MASTER'S THESIS

Reliable Multicast via Satellite

by Stephen M. Payne
Advisor: John S. Baras

CSHCN M.S. 99-4
(ISR M.S. 99-7)

Sponsored: Advanced Telecommunications &Information Distribution Research Program

ABSTRACT

Title of Thesis:	RELIABLE MULTICAST VIA SATELLITE
Degree candidate:	Stephen Marshall Payne
Degree and year:	Master of Science, 1999
Thesis directed by:	Professor John S. Baras
	Department of Electrical and Computer Engineering

Many different reliable multicast protocols have been proposed and analyzed in the current literature. With advances in satellite technology, satellites have become more used within commercial networks. Since satellites are naturally a broadcast medium, multicast communications have the potential to greatly benefit from their wide-scale deployment. The performance of reliable multicast protocols needs to be studied and well understood over networks including satellite links. Most of the analysis performed on these protocols have dealt with bandwidth usage, buffer requirements, and processing delay. Very few studies address the transmission delay incurred from using reliable multicast protocols. As delay becomes a larger issue in reliable multicast applications, performance evaluation with respect to this metric becomes important.

An existing hybrid error control protocol that combines packet level parity retransmissions with ARQ type feedback was studied under a variety of different conditions. Additionally, several modifications were made to the protocol and their performance in terms of bandwidth and delay were studied. The protocols were studied both with and without local recovery schemes. For non-local recovery schemes, the use of autoparity was examined and was shown to decrease recovery latency at the cost of additional bandwidth usage. The effects of different estimation schemes coupled with autoparity usage were investigated and results were compared. Simplistic adaptive mechanisms where the parity provided during each transmission is adjusted based upon observed packet loss statistics used with a local recovery scheme were found to offer the best overall results in terms of reducing recovery latency and satellite bandwidth usage.

RELIABLE MULTICAST VIA SATELLITE

by

Stephen Marshall Payne

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
1999

Advisory Committee:

Professor John S. Baras, Chair
Assistant Research Scientist M. Scott Corson
Associate Professor Leandros Tassiulas

DEDICATION

To my parents, for their continual love and support.

ACKNOWLEDGEMENTS

I would like to express my gratitude to Professor John S. Baras for his advice and support throughout my graduate studies at the University of Maryland. Additionally, I would like to thank both Dr. Corson and Dr. Tassiulas for kindly consenting to serve on the advisory committee and review this thesis.

I would like to thank Allison for her patience, support and most importantly love.

I would like to thank all of my peers within the department. The support, comradeship, and insightful feedback played an important role in my completion of this thesis. In no particular order, I would like to especially acknowledge Brad Barrett, John Mitchell, Manish Karir, Mingyan Liu, Vigay Bharadwaj, and Ravi Vaidyanathan. They played a significant role in shaping this thesis and "kept the laughter rolling" during the late evenings spent in various computer laboratories.

Finally, I would also like to express my appreciation for the support received from several sponsors throughout my studies at the University of Maryland. This work was supported in part by the ATIRP Fellowship in the Advanced Telecommunications /Information Distribution Research Program (ATIRP) Consortium sponsored by the U.S. Army Research Laboratory under the Federated Laboratory Program, Cooperative Agreement DAAL01-96-2-0002. I also benefited from an exciting summer CSHCN internship at the Army Research Laboratory.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. Introduction

Multicast networking and its corresponding applications are becoming increasingly popular solutions for new suites of Internet products and services. In multicast networking, data is transmitted from either one sender to many recipients or many senders to many recipients. Multicast applications offer many appealing benefits when trying to disseminate information to a large group of users. Perhaps, the most important benefit is the increased multicast bandwidth efficiency over unicast and broadcast counterparts. There are two broad classifications of multicast applications; those that require reliable multicast and those that do not require this capability. Protocols enabling both classes of multicast applications need to be studied in a variety of different network topologies.

