# Proactive Hybrid FEC/ARQ Scheme for Reliable Multicast

Ji Li

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

November 2002

# ABSTRACT

## Proactive Hybrid FEC/ARQ Scheme for Reliable Multicast

**Ji Li**

The application of FEC to multicast communication has seen a significant development in years. However, there are still numerous questions about practical implementations that make this field interesting for research. This work is an attempt to address some of these problems.

In the first part of this investigation, the effect of different error control techniques to the multicast communication is analyzed. A new proactive hybrid FEC/ARQ technique is tested in the lab. This technique is based on the RSE code (Reed –Solomon Erasure Correcting Code) which mainly deals with erasures of network at packet level. The sender sends parity-encoded RSE repairs along with original data. The redundant information will allow the reconstruction of some amount of missing data at the receivers. If these proactive repairs are insufficient to reliably deliver the data to receivers, additional RSE repairs are obtained via receiver requests for the transmission (ARQ) of additional repairs.

The thesis also proposes a reliable multicast implementation scheme, and develops a laboratory test-bed for this scheme. Multicast tree is built and maintained by adopting some control messages for multicast session establishment. According to dynamically established multicast routing table, data packets will be propagated from

senders to receivers through Rendezvous Points. The above-mentioned proactive hybrid FEC/ARQ technique is applied to this scheme. The performance of the proposed scheme is compared to that of the traditional ARQ scheme. Significant improvements are achieved in terms of transmission time, feedback implosion, and bandwidth utilization.

# ACKNOWLEDGEMENTS

It is with the utmost sincerity that I express my thanks to my thesis supervisor, Dr. Ahmed Elhakeem, for originally proposing the research area and subsequently guiding me through the various stages of the thesis development. I would also like to thank Dr. Michel Kadoch, Dr. Maria Bennani of École de technologie supérieure, Université du Québec for their continuously support and encouragement.

I am also grateful for the constant support provided by my family and friends. I would like specially thank Qingfen Xu, Donghui Chen, Zuowen Wan, Ahbudula Alwehaibi for their helpful comments and suggestions.

This work was supported by a CRD grant from NSERC/BELL Canada, and the completion of this research was made possible thanks to Bell Canada's support through its Bell University Research Program.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ACRONYMS

| | |
|---|---|
| **ACK** | ACKnowledgment |
| **AER** | Active Error Recovery |
| **AFDP** | Adaptive File Distribution Protocol |
| **ALF** | Application Level Framing |
| **ARM** | Active Reliable Multicast |
| **ARQ** | Automatic Repeat request |
| **BGP** | Border Gateway Protocol |
| **BGMP** | Border Gateway Multicast Protocol |
| **CBT** | Core Based Trees |
| **CRC** | Cyclic Redundancy Check |
| **DVMRP** | Distance Vector Multicast Routing Protocol |
| **ERS** | Expanded Ring Search |
| **FEC** | Forward Error Correction |
| **IETF** | Internet Engineering Task Force |
| **IGMP** | Internet Group Management Protocol |
| **IP** | Internet Protocol |
| **LAN** | Local Area Network |
| **LBRM** | Log-Based Receiver-reliable Multicast |
| **LMS** | Lightweight Multicast Service |
| **MAC** | Medium Access Control |
| **MBGP** | Multiprotocol BGP4/Multicast BGP |
| **MBone** | Multicast Backbone |
| **MDP** | Multicast Dissemination Protocol |

| | |
|---|---|
| **MOSPF** | Multicast Extensions to OSPF |
| **MPEG** | Motion Picture Experts Group |
| **MSDP** | Multicast Source Discovery Protocol |
| **MTP** | Multicast Transport Protocol |
| **NAK (NACK)** | Negative ACKnowledgement |
| **NAPP** | Negative Acknowledge with Periodic Polling |
| **NIC** | Network Interface Card |
| **OSI** | Open Systems Interconnection |
| **OSPF** | Open Shortest Path First |
| **PIM** | Protocol Independent Multicast |
| **PGM** | Pragmatic General Multicast |
| **QoS** | Quality of Service |
| **RBP** | Reliable Broadcast Protocol |
| **RIP** | Routing Information Protocol |
| **RM** | Reliable Multicast |
| **RMANP** | Reliable Multicast Active Network Protocol |
| **RMCM** | Reliable Multicast for Core-based Multicast Trees |
| **RMTP** | Reliable MTP |
| **RSC** | Reed-Solomon Code |
| **RSE** | Reed-Solomon Erasure Code |
| **RTCP** | Real-Time Control Protocol |
| **RTP** | Real-Time Transport Protocol |
| **RTT** | Round Trip Time |
| **SRM** | Scalable Reliable Multicast |
| **TCP** | Transmission Control Protocol |
| **TMTP** | Tree-based MTP |
| **TP** | Turning Point |

| | |
|---|---|
| **TPDU** | Transport Protocol Data Unit |
| **TTL** | Time to Live |
| **UDP** | User Datagram Protocol |
| **WAN** | Wide Area Network |
| **XTP** | Xpress Transport Protocol |

# Chapter 1

# Introduction

## 1.1 Issues in Multicast Communications

Traditional network computing applications involve communication between two computers. However, some important emerging applications such as LAN TV, desktop conferencing, corporate broadcasts, and collaborative computing require simultaneous communication between groups of computers. This process is known generically as multicast communications.

In multicast communication, messages are concurrently sent to multiple destinations, all members of the same multicast group. Mechanisms to support such a form of communication are becoming an increasingly important component of the design and implementation of distributed systems. One of the core issues that need to be addressed as part of providing such mechanisms is the issue of reliable multicast. An RM protocol either confirms with the upper-layer application that the delivery of multicast datagrams is error-free and in sequence, or informs the application the failure of delivery.

Reliability in computer communication is achieved either by using reliable

transport protocols such as TCP, or by implementing the required feathers directly in the application. In both cases, the unavoidable packet losses are recovered by using Automatic Repeat reQuest (ARQ) or Forward Error Correction (FEC) techniques, or their combination. ARQ techniques are used in unicast communication, but they do not scale well to multicast with large groups of receivers, since segment losses tend to become uncorrelated thus greatly reducing the effectiveness of retransmissions. In such cases, FEC techniques can be used, consisting of the transmission of redundant packets to allow the receivers to recover from independent packet losses.

In recent years, some research [1, 2, 3] proposed a new technique that uses a novel combination of FEC/ARQ to reliably deliver data to a large receiver set with an emphasis on reducing delay and local recovery. These applications include high quality, non-interactive audio and video, and stock quote dissemination. In this thesis, we will present a hybrid ARQ/FEC approach that the sender encodes data and transmits the original data along with some redundant data. If a receiver detects losses, which cannot be recovered from the data received from the sender, then the receiver requests for the transmission (ARQ) of additional FEC repairs that it needs to fully recover the original data. The benefit of this approach is that the sender and receivers need to be only aware of the number of lost packets and not their sequence numbers.

## 1.2 Scope and Objectives

The major objective of this research falls into two parts. In the first part, we propose a new multicast implementation scheme which will simply utilize various control messages to build and maintain multicast trees. In the second part, we will apply above mentioned proactively hybrid FEC/ARQ technique to reduce the latency of reliable delivery of data to multiple receivers, to reduce feedback implosion, and to improve bandwidth utilization.

The aims of this thesis are as follow:

- Build and maintain multicast tree for multicast communication.
- Apply proactive hybrid FEC/ARQ to assure reliability of multicast.
- Provide time-bounded delivery with low delay and ordering; guarantee dynamic loss detection and recovery.
- Promise shorter transmission time, decreased NAK traffic, and improved bandwidth utilization.

## 1.3 Thesis Organizations

This thesis is organized as follows. In *Chapter 2*, two basic techniques for error control are presented: ARQ and FEC. We also introduce Reed-Solomon Erasure Correcting Codes and its application for the proactive hybrid FEC/ARQ technique. In *Chapter 3*, we introduce the basis of multicast and review the concerning of reliable

multicast. In *Chapter 4*, we describe a multicast implementation scheme by using the proactive hybrid FEC/ARQ technique. Some performance test results are also shown in this chapter. We give conclusions and suggestions for future work in the last chapter.

# Chapter 2

# Proactive Hybrid FEC/ARQ

This chapter presents the basic proactive hybrid FEC/ARQ approach.

There have been a growing number of applications that could benefit from a reliable multicast transport services. To deal with loss, two well known techniques exist: ARQ (Automatic Repeat Request), which retransmits the lost data upon request, and FEC (Forward Error Correction) which transmits redundant data, called parity data along with the original data.

When integrated with ARQ, FEC provides a substantial reduction in the usage of network resources. Moreover, increased processing efficiency is achieved by the sender and receivers, even when FEC is implemented in software.

## 2.1 ARQ

The most common techniques for error control are based on some or all of the following ingredients [4]:

- **Error Detection**: Damaged or lost data can be detected, for example CRC.
- **Positive acknowledgement**: The destination returns a positive acknowledgement to successfully received error-free data.
- **Retransmission after timeout**: The source retransmits data that has not been acknowledged after a predetermined amount of time.
- **Negative acknowledgement and retransmission**: The destination returns a negative acknowledgement to data in which an error is detected. The source retransmits such data.

Collectively, these mechanisms are all referred to as automatic repeat request (ARQ). Three version of ARQ are in common use:

- Stop-and-wait ARQ.
- Go-back-N ARQ.
- Selective-reject ARQ.

**Stop-and –Wait ARQ**

Stop-and-wait ARQ is based on the stop-and-wait flow control technique. The source station transmits single frame and then must await an acknowledgment (ACK). No other data frames can be sent until the destination station's reply arrives at the source station.

The frame transmitted by the source could suffer an error. If the error is detected

by the destination, it discards the frame and sends a negative acknowledgement (NAK), causing the source to retransmit the damaged frame. On the other hand, if the transmitted frame is so corrupted by noise as not to be received, the destination will not respond. To account for this possibility, the source is equipped with a timer. After a frame is transmitted, the source waits for an acknowledgment (ACK or NAK). If no recognizable acknowledgment is received during the timeout period, then the frame is retransmitted. Note that this system requires that the source maintains a copy of a transmitted frame until an ACK is received for that frame.

**Go-Back-N ARQ**

The form of error control based on sliding-window flow control that is most commonly used is called go-back-N ARQ. In this method, a station may send a series of frames sequentially numbered modulo some maximum value. The number of unacknowledged frames outstanding is determined by window size, using the sliding-window flow control technique. While no errors occur, the destination will acknowledge incoming frames as usual. If the destination station detects an error in a frame, it sends a negative acknowledgment for that frame. The destination station will discard that frame and all future incoming frames until the frame in error is correctly received. Thus, the source station, when it receives a negative acknowledgment, must retransmit the frame in error plus all succeeding frames that were transmitted in the interim.

**Selective-reject ARQ**

With selective-reject ARQ, the only frames retransmitted are those that receive a NAK or which time out. Selective-reject ARQ would appear to be more efficient than the go-back-N approach, since it minimizes the amount of retransmission. On the other hand, the receiver must contain storage to save post-NAK frames until the frame in error is retransmitted, and contain logic for reinserting that frame in the proper sequence. The transmitter, too, will require more complex logic to be able to send frames out of sequence. Because of such complications, the selective-reject ARQ is rarely implemented.

## 2.2 Forward Error Correction (FEC)

### 2.2.1 Introduction

FEC techniques are generally based on the use of certain error detection and correction codes [5] to add redundant (parity) data to the data stream to allow some of the transmission errors or loss packets to be corrected at the receiver without having to ask for retransmission from the sender. These codes have been studied for a long time and are widely used in many fields of information processing, particularly in telecommunications systems. In the context of computer communications, error detection is generally provided by the lower protocol layers which use checksums (e.g. Cyclic Redundancy Checksums (CRCs)) to discard corrupted packets. After such link layer processing, the upper protocol layers have mainly to deal with erasures, i.e. missing packets in a stream.

Erasures originate from uncorrectable errors at the link layer (but those are not frequent with properly designed and working hardware), or, more frequently, from congestion in the network which causes otherwise valid packets to be dropped due to lack of buffers. Erasures are easier to deal with than errors since the exact position of missing data is known.

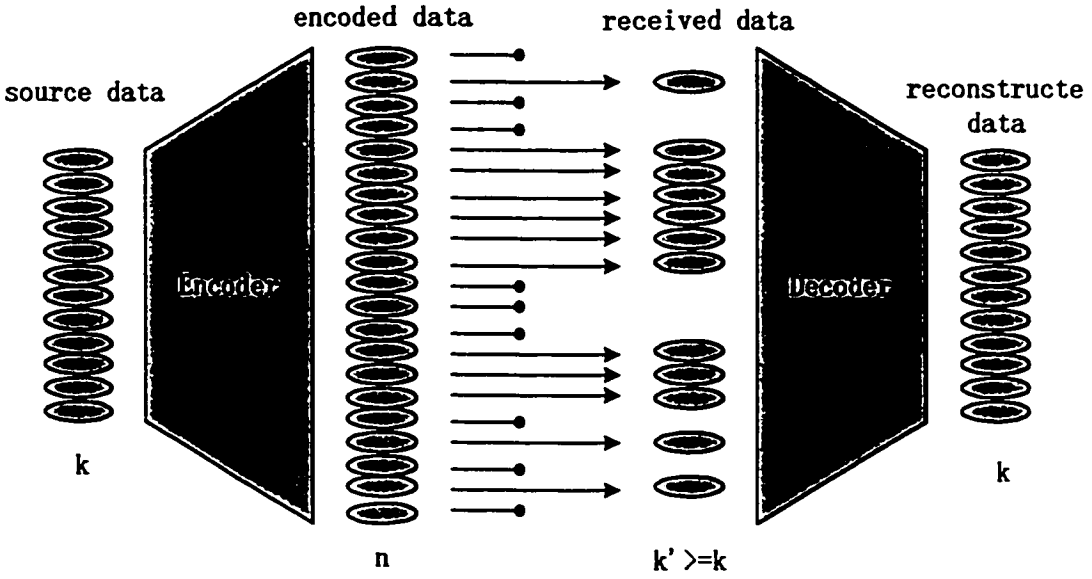



**Figure 2.1: A graphical representation of the encoding/decoding process**

The computations necessary for erasure recovery are slightly simpler than those for full error recovery. As an example of FEC, an (n, k) block erasure code (or, the corresponding encoder) takes k source packets and produces n >k encoded packets in such a way that any subset of k encoded packets (and their identity) allows the reconstruction of the source packets in the decoder (Figure 2.1). A code is called systematic when the encoded packets include the source packets at the beginning of code

word. Systematic codes are much cheaper 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 range of applications, which can take advantage of FEC techniques, is extremely wide. The bandwidth, reliability and congestion control requirements of these applications vary widely, as well as the communication patterns. However, the common requirement is for an improved behavior in presence of packet losses, to whatever reason they depend upon. FEC gives an increased resilience to losses, and can be superior to ARQ when the number of packets to transmit is sufficiently large that the encoding overhead is not exceedingly larger than the expected loss rate. For a proper choice of parameters, FEC can make the residual packet loss rate so small that reliable transfers can be achieved without the need for a feedback channel. This is an extremely important feature for the development of scalable multicast protocols. Additionally, by avoiding the need of an explicit retransmission to recover missing data, the jitter in communication latency can be significantly reduced, making the transport better suited to the transfer of real-time data.

In unicast applications, reducing the amount of feedback necessary for reliable delivery is generally useful to overcome the high delays incurred with ARQ in presence of long delay paths. Also, FEC can be used in presence of asymmetrical communication links, e.g. when uplink communication is expensive and must be minimized.

Reliable multicast communication is the area where FEC can be most beneficial. In fact, ARQ scales poorly with the number of receivers because of two reasons: first, the sender might have to deal with an exceedingly large number of ACKs (or NAKs) from the receivers (the "ACK implosion" phenomenon); second, as the number of receivers grows, so does the probability that losses at different receivers become uncorrelated, to the point that most if not all the packets need to be retransmitted at least once, with a large impact on performance.

### 2.2.2 Desired FEC Characteristics

The desired characteristics for FEC are summarized below [6]:

(a) BURST CORRECTION: Errors are in multiples of the packet length and are coincident with packet boundaries

(b) ERASURE CORRECTION: Most errors are because of congestion: resulting in packet erasures.

(c) ADAPTABILITY: The channel error statistics and application requirements (for delay, throughput and reliability) vary over a wide range in a short period of time.

(d) LOW REDUNDANCY: The number of errors can be very small and the block size can be very large.

(e) LOW LATENCY: Some application need small encoding/decoding delay.

(f) HIGH THROUGHPUT: Applications need throughput of 1 gigabit per second.

(g) LOW COMPLEXITY: Implementation should be economical.

## 2.3 Reed-Solomon Erasure Correcting Codes

The Burst Erasure correcting Code described in this section is based on the well-known Reed-Solomon burst error correcting code (RSC). The major difference is that it can only correct erasures. This Reed-Solomon Erasure correcting code (RSE) produces identical codewords, with the same parameters.

### 2.3.1 RSE Codes Theory

Erasure codes are generally used to reduce the loss rate on a noisy channel. If p is the loss rate for a single data item, using an (n, k) code the residual loss rate decreases roughly as $p^{n-k}$. Depending on the actual needs, this number can be made arbitrarily small.

The mathematical principle behind this coding is linear algebra over finite Galois fields [7, 8]. The $k$ original segments are viewed as the coefficients (in the form of vectors) of a polynomial function of degree $k$ - $1$. Redundant segments can be generated by evaluating the polynomial function at $n$ different points. Any $k$ out of the $n$ values fully specifies the polynomial, effectively regenerating its coefficients.

At the encoding side, denote the $i$-th byte of the $k$ segments as $b_1, b_2, \ldots, b_k$. They [7,8] define a polynomial function $F(X) = b_1 + b_2 X^1 + \ldots + b_k X^{k-1}$ .

Then $f_j = F(p^{j-1})$ is the $i$-th byte of the $j$-th redundant FEC packets where $p$ is a *primitive element* of the Galois Field $GF(2^s)$ and $j \in \{1,..., h\}$. This encoding process can be summarized into a linear transform: $y = G\,x$ where $x$ is the input vector of size $k$ $<b_1, b_2, \ldots, b_k>$ and $y$ is the output vector of size $n$ $<b_1, b_2, \ldots, b_k, f_1, f_2, \ldots, f_h>$. The *generator matrix* $G$ is of size $n \times k$, with the top $k$ rows being the identity matrix and the bottom $h$ rows being $\{p^{(i-1)(j-1)}$ */column index i=1, ..., k and row index j=1, ..., h*$\}$. For any segment length, the same transform is applied to every byte of the $k$ segments to generate the $n$ FEC encoded segments.

Primitive element $p$ of a Galois Field is a value whose powers generate all the non-zero elements of the field. Powers of the primitive element repeat with a period of $q - 1$ where $q$ is the number of elements in the field. In particular, $p^{q-1} = p^0 = 1$. It follows that division by any value $x$ (which can be expressed as $p^i$) is equivalent to multiplication by $p^{-i}$, i.e., $p^{q-1-i}$. This feature is especially useful when inverting the generator matrix $G$ (requiring division) in order to decode FEC packets. This scheme works for code lengths other than 8. We choose 8 as the code length just for simplicity.

At the decoding side, the original $k$ segments can be recovered by solving the following linear equation: $y' = G'\,x$, where $y'$ is a subset of any $k$ elements in $y$ that are available at the receiver and $G'$ is a sub-matrix of $G$ consisting only rows that correspond to the elements in $y'$. To solve this equation, the decoder first inverts the sub-matrix $G$ and then obtains $x = G'^{-1} y'$. In fact, elements in $x$ that have already arrived do not need

to be regenerated, an advantage of the systematic code.

While hardware implementation for RSE code is faster than software implementation, hardware FEC is more common in telecommunication equipment rather than in PCs and workstations for end-to-end transport protocols. However, Rizzo et al. [9] showed that software RSE is becoming feasible as the CPU speed increases. For $k$ values under 32, his software implementation can encode at 10 megabytes/second and decode at 9 megabytes/second on even a low-end PC.

