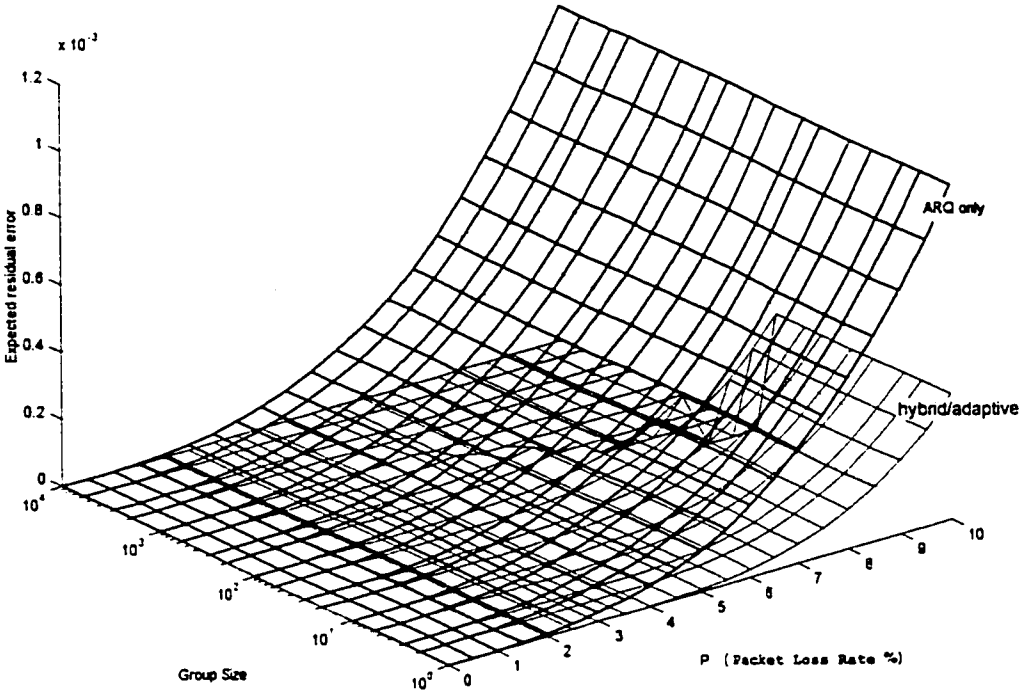

Figure 5.10 Comparison of expected residual error (hybrid/adaptive vs. ARQ only)