Since satellites are naturally a broadcast medium, multicast communications have the potential to greatly benefit from their wide-scale deployment. Such wide-scale deployment of satellites requires that multicast protocols be evaluated over networks containing these links. Besides different error characteristics than their terrestrial counterparts, satellite links suffer from different delay (normally higher) characteristics. Reliable multicast protocols' performance needs to be studied and well understood over networks including satellite links. However, most existing analyses (e.g. [8], [17], [24], [25], [26]) of reliable multicast protocols have dealt with bandwidth usage, buffer requirements, and processing delay. Fewer studies (e.g. [17], [27]) address the propagation delay incurred from using reliable multicast protocols. As low latency reliable multicast applications emerge, delay becomes an important performance metric.

This thesis focuses on the latency incurred when reliable multicast protocols are used over satellite links.

When considering the propagation delay implications of reliable multicast protocols over satellite links, error recovery is a crucial issue. In an Automatic Repeat Request (ARQ) protocol, a feedback channel is required for proper protocol function. However, due to the high latency over satellite links, such protocols experience significant throughput degradation (e.g. the TCP degradation observed over high latency links [1]). For this reason, Forward Error Correction (FEC) schemes appear to be the more promising of the two approaches. Unfortunately, pure FEC does not require a feedback channel and therefore does not guarantee reliability. A hybrid approach that uses parity packets to reduce the residual packet error probability and a feedback mechanism to ensure reliability combines the advantages of both FEC and ARQ schemes to form a more robust protocol. There are variations of hybrid error control (HEC) protocols that offer additional potential benefits. One important variation uses local recovery to limit the amount feedback to and the number of transmissions from the source.

Prior to studying generic HEC protocols, a network model must be developed. The set of receivers is subdivided into R connectivity clusters. Each connectivity cluster has a privileged receiver (PR) that has satellite communication capabilities. There are two topological scenarios in which; 1) the connectivity clusters are not connected via medium other than the satellite, and 2) the connectivity clusters are connected via medium other than the satellite. In scenario 1, a source sends multicast data over a satellite to R connectivity clusters. As these connectivity clusters are isolated from each another, only global recovery schemes in which retransmissions occur only from the source are feasible. In scenario 2, the connectivity clusters are terrestrially connected. Since these connectivity clusters overlap, local recovery schemes in which retransmissions potentially occur from non-source nodes can be considered as viable solutions. In both scenarios, communications between the sender and the receivers must initially traverse a satellite link and no terrestrial link exists between the source and the multicast group.

With the definition of the network scenarios, the generic protocols are studied in a variety of ways. The protocols are studied in the unconnected scenario and then later studied in the connected scenario with local recovery. In the unconnected scenario, the use of autoparity (i.e. – unsolicited parity packets transmitted by the source on the initial transmission round of each group of data) is examined and its effects on the recovery latency and bandwidth usage are presented. Secondly, the effects of the maximal packet loss probability, channel estimation scheme [6] coupled with autoparity usage is investigated. The examination of the two aforementioned items are first studied with the assumption that the source can generate an infinite amount of parity packets. Both mathematical analysis and simulations are used to study these schemes under the infinite parity assumption. Then this assumption is removed and simulations are used to examine the effects of different amounts of finite parity. Before continuing to study the connected scenario with local recovery schemes, simplistic adaptive mechanisms where the parity provided during each transmission is adjusted based upon observed packet loss statistics.

In the connected scenario, the addition of local recovery mechanisms is studied with regard to their impact on the delay caused by the satellite link and satellite bandwidth usage. Local network usage is also taken into account as the associated costs of using such local recovery schemes. Some of the studies performed for the unconnected scenario apply to the connected scenario and are studied in conjunction with the local recovery mechanism. The first local recovery mechanism to be examined alternated between one global and one local transmission round. Subsequent studies look into potential local recovery adaptive mechanism as well as using different local-global transmission round ratios (such as 2 local for every global round).

The remainder of the thesis is organized into five sections. The next section contains an overview of multicast communications and the error control mechanisms and architectures used in reliable multicast protocols. In section 3, a detailed problem statement is developed. The two scenarios are then studied and their results are presented in sections 4 and 5. Section 6 concludes the thesis by summarizing results and presenting areas for future work.