### 2.3.2 RSE Characteristics

This section compares the RSE with other codes [6].

a) BURST CORRECTION

The RSE has the same burst error characteristics as the RSC. However, because it only tries to correct erasures, it is able to correct up to the number of redundant symbols sent (n-k)-which is twice that of the equivalent error correcting RSC.

b) ERASURE CORRECTION

The RSE can correct only erasures and detect errors. If all (n-k) symbols are used for erasure correction, there are no additional symbols to detect any additional symbol errors. If a word is received in error, erasure correction multiplies the error.

c) ADAPTABILITY

If m is picked sufficiently large, so $n < 2^m$; then, (n-k) and n can be varied almost arbitrarily.

d) LOW REDUNDANCY

The small number of errors expected on future fiber networks, makes it desirable for the code to operate efficiently with very low redundancy.

e) LOW LATENCY

For the RSE the decoding can begin as soon as k good symbols of the block are received. There is no interleaving. By keeping n small, the impact of block coding on latency can be kept down to acceptable levels.

f) HIGH THROUGHPUT

The RSE is capable of throughput over 1 gigabit per second for hardware implementation. [6]

g) LOW COMPLEXITY

The RSE is implemented using regular lower complexity systolic chip architecture than RSC. [6]

## 2.4 Proactive Hybrid FEC/ARQ Technique

An increasing number of applications, such as distributed interactive simulation, live auctions, distributed games and collaborative systems, require the network to provide a reliable multicast service. This service enables one sender to reliably transmit data to multiple receivers. Reliability is traditionally achieved by having receivers send negative

acknowledgments (NAKs) to request from the sender the retransmission of lost (or missing) data packets. However, this Automatic Repeat request (ARQ) approach results in the well-known NAK implosion problem at the sender. Many reliable multicast protocols have been recently proposed to reduce NAK implosion. But, the message overhead due to NAK requests remains significant. For this reason, the combination of ARQ and FEC for reliable multicast has been proposed in recent years. Next, we will present two hybrid FEC /ARQ techniques.

### 2.4.1 Hybrid FEC/ARQ I and II

FEC erasure correction can be layered underneath or integrated with an ARQ-based reliable transport protocol to improve the efficiency of the transport protocol. For example, instead of retransmitting individual lost segments, the source transmits FEC packets either proactively with the original data or in response to a NAK (or ACK timeout)[10]. Receivers can use the FEC packets to recover any losses in a range of data segments over which the FEC packets are computed.

The above scheme, call hybrid FEC/ARQ, is classified into two categories, namely *hybrid FEC/ARQ I* (or proactive FEC) and *hybrid FEC/ARQ II* (or reactive FEC) [6]. Note that many earlier studies such as [10] and [11] focus on hybrid FEC/ARQ with a combination of error detection and error correction in transmitting and receiving each "codeword". We, however, describe hybrid FEC/ARQ only in the context of erasure

correction of data segments.

*Reactive FEC*, i.e., *hybrid ARQ II*, was proposed in [11]. FEC packets are transmitted only as retransmissions, either upon timeout waiting for an ACK or in response to a NAK. Retransmissions are no longer the original segments that the requester has lost but FEC packets that can help solve the linear equations for any lost segments in the transmission group. Consequently, a NAK indicates not the sequence number of one lost segment but the total number of lost segments in one transmission group. The source then picks the maximum number of losses reported by all the NAKs for the transmission group and multicasts that amount of FEC packets to the multicast group.

However, as Nonnenmacher et al. [3] pointed out, receivers with high loss rates dominate the performance of reactive FEC, even if they are a very small fraction of the multicast group, because the source has to retransmit according to the worse case. Besides, the responsibility of retransmission is centralized at the source. As for these cases, the proactive FEC is subject to lower possibility for NAK implosion than reactive FEC.

### 2.4.2 Proactive Hybrid FEC/ARQ Technique

The proactive hybrid FEC/ARQ technique uses a combination of ARQ and FEC to reliably deliver data to a set of receivers. The novel aspect of this technique is to

proactively transmit FEC repair packets. In proactive FEC (also referred to as layered FEC) [6], the source transmits extra FEC packets along with the original data of each transmission group, regardless of the receiver's feedback. Retransmissions are still necessary to ensure reliability but are fewer because proactive FEC effectively reduces the end-to-end loss rate.

We now describe the basic technique used by the sender and the receivers to transmit a block of k data packets. The sender multicasts the k data packets during the first round, with zero, one, or more FEC repair packets. We say that any repairs transmitted at this time are transmitted proactively, because it is not known whether or not receivers will use them. Due to potential loss inside the network, each receiver receives a subset of these transmitted packets. The subsets received by the various receivers need not be identical since a multicast packet may be lost at various points within the network. After the sender's first round transmission ends, each receiver assesses the total number of distinct packets that it has received. If receiver i has received $k_i$<k packets belonging to the block, it sends a request to the sender to transmit FEC repair packets. Henceforth, we will refer to this request as a NAK, since such a request acknowledges that insufficient number of packets was received. For now, we assume that a NAK specifies only the number of additional FEC repair packets desired by that receiver for that block: the use of FEC obviates the need for the sender to know specifically what packets have been received by the receiver. By requesting more that

k-$k_i$ additional repairs, a receiver is requesting repairs proactively. In the next round, the sender transmits $\max_{i \in R}(k - k_i)$ additional repairs. This process repeats until the receivers have obtained the k packets. An entire stream of data is transmitted by segmenting the data packets into blocks, and applying the above steps to each block.

# Chapter 3

# Reliable Multicast

Multicast enables many new types of applications and reduces network congestion and server loads. Multicast products and services are receiving widespread industry attention because of their potential benefits. Advances are being made in areas such as reliable multicasting, real-time applications support, and network management and diagnosis. This chapter introduces the technical concepts and mechanisms of multicast in Section 3.1. While sections 3.2 reviews the literature concerning reliable multicast.

## 3.1 Overview of Multicast

IP multicast [10] is the transmission of an IP datagram to a "host group", a set of zero or more hosts identified by a single IP destination address. A multicast datagram is delivered to all members of its destination host group. The multicast source does not need to know the group membership. As defined in [12], multiple sources may send to the same host group and group members do not need to know where the potential sources are. This is called the "many-to-many" multicast service model.

### 3.1.1 Introduction to Multicast and its advantages

"Multicasting provides an efficient way of disseminating data from a sender to a group of receivers." [5] Data destined for the receivers in a multicast group is sent to a single multicast address instead of being addressed separately to each receiver. Only a single copy of the data is sent by the source, the appropriate number of copies is made by each router on the path from a source to the receivers in a multicast group (Figure 3.1).

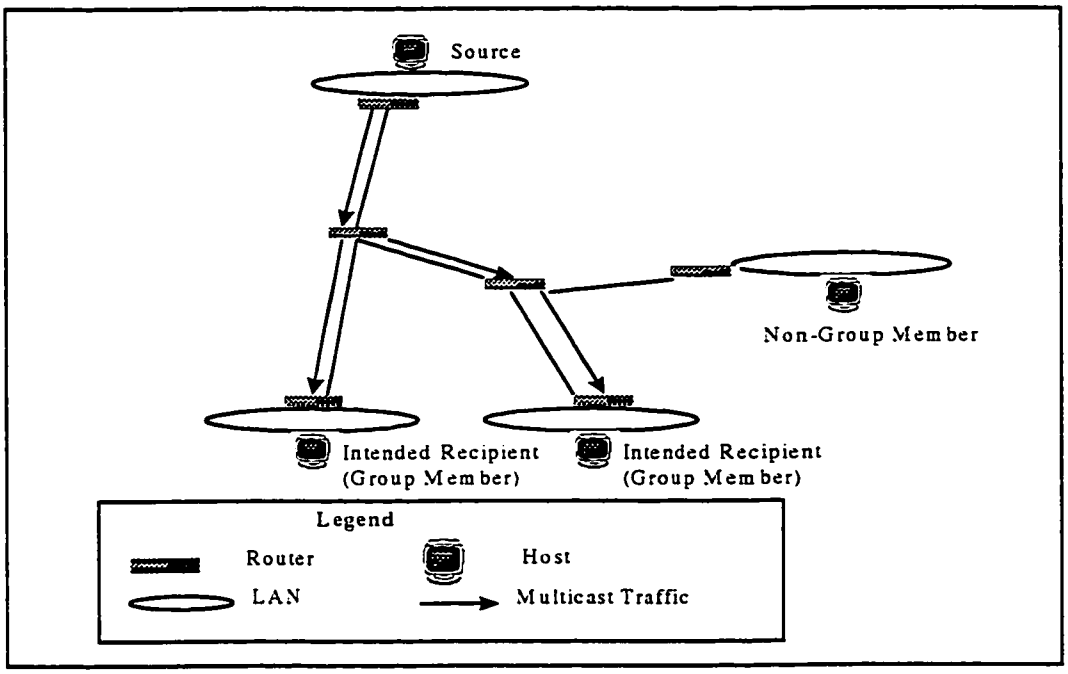



**Figure 3.1. Example of Multicasting.**

Broadcast packets are forwarded on all interfaces of a router except the incoming one. Broadcasting over an internet is not practical since data forwarded over links not leading to intended receivers wastes bandwidth (Figure 3.2).

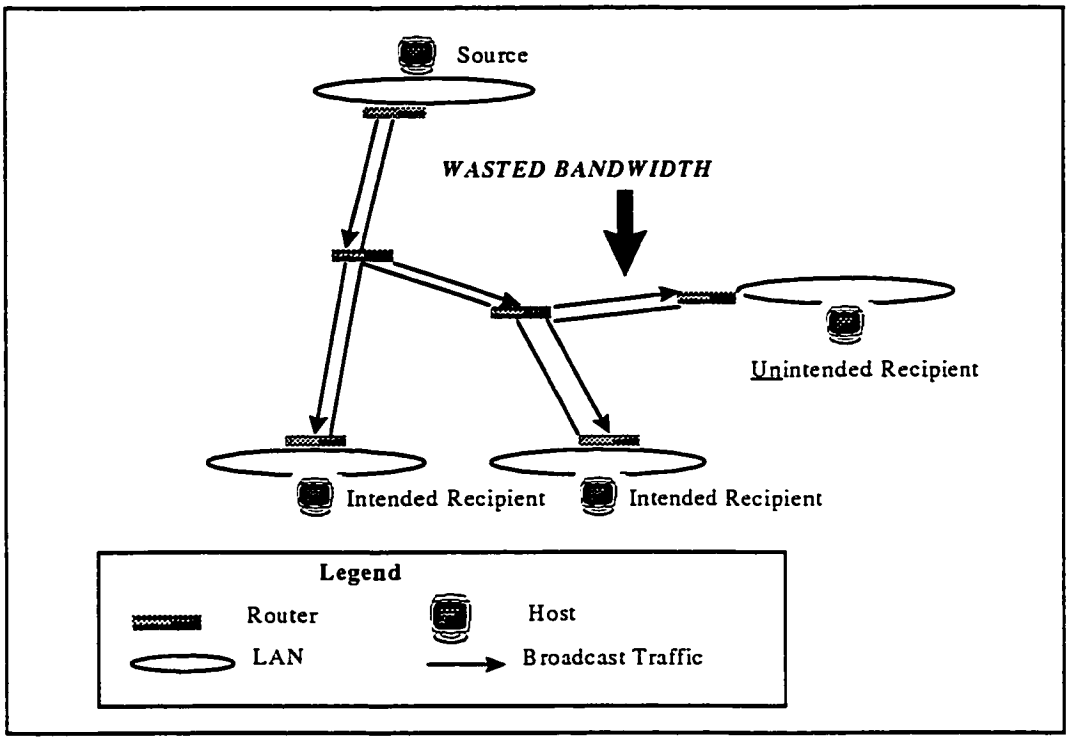



**Figure 3.2.Example of Broadcasting**

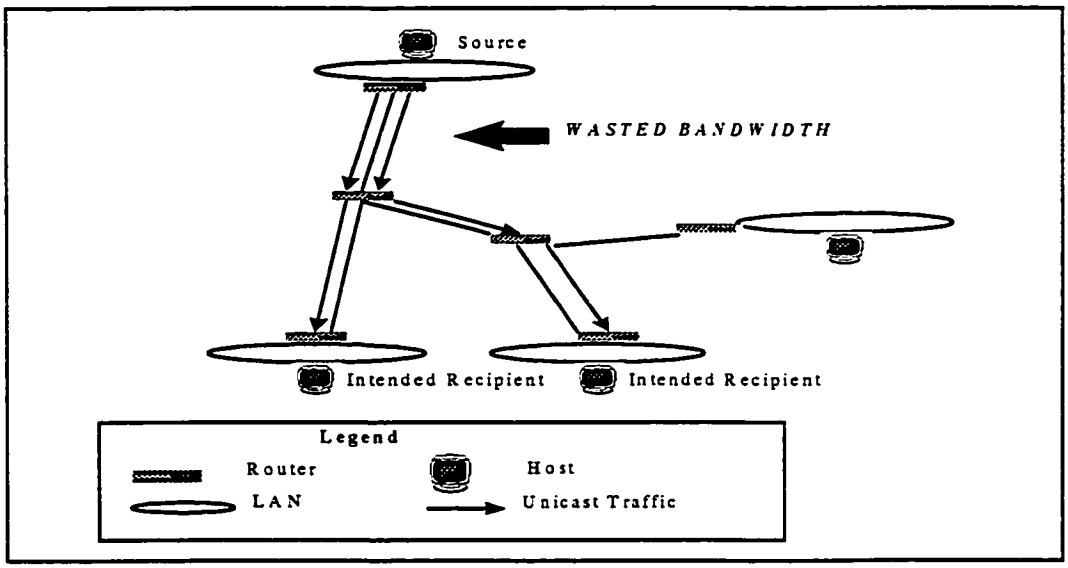



**Figure 3.3.Example of Multiple Unicast Connections**

Bandwidth is also wasted when separate copies of the data are made for each receiver because the same data will traverse at least one link as many times as there are

receivers (Figure 3.3). Routers between the source and the intended receivers may also have to process packets for the same data more than once. Finally, "if connection-oriented service is desired, a single one-to-many connection will most likely be faster and less costly to set up than multiple one-to-one connections." [13].

The advantages of multicast are [14]:

- **Scalability**...Scales to an unlimited number of users
- **Reduced costs**...Cheaper equipment and access line
- **Increased speed**...Increases the delivery speed
- **Saved Bandwidth**...fewer copies of data are needed.

Therefore, there are many applications which could benefit from the efficiency of multicasting. Commanders at different echelons may participate in a group planning session made possible by many-to-many multicasting, where each participant acts as both a sender and a receiver. The product of this planning session may be distributed to subordinate commanders using a multicast protocol. Concurrent updates of geographically separated logistics databases may be multicast after a warehouse inventory is completed. Multicast protocols may also be used to distribute intelligence estimates or satellite imagery to intelligence consumers.

### 3.1.2 Multicast Group Concept

Multicast is based on the concept of a group [15]. An arbitrary group of receivers expresses an interest in receiving a particular data stream. This group does not have any

physical or geographical boundaries—the hosts can be located anywhere on the Internet. Hosts that are interested in receiving data flowing to a particular group must join the group using IGMP. Hosts must be a member of the group to receive the data stream. Multicast/group communications means one to many as well as many to many. A group is identified by a class D IP address (224.0.0.0 to 239.255.255.255).

The group model is an open model. Firstly, anybody can belong to a multicast group without authorization, and a host can belong to many different groups with no restriction. Secondly, the group is dynamic, a host can subscribe to or leave at any time. Finally, a host (source/receiver) does not know the number/identity of members of the group

### 3.1.3 The MBone

The Internet Multicast Backbone (MBone) is an interconnected set of subnetworks and routers that support the delivery of IP multicast traffic. The goal of the MBone is to construct a semi-permanent IP multicast testbed to enable the deployment of multicast applications without waiting for the ubiquitous deployment of multicast-capable routers in the Internet [16].

The MBone has grown from 40 subnets in four different countries in 1992, to more than 3400 subnets in over 25 countries by March 1997. With new multicast

applications and multicast-based services appearing, it seems likely that the use of multicast technology in the Internet will keep growing at an ever-increasing rate [17].

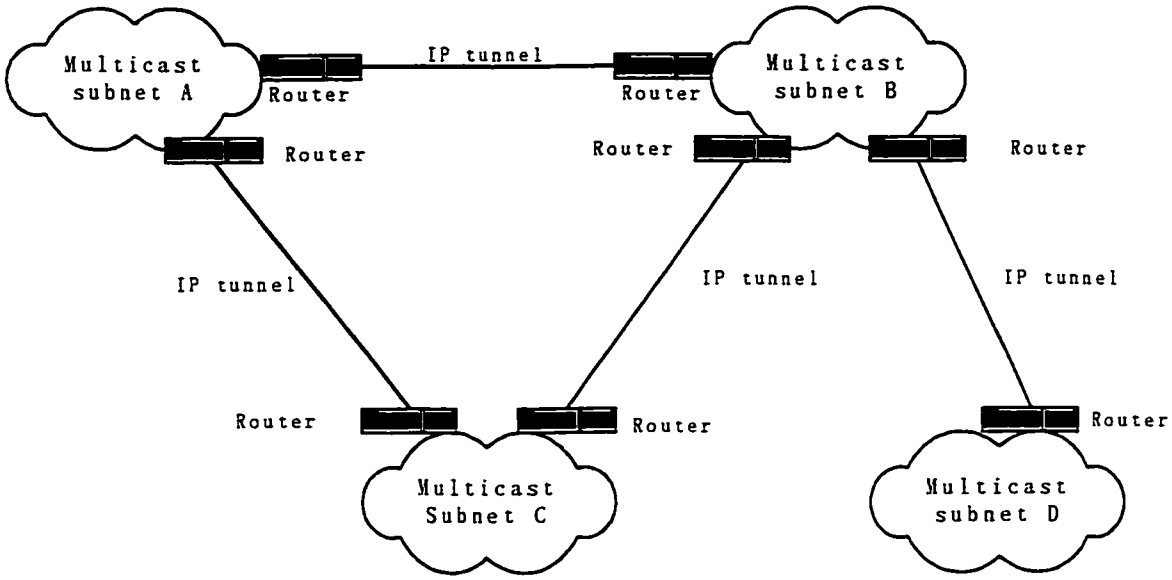



**Figure 3.4 Architecture of Mbone**

The MBone is a virtual network that is layered on top of sections of the physical Internet. It is composed of islands of multicast routing capability connected to other islands by virtual point-to-point links called "tunnels." The tunnels allow multicast traffic to pass through the non-multicast-capable parts of the Internet. Tunneled IP multicast packets are encapsulated as IP-over-IP (i.e., the protocol number is set to 4) so they look like normal unicast packets to intervening routers. The encapsulation is added on entry to a tunnel and stripped off on exit from a tunnel. This set of multicast routers, their directly-connected subnetworks, and the interconnecting tunnels comprise the MBone.

Since the MBone and the Internet have different topologies, multicast routers execute a separate routing protocol to decide how to forward multicast packets. The majority of the MBone routers currently use the Distance Vector Multicast Routing Protocol (DVMRP), although some portions of the MBone execute either Multicast OSPF (MOSPF) or the Protocol-Independent Multicast (PIM) routing protocols.

### 3.1.4 Multicast Addressing

IP multicast addresses specify a "set" of IP hosts that have joined a group and are interested in receiving multicast traffic designated for that particular group. IPv4 multicast address conventions are described in the following sections [18].

#### 3.1.4.1 Class D Address

The Internet Assigned Numbers Authority (IANA) controls the assignment of IP multicast addresses. IP Multicast uses Class D Internet Protocol addresses, those with 1110 as their high-order four bits, to specify multicast host groups. Therefore, all IP multicast group addresses fall in the range from 224.0.0.0 through 239.255.255.255.

The IANA has reserved addresses in the range 224.0.0.0/24 to be used by network protocols on a local network segment. Packets with these addresses should never be

forwarded by a router. Packets with link local destination addresses are typically sent with a time-to-live (TTL) value of 1 and are not forwarded by a router.

**Table 3.1 A summary of the multicast address ranges**

| Multicast Address Range Assignments Description | Range |
|---|---|
| Reserved Link Local Addresses | 224.0.0.0/24 |
| Globally Scoped Addresses | 224.0.1.0 to 238.255.255.255 |
| Source Specific Multicast | 232.0.0.0/8 |
| GLOP Addresses | 233.0.0.0/8 |
| Limited Scope Addresses | 239.0.0.0/8 |

Network protocols use these addresses for automatic router discovery and to communicate important routing information. For example, Open Shortest Path First (OSPF) uses the IP addresses 224.0.0.5 and 224.0.0.6 to exchange link-state information.