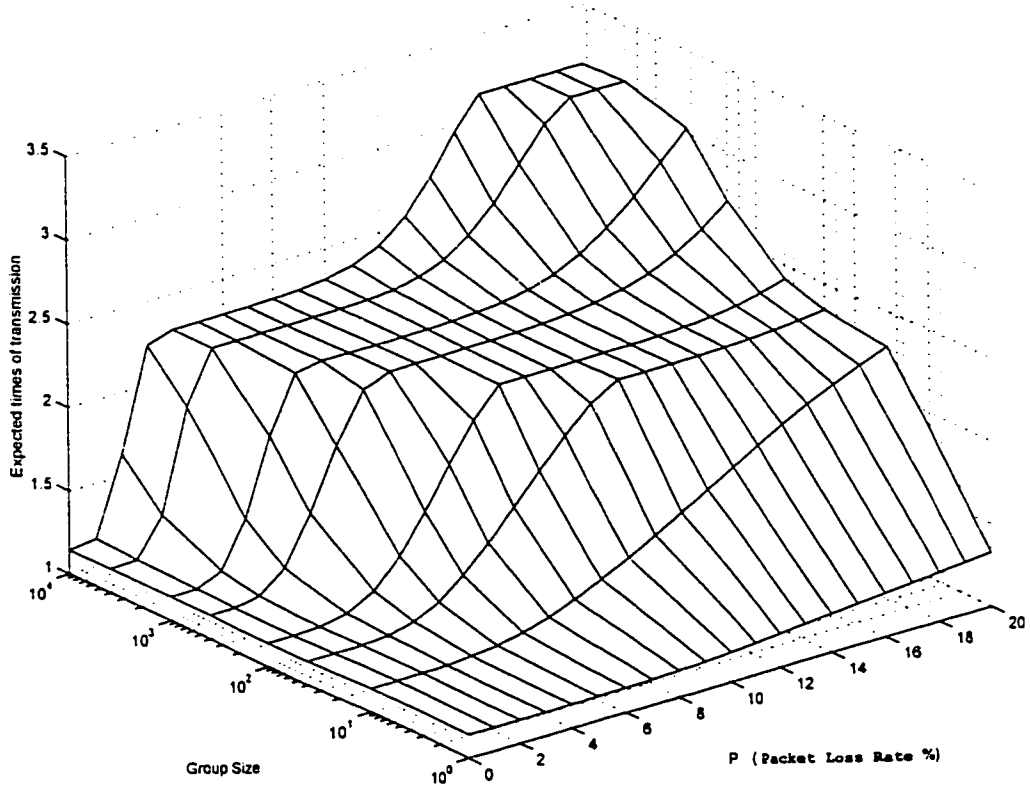

Figure 5.11 Expected transmission times (hybrid/adaptive scheme, high loss)

decoder recovers most of the losses, only one transmission and very few retransmissions take place. The first slop shows the increase of the number of retransmissions for the one-copy policy. When the number of retransmissions reaches certain threshold, the technique switch to use a 2-copy policy. The flat part in the middle demonstrates the situation when the RS decoder recovers most of the losses, two transmissions and very few retransmissions take place. The second slop shows the increase in the number of retransmissions while the two-copy scheme prevails.

Figure 5.12 shows the expected number of NAKs generated per block in the adaptive/hybrid scheme under a high loss rate (equations used are same with that of Figure 5.7).

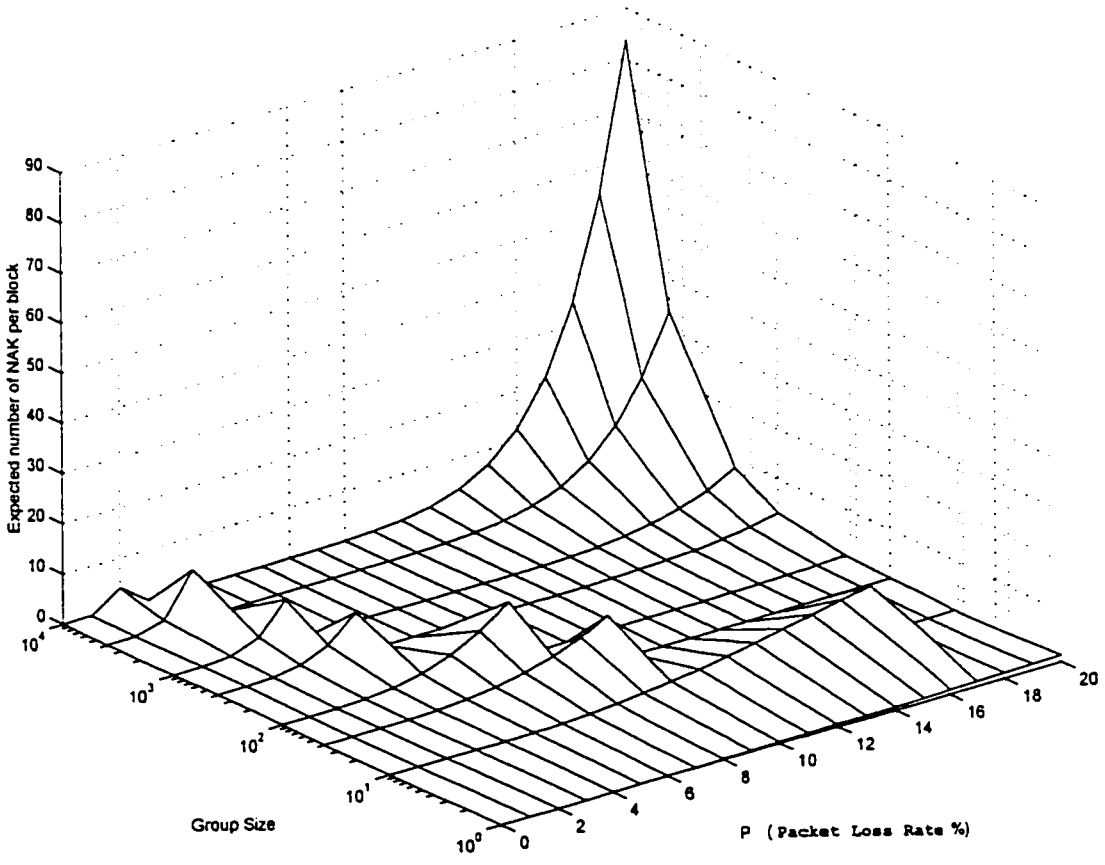

Figure 5.12 Expected number of NAKs generated per interleaved block (hybrid/adaptive scheme, high loss)