2. Background

Network communications can be broadly classified based upon the number of senders, the number of receivers, and the association between these two types of network entities. Point-to-point communications occur between one sender and one receiver. Classical unicast services belong to the point-to-point communication class. When communications occur between one sender and a set of receivers, it is referred to as point-to-multipoint. Point-to-all communications occur between a single receiver and all associated partners. An example of point-to-all communication is a broadcast service over a local area network. Communication between a group of network entities able to both send and receive data is known as a multipoint-to-multipoint system. [4]

Whereas unicast services enable point-to-point communications, multicast services enable both point-to-multipoint and multipoint-to-multipoint communications. Some multicast services do not require reliable communication such as distributed gaming, distance learning applications, or distributed interactive simulations[15]. Other applications require reliable multicast service. These applications can be categorized based upon their data content (i.e. data-only or multimedia) and their delay constraints (real-time or non-real-time). For example, video-replication is a data-only, non-real-time application. Financial stock quote dissemination is considered a data-only, real-time application. One data-only, non-real-time application is database replication [15].

The number of applications that can benefit from reliable multicast transport services is growing as the number of Internet applications increases. The military also has the need for a reliable multicast transport service that can deliver important information to many users/units deployed upon a battlefield. This information could include terrain maps, intelligence information, and orders of battle.

2.1 Multicast versus Unicast

If a unicast service were used to provide point-to-multipoint communications, then the source must generate a copy of a particular packet to be sent to each individual receiver. Thus the source is required to copy and address each duplicate packet with a different address. Rather than generate a duplicate packet for each receiver, multicast services first establish a cycle-free routing tree connecting the sender with the set of receivers. When sending a packet, the source addresses the packet to the multicast group connected by the multicast tree. Every node in the tree forwards an arriving packet to its children. Therefore, the source only sends a packet to its direct children. Upon receipt of a packet, each node forwards a copy to each of its children. Two major advantages of multicast transmission are simplicity of addressing and more efficient use of bandwidth.

As multicast services enable point-to-multipoint and multipoint-to-multipoint communications, they have a different set of requirements than unicast services. These requirements can be subdivided into five different areas; addressing, group membership, routing, network heterogeneity, and error control mechanisms. Multicasting requires an aggregated address per group enabling many anonymous members to participate in a session; whereas unicasting requires that the sender knows only the receiver's address. In multicast communications, there is a concept of group membership. Group membership can remain static or may dynamically change over a multicast session. The complexity of routing in multicast environments is generally higher than in their unicast counterparts. As group membership dynamically changes, the complexity of routing further increases. The routing protocol’s efficiency impacts multicast service performance. Multicast groups often contain receivers that experience different network characteristics. Network heterogeneity impacts the quality of service guarantees, the negotiation of any session parameters, and the flow and congestion control. Both multicast and unicast communications need error control mechanisms to recovery from erroneous data. However, these mechanisms are especially important when the multicast service requires reliable delivery of data.

There are five basic issues involved when considering reliable multicast protocols [23]. The first issue, the request implosion problem, occurs when the loss of a packet results in many simultaneous repair requests. Such a phenomenon can overwhelm the sender and possibly other receivers depending upon how the feedback is transmitted to the group. The second issue, duplicate replies, occurs when multiple repairs are sent from group members responding to a request (this happens primarily in local recovery situations). The third issue, recovery latency is the time required for a repair to be received after a loss is detected. Recovery isolation is very important in correlated-loss situations where a loss on an upstream link causes a group of receivers to lose the same packet. The fourth issue, recovery isolation, is the desire to isolate non-local nodes from receiving local repairs. Finally, the issue of adaptability to dynamic membership changes addresses the time required to return to a steady state after group membership or topological changes.

There has been a great deal of research focused on the many aspects of multicast and reliable multicast protocols. More recently, there has been an explosion of proposed protocols and schemes to provide reliable multicast services. This thesis primarily focuses on error control mechanisms and architectures and their impact upon the recovery latency in networks containing satellite links. Prior to studying this impact, a detailed overview of the main thrusts of research on error control mechanisms and architectures is presented in the following sections.