Addresses in the range from 224.0.1.0 through 238.255.255.255 are called globally scoped addresses. These addresses are used to multicast data between organizations and across the Internet. And the range 239.0.0.0 to 239.255.255.255 is reserved for administrative scooping (RFC 2365). This is where groups of routers agree

an address range to and from which multicast traffic is prevented from entering or leaving a defined zone.

**Table 3.2 List of some well-known link local IP addresses**

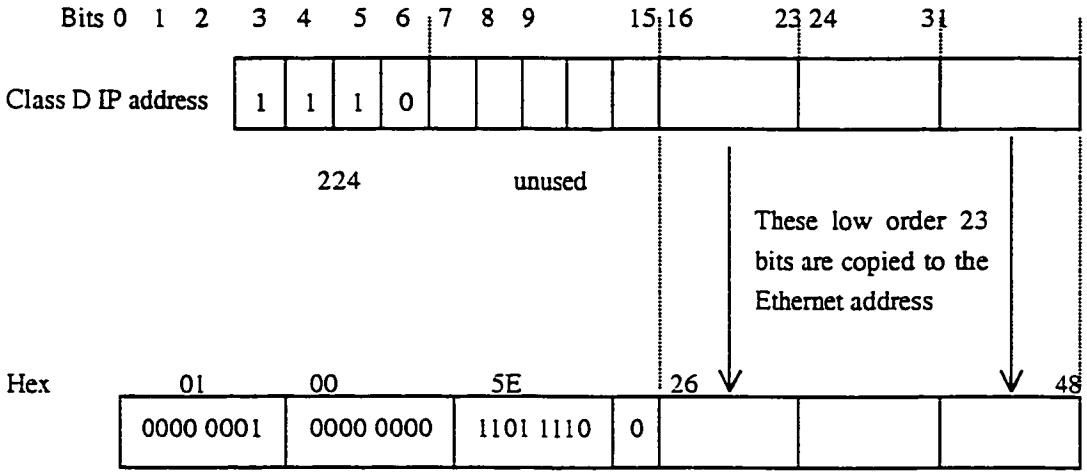
| Examples of Link Local Addresses IP Address | Usage |
|---|---|
| 224.0.0.1 | All systems on this subnet |
| 224.0.0.2 | All routers on this subnet |
| 224.0.0.5 | OSPF routers |
| 224.0.0.6 | OSPF designated routers |
| 224.0.0.12 | Dynamic Host Configuration Protocol (DHCP) server/relay agent |

### 3.1.4.2 Mapping a Class D Address to an IEEE-802 MAC Address

The IANA owns a block of Ethernet MAC addresses that start with 01:00:5E in hexadecimal format. Half of this block is allocated for multicast addresses. The range from 0100.5e00.0000 through 0100.5e7f.ffff is the available range of Ethernet MAC addresses for IP multicast.

The mapping between a Class D IP address and an IEEE-802 (e.g., FDDI, Ethernet) MAC-layer multicast address is obtained by placing the low order 23 bits of the Class D address into the low-order 23 bits of IANA's reserved MAC-layer multicast

address block (see Figure 3.5). This simple procedure removes the need for an explicit protocol for multicast address resolution on LANs akin to ARP for unicast. All LAN stations know this simple transformation, and can easily send any IP multicast over any IEEE-802-based LAN.




Ethernet multicast address

**Figure 3.5 Ethernet address mapped on to the Class D multicast addresses**

### 3.1.4.3 Transmission and Delivery of Multicast Datagrams

When the sender and receivers are members of the same (LAN) subnetwork, the transmission and reception of multicast frames is a straightforward process. The source station simply addresses the IP packet to the multicast group, the network interface card maps the Class D address to the corresponding IEEE-802 multicast address, and the frame is sent.

Receivers that wish to capture the frame notify their MAC and IP layers that they want to receive datagrams addressed to the group. Things become somewhat more

complex when the sender is attached to one subnetwork and receivers reside on different subnetworks. In this case, the routers must implement a multicast routing protocol that permits the construction of multicast delivery trees and supports multicast packet forwarding. In addition, each router needs to implement a group membership protocol that allows it to learn about the existence of group members on its directly attached subnetworks.

## 3.1.4 The Protocol Stack of Multicast

### 3.1.4.1. Introduction

Multicast capabilities are implemented in several protocol layers. The functions related to multicasting at the involved layers of the protocol stack (Figure 3.6) are introduced in the sections which follow.

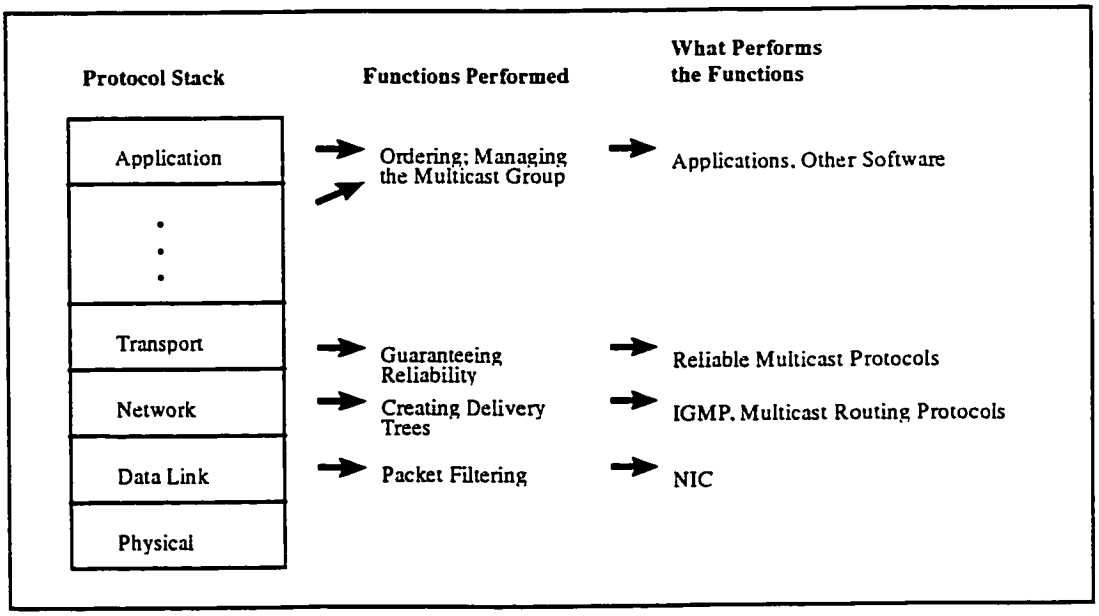



**Figure 3.6. How Different Layers of the Protocol Stack Contribute to Multicasting.**

In the link layer there is of course a need to support multicast addresses. In the network layer there is e.g. the IGMP protocol for handling joins and leaves to and from multicast groups and multicast routing protocols, such as MOSPF, DVMRP, and PIM.

The transport layer handles reliability such as in TCP and this is the case also when it comes to multicast. Most of the reliable multicast protocols are found in the transport layer. Some protocols operate also in the application layer. These are designed to take into account also the semantics of the multicast packets and to handle application specific requests on data. These protocols build on the ALF [12] principle. It should be pointed out that the border between what is considered transport or application layer is very thin.

#### 3.1.4.2. Data Link Layer

The two functions essential to multicasting that are performed at the data link layer relate to group addressing. First, protocols in this layer distinguish between group addresses and other addresses. The Ethernet, Token Ring, Token Bus, and Fiber Distributed Data Interface (FDDI) Media Access Control (MAC) protocols each provide some means of indicating if an address is a group address [13].

Second, if a host belongs to a multicast group, the Network Interface Card (NIC) will deliver MAC frames with the appropriate group address to the host. Filtering packets with the NIC is more efficient than filtering them with higher layer software since "...doing it in software is orders of magnitude slower." [13]

### 3.1.4.3 Network Layer

Two types of network layer protocols perform complimentary multicast functions. The Internet Group Management Protocol (IGMP) informs a host's immediately neighboring router of its desire to join or leave a multicast group. Multicast routing protocols build delivery paths from a sender to routers with group members on their attached networks.

***Dynamic Registration***--There must be a mechanism that informs the network that the computer is a member of a particular group. Without this information, the network would be forced to flood rather than multicast the transmissions for each group. For IP networks, the Internet Group Multicast Protocol (IGMP) is an IP datagram protocol between routers and hosts that allows group membership lists to be dynamically maintained. The host sends an IGMP "report", or join, to the router to join the group. Periodically, the router sends a "query" to learn which hosts are still part of a group. If a host wants to continue its group membership, it responds to the query with a report. If the host sends no report, the router prunes the group list to minimize unnecessary transmissions. With IGMP V2, a host may send a "leave" message to inform the router that it no longer is participating in a multicast group. This allows the router to prune the group list before the next query is scheduled, minimizing the time period in which wasted transmissions are forwarded to the network.

***Multicast Routing***--The network must be able to build packet distribution trees that specify a unique forwarding path between the subnet of the source and each subnet containing members of the multicast group. A primary goal in distribution trees construction is to ensure that at most, one copy of each packet is forwarded on each branch of the tree. This is accomplished by constructing a Spanning Tree rooted at the designated multicast router of the sending host, providing connectivity to the designated multicast routers of each receiving host. For IP multicast, the IETF has offered several multicast routing protocols for consideration. These include: the Distance Vector Multicast Routing Protocol (DVMRP), Multicast extensions to OSPF (MOSPF), Protocol-Independent Multicast (PIM), and Core-Based Trees (CBT). Multicast routing protocols build distribution trees by examining a unicast reachability protocol's routing table. Some protocols use the unicast forwarding table, including PIM and CBT. Alternatively, other protocols use their own private unicast reachability routing tables. DVMRP uses its own distance vector routing protocol to determine how to build source-based distribution trees. Similarly, MOSPF, uses its own link state database to build source-based distribution trees.

Multicast routing protocols fall into two categories: Dense-mode (DM) and Sparse-mode (SM). DM protocols assume that almost all routers in the network will need to distribute multicast traffic for each multicast group (for example, almost all hosts on the network belong to each multicast group). Accordingly, DM protocols build

distribution trees by initially flooding the entire network and then pruning back the small number of paths without receivers. SM protocols assume that relatively few routers in the network will be involved in each multicast. The hosts belonging to the group are widely dispersed, as might be the case for most multicasts in the Internet. Therefore, SM protocols begin with an empty distribution tree and add branches only as the result of explicit requests to join the distribution. The DM protocols, MOSPF, DVMRP, and PIM-DM [19], are most appropriate in LAN environments with densely clustered receivers and the bandwidth to tolerate flooding, while the SM protocols, CBT and PIM-SM [20], are generally more appropriate in WAN environments. PIM is also capable of functioning in Sparse-Dense mode by adjusting its behavior to match the characteristics of each receiver group.

The above protocols can not be used directly for interdomain multicast. To perform interdomain multicast, several solutions are proposed and developed, which are including MBGP, MSDP and BGMP.

#### 3.1.4.4 Transport Layer

Although multicast routing protocols provide best effort delivery of multicast datagrams on the Internet, many multicast applications have requirements beyond this. Therefore, various multicast transport protocols are proposed on top of the multicast routing protocols to meet the needs of different applications. In [21], they are classified according to the kind of applications they support. "General purpose" protocols, such as

RBP, MTP, and XTP, are designed to provide a general solution to the group communication problem. They represent the earlier stage of the multicast transport protocol deployment. It is realized later that a single generic protocol cannot meet the requirements of all multicast applications and, thus, most recent protocols are designed with some specific applications in mind. Some protocols are designed for multipoint interactive applications, such as RTP/RTCP and SRM, while others support data dissemination services, such as the MDP and the AFDP.

Multicast transport protocols serve two major functions, namely, providing reliability and performing flow and congestion control. There are different definitions of reliability for different multicast applications. For example, total reliability is more suitable for reliable bulk-data transfer such as file distribution, while semi-reliability and time-bounded reliability are designed for loss-tolerant real-time applications such as video conferencing.

#### 3.1.4.5. Higher Layers

For some protocols, certain aspects of reliability are handled by layers higher than the transport layer. Ordering and group management, for example, may be the responsibility of higher layers. Some protocols also involve higher layers in failure recovery and flow control.

## 3.2 Reliable Multicast

A multicast datagram is delivered with the same "best-effort" reliability as unicast datagrams, i.e., the datagram is not guaranteed to arrive intact at all members or in the same order relative to other datagrams. Therefore, just as unicast applications require TCP on top of IP unicast, a multicast application requires a reliable multicast (RM) protocol on top of IP multicast. An RM protocol either confirms with the upper-layer application that the delivery of multicast datagrams is error-free and in sequence, or informs the application the failure of delivery. An RM protocol can employ sequence numbers, checksums, positive acknowledgements (ACKs) and/or negative acknowledgements (NAKs) to achieve the reliability goals. Large-scale reliable multicast faces the challenges of feedback implosion and the "crying baby" [22].

### 3.2.1 Definition of Reliability

To introduce the work on reliable multicast, we first need to give a definition of "reliability." We define broad sense "reliability," which comprises the following three aspects.

The first is Error-Free Delivery [5]. It refers to delivering all data to all receivers eventually. This is the narrow sense "reliability" used in Internet multicast. "All receivers" may or may not include late-join or temporarily partitioned receivers. If they are included, at least the sender needs to keep all transmitted data during the multicast

session. The second property, atomicity [23], requires that all group members deliver a message to the application within a specified interval begun either after one group member has delivered the same message or after a majority of group members have delivered the message. The final property is ordering, which may range from no guarantees of ordering, to a guarantee that messages are always delivered consistently even in the presence of network failures [23].

### 3.2.2 The Use of FEC and ARQ for Reliable Multicast

There are many ways to provide reliability for transmission protocols. A common method is to use ARQ, automatic request for retransmission. With ARQ, receivers use a back channel to the sender to send requests for retransmission of lost packets. ARQ works well for one-to-one reliable protocols, as evidenced by the pervasive success of TCP/IP. ARQ has also been an effective reliability tool for one-to-many reliability protocols, and in particular for some reliable IP multicast protocols. However, for one-to-very-many reliability protocols, ARQ has limitations, including the feedback implosion problem because many receivers are transmitting back to the sender, and the need for a back channel to send these requests from the receiver. Another limitation is that receivers may experience different loss patterns of packets, and thus receivers may be delayed by retransmission of packets that other receivers have lost that but they have already received. This may also cause wasteful use of bandwidth used to retransmit

packets that have already been received by many of the receivers.

FEC codes provide a reliability method that can be used to augment or replace other reliability methods, especially for one-to-many reliability protocols such as reliable IP multicast. In this thesis, we focus on the use of parity encoding techniques to reduce the latency of reliable delivery of data to multiple receivers and to reduce feedback implosion. Such techniques are useful in applications that deliver real-time information to a large receiver set and can tolerate small transmission delays. These applications include high quality, non-interactive audio and video, and stock quote dissemination.

In the general literature [25], FEC refers to the ability to overcome both erasures (losses) and bit-level corruption. However, in the case of an IP multicast protocol, the network layers will detect corrupted packets and discard them or the transport layers can use packet authentication to discard corrupted packets. Therefore the primary application of FEC codes to IP multicast protocols is as an erasure code. The payloads are generated and processed using an FEC erasure encoder and objects are reassembled from reception of packets containing the generated encoding using the corresponding FEC erasure decoder. Therefore, as long as the receiver receives enough distinct FEC packets, it can reconstruct the original payload segments via FEC decoding. Section 2.3 describes RSE erasure correction algorithms in detail.

The input to an FEC encoder is some number k of equal length source symbols. The FEC encoder generates some number of encoding symbols that are of the same

length as the source symbols. The chosen length of the symbols can vary upon each application of the FEC encoder, or it can be fixed. These encoding symbols are placed into packets for transmission.

The number of encoding symbols placed into each packet can vary on a per packet basis, or a fixed number of symbols (often one) can be placed into each packet. Also, in each packet is placed enough information to identify the particular encoding symbols carried in that packet. Upon receipt of packets containing encoding symbols, the receiver feeds these encoding symbols into the corresponding FEC decoder to recreate an exact copy of the k source symbols. Ideally, the FEC decoder can recreate an exact copy from any k of the encoding symbols. Table 3.3 compares the major differences using pure FEC and ARQ in reliable multicast protocols.

**Table 3.3 Comparison Between Pure FEC and ARQ in Reliable Multicast Protocols**

| FEC | ARQ |
|---|---|
| Suitable for large groups with large Round-Trip Times(RTTs), or when the feedback channel is unavailable | Suitable for small groups with feedback channel |
| Suitable for networks with homogeneous loss probability | Suitable for networks with heterogeneous loss probability |
| Efficient in overcoming independent loss | Efficient in overcoming shared loss |
| Suitable for real-time interactive applications | Suitable for non-interactive applications |
| Only provides semi-reliability | Provides total reliability |

In chapter 4, we will examine an approach that sends parity-encoded loss repairs proactively, i.e. repairs are sent before it is known whether the transmission of the repair is necessary. If these proactive repairs are insufficient to reliably deliver the data to receivers, additional FEC repairs are obtained via receiver requests for the transmission (ARQ) of additional repairs. We find that the bandwidth used by the sender for data and repair transmission does not increase significantly in comparison to that used by a non-proactive protocol, as long as the number of packets transmitted proactively is kept below a certain bound. At the same time, there is a significant reduction in both the expected delay of reliable delivery, as well as the amount of feedback transmitted toward the sender.

### 3.2.3 Summary of Reliable Multicast Protocols and Mechanisms

In this section, we give the list of major reliable multicast protocols and their main properties in table 3.4 [5, 21, 22, 24].

### 3.2.4 Challenges and Solutions

#### 3.2.4.1 Security

As IETF requirements (RFC 2357) [26], the first challenge will be security, which is addressed by MSEC/SMUG working groups. There are many different needs (authorization, confidentiality...) for reliable multicast that are much more complex than

unicast security. For example, a member leaving a group must not be able to further listen to data, and to update the session key in a scalable/efficient way.

**Table 3.4 Major Reliable Multicast Protocols**

| Protocols | Main properties |
|---|---|
| SRM | Receiver-initiated NACK-based protocol, using random timers to suppress NACKs and retransmissions. |
| LBRM | Receiver-initiated tree-based protocol, using distributed logging servers for retransmission, including a variable heartbeat mechanism for fast loss detection, and dynamically choosing between multicast and unicast for retransmission. |
| RMTP | Receiver-initiated tree ACK-based protocol with selective retransmission. Statically chosen DRs aggregate ACKs and perform local recovery. |
| RMTP-II | Tree ACK-based protocol with NACK and FEC options. The control tree consists of a top node, DRs and receivers, supporting unordered, source-ordered, and time bounded delivery, few-to-many multicast, asymmetrical networks, and different membership control. |
| TMTP | Tree NAPP-based protocol. The tree is organized using ERS with bounded fan-out. TTL scoping is used for local recovery. |
| LMS, Search Party, and RMCM | Using router support to deliver repair request, LMS uses pre-selected replier link at each TP; Search Party selects replier links in a probabilistic manner; RMCM is similar to LMS, but designed for bi-directional shared multicast routing trees. |
| ARM, AER, PGM, and RMANP | Using multicast routing tree for local recovery, relying on active service provided by active routers (or their collocated servers). RMANP is implemented on an active network and defines different types of capsules for different operations. |
| Lorax | Shared ACK tree-based protocol which organizes the ACK tree using ERS. Nodes on the tree are labeled using a hierarchical scheme. |

### 3.2.4.2 Congestion Control and Flow Control

Flow control and congestion control are among the fundamental problems of Internet multicast. This is an active research area with many challenges and open issues.

***Multicast Flow Control***: A set of techniques that match the data transmission rate to the capacities of receivers and to the service rates in the paths leading to those receivers.

***Multicast Congestion Control***: A set of techniques that regulate the data transmission rate in response to network conditions and the principles and mechanisms of sharing congested links among many sessions.

The difference between congestion control and flow control in the unicast scenario: congestion control is a global issue and it ensures that the subnet is able to carry the offered traffic, while flow control is used to prevent a sender from transmitting data too fast as to overwhelm the receiver. In multicast, since there are many receivers, flow control must meet the requirements of many receivers, while congestion control has to deal with the fairness issues not only among multicast sessions, but also between multicast sessions and other traffics such as TCP flows.