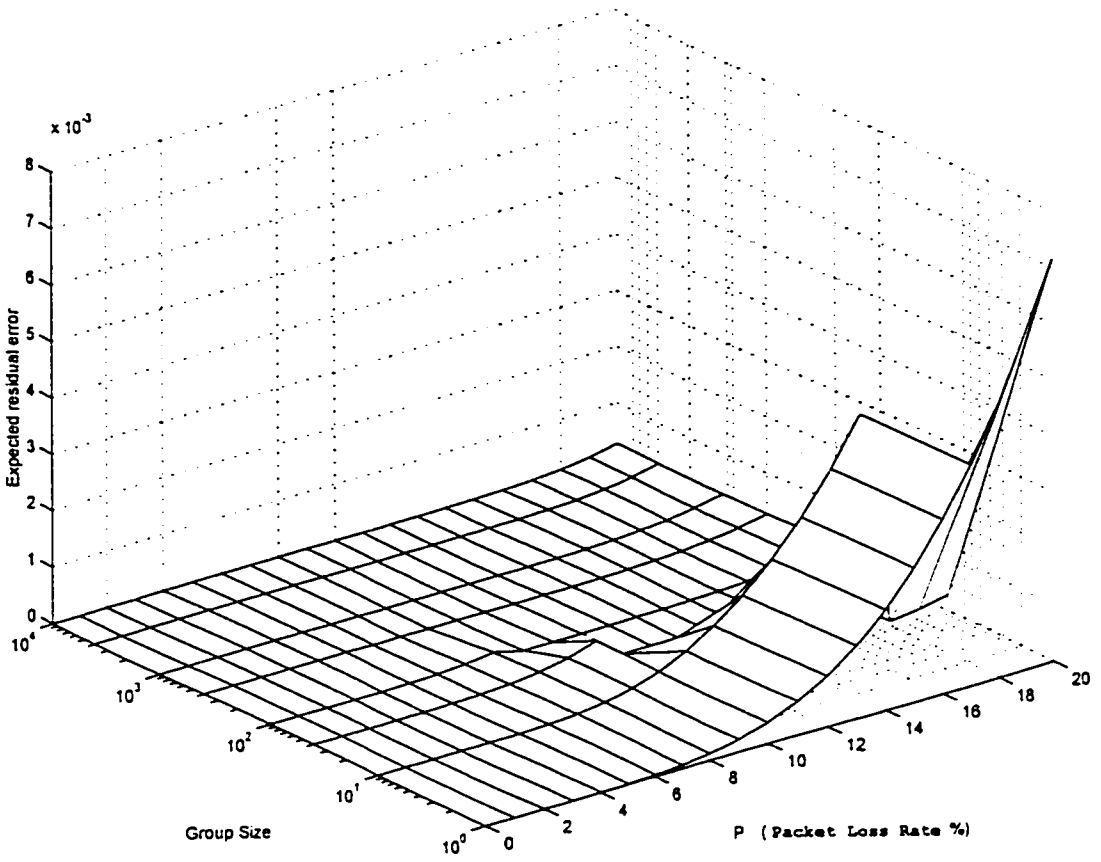

Figure 5.13 Expected residual error (hybrid/adaptive scheme, high loss)

Figure 5.13 shows the expected residual errors of the adaptive/hybrid scheme under a high loss rate (equations used are same with that of Figure 5.9).

By now, we have seen how appropriate amount of redundancy combined with FEC reduces feedback and bandwidth as compared with simple ARQ schemes in reliable multicast. By costing less RTT, the new adaptive/hybrid scheme can also help to meet real-time guarantees.

5.3 Policy decision-making method

We seem to have found an easy way to achieving fewer transmissions, less NAKs, less residual errors, less delay, and less delay jitter in a multicast environment.

But this new adaptive/hybrid scheme is still incomplete. It cannot be true before we find a real-time, low-cost, scalable, simple and effective policy decision-making method (e.g. how one switches from policy one to policy two in practice). Therefore, we need to explore this a little further.