2.2 Error Control Mechanisms and Architectures

This section deals with the mechanisms and architectures used by reliable multicast protocols to recover from packet errors or packet losses. Error control architectures can be classified based upon group member participation in error recovery. In centralized-error recovery (CER) architectures all retransmissions are sent from the original source; whereas in distributed error recovery (DER) architectures other intermediate nodes are allowed to participate in the retransmission of packets [17]. Within each of these architectures, there are two mechanisms that can be used to provide error recovery; Automatic Repeat Request (ARQ) and Forward Error Correction (FEC). Additionally, these two mechanisms can be combined to form hybrid schemes that provide more robust reliability. This section discusses error control mechanisms and explores these mechanisms within different error control architectures.

2.2.1 Error Control Mechanisms

2.2.1.1 Automatic Repeat Request

Automatic Repeat Request is a mechanism that recovers lost packets using retransmissions initiated by feedback. There are two basic forms of ARQ protocols; sender-initiated and receiver-initiated [26]. In sender-initiated protocols, the sender must ensure the reliable delivery of data to the entire multicast group via a multicast tree.

Upon sending a particular packet, the sender starts a timer. Each receiver that correctly receives the packet sends a positive acknowledgement (ACK) to the sender. For each transmitted packet, the sender maintains a list of the receivers from which it has received an ACK. When the timer expires, the sender determines if all of the receivers in the group have acknowledged the packet. If at least one receiver is missing the packet, the packet is retransmitted and the timer restarted.

There are several disadvantages that make sender-initiated protocols undesirable for multicast environments. First, the sender must maintain packet state information for each receiver resulting in increased processing requirements for large multicast groups. Second, updating this state information for each receiver requires a significant amount of feedback and requires greater processing requirements at the source. This feedback results in additional network congestion on links surrounding the sender. Therefore, sender-initiated protocols are limited in that they do not scale well to support large multicast groups. [26]

Whereas sender-initiated protocols place the responsibility of reliable delivery upon the sender, receiver-initiated protocols place most of this responsibility upon the receivers [26]. In receiver-initiated protocols, the sender continually transmits new data packets over a multicast tree until a negative acknowledgement (NAK) is received. A NAK is generated when a receiver detects a lost packet. Packet loss detection is performed by observing “gaps” in the received packets sequence numbers. If a “gap” is detected, then a NAK is generated. When sending a NAK, the receiver will start a NAK retransmission timer. If this timer expires before the correct reception of the requested packet, then the receiver resends a NAK to the sender.

Generating NAKs in such a manner results in the request (or feedback) implosion problem. This phenomenon occurs when a significantly large number of negative acknowledgements are transmitted to the sender. These feedback messages must traverse links immediately surrounding the sender and thus result in network overload and congestion. Therefore, a NAK suppression mechanism is desirable so that the number of NAKs actually reaching the sender is significantly decreased (ideally down to 1). Most suppression mechanisms use timers in conjugation with multicast NAKs. In such a mechanism, any receiver detecting a loss waits for a random amount of time prior to sending a NAK. Although, the characteristics of the timer are not discussed, they are important to the performance of the suppression mechanism [19]. If no other NAK for the same packet is received within the time interval, then the receiver proceeds and multicasts the NAK to the entire multicast tree (includes the sender and all other receivers). If a receiver receives a NAK prior to its timer expiration, then it suppresses its NAK. This particular receiver sets its NAK retransmission timer so that the feedback cycle resets if the request is not fulfilled. Such a mechanism enables the receivers to coordinate NAK generation and thus reduce the number of feedback messages received at the source. Some form of NAK suppression mechanism is assumed in subsequent discussions.