The challenge to the IETF is to encourage research and implementations of reliable multicast, and to enable the needs of applications for reliable multicast to be met as expeditiously as possible, while at the same time protecting the Internet from the congestion disaster or collapse that could result from the widespread use of applications with inappropriate reliable multicast mechanisms. Because of the setbacks and costs that

could result from the widespread deployment of reliable multicast with inadequate congestion control and flow control, the IETF must exercise care in the standardization of a reliable multicast protocol that might see widespread use.

#### 3.2.4.3 Scalability

There are many problems arising with a large number of receivers for reliable multicast. The first problem is ***scalable control traffic*** because ACK or NAK implosion might happen. Several solutions [26] are addressed in the following:

*Solution 1*: Feedback suppression at the receivers: each node picks a random backoff timer, send the NAK at timeout if loss not corrected.

*Solution 2*: Proactive FEC (forward error correction): send data plus additional FEC packets, any FEC packet can replace any lost data packet.

*Solution 3*: Use a tree of intelligent routers/servers, use a tree for ACK aggregation and/or NAK suppression. see PGM [27].

The second problem is ***scalable retransmissions***, if each receiver experience 1% packet loss, each packet will be sent several times.

*Solution 1*: Use proactive/reactive FEC.

*Solution 2*: Use a tree of retransmission servers: a receiver can be a retransmission server if it has the data requested.

#### 3.2.4.4 Heterogeneity

Another major challenge for multicast communications is the heterogeneity of group members and network capacities. There are a lot of possible solutions [26] for this, a few are listed below:

*Solution 1*: adjust transmission rate to the slowest receiver without going below a given threshold, but high end receivers won't be happy in presence of very slow receivers. It requires a group management scheme.

*Solution 2*: use various homogeneous receiving groups, better but at the cost of extra traffic. It requires a group management scheme too.

*Solution 3*: use multi-rate transmissions (ALC/LCT), addressed by the RM protocol itself.

### 3.2.5 Current IETF standardization work

Not one single technique will prove to solve all the complex problems of reliable multicast. Hybrid protocols for reliable multicast have used combinations of different error correcting schemes, such as combining repeating protocols with redundancy.

The IETF working group on reliable multicast transport (RMT) [24] is aiming at standardize reliable multicast. Recognizing the problems with a one-size-fits-all solution, the RMT WG is going to present three protocols, a NAK based, a tree-based ACK, and an asynchronous layered coding protocol. This is currently all work in progress.

In RMT, protocols are broken down into building blocks (BB) and protocol

implementations (PI). The BBs describe algorithms and interfaces towards other BBs and PIs. A PI, on the other hand, constitutes a set of BBs and other functions which make up a complete instance of a protocol. The three proposed protocols are list below:

**Flat NORM:** For small to medium sized groups; simplicity; uses NAK.

**Hierarchical TRACK:** For medium sized to large groups; requires tree building.

**Layered ALC:** For all sizes of groups; unlimited scalability.

#### 3.2.5.1 NORM

The NAK based protocol, NORM [28, 29] will incorporate means for NAK avoidance and dynamically adjustable timers. It will cater to measurement of the greatest round trip time for adjustment of protocol state and back-off timers. There will be dedicated header field to gather such information as to let NORM adapt to changing network conditions. NORM will also allow for proactive as well as reactive FEC. Though NORM is a NAK based protocol, ACK's can also be used by explicit configuration.

This protocol is designed to provide end-to-end reliable transport of bulk data objects or streams over generic IP multicast routing and forwarding services. NORM uses a selective, negative acknowledgement mechanism for transport reliability and offers additional protocol mechanism to conduct reliable multicast sessions with little "a priori" coordination among senders and receivers. It is capable of operating with both reciprocal multicast routing among senders and receivers and with asymmetric connectivity (possibly a unicast return path) from the senders to receivers. The protocol

offers a number of features to allow different types of applications or possibly other higher level transport protocols to utilize its service in different ways. The protocol leverages the use of FEC-based repair and other IETF reliable multicast transport (RMT) building blocks in its design. (Fig.3.7)

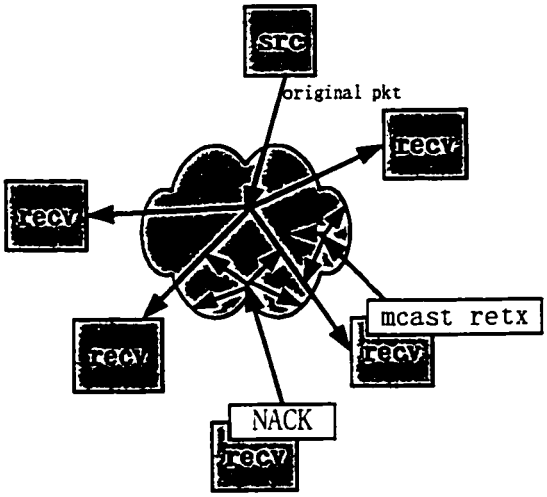



Figure 3.7 NORM

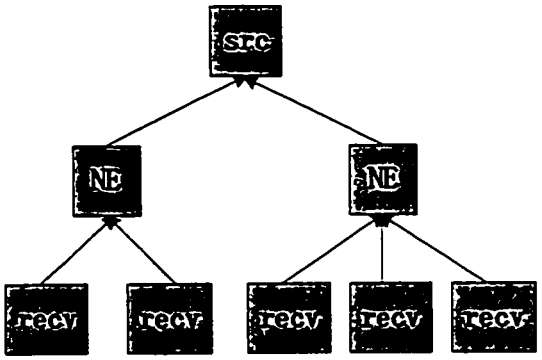



Figure 3.8 TRACK

### 3.2.5.2 TRACK

The tree-based PI is TRACK [30], which supports NAKs, proactive and reactive FEC and tree-based ACKs. Via General Router Assist, local repairs are also supported. Repair Heads correspond to the dedicated receivers or repair servers found in other protocols mentioned earlier.

TRACK has also a resemblance of the obsolescence property proposed in [24] in this sender-controlled recovery window. In each data packet, the sender may indicate that

packets older than the given sequence number, are obsolete.

The TRACK PI is based on Tree Based Acknowledgment, a tree offers assistance services for NAK suppression, ACK aggregation, retransmissions (or a subset of them) for medium to large groups (Fig.3.8). Building blocks required (or optionally used) by the TRACK PI, like the NACK PI (NACK, FEC, CC, security) plus GRA (Generic Router Assistance) for tree management. For Instance, the *CISCO's PGM (pragmatic general multicast)* build a tree of NE (Network Elements) (server or router) that perform:

- *ACK* aggregation along the tree
- NAK suppression along the tree
- Localized retransmission in a subset of the tree
- Retransmission (if data is cached)
- FEC possible for increased scalability/lower Latency.

### 3.2.5.3 ALC (Asynchronous Layered Coding)

A massively scalable reliable content delivery protocol, hereafter referred to as ALC. ALC combines the LCT building block, a multiple rate congestion control building block that is in compliance with RFC2357 [26] and the FEC [31] building block to provide congestion controlled reliable asynchronous delivery of content to an unlimited number of concurrent receivers from a single sender.

The layered protocol instantiation, LCT the layering supports scalability with

regards to link and host capacity, as well as congestion control and error recovery. As in [14] a receiver can obtain additional parity packets by subscribing to more layers, which can result either in a faster decoding on account of a faster reception of a required number of packets, or an error recovery by acquiring a sufficient amount of parity information.

The ALC PI is based on multi-rate transmission plus proactive FEC. It is entirely "receiver-oriented" for maximum scalability (several millions...) (Figure 3.9). ALC targets multicast file transfer and can easily handle hierarchical video coding for real-time streaming...etc.

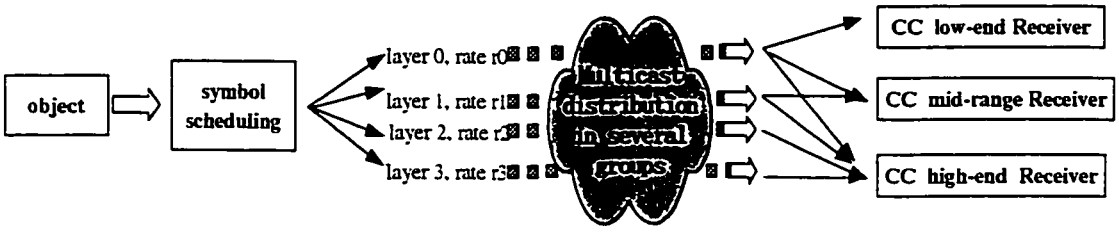



**Figure 3.9 ALC**

# Chapter 4

# Implementation Scheme for Reliable Multicast

This chapter presents the detailed technical information concerning our proactive hybrid FEC/ARQ multicast implementation scheme in section 4.1. In order to test this scheme, we set up a test-bed in laboratory Lagrit at ETS, Montreal with equipment purchased by a joint CFI grant. Many experiments have been conducted on this test-bed. Some test results and explanations are presented in section 4.2.

## 4.1 Implementation scheme for reliable multicast

### 4.1.1 Terminology

This specification uses a number of terms to refer to the roles of routers participating in our scheme. The following terms have special significance for this scheme:

***Rendezvous Point (RP)***: An RP is a router that has been configured to be used as the root of the distribution tree for a multicast group. Join messages from receivers for a group are sent towards the RP, and join messages from senders are also sent to the RP so that receivers can discover who the senders are, and start to receive traffic destined for the

group.

***Designated Router (DR):*** A shared-media LAN like Ethernet may have multiple multicast routers connected to it. If the LAN has directly connected hosts, then a single one of these routers, the DR, will act on behalf of those hosts with respect to the same multicast protocol. A single DR is elected per LAN using a simple election process.

***Unicast Routing Table (URT):*** This table specifies the outgoing interface for a data packet to next hop towards the destination.

***Multicast Routing Table (MRT)***: This is the multicast topology table, which is typically derived from the unicast routing table. In this scheme, the MRT is used to decide where to send data packets. A secondary function of the MRT is to provide routing metrics for destination addresses.

***Upstream:*** Towards the root of the tree. The root of tree may either be the source or the RP depending on the context.

***Downstream:*** Away from the root of the tree.

***Iif:*** Incoming interface of certain node.

***Oif:*** Outgoing interface of certain node.

### 4.1.2 Overview

This section provides an overview of this implementation scheme for reliable multicast. It is intended as an introduction to how the scheme works.

The scheme relies on multicast routing table to propagate data along multicast tree from one router to another router. This routing table is called the MRT. The routes in

this table may be taken directly from the Unicast Routing Table, or it may be modified by control messages such as Join/Prune messages. Regardless of how it is created, the primary role of the MRT in the scheme is to provide the next hop router along a multicast-capable path to each destination subnet. The MRT is used to determine the next hop neighbor to which any Join/Prune message is sent. Data flows along the reverse path of the Join messages. Thus, in contrast to the URT which specifies the next hop that a data packet would take to get to some subnet, the MRT gives reverse-path information, and indicates the path that a multicast data packet would take from its origin subnet to the router that has the MRT.

This scheme is able to propagate data packets from sources to receivers through rendezvous point. This is essentially done in five phases, although as senders and receivers may come and go at any time. The last three phases may occur simultaneously.

**Phase A: Sender and Receiver Join**

In phase A, initially every RP will express its interest in handling traffic for a multicast group. It does this by periodically flooding Type 3 message to all its outgoing interface by using UDP socket, every neighbor router will repeat the same process, it will flood every Type 3 message except the port it received from. Once a specific sender unicasts Type 1 message to all RPs, it will wait for ACK from anyone of them. If it receives at least one ACK, the sender will return an ACK to this RP which will handle its session from now on. Other RPs will not receive replied ACK from sender, so they will

timeout and will not handle this session. This process of sender joining to the multicast group is called sender registering.

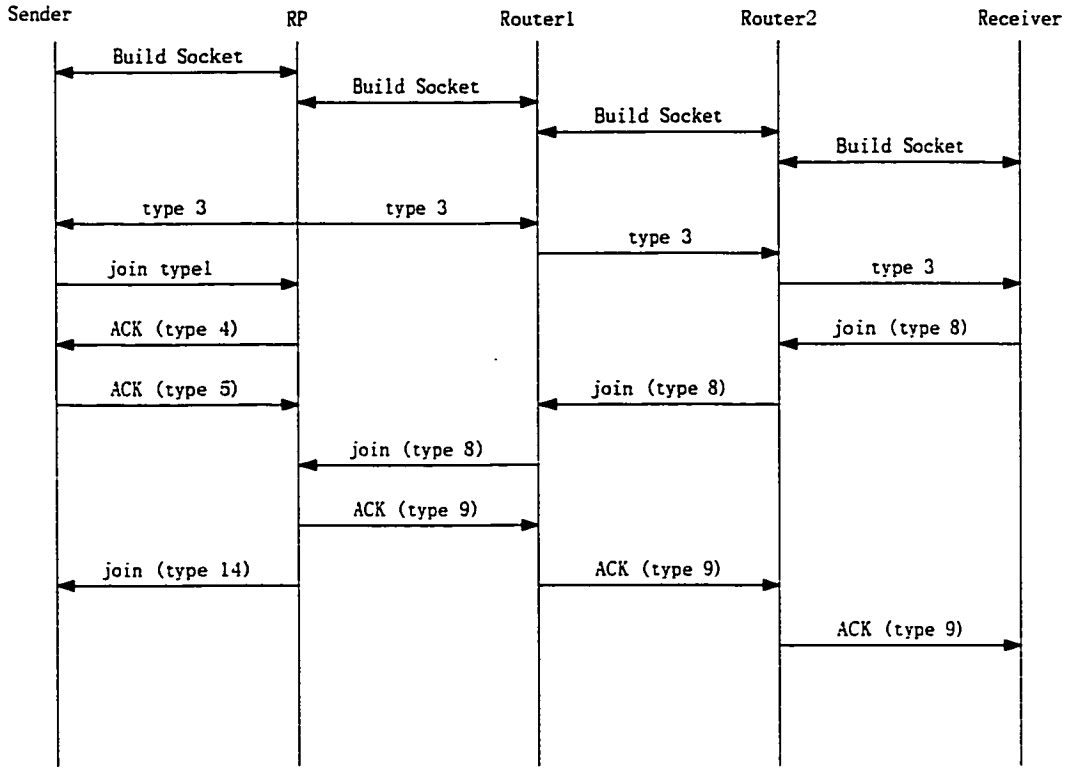



**Figure 4.1 Phase A: Sender and Receiver Join**

Simultaneously, a multicast receiver expresses its interest in receiving traffic destined for a multicast group. Typically it does this by using Type 8 message (receiver join). On receiving the receiver's expression of interest, the nearby DR then sends a Join message towards the RP for that multicast group. This Join message travels hop-by-hop towards the RP for the group, and in each router it passes through, multicast tree for this group is instantiated. Eventually the Join message either reaches the RP, or reaches a router that already owns this Join for that group. When many receivers join the group,

their Join messages converge on the RP, and form a distribution tree for this group that is rooted at the RP. This is known as the RP Tree .

**Phase B: Data Transmission**

After phase A, a multicast data sender just starts sending data destined for a multicast group. The sender's local router (DR) takes those data packets, unicast-encapsulates them, and sends them directly to the RP. The RP receives these encapsulated data packets, decapsulates them, and forwards them following the multicast

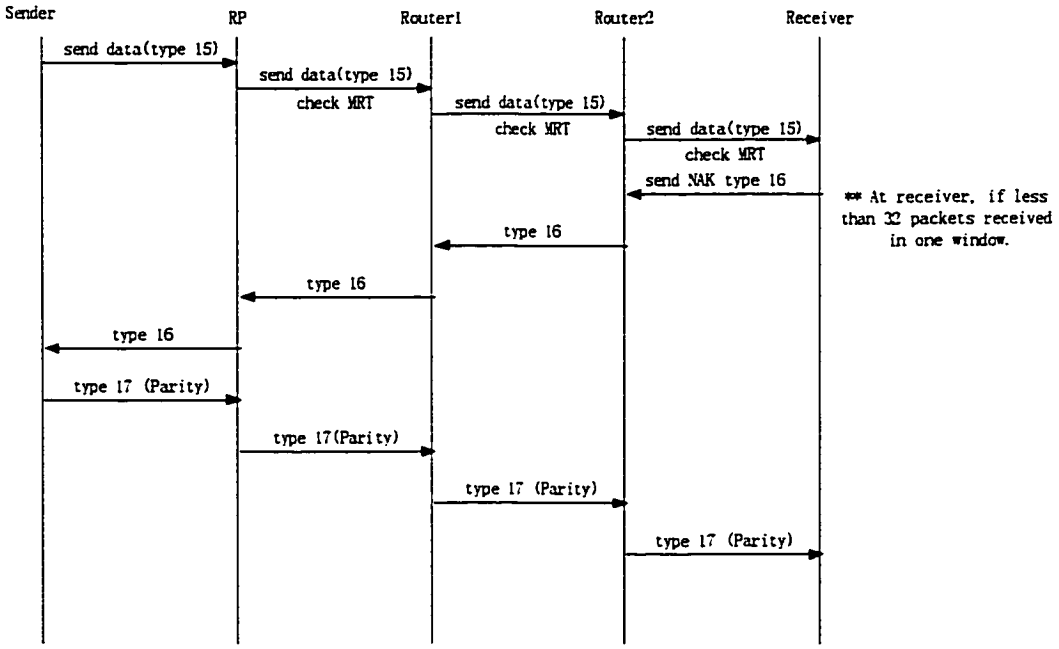



**Figure 4.2 Phase B: Data Transmission**

tree in the routers , being replicated wherever the multicast tree branches, and eventually reaching all the receivers for that multicast group. Once sender receives NAK (Type 16) from receiver, it will send retransmission packet (Type 17) to receiver by unicast.

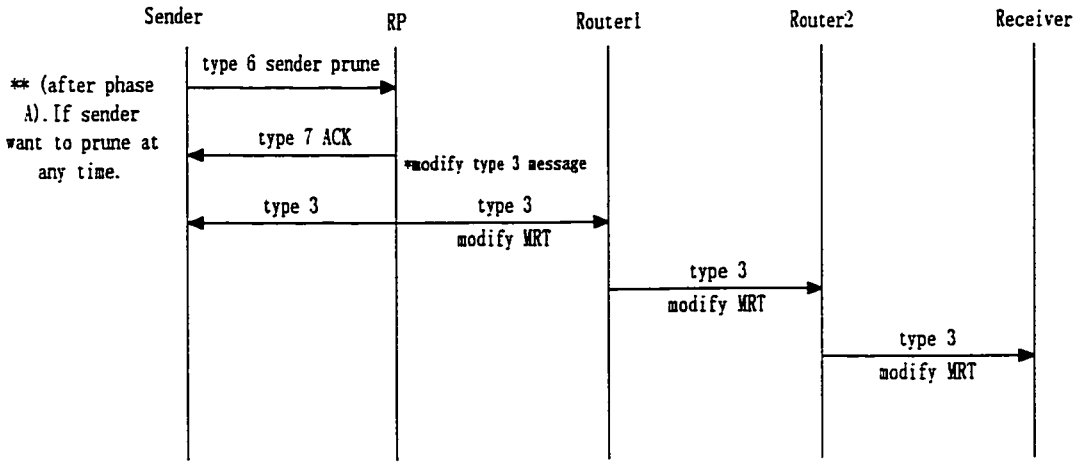



**Figure 4.3 Phase C: Sender Prune**

### Phase C: Sender prune

When sender wants to quit the group, it will send prune message (Type 6) to RP by unicast. After sender receives ACK (Type 7) from RP, RP will modify and flood its Type 3 message accordingly and routers will update their multicast routing tables.