For a certain size of group, the total number of NAKs received per block at the sender reflects the condition of network. Therefore, one can decide to switch to 2-copy scheme by counting the number of NAKs received at sender. Figure 5.14 shows the new hybrid/adaptive policy decision-making threshold selected by investigating of Figure 5.3 and estimating the number of transmissions and NAKs at the intersection of the two

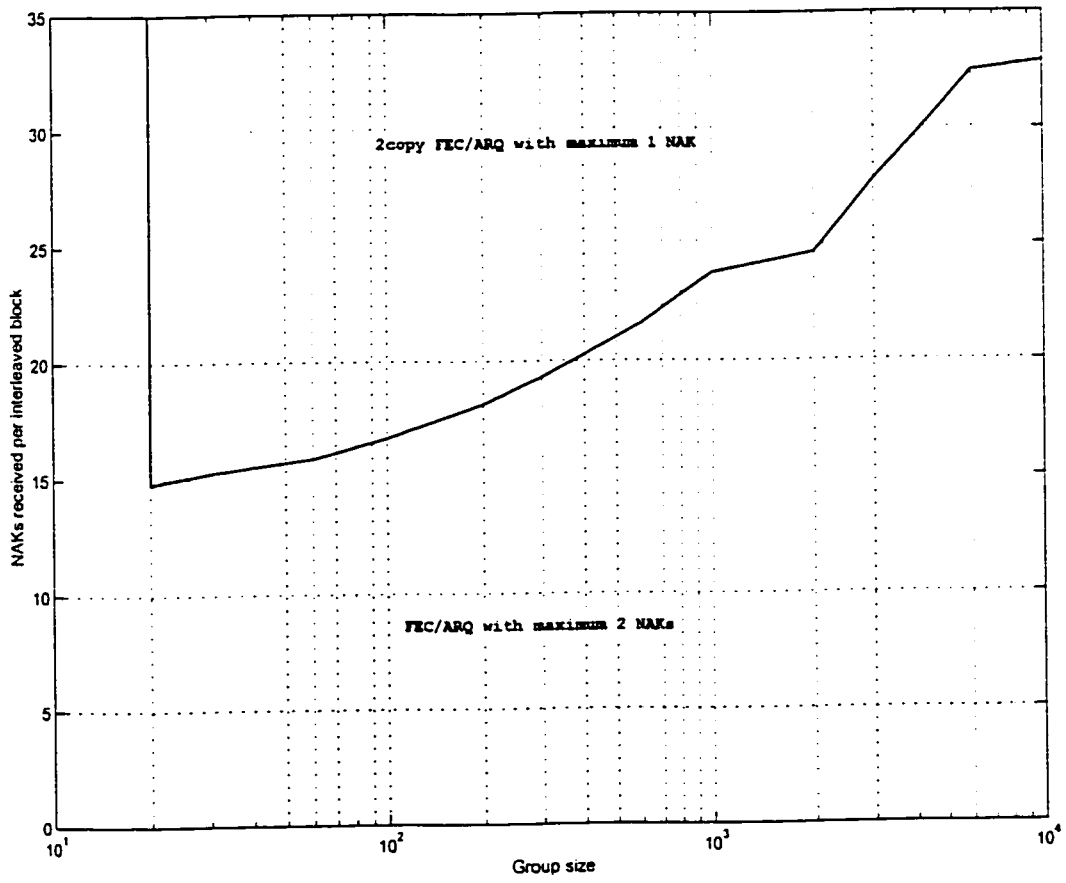

Figure 5.14 Hybrid/adaptive scheme decision making threshold

policies. When the group size changes from 20 to 10000, the threshold changes from 15 to 33 only. This result shows that the threshold is not very sensitive to the size of the multicast group. This means, even if we cannot make a really good approximation of the independent-loss-equivalent group size, we can still make a good decision in switching between the one and two copy policies.

For implementation of our new adaptive/hybrid scheme, the sender knows the network condition by counting NAKs generated per window and measuring RTT changes. It switches to 2-copy FEC/ARQ status when NAKs exceed the threshold for the current independent-loss-equivalent group size. It rolls back to one-copy FEC/ARQ status when times out.

5.4 Discussions of adaptive/hybrid scheme

The 2-copy plus FEC scheme increases the successful transfer probability of a packet and, consequently, decreases the required number of retransmission requests. Thus, this strategy reduces the effects of NAK implosion for a large number of receivers. Moreover, it also reduces the packet transmission delay, since copies of every packet are sent in advance.