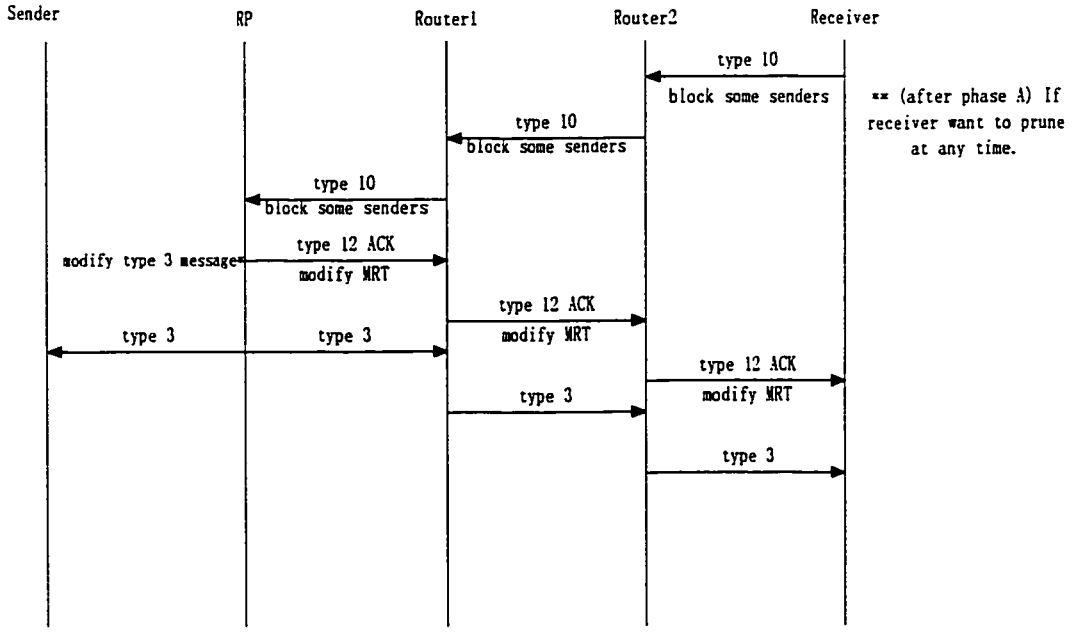



**Figure 4.4 Phase D: Receiver Prune**

**Phase D: Receiver Prune**

When all receivers on a leaf-network leave the group, the DR will send a receiver prune message (Type 10) towards the RP for that multicast group. After receiving ACK (Type 12) from upstream router, the downstream router will modify its MRT, accordingly RP will change its Type 3 message and flooding. However if the prune message is not sent for any reason, the state will eventually time out.

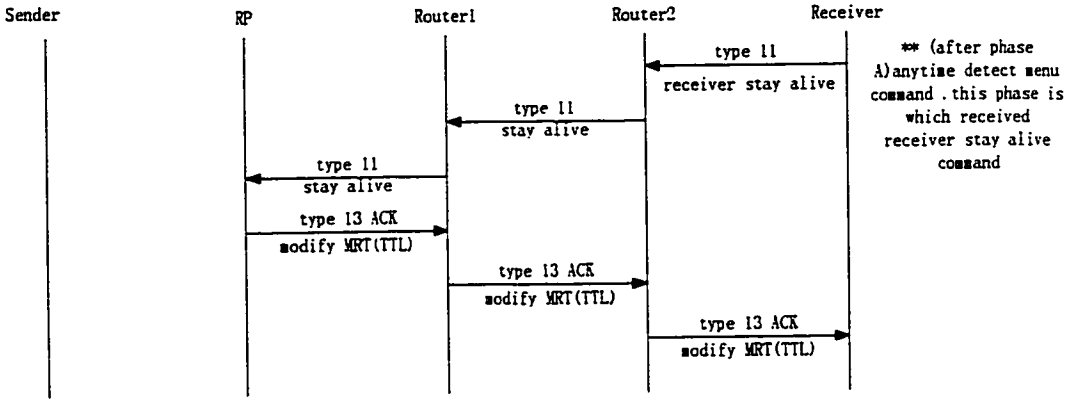



**Figure 4.5 Phase E: Receiver Stay Alive**

**Phase E: Receiver Stay Alive**

Receiver stay alive messages (Type 11) are sent periodically so long as the receiver remains in the group. After receiving ACK (Type 13) from upstream router, the downstream router will modify its MRT, accordingly RP will change its Type 3 message and flooding.

### 4.1.3 Control Messages Format

This scheme proposes several control messages following some ideas of PIM sparse mode routing protocol to establish and maintain multicast tree. However, quite different from PIM, this scheme is mainly working on application layer so as to be easily deployed in practical way.

**Type 1**: Source sends join message to RP by unicast, repeat for each session.

| Type 1 | Sender ID | RP ID | Max_hops | No_groups | Group_id | ... |
|---|---|---|---|---|---|---|

**Figure 4.6.1 Type 1 message format**

Note: No_groups in this control message means the number of groups, and the next field Group_id indicates the address of each group. In this scheme, we suppose to have several groups and sender can join different groups. Moreover, we have more than one RP in this scheme, and each RP will handle one multicast session.

**Type 3**: RP advertises each session it is handling or willing to handle.

| Type 3 | Flag | RP ID | No_groups | Group_id | ... | Length_list | Sender_id | ... | Hop count |
|---|---|---|---|---|---|---|---|---|---|

**Figure 4.6.2 Type 3 message format**

Note: Type 3 control message is built at each interested RP and sent repeatedly (periodically flooded) to all its outgoing interfaces by using UDP.

**Type 4**: ACK for Type 1, from RP to sender (unicast). Same format as Type 1.

**Type 5**: ACK for Type 4, from sender to RP (unicast). Same format as Type 1.

**Type 6**: Sender prune message, from sender to RP. Same format as Type 1.

**Type 7**: ACK for Type 6, from RP to pruning sender. Same format as Type 1.

After sending Type 7 message, RP will modifies and floods its Type 3 message, and routers update their multicast routing tables.

**Type 8:** Receiver sends join message to the nearby DR for all requested senders, then to upstream routers in one session.

| Type 8 | Recnode_id | Receiver_id | RP ID | Group_id | No_senders | Sender_id | ... |
|---|---|---|---|---|---|---|---|

**Figure 4.6.3 Type 8 message format**

Each intermediate router will repeat this message to upstream router until it reaches RP or a router where this session is confirmed in its MRT. This router does not transmit Type 8 any more to upstream routers.

**Type 9**: ACK for Type 8. Same format as Type 8.

If first confirmed router receives Type 8 message, it will return an ACK downstream, which will be propagated all the way until the DR of the requesting receiver is received. During this process, all the intermediate routers will confirm with setting this session as "confirmed" in the multicast routing table.

Note: hop count of Type 3 message is increment by 1 at each router before this message is flooded to all ports, except the receiving port.

**Type 10**: Receiver prune message, sent from receiver to router. Same format as Type 8.

The router will update its table accordingly by eliminating this receiver from the list of receivers interested in those senders.

**Type 11**: Receiver stay alive message, from receiver to router. Same format as Type 8. Once this packet is received at the router, the corresponding TTL is initiated or refreshed in the routing table.

**Type 12**: ACK for Type 10, from router to receiver. Same format as Type 8.

**Type 13**: ACK to Type 11. Same format as Type 8.

**Type 14**: RP informs sender to send data.

| Type 14 | Sender ID | RP ID | Group_id |
|---|---|---|---|

**Figure 4.6.4 Type 14 message format**

As soon as RP receives at least one Join message from one of the downstream routers (notice that this router will not send a Join message to RP unless it receives Join message from downstream routers or receivers), this Type 14 message would be transmitted to the applicable senders from RP.

**Type 15:** Data packet, from sender to receiver.

| Type 15 | Sender ID | RP ID | Group_id | Data |
|---|---|---|---|---|

**Figure 4.6.5 Type 15 message format**

**Type 16**: NAK for Type 15, from receiver to sender.

| Type 16 | Sender ID | RP ID | Group_id | Recnode_id | Receiver_id | Seq_Num |
|---|---|---|---|---|---|---|

**Figure 4.6.6 Type 16 message format**

**Type 17**: Retransmission packet, from sender to receiver.

| Type 17 | Sender ID | RP ID | Group_id | Recnode_id | Receiver_id | Data |
|---|---|---|---|---|---|---|

**Figure 4.6.7 Type 17 message format**

Note: all above fields are one byte, except the data field in Type 17 and Type 15 are 1029 bytes in length.

### 4.1.4 Experimental Networking Topology

The topology of our networking system is shown as figure 4.7. In this experiment, we utilize PCs with several LAN cards (each PC has maximum 4 LAN cards) to work as routers. LAN cards connect every two routers. In order to utilize PCs capacity sufficiently, we install two user programs and one router program in one PC which runs Linux operating system (Redhat 7.1). Hence, each PC is working as router and two users. Moreover, our implementation program is working on application layer.

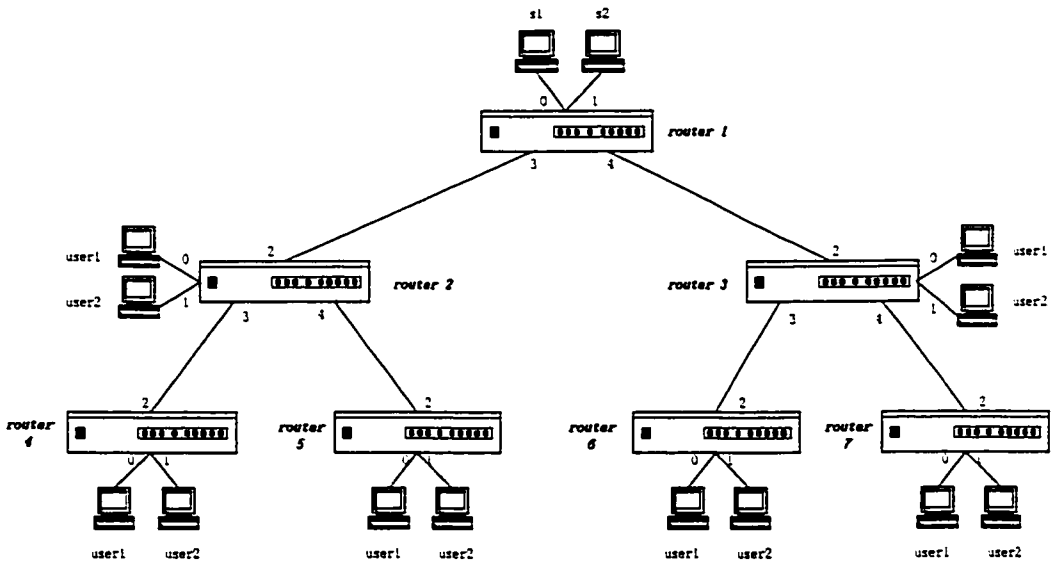



**Figure 4.7: Experimental Networking Topology**

A simple configuration of seven PCs (Five Pentium III 600Mhz with 64 MB

memory and Two Pentium II 233Mhz with 64 MB memory), connected to each other by using 10/100 BaseT Ethernet NIC. IPwave 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). Many experiments were conducted for different amounts of packet loss. We choose this binary tree because it is the most common topology for multicast communication.

Since sender, receiver, router and RP have to fulfill different tasks, our implementation will be mainly four different programs (C code). We adopt a sort manager to divide all these four programs according to their task.

First of all, the program will be started from RP, the RP will advertise each session it is handling or willing to handle. According to the unicast routing table, Type 3 control message is built at each interested RP and sent repeatedly (periodically flooded) to all its interfaces by using UDP socket. Every router will repeat the same process, i.e. it will flood received Type 3 message except the port it received from, and change MRT as RP change its Type 3 message if necessary. Secondly, sender unicasts to each potential RP by using control message of Type 1, and repeats this process for each session.

It is 3-way handshake between sender and RPs. After receiving Type 1, each RP will answer by ACK and sent by unicast (Type 4 control message). If sender cannot receive any ACK from RP, it will timeout and try again. If sender receives ACK from RP, it will send Type 5 message to RP by unicast.

### 4.1.5 Several Important Issues for this Implementation Scheme

#### 4.1.5.1 Establish Multicast Routing Table

In this scheme, Multicast Routing Table will be established dynamically according to group, source, destination, incoming interface, outgoing interface...etc, which is shown as table 4.1. The structure of link table is adopted for the establishment of the multicast routing table. The pointers of source and group will search the link table in order to find the multicast routing entry. The multicast routing table firstly will be initialized, and then it will be modified according to the different type of control message such as join or prune message...etc.

**Table 4.1:Multicast Routing Table at Certain Router**

| Group | Source | RP | Iif | Hop counter | TTL | | Next node or users | Oif | TTL | Confirm |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | S1 | 1 | 2 | 2 | 300 | | 4,5 | 3,4 | 300 | Y |
| 2 | S2 | 1 | 2 | 2 | 200 | | 5 | 4 | 200 | ? |
| | | | | | | | | | | |
| | | | | | | | | | | |

Note: The content about this table is just an example. We assume:

1. This node is router 2.(see figure 4.7)
2. S1 and S2 are senders in router 1. S1 for group 1. S2 for group 2.
3. Receiver1 (in router 4) has joined the group 1.
   Receiver2 (in router 5) has joined the group 1.
4. Receiver3 (in router 5) has joined the group 2.

Iif: Incoming interface of this node; Oif: outgoing interface of this node.
Confirm means this multicast routing information has been acknowledged by upstream router.

### 4.1.5.2 Unicast Routing Table

The unicast routing table will be established statically which is shown as table 4.2. In this table, we list all sources and destinations, and the next hop. For example, if a message is sent from node 3 to node 2, it will be sent to node 1 (next hop) through Oif 2 according to this unicast routing table. This table shows the shortest path between source and destination and will also be used to build multicast routing table.

**Table 4.2: Unicast Routing Table**

| S / D | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | | 7 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Oif | N | Oif | N | Oif | N | Oif | N | Oif | N | Oif | N | Oif | N |
| 1 | --- | --- | 2 | 1 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 3 | 2 | 3 |
| 2 | 3 | 2 | --- | --- | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 3 | 2 | 3 |
| 3 | 4 | 3 | 2 | 1 | --- | --- | 2 | 2 | 2 | 2 | 2 | 3 | 2 | 3 |
| 4 | 3 | 2 | 3 | 4 | 2 | 1 | --- | --- | 2 | 2 | 2 | 3 | 2 | 3 |
| 5 | 3 | 2 | 4 | 5 | 2 | 1 | 2 | 2 | --- | --- | 2 | 3 | 2 | 3 |
| 6 | 4 | 3 | 2 | 1 | 3 | 6 | 2 | 2 | 2 | 2 | --- | --- | 2 | 3 |
| 7 | 4 | 3 | 2 | 1 | 4 | 7 | 2 | 2 | 2 | 2 | 2 | 3 | --- | --- |

Source node

Destination node

Next Hop node and *outgoing* interface of source node

S: Source node Oif : outgoing interface of source node

N:Next node D: Destination node

#### 4.1.5.3 Join/Prune Message

The mechanism of Join/Prune message allows a host (sender or receiver) to inform its local router that it wishes to receive transmissions addressed to a specific multicast group or leave this group. Also routers periodically query the LAN to determine if any group members are still active. A Join/Prune message consists of a list of groups and a list of joined or pruned sources for each group. When processing a received Join/Prune message, each joined or pruned source for a group is effectively considered individually. Since sender and receiver can join or prune at any time, we utilize non-blocking I/O processing (select) of Linux socket to respond each request.

The MRT will be modified according to the received Join/Prune message. Based on this, a router is able to determine which (if any) multicast traffic needs to be forwarded to its "leaf" subnetworks.

#### 4.1.5.4 Trees with Rendezvous Points

In this scheme, the trees are rooted at Rendezvous Points. The trees consider multiple senders and receivers. Corresponding algorithms are inherently based on the establishment of rendezvous points in the network. These rendezvous points are familiar with group membership. They receive the required information, for example, they are notified if a new member has joined a group. When this information is forwarded, it passes through all routers located between the new member and the rendezvous point.

This allows those routers to extract whatever information is relevant to them, such as group membership. A router therefore stores information that provides details about the group membership of subsequent receivers. If no group member is located on that side of the router, the link is inactive.

The basic procedure for these trees initially requires the selection of a rendezvous point for a group. The members of the group issue the appropriate data units in order to register at the rendezvous point. Routers located on the path between group member and rendezvous point forward this register data unit toward the rendezvous point. All they need is information about the group. Trees with rendezvous points do not require that the data sources (i.e., the senders) are members of the group because they are only concerned with routing aspects (i.e., with the destination of data units). The data flow is as follows: the data units are forwarded from the sender to the rendezvous point. From there they are distributed to the receivers of the group.

#### 4.1.5.5 Socket Management

For each node, we build at least 3 Linux UDP sockets, two for two users, and others for its interfaces. Each interface is bound to a different port. Before exchanging all the control messages, hello messages are sent on each multicast-enabled interface. This allows a router to learn about the neighboring routers on each interface.

#### 4.1.5.6 Proactive FEC/ARQ in this scheme

We have mentioned the basic proactive FEC/ARQ approach in Chapter 2. Now we describe how this technique is used in this scheme. Since the regular network loss is less than 10 %, we choose 4 parity packets in order to endure 10 % loss for the network. The sender multicasts the 32 data packets during the first round, with 4 RSE repair packets. After the first round transmission ends, each receiver assesses the total number of packets that it has received. If receiver i has received $k_i < 32$ packets belonging to the block, it sends NAK to the sender for FEC repair packets. The NAK specifies $32 - k_i$ additional repair packets which will be transmitted in the next round by the sender. This process repeats until the receivers have obtained the 32 packets.

**Table 4.3:RSE code specification in the scheme**

| | |
|---|---|
| Symbol Size | 8 bits |
| Original data packet number in one block | 32 packets |
| Proactive packet number | 36 packets (32+4) |
| Packet Size | 1024 bytes |

Note: The code used in this scheme operates in GF($2^8$).

### 4.1.6 Main program flow charts

Figures 4.8 to figure 4.11 show sender, receiver, router and RP main flow charts.

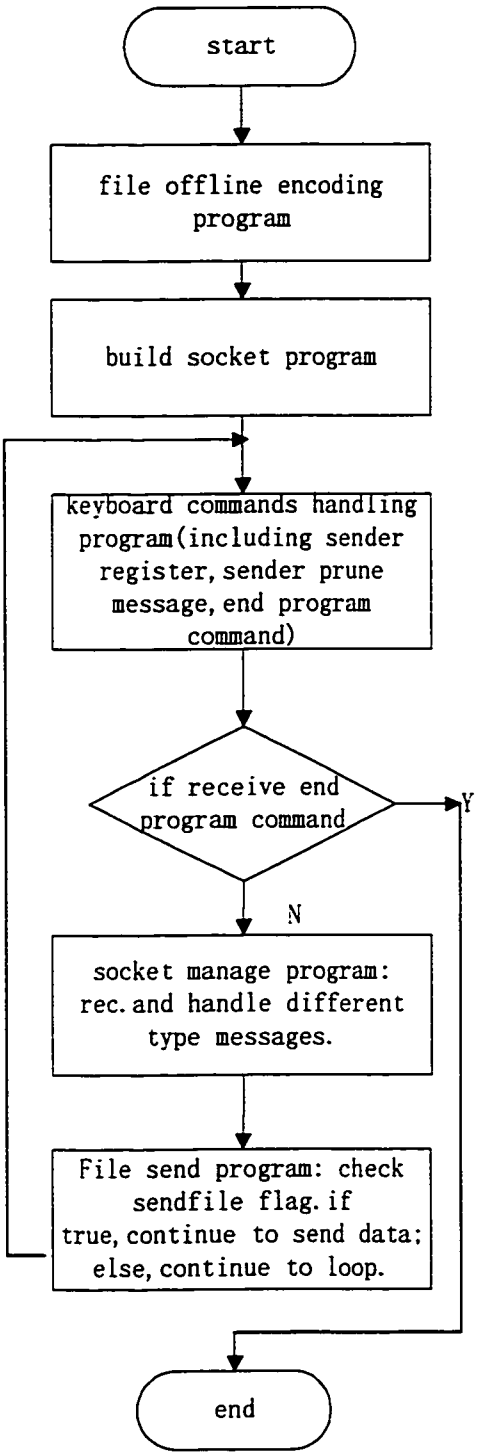



Figure 4.8 Sender Main Program Flow Chart

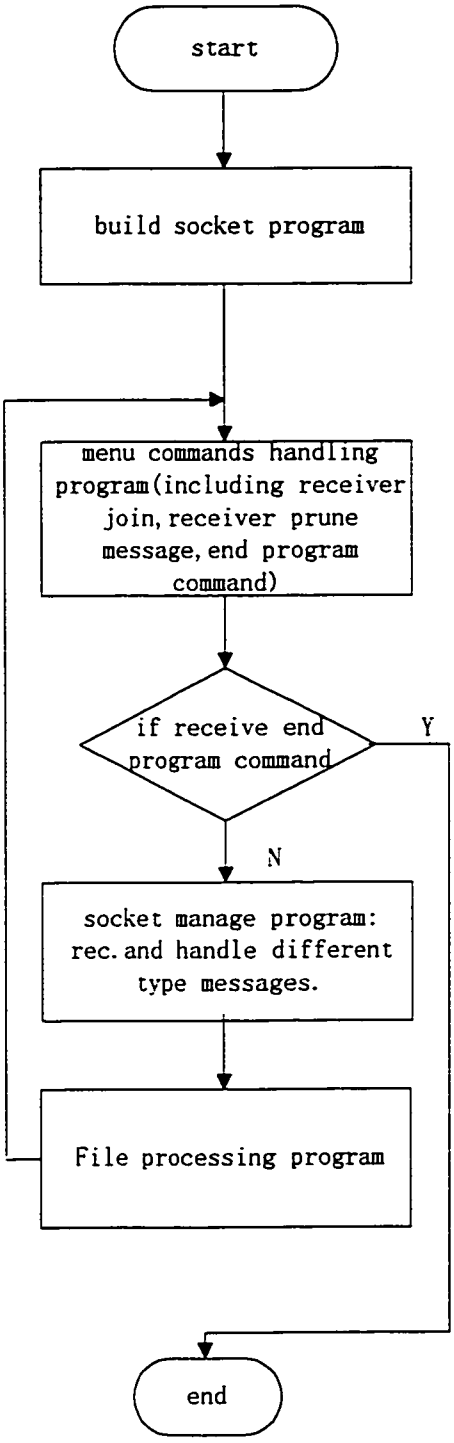



Figure 4.9 Receiver Main Program Flow Chart

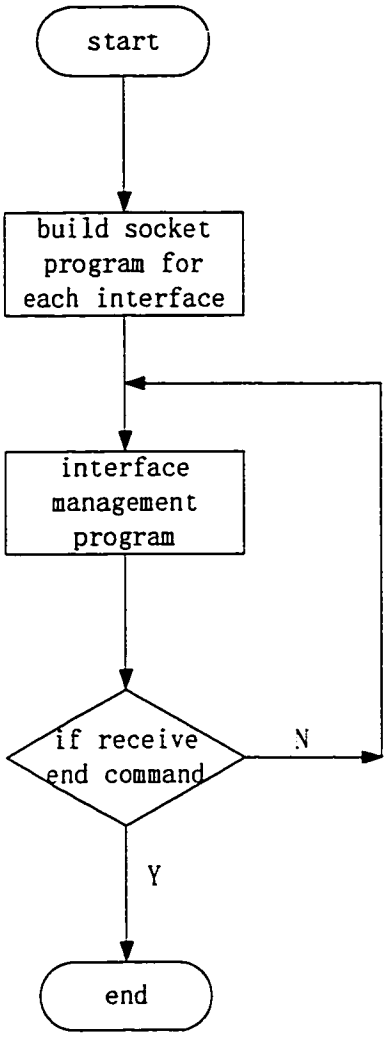



Figure 4.10 Router main program flow chart

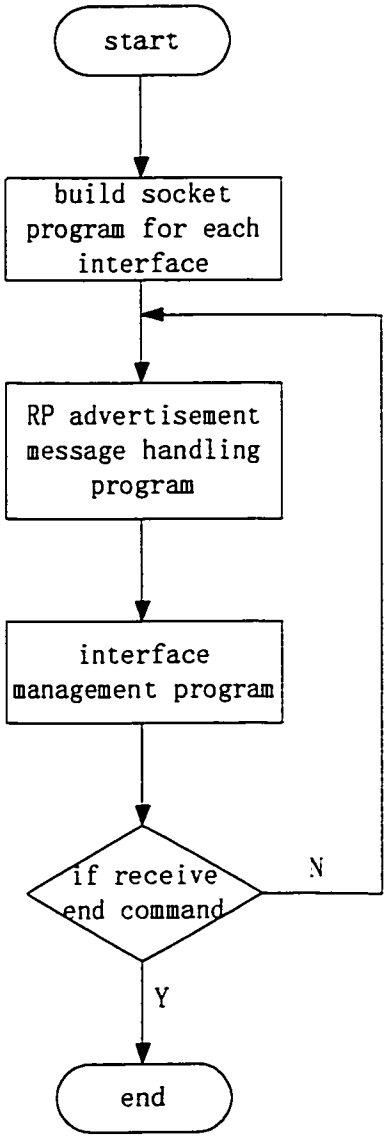


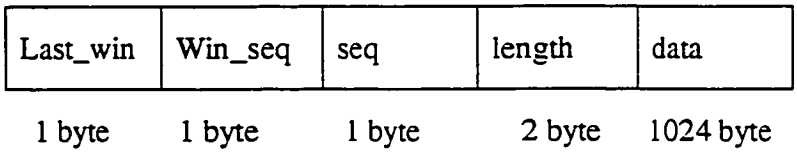

Figure 4.11 RP main program flow chart

### 4.1.7 Detailed Processing of the Procedure

#### 4.1.7.1 Offline Encoding

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 voice... multicast session or one of the intermediate routers.

**Figure 4.12: Data Packet format**




The format of each packet is shown as Figure 4.12, the first field of Last_win indicates the last window of the whole file, if Last_win equal to 1, the receiver will know it's the last window of the file, and vice versa. Since we can't guarantee the size of the last window is as same as the regular window, we have to treat the last window specially. Therefore, we give a field to indicate the last window in this scheme. Each packet of data, 1024 bytes is given a sequence number in the range 1 to 64, the sequence number field in the corresponding field is preceded by a Win_seq field which we use to indicate the number of windows.

After reading 32 packets data from the original file, we feed them to the Reed Solomon encoder routine which generate 32 parity packet according to these 32 data packets, and add 1 byte packet sequence to each packet. Since this file will be divided

into windows, we add 1 byte window sequence (0 to 63 packets). The output of RSE encoder will be written to temporary file. Next we check if it is the end of file. At the end of the file, we close the file; otherwise, the loop will be continued.

Much of the processes above are done offline in multicast applications, i.e. the whole MPEG or voice file ...etc is FEC encoded as above and stored before starting the multicast in real time. The process of mixing repairs and original data packets has to be executed in real time though once the multicast session starts. The Negative Acknowledgment NAK (Type 16) control message format is also shown at section 4.1.3. 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.

#### 4.1.7.2 Sending File, Processing Retransmission

The server routine groups up to 2 packets in one UDP data unit, i.e. TPDU. Since we adopt proactive FEC technique, we will send 4 out of 32 parity packets along with the 32 original data packets in order to reduce retransmission. These 36 packets in one window will be sent from sender to receiver along the multicast tree.

Because of this proactivity, if any 4 packets are lost, RSE decoders can conceal up to 36-32 = 4 lost packets. Such loss is not detrimental and will not result in retransmission requests. For loss concealment 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 [9].

Upon received NAK (Type 16) from receiver, the sender will get the max number of loss from NAK buffer, set m=max number of loss, then the sender will select the m parity packets of this window which differ from the parity packets which have been sent before from the temporary file. The sender will capsulate the parity packets into Type 17 message packets, fill them into the send buffers, and send to the RP by using UDP socket. Priority is given to retransmission packets (Type 17), and data packets (Type 15) are transmitted only when there is no repair packet to be sent.

#### 4.1.7.3 Propagate Data Packet

Each TPDU will be encapsulated and decapsulated at each node, and propagated by multicast (Type 15). If one router received Type 15 from upstream router, it will check the MRT, find Oifs to next downstream routers or receivers. Then it will fill the Type 15 messages into the send buffer of each Oif which has been found in MRT, and send the message to the next downstream router by using the socket of the Oif. For Type 17, the router will repeat the same process.

If the router receives NAK message from receiver or downstream router, it will check unicast routing table to find Oif to next node towards Rendezvous Point. Next it will fill the Type 16 message into the send buffer to the next node by using the socket of the Oif. Finally, NAK message will be sent to corresponding senders from RP.

#### 4.1.7.4 Data Processing at Receiver

At receiver, the type field for the received message is checked at first. If the type field indicates a data message (Type 15), this packet is sent to the temporary buffer according to the window sequence one by one. If the first 32 data packets have been received (before the expiry of a certain time out on the timer), the first 5 bytes of each packet are stripped and the following 1024 data bytes delivered to the receiver buffer. If any data loss happened and at least 32 packets are received, RSE decoding will be tried. The successfully decoded packets will be delivered to the receiver buffer. Some losses may happen when some packets arrive too late. In order to avoid this kind of loss in certain time bound, we open 4 window buffers to receive data packets. If the fourth window starts to receive data packets, data packets of the first window in the receiver buffer will be written to file. If less than 32 packets are received, we calculate the number of retransmission required, then a NAK (Type 16, parity request) is formulated and send to the sender by unicast.

If the type field indicates a retransmission packet (Type 17), the parity packets are stored to the corresponding window buffer and we count the number of the packets. As soon as the counter of the packets in this window is equal to 32, we continue the RSE decoding process and writing file process as we mentioned above.

The constant length of 1024 bytes, the type, and the sequence fields of each packet will enable correct parsing and processing at the client side. If the proactive parity

packet is 4, then each window can have 4 packets loss without any retransmission. Since RSE decoder can decode any 32 packets to recover the original data packet, the NAK buffer just needs to indicate the maximum number of requested repair packets, and it does not need to tell the sender which specific packet should be retransmitted. This significantly reduces the length of NAK buffer.

## 4.2 Description of the Experimental Results

Many experiments have been conducted for different group size, amounts of packet loss, and proactive parity packets at different transmission rate. The results are presented in this section.

### 4.2.1 Unicast Communications

Figures 4.13 ~ 4.15 show a sample of the obtained results by unicast communication. Fig. 4.13 shows the percentage of packets for which the RSE decoder was called. For low data rates and /or low random loss, this percentage is low but steadily rises as the induced loss increase. Figure 4.14 shows the percentage of windows generating a NAK did not increase much due to the fact that many data packets were recovered by the proactively transmitted parity packets (shown as Figure 4.13).

Figure 4.15 shows that very few retransmissions are possible even at high data rates, and high loss probabilities due to RSE decoding.

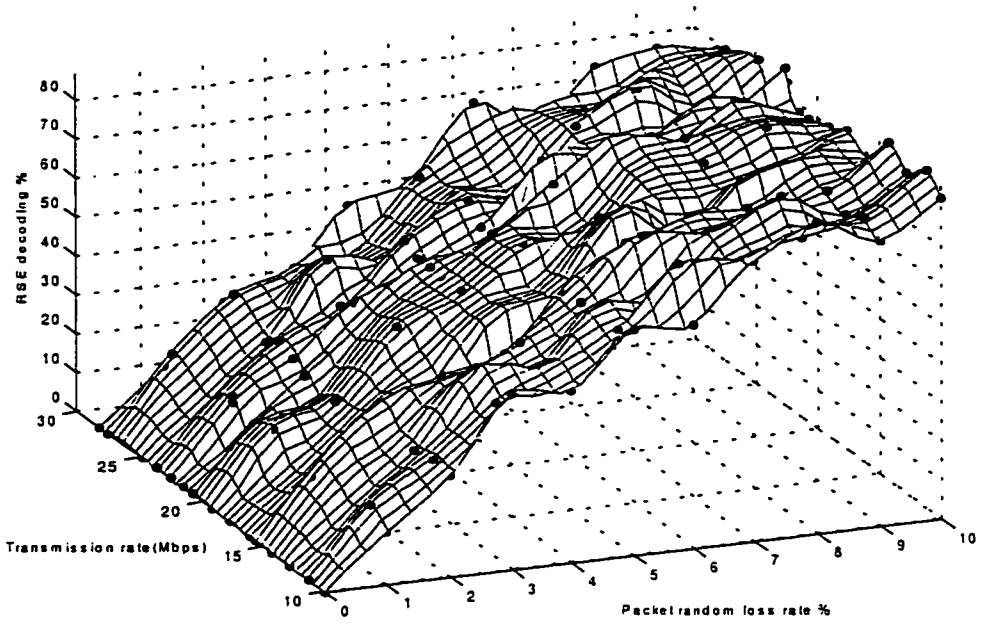



**Figure 4.13 Percent of RSE decoder being called vs. rate and random loss for the unicast hybrid FEC/ARQ scheme**

Figure 4.14~Figure 4.15 also show that the processing constraint appears after the transmission rate exceeds 28 Mbps. It should be noted that the NAK and retransmissions happen when random loss rate is around 4% although loss correction capability of RSE code in this scheme is higher than 10%. It is because random loss in each window is not uniformly distributed.

We also compared this RSE code with RSC code and BCH code for unicast communication. We found that RSE is stronger for erasure correction, its encoding and decoding algorithm is comparatively simple, and it is convenient for multicast communications.

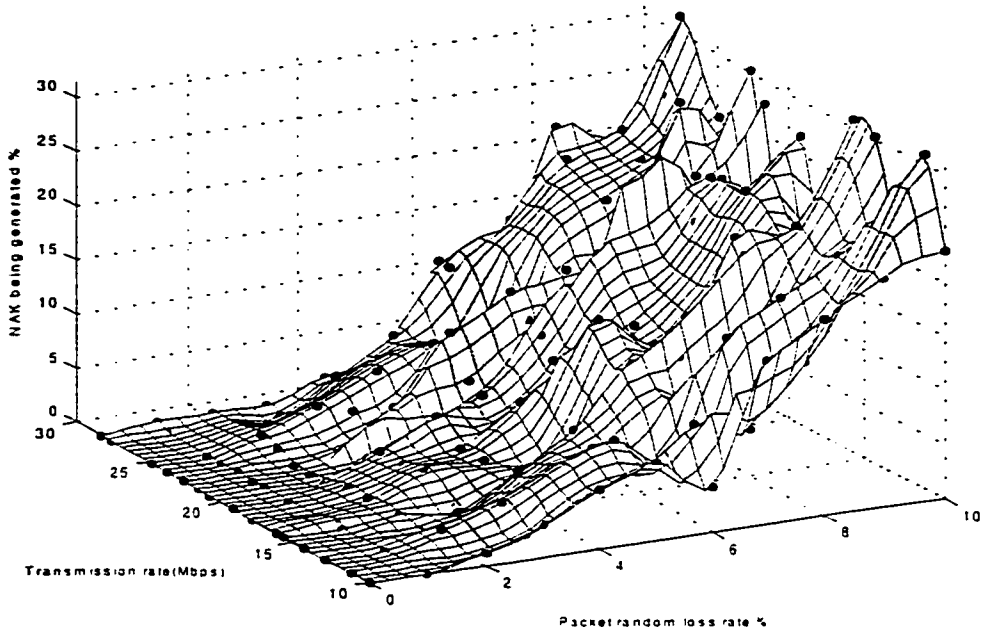



**Figure 4.14 Percent of windows generated NAKs vs. rate and random loss for the unicast hybrid FEC/ARQ scheme**

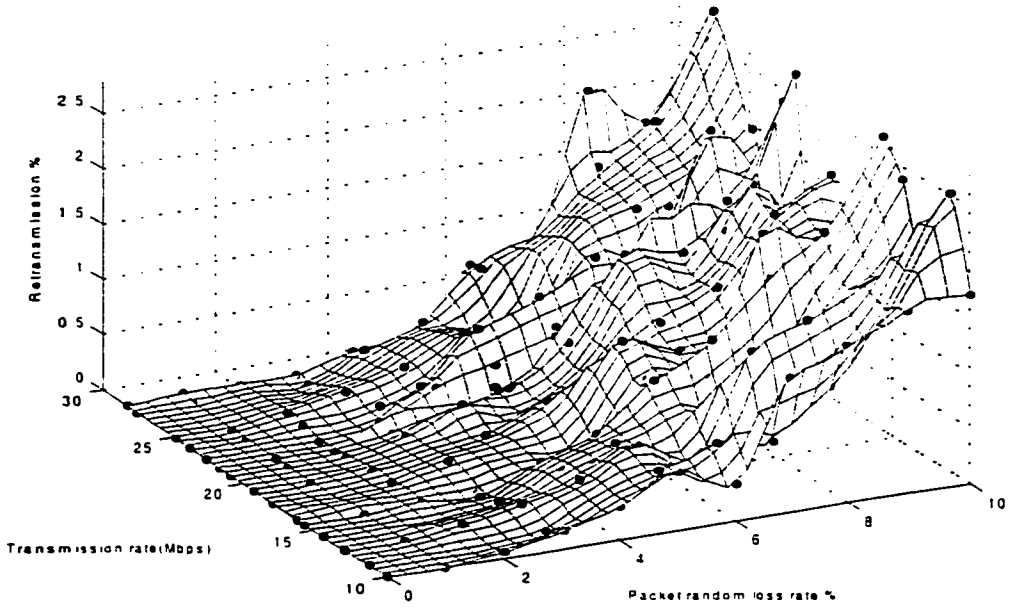



**Figure 4.15 Percent of retransmissions vs. rate and random loss for the unicast hybrid FEC/ARQ scheme**

### 4.2.2 Multicast Communications

In this section, we will compare the hybrid FEC/ARQ technique with the pure ARQ technique for multicast communications. ARQ is generally used in unicast protocols. Since no manipulations of the data stream are required, and only missing packets are retransmitted, ARQ is extremely inexpensive for unicast. In presence of losses, when the bandwidth-delay product approaches the sender's window, ARQ might result in a reduced throughput; also, data recovery requires at least an additional RTT, which can make this technique unattractive when high latencies cannot be tolerated. In addition, the effect of ARQ decreases in multicast communication when the number of (groups of) clients with uncorrelated losses grows.

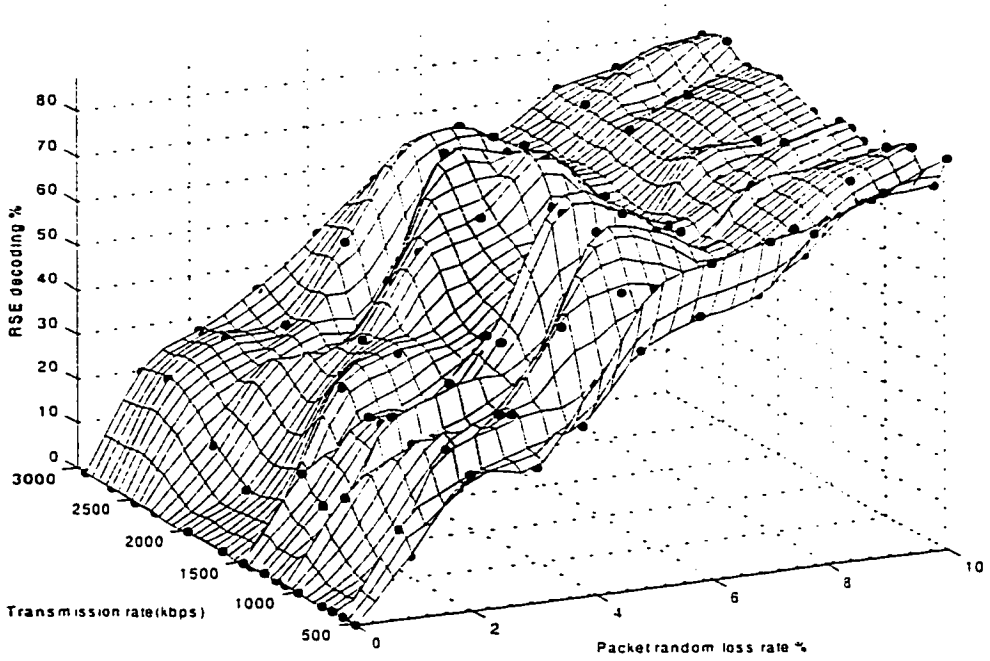



**Figure 4.16 Percent of RSE decoder being called vs. rate and random loss for the multicast hybrid FEC/ARQ scheme (group size 2)**

Figures. 4.16 ~ 4.20 show a sample of the obtained results by multicast communication which has two receivers in one group. Fig. 4.16 shows the percentage of packets for which the RSE decoder was called. For low data rates and /or low random loss, this percentage is low but steadily rises as the induced loss increase. Figure 4.17 and Figure 18 shows the percentage of windows generating a NAK respectively for hybrid FEC/ARQ and pure ARQ technique. Here we can easily see that hybrid FEC/ARQ greatly reduces the number of NAKs compared to the pure ARQ technique because many data packets were recovered by the proactively transmitted parity packets. This will mitigate so called NAK implosion problem. And we find that by using hybrid FEC/ARQ technique, the transmission rate can be achieved at higher level.