Hybrid ARQ where initial transmission is protected by redundant information provides improved delay characteristics. In addition, our adaptive/hybrid scheme provides further improvement in delay characteristics. It helps to delivery packets to the receivers correctly and efficiently before the deadlines. At same time, the policy will cause much less number of NAKs, which is a very important factor in scalability of multicast.

Chapter 6

Conclusions and Future Work

6.1 Concluding remarks

We have presented a combination of RS FEC/ARQ and interleaving techniques, which provide good QoS experimental results in terms of final efficiency. We have chosen selective repeat as our ARQ scheme to minimize the load at both the sender and the network. Due to the indeterminacy of RTT, the limit of processing ability of PC and the complexity of RS decoding, receivers can only cope with certain transmission rate at certain loss rate. Not to make things worse, we re-transmitted only original data. Which means there was no RS decoding again after re-transmission. Matching well with our analytical results, our laboratory work showed the practical feasibility of a software implementation of the hybrid FEC/ARQ and interleaving scheme. The memory consumed by the hybrid scheme at the client side is less than 200kB. This scheme will help to multicast stream media to up to 10000 ordinary PC receivers at a streaming rate of 1.5Mbps per receiver.

We have presented comparative performance analysis of the hybrid FEC/ARQ scheme and the ARQ only scheme and have shown the influence of varying sending

rates, group sizes, and packet loss probabilities. We have evaluated the advantages achieved by our schemes in terms of the average transmission times, average number of NAKs generated and expected residual errors.

For scalable multicast applications requiring low-latency or having real-time deadline requirements, we derived an adaptive/hybrid scheme step by step from our analytical results.

We have found hybrid FEC/ARQ better not only in terms of less transmission, but also in terms of less NAKs and lower residual error rate. The hybrid FEC/ARQ scheme is applicable when the packet loss rates are less than 4%. In addition, to adapt to higher loss rate, a two-copy-FEC-ARQ scheme was adopted. We have shown an adaptive media stream multicast policy that extends scalability further. Finally, the way to choose among different schemes has been discussed.

We have demonstrated how appropriate amount of redundancy combined with coding decreased the bandwidth overhead, eliminated NAK implosion, reduced the residual error rate, and helped to meet real-time guarantees. Scalability is achieved for large number of receivers up to 10 thousand while the independent loss rate is up to 20 percent in a real-time multicast.

One important factor of these schemes is that all their performances are supposed to be built on top of a low-cost platform. Our low-cost goal means no special hardware at sender and receivers, no special routers and good bandwidth efficiency.

Even though the assumption of a homogeneous multicast tree is only the best case for FEC reliable multicast schemes, the test and analytical results will still give us some guidance in implementation of real application.

In this work, the new hybrid FEC/ARQ scheme is the result of our project group meetings. I completed a simple end-to-end implementation test and the analysis on group communication performance (chapter 3, 4, 5).

6.2 Suggestions for further research

With shared loss and asymmetric tree topology the whole repair efficiency may be explored further.

The relation among the pattern of burst loss, the interleave depth of the RS codes, the best RS codes length, the processing ability of receivers and the delay sensitive characteristic of application may be explored further.

One should notice that the two-copy policy is only a complement policy to our hybrid FEC/ARQ scheme. When the status keep in 2-copy most of the time, an FEC codes with more redundant may be chosen because pure FEC codes are more efficient than 2-copy plus FEC that we investigated.

The way to embed our hybrid FEC/ARQ scheme to the intermediate routers and tunnels to reduce residual error and enhance reliability may be explored further.

One may also start to test this adaptive/hybrid schemes in large-scale by implementing a real server and a free downloadable client software with statistics functions. Some implementation optimizing suggestions are as follow:

Sender periodically updates a log file of receivers' ID, NAK's sequence number and RTTs by measuring their NAK arrive times. A process at sender checks this file periodically and updates the independent loss equivalent group size accordingly. By time, the process knows the receiver with least RTT who shares the common loss with

other receivers. Then, the process sends NAK suppress messages to those receivers and leave the one with least RTT to keep informing packet losses. The receivers received NAK suppress message will keep silence until times-out or experiencing a loss increase, which may be caused by some group members' leaving.

This scheme reduces feedback further without the need of special routers to aggregate of feedback.

After the sender's process sends NAK suppression messages, the process should update the independent loss equivalent group size accordingly.

The mechanism used for measuring the round trip time between a receiver and the sender is as follow. After a block is sent at sender, a timestamp is locally recorded for the sequence number. When an NAK is received for the sequence number, the difference between the current time and the recorded timestamp gives RTT.