Figure 4.19 and Figure 4.20 show that FEC/ARQ technique needs much less retransmissions than ARQ technique in our scheme.

Furthermore, by using RSE decoder, the receiver only need to request the number of additional parity packets for FEC/ARQ, but for pure ARQ technique, the receiver has to indicate every sequence number of lost packets.

The performance test result will not be quite different if we change the window size because we calculate the percentage of NAKs and retransmissions. If the window size increase, the number of NAKs will be less, accordingly the total number of window will be less too. However, we can not use the whole file as one window, that will cause too much delay.

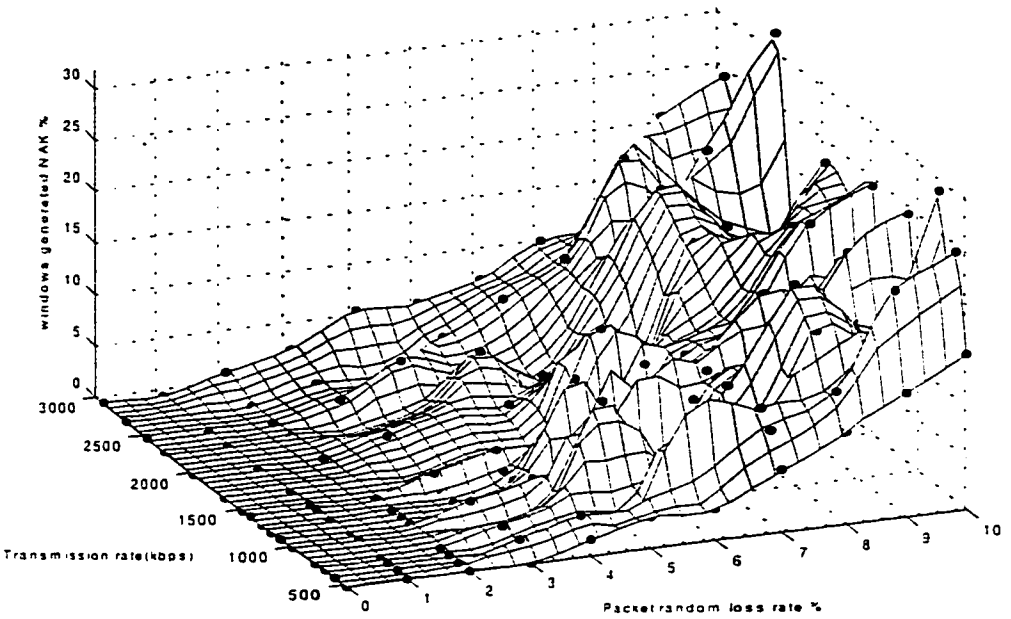



**Figure 4.17 Percent of windows generated NAK vs. rate and random loss for the multicast hybrid FEC/ARQ scheme (group size 2)**

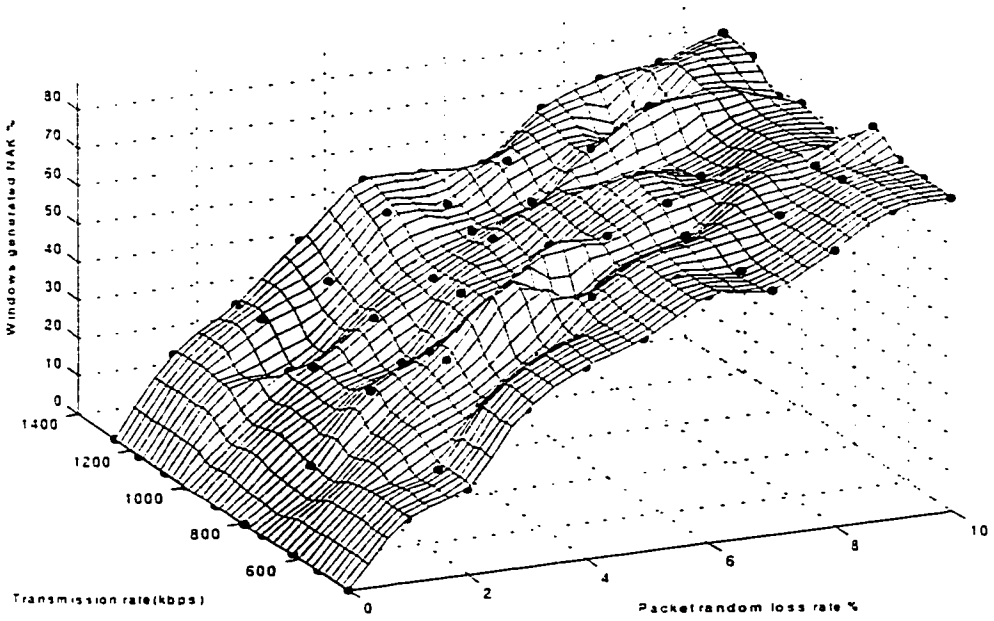



**Figure 4.18 Percent of windows generated NAK vs. rate and random loss for the multicast ARQ scheme (group size 2)**

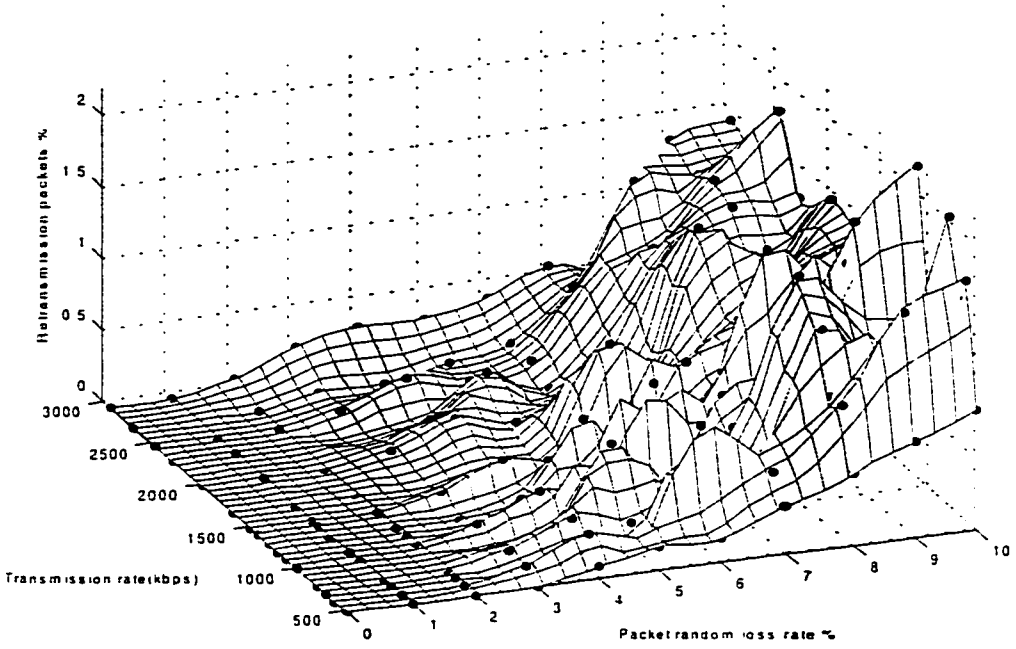



**Figure 4.19 Percent of retransmissions vs. rate and random loss for the multicast hybrid FEC/ARQ scheme (group size 2)**

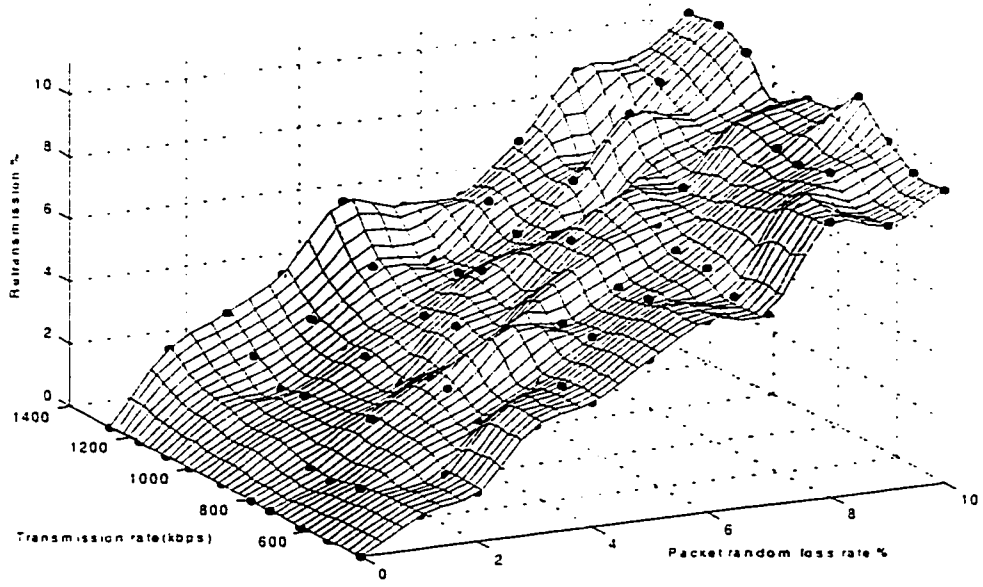



**Figure 4.20 Percent of retransmissions vs. rate and random loss for the multicast ARQ scheme (group size 2)**

Figures. 4.21 ~ 4.25 show a sample of the obtained results by multicast communications (Group Size 10) which are similar to Figures 4.16~4.20. Fig. 4.21 shows the percentage of packets for which the RSE decoder was called. Compare Figure 4.22 with Figure 4.23, we find the percentage of windows generating a NAK did not increase much due to the fact that many data packets were recovered by the erasure decoding capability of RSE codes in Figure 4.21. If there is a large number of receivers that experience packet loss of a magnitude, the sender will instead face a possible NAK implosion for the pure ARQ technique, and the hybrid FEC/ARQ technique will greatly decrease the number of NAK being generated.

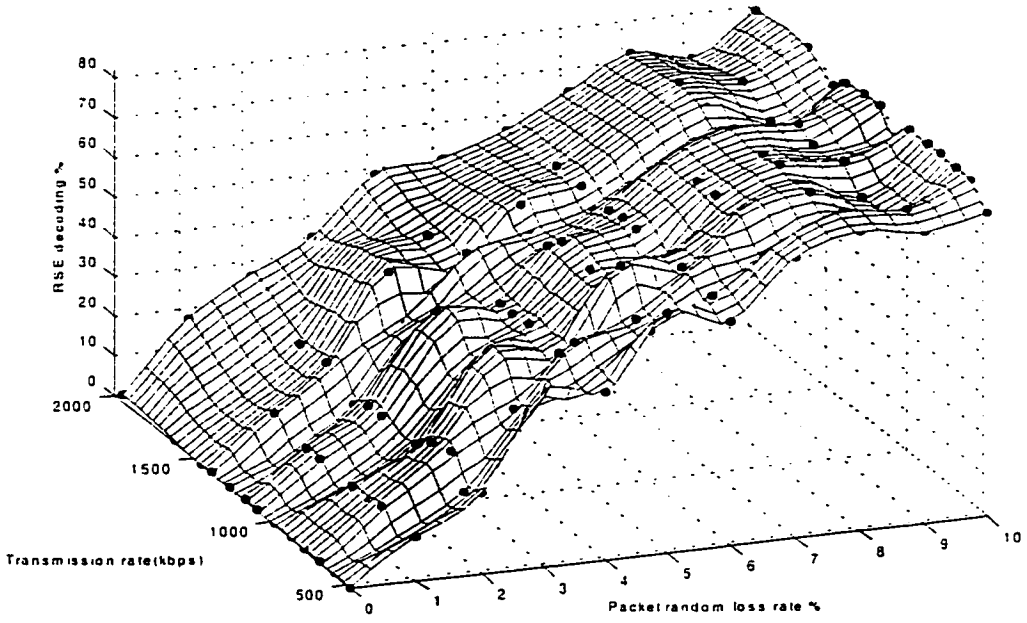



**Figure 4.21 Percent of RSE decoder being called vs. rate and random loss for the multicast hybrid FEC/ARQ scheme (group size 10)**

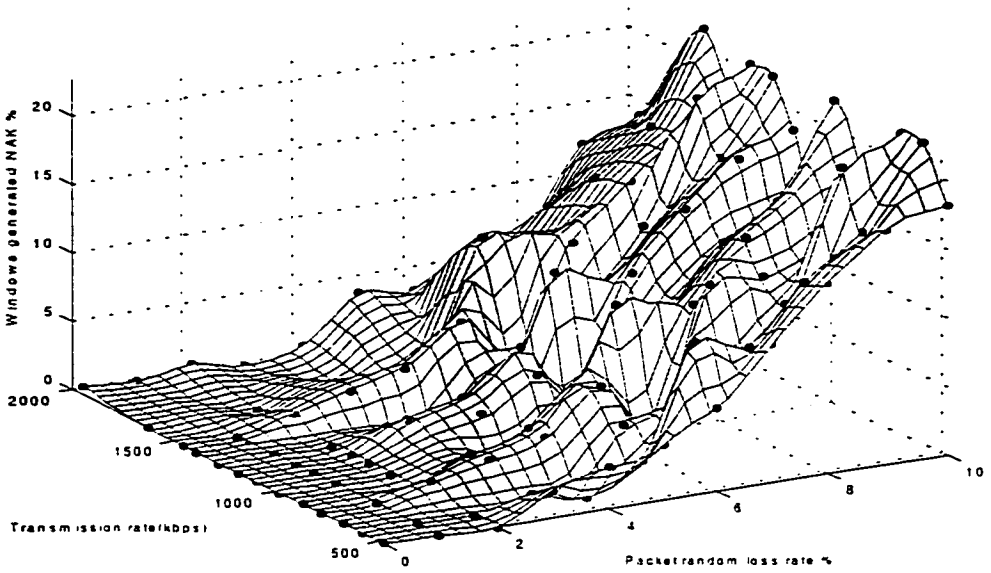



**Figure 4.22 Percent of windows generated NAK vs. rate and random loss for the multicast hybrid FEC/ARQ scheme (group size 10)**

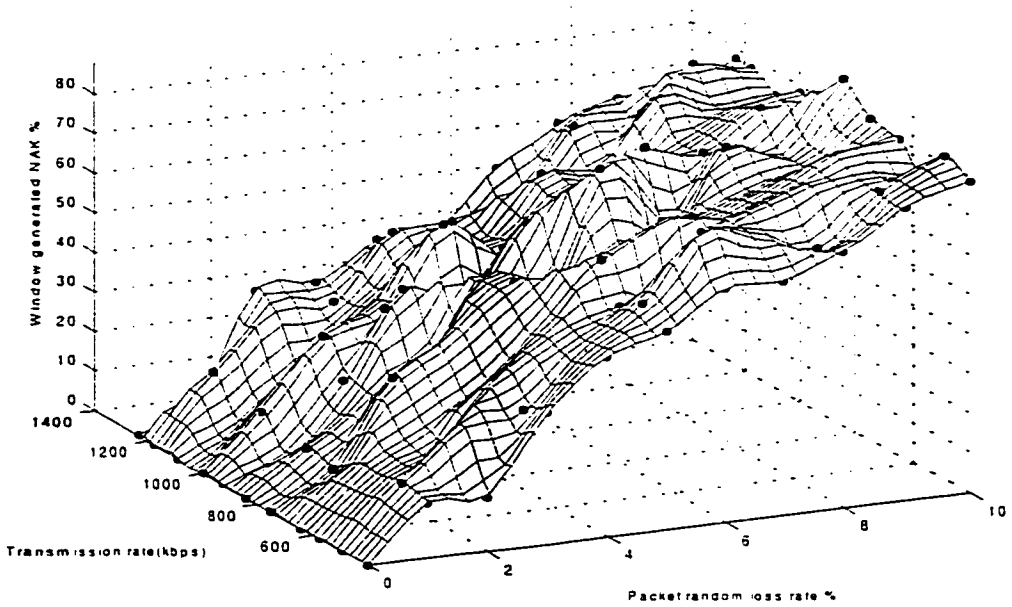



**Figure 4.23 Percent of windows generated NAK vs. rate and random loss for the multicast ARQ scheme (group size 10)**

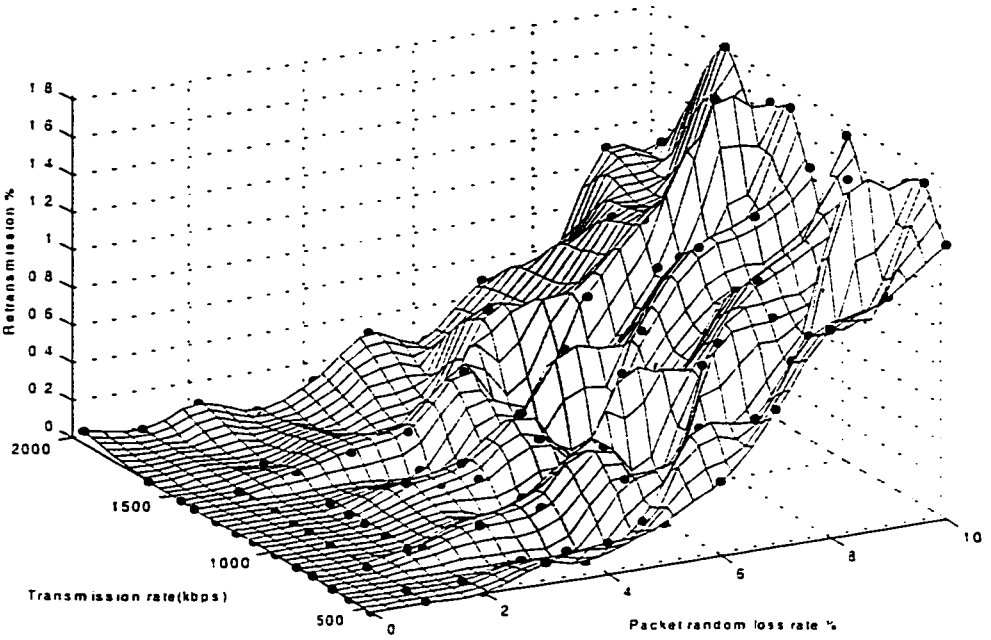



**Figure 4.24 Percent of retransmissions vs. rate and random loss for the multicast hybrid FEC/ARQ scheme (group size 10)**

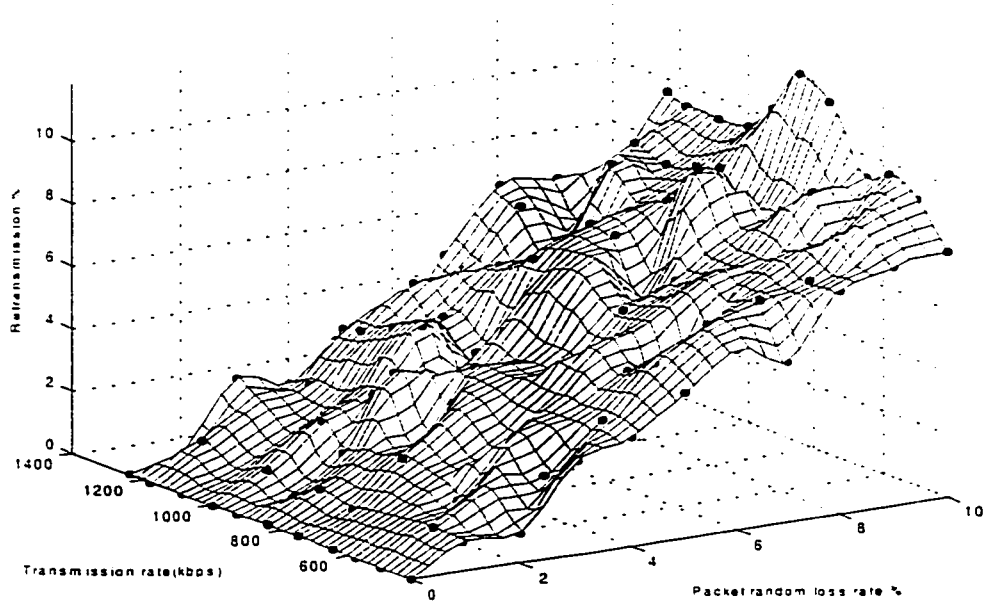



**Figure 4.25 Percent of retransmissions vs. rate and random loss for the multicast ARQ scheme (group size 10)**

Figure 4.24 and Figure 4.25 show that much fewer retransmissions are needed for the hybrid FEC/ARQ technique because of the erasure decoding capability of RSE codes.

In Figure 4.26, we compare the transmission time of hybrid FEC/ARQ against that of pure ARQ. The transmission time is defined as the time it takes from the start of the multicast transmission until all receivers are able to reconstruct the original data transmitted which including all retransmission and processing, window waiting, FEC decoding, ARQ ... etc. Due to the proactively transmitted parity packets, many lost packets can be recovered at receivers; receivers will generate much less NAK as we showed in above figures. Thus, total transmission time could be saved greatly.

The advantage of hybrid FEC/ARQ is especially evident when the number of receivers increases. Moreover, by using RSE decoder, receivers do not recover their lost packets by waiting for retransmissions as in ARQ. Rather, receivers can recover their losses as they receive enough "fresh" repair packets from the sender. The relative reduction in transmission time is more pronounced when the number of receivers increases.

The transmission time for hybrid FEC/ARQ and pure ARQ techniques in figure 4.26 are tested at 4 % random loss rate and 1 Mbps transmission rate.

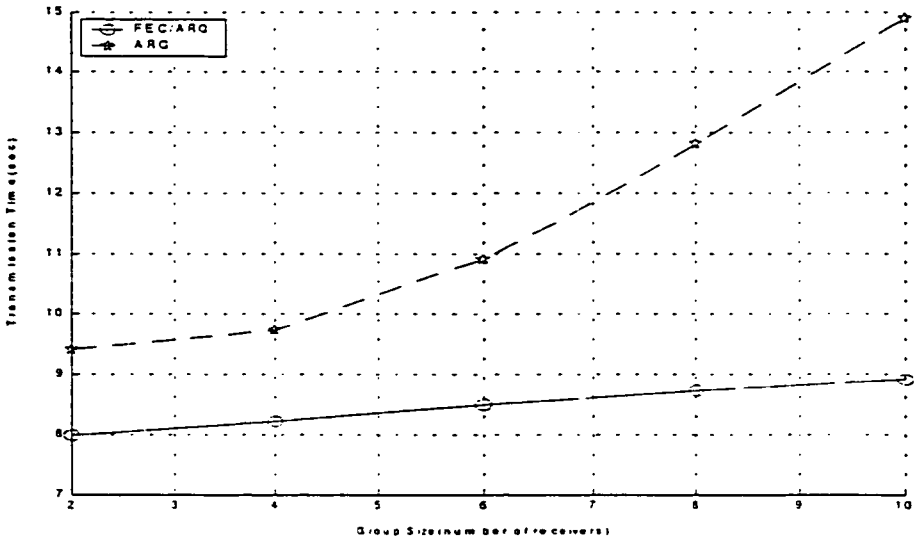



**Figure 4.26 Transmission Time vs. Group Size**

### 4.2.3 Study for Different Proactive Factor

Figure 4.27~4.28 show the reflection of NAK and retransmission with different proactive factor with group size 2 and at 1 Mbps transmission rate. We find NAKs and retransmission packets decrease as proactive parity packets increase when random loss rate less than ten percent. Thus, properly chosing proactive factor will improve the efficiency of proactive FEC/ARQ approach.

Of course, if the network condition is becoming worse, we need more efficient flow control and congestion control technique instead of purely increasing proactive parity packets. And we could also adopt some mechanisms to adjust the proactive factor automatically in order to meet different network conditions.

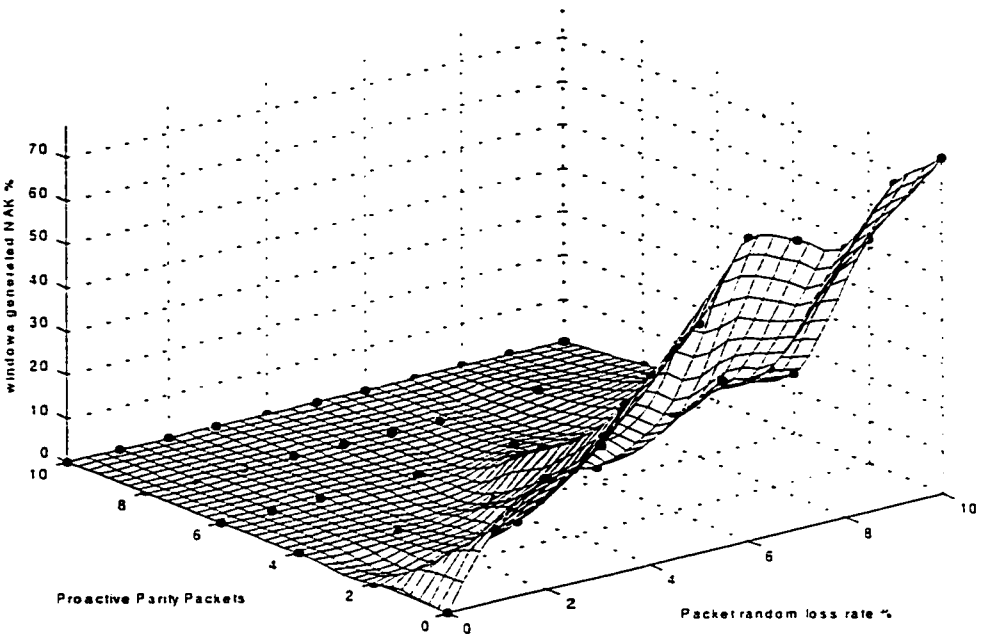



**Figure 4.27 Percent of windows generated NAK vs. proactivity and random loss for the multicast hybrid FEC/ARQ scheme (group size 2)**

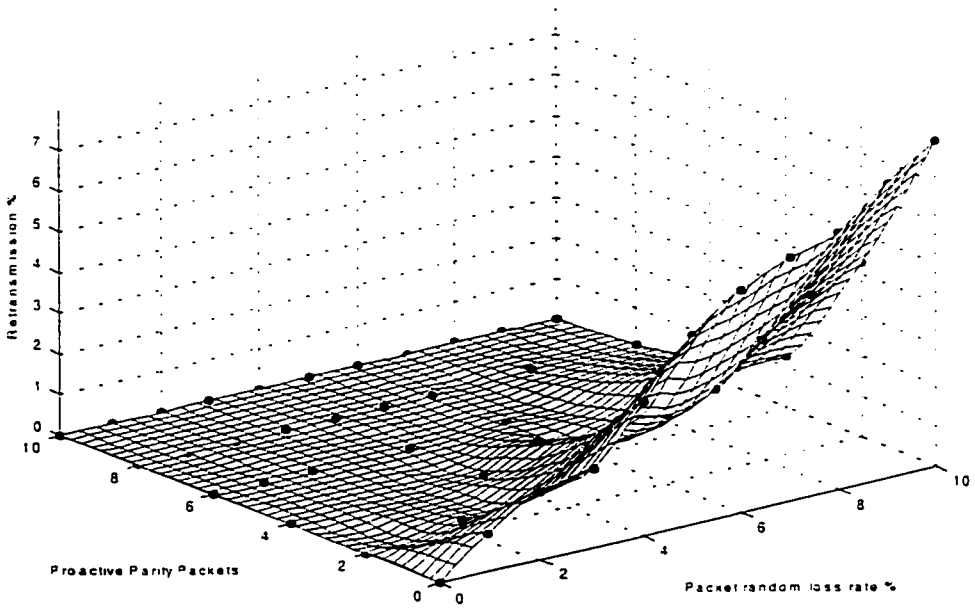



**Figure 4.28 Percent of retransmissions vs. proactivity and random loss for the multicast hybrid FEC/ARQ scheme (group size 2)**

### 4.1.4 Effect of Group Size for Performance

Figure 4.29 and Figure 4.30 show that the NAKs and retransmissions will be increased with the growth of group size. Compared to ARQ, FEC/ARQ results in significantly fewer number of NAKs and retransmission packets.

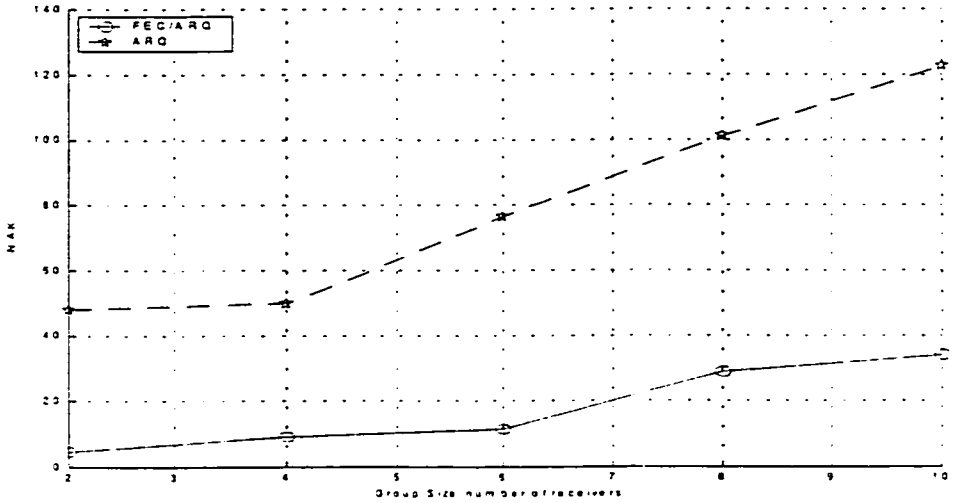



**Figure 4.29 Total number of NAKs vs.Group Size**

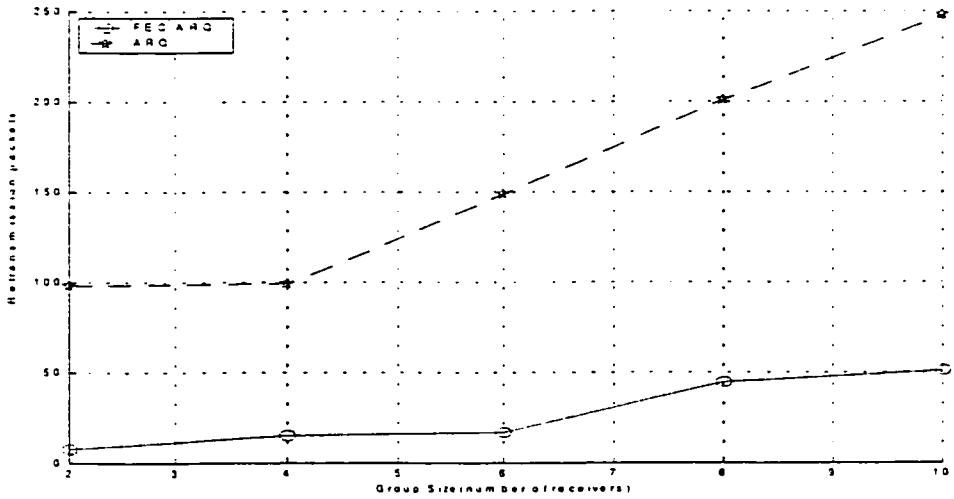



**Figure 4.30 Total number of retransmission packets vs. Group Size**

The test results for figure 4.29 and figure 4.30 are based on 7 % random loss rate and 1 Mbps transmission rate. For group size less than 4, only the second level nodes in our experimental topology (figure 4.7) will handle the multicast traffic. Hence, the slope in these two figures didn't change much during this period. If group size is greater than 4, the third level nodes in the topology will start to handle the multicast traffic, this will greatly increase NAKs or retransmissions than before. Therefore, the slope is change suddenly at this point.

## 4.3 Summary

In this chapter, we have developed a new multicast implementation scheme using a hybrid proactive FEC/ARQ technique, and established a test-bed at ETS in order to test this scheme. Firstly, we described how to build and maintain multicast tree for multicast communication. Next, we discussed how hybrid proactive FEC/ARQ technique is applied to the multicast communication and its positive effect for QoS. Finally, we compared the performance of FEC/ARQ with that of ARQ and concluded that FEC/ARQ is more effective than ARQ for reliable multicast in a practical way. Our preliminary evaluation of hybrid proactive FEC/ARQ technique suggests that it promises shorter transmission time, decreased NAK traffic, and improved bandwidth utilization (reduced retransmission packets) when compared to the traditional ARQ reliable multicast technique. Meanwhile, we also investigated the effects of proactive factor and group size for reliable multicast scheme.

# Chapter 5

# Conclusions and Future Work

## 5.1 Concluding Remarks

In this thesis, we have developed a reliable multicast implementation scheme, and we have presented a combination of RSE FEC/ARQ technique, which provides good QoS experimental results for this scheme in terms of transmission time, NAK traffic, and bandwidth utilization. In this thesis, we have also presented the benefit of multicast, and the need for reliable multicast.

In the first part of the contribution, we have proposed a reliable multicast implementation scheme, and developed a laboratory test-bed to test the performance of this scheme. In order to create and maintain multicast group, we adopt several control messages to build the multicast routing table which is based on unicast routing table. By using these control messages, senders and receivers can join or leave the group at any time. After the multicast tree was built, senders will propagate data packets to all receivers in this group through the Rendezvous Point.

In the second part of the contribution, we have applied proactive hybrid FEC/ARQ to this multicast scheme in order to achieve reliability and better efficiency.

We add a certain FEC redundancy to original data packets. These redundancy packets could allow receivers to recover the original data even under partial loss. ARQ will be applied if the redundancy information is not enough. From our laboratory test result, it has been shown that it provides shorter transmission time, significantly reduced NAK and better bandwidth utilization. In addition, we have examined the effect of proactive factor and group size for reliable multicast.

## 5.2 Suggestions for Future Work

First of all, since NAK suppression technique for multicast tree becomes more and more popular, we could use routers to take an active part in the protocol processing in order to sustain scalability. Like PGM [27], the network element maintains a repair state for every NAK it receives.

Secondly, another possible direction for future work of this scheme is to have all intermediate routers equipped with function of FEC and ARQ. It can be expected that the network bandwidth will be saved. However, a solution would have to be found for the tradeoff between bandwidth and router processing ability.

In addition, the investigation of where to use FEC in multicast tree could be addressed. In [32], Nonnenmacher and Biersack demonstrated that the effect of FEC on the multicast tree is strongly dependent on shared links. Since they only have theoretical analysis, we could have more experimental results for this.

Finally, in our scheme, sender might adjust proactive factor automatically according to the number of NAK or required retransmission packets, which is similar to a new multicast protocol (Adaptive Reliable Multicast) [33]. If the network condition is good, the sender will not transmit too much redundancy. This will greatly reduce the amount of data transmission and reduce the time it takes for all receivers to receive the data intact (without loss). Moreover, the relation among the pattern of loss, the best RSE packet size, the processing ability of receivers and the delay sensitive characteristic of application may be explored further. The hardware implementation for this scheme can also be considered.

# References


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

[2] J. P. Macker, "*Reliable multicast transport and integrated erasure-based forward error correction*", IEEE MILCOM, November 1997.

[3] J. Nonnenmacher, E. W.Biersack, D. Towsley, "*Parity-based loss recovery for reliable multicast transmission*", SIGCOMM'97, September 1997.

[4] William Stallings, "*Data & Computer Communications,6th edition*", Prentice Hall, 2000.

[5] John C. Lin, Sanjoy Paul, "*RMTP: A Reliable Multicast Transport Protocol*", Proceedings of IEEE INFOCOM '96, March 1996, pp. 1414 - 1424. Also available at *http://hill.lut.ac.uk/DS-Archive/MTP.html*

[6] Anthony J. McAuley, "*Reliable Broadband Communication Using a Burst Erasure Correcting Code*", ACM SIGCOMM '90, September 1990.

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

[8] R.J. McEliece, "*The Theory of Information and Coding*", Addison Wesley, 1997.

[9] L. Rizzo, L. Vicisano, "*Effective erasure codes for reliable computer communication protocols*", ACM Computer Communication Review, April 1997.

[10] R.H. Deng, "*Hybrid ARQ Schemes for Point-to-multipoint Communication over Nonstationary Broadcast Channels*", IEEE transactions on communications, vol. 41, No. 9,September 1993.

[12] D.Clark and D.Tennenhouse, " *Architechtural consideration for a new generation of protocol*", in Proceeding of SIGCOMM, Philadelphia, PA, 1990, pp. 201-208.

[13] S.E. Deering, "*Host extensions for IP multicasting*", RFC 1112, August 1989.

[14] V. Roca, B. Mordelet, "*Improving the efficiency of a multicast file transfer tool based on ALC*", INRIA Research Report number 4411, March 2002.

[15] S.E. Deering, B. Cain, A. Thyagarajan, "*Internet group management protocol, version3*", Internet Draft draft-ietf-idmr-igmp-v3-00.txt, work in progress, December 1997.

[16] T. Maufer, C.Semeria, 3Com Corporation, " *Introduction to IP Multicast Routing*", draft- ietf-mboned-intro-multicast-01.txt, March 1997.

[17] C. K. Miller, "*Multicast Networking and Applications*", (c) 1999 Addison Wesley Longman, Reading MA ISBN 0-201-30979-3.

[18] Vicki Johnson, Marjory Johnson "*How IP Multicast Works-an IP Multicast Initiative White Paper*", Web: www.ipmulticsta.com, visited in Nov. 2001.

[19] Andrew Adams, Jonathan Nicholas, William Siadak, "*Protocol Independent Multicast–Dense Mode (PIM-DM): Protocol Specification*", draft-ietf-pim-dm-new-v2-01.txt, February 15, 2002.

[20] Bill Fenner, Mark Handley, Hugh Holbrook, Isidor Kouvelas "*Protocol Independent Multicast–Sparse Mode (PIM-SM): Protocol Specification*", draft-ietf-pim-sm-v2-new-04.txt, November 21, 2001.

[21] K. Obraczka, "*Multicast transport protocols: A survey and taxonomy*" IEEE Commun. Mag., vol. 36, pp.99-102, Jan.1998.

[22] S. Floyd et al, " *A Reliable Multicast Framework for Light-weight Sessions and Application level Framing*". A later version of ACM SIGCOMM paper (ftp://ee.lbl.gov/papers.srm1.tech.ps.Z), November 1995.

[23] Hector Garcia-Molina, and Annemarie Spauster, "*Ordered and Reliable Multicast Communication*", ACM Transactions on Computer Systems, Vol. 9, No. 3, August 1991, pp. 242 - 272.

[24] "*Reliable Multicast Transport (RMT) charter*", http://www.ietf.org/html.charters/rmt-charter.html .

[25] Stefan Elf, Peter Parnes, "*A Literature Review of Recent developments in Reliable Multicast Error Handling*", ISSN: 1402-1528, Jan. 2001

[26] A. Mankin, A. Romanow, S. Bradner, V. Paxson., "*IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols*", RFC2357, June 1998.

[27] L.Rizzo, "*PGMCC: A TCP-friendly single-rate multicast congestion control scheme*", Proceedings of SIGCOMM 2000, Stockholm Sweden, August 2000.

[28] B. Adamson, C. Bormann, M. Handley, J. Macker, "*NACK-oriented reliable multicast protocol (NORM)*", RMT Working Group, draft-ietf-rmt-pi-norm-04.txt, March 2002.

[29] B. Adamson, C. Bormann, M. Handley, J. Macker, "*NACK-oriented reliable multicast (NORM) protocol building blocks*", RMT Working Group, draft-ietf-rmt-bb-norm-02.txt, July 2001.

[30] B. Whetten, D.M. Chiu, M. Kadansky, G. Taskale, "*Reliable multicast transport*

*building block for TRACK*", RMT Working Group, draft- ietf-rmt-bb-track-01.txt, March 2001.

[31] M. Luby, J. Gemmell, L. Vicisano, L. Rizzo, M. Handley, J. Crowcroft, "*Forward Error Correction building block*", Internet Draft draft-ietf-rmt-bb-fec-06.txt, January 2002.

[32] J. Nonnenmacher, E.W. Biersack, "*Reliable multicast: Where to use FEC* ", IFIP 5th International Workshop on Protocols for High Speed Networks (PfHSN'96), October 1996.

[33] Jaehee Yoon, Azer Bestavoros, Ibrahim Matta, "*Adaptive Reliable Multicast*", Proceedings of IEEE International Conference on Communications,vol.3, pp.1542-6, 2000.