After switching to the 2-copy scheme, the adaptive/hybrid policy needs to roll back to 1-copy scheme using a timer. The rollback timer's setting depends on the network congestion status changes' pattern. When the status changes slowly, which depends on factors like daytime, night or rush hours, the rollback timer could be chosen longer. Choosing longer timers helps to reduce processing load.

Appendix A

Hybrid FEC/ARQ Pseudo C Code

```
/* pseudo C code for the client and server process */
/* server side: */

/*  TimeOut events: */
   switch(type){
      case re-transmission
         re-transmit(repair_data);
      case transmission
         transmit(fresh_data);
         if (not_end_of_stream)
            add_timer(transmission);  /* for rate-control */
         else
            add_timer(send_last_data);
      case send_last_data
         transmit(last_data);
         prepare_close_session();
      }
  main loop:
   {
      while(select(req or NAK)) recvfrom(); /* select receive from sockets */
      switch(type){
         case(req)
            add_timer(transmission);
            break;
```

```
        case(NAK)
          if(seq_curent - seq_nak < limit) add_timer(re-transmission);
          break;
        otherwise warning();
      }
    }

/* client side: */

/* TimeOut events:  */
/* the only type of TimeOut event here is send_NAK */

    build(NAK_packet);  /* include seq and CRC */
    send(NAK_of_the_window);
    NAK_counter + 1;
    if(NAK_counter < NAK_MAX_counter)
      add_timer(NAK_timeout);

/*  main loop: */
    while(recvfrom(server))
    {
     switch(type){
      case(fresh_data)
        update(NN_counter);
        update(KK_counter);
        if(new_window_data)
          {
          record(timestamp);   /* we may have a timer start here for NAK */
          if(NN_counter < K)
            add_timer(NAK_of_window);
            /* NAK event check NN_counter again   */
                    /*  and may add_timer for another NAK. */
```

```
    }
 if(NN_counter >= K)
    {
     stop_RS(last_window);
     copy current_win_lostlist to window_lostlist;
     update(lost_window_index);
    }
 deinterleave();
 break;
case(repair_data)
 check_busy_window:
 if(seq_of_window == current_seqh)
    {
       deliver();
       updata lost_list & flags
    }
 else {
              check(window_lostlist and seq);
      if(match){
                        if(!dup && KK_counter == K)
                          {
                          CRC_check();
                           deliver();
                          update(list&flags);
                          }
                        }
     else discard() /* expired */
     }
 break;
case(last_data)
 if(out_of_order)
    NAK_last();
```

```
 else {
    CRC_check();
    deliver();
    if(last_len == received_last)
    close_session();
   }
 break;
otherwise warning();
} /* end_of_switch_type */
if((KK_counter == 223) && !(need_RS_recover_from_error == 1))
   {
   CRC_1word();
   if(CRC == 0)
      {
      deliver();
      curent_window ++;
      }
   else
      need_RS_recover_from_error = 1;
   }
else if(NN_counter >= 223) && !need_RS_recover_from_error \\
                    && (NN_counter_nextwindow < 223)
   { RS_decode_1word();
    CRC_check();
    if(CRC == 0){
                          deliver();
                 }
    else need_RS_recover_from_error = 1;
   }
if (need_RS_recover_from_error && NN_counter > 234)
   {
         RS_decode_1word();
```

```
        if(CRC == 0)
          {
                  deliver();
          need_RS_recover_from_error = 0;
          }
        /* else may add_timer(NAK); */
        }
      if(NN_counter_nextwindow > 223)
        {
              move_onto_next_window;
         report_error;
        }
      if(RS_decode_flag)
        {
           RS_decode_block();
           CRC_check();
           deliver();
           }
    }
}
```

References

[1] Vicki Johnson and Marjory Johnson, "How IP Multicast Works - An IP Multicast Initiative White Paper", Web: www.ipmulticast.com, visited in Nov. 2001

[2] M. Hofmann, "A Generic Concept for Large-Scale Multicast", In B. Plattner, editor, Proc.International Zuerich Seminar, Volume 1044 of LNCS, pp.95-106, Springer Verlag, February 1996

[3] H. Eriksson, "MBONE: The Multicast Backbone", Communications of the ACM, Vol. 37, No. 8, pp.54-60, August 1994

[4] R. Braudes and S. Zabele, "Requirements for multicast protocols", Request for Comments(Informational) RFC 1458, Internet Engineering Task Force, May 1993

[5] S. Armstrong, A. Freier, and K. Marzullo, "Multicast Transport Protocol", Request for Comments (Informational) RFC 1301, Internet Engineering Task Force, February 1992

[6] E. Klemmer, "Subjective Evaluation of Transmission Delay in Telephone Conversations", Bell Systems Technical Journal, vol.46, pp.1141-1147, July 1967

[7] D. Towsley, J. Kurose, and S. Pingali, "A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols", IEEE Journal on Selected Areas in Communications, 15(3) pp.398-406, 1997

[8] S. Pejhan, M. Schwartz, and D. Anastassiou, "Error Control Using Retransmission Schemes in Multicast Transport Protocols for Real-Time Media", IEEE/ACM Transactions on Networking, 4(3) pp.413-427, June 1996

[9] S. B. Vicker, "Error Control Systems for digital communications and storage" Prentice-Hall, 1995.

[10] J. Nonnenmacher, E. W. Biersack, and D. Towsley, " Parity-Based Loss Recovery for Reliable Multicast Transmission", IEEE/ACM Transactions on Networking, Vol. 6, No.4, pp.349-361, August 1998

[11] Rizzo, L., and Vicisano, L., "Effective Erasure Codes for Reliable Computer Communication Protocols", ACM SIGCOMM Computer Communication Review, Vol.27, No.2, pp.24-36, Apr 1997

[12] J.H.Jeng , and T.K.Truong, "On Decoding of both Errors and Erasures of a Reed-Slolomon Code using an Inverse-Free Berlekamp-Massey Algorithm, IEEE Transactions on Communication, Vol. 47, No. 10, October 1999

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

[14] B. Rajagopalan, "Reliability and Scaling Issues in Multicast Communication", Proceedings of ACM SIGCOMM 92, pp.188-198, September 1992

[15] C. Papadopoulos, G. Parulkar, and G. Varghese, "An Error Control Scheme for Large-Scale multicast Applications", Proceedings IEEE INFOCOM'98 Conference on Computer Communications, vol.3 pp.1188-96, March 1998

[16] D. Rubenstein, J. Kurose , and D. Towsley, "A Study of Proactive Hybrid FEC/ARQ and scalable Feedback Techniques for reliable, real time Multicast ", Computer Communication Journal , Elsevier Publisher, Vol. 24, pp.563-574, 2001

[17] B.N. Levine, and J.J. Garcia-Luna-Aceves, "A Comparison of Reliable Multicast Protocols", Multimedia Systems(ACM/Springer), Vol. 6, No.5, August 1998

[18] C. Diot, W. Dabbous, and C. J, "Multipoint Communication: A Survey of Protocols, Functions and Mechanisms", IEEE Journal on Selected Areas in Communications, 15(3), pp.277-290, April 1997

[19] G. Carle, E. Biersack, "Survey on Error Recovery for IP-based Audio-Visual Multicast Applications", IEEE Network Mag., pp.24-36, Nov./Dec. 97

[20] J. P. Macker, "Reliable multicast transport and integrated erasure-based forward error correction", ProceedingsIEEE MILCOM, p.973-7, vol.2, November 1997

[21] Luby, M., Vicisano, Gemmell, J., L., Rizzo, L., Handley, M., Crowcroft, J., "The use of Forward Error Correction in Reliable Multicast", Internet draft draft-ietf-rmt-info-fec-01.txt, October 2001

[22] Rizzo, L., "On the Feasibility of Software FEC", DEIT Tech Report, http://www.iet.unipi.it/~luigi/softfec.ps, Jan 1997

[23] R. Aiello, E. Pagani and G. P. Rossi, "Design of a Reliable Multicast Protocol", Proceedings of IEEE INFOCOM '93, pp.75-81, March 1992

[24] Danzig, P, "Flow Control for Limited Buffer Multicast", IEEE Transactions on Software Engineering,Vol. 20, No. 1, pp.1-12, Jan 1994

[25] J. Rosenberg and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction", Request for Comments 2733, Internet Engineering Task Force, Dec. 1999

[26] K. Obraczka "Multicast Transport Mechanisms: A Survey and Taxonomy", IEEE Communications Magazine, Vol. 36 No. 1, January 1998

[27] D. Villela, O. Duarte, "Improving scalability on reliable multicast communications", ELSEVIER Computer Communications Vol. 24, pp.563-574, 2001