2.2.1.2 Forward Error Correction

Whereas ARQ is a reactive mechanism activated by packet losses during transmission, FEC is a proactive mechanism that transmits redundant data so receivers can reconstruct the original message even in the presence of communication errors. This statement offers some intuitive feel of the tradeoffs present when considering ARQ or

FEC as error control mechanisms. Since ARQ schemes only retransmit data when errors occur, they waste little bandwidth at the cost of the delay incurred by waiting for feedback. FEC mechanisms transmit redundant data in anticipation of errors thereby reducing feedback delay. However, this redundancy may waste bandwidth over relatively error-free links.

The process of generating redundant data (or parity) requires the processing of an entire data stream. Such an encoding process is computationally expensive. The high computational cost of FEC implementations is not a concern in bit level communication systems where the encoder/decoder is usually implemented in dedicated hardware. At this level hardware implementation is usually much cheaper than having a feedback channel. In computer communications, the feedback channel often requires little overhead and FEC requires a noticeable processing overhead for the host systems. However, as shown by Rizzo in [24], packet level FEC can be effectively implemented with software.

In coding theory there are two types of errors; corruption and erasure. Corruption of data occurs when bits are corrupted, whereas the erasure of data occurs when whole packets are lost [17]. In multicast protocols, the use of FEC mechanisms is mainly restricted to the recovery from erasures thereby reducing the effect of packet loss at different receivers. In other words, as long as a receiver collects a sufficient number of different packets, reconstruction of the original data is possible independent of the received packets' identity. For example, k data packets can be encoded to produce n packets (where $n > k$) so that if any k-subset of the n packets are correctly received, then the data packets can be reconstructed. By producing $h = n – k$ parity packets and sending these packets with the *k* data packets, the residual error rate is reduced. This property of FEC mechanisms improves reliable multicast protocols scalability irrespective of the actual loss pattern at each receiver [17]. Additionally, the reduction in the residual loss rate (after decoding) largely reduces the need to send feedback to the sender, thus minimizing the use of the channel and simplifying feedback handling.

There are a variety of different ways to perform FEC encoding on a packet stream. The following description of the RSE schemes mirrors the one presented in [17]. Other coding descriptions are developed in [14] and [24]. In most cases, a Reed-Solomon erasure (RSE) code is used to generate the redundant packets. As previously stated, *k* represents the number of data packets of length P bits. These data packets are represented as $d_1, d_2, \ldots d_k$. The RSE encoder takes $d_1, d_2, \ldots d_k$ and produces parities $p_1, p_2, \ldots p_{n-k}$. For the purpose of coding, consider the vector $\overline{d} = [d_1, d_2, \ldots d_k]$ of data packets as elements of the Galois field $GF(2^P)$. Given the primitive element α of $GF(2^P)$, the (n,k) matrix G′ = [g $_{\mathrm{i,j}}$] with elements in $GF(2^P)$ is defined as

$$g_{i,j} = \alpha^{i \cdot j} \qquad 0 \le i \le 2^P - 2, \quad 0 \le j \le k-1.$$

The basic RSE encoder can produce up to $n = 2^P - 1$ FEC packets as components of

$$\overline{y}' = [y_1', \ldots, y_k'] = G' \overline{d}^T$$

The matrix G' has the property that any *k* out of the n row vectors are linearly independent. Therefore, at the RSE decoder, any k components of $\overline{y}'$ are sufficient to uniquely specify $d_1, d_2, \ldots d_k$.

Since the basic RSE scheme is not a systematic code, the data packets $d_1, d_2, \ldots d_k$ are not part of $\overline{y}'$. Consequently, the RSE decoder must always solve k simultaneous linear equations to retrieve the data packets from *k* components of $\bar{y}'$. This decoding complexity can be avoided by using Gaussian elimination on the matrix G' prior to encoding. This operation modifies the first k row vectors of G' into a (k,k) identity matrix and creates a new matrix *G*. Using this *G* in the encoding $\bar{y} = G\bar{d}^T$, the first *k* components of $\bar{y}$ are copies of $d_1, d_2, \ldots d_k$. The remaining *n-k* components of $\bar{y}$ are the parities $p_1, p_2, \ldots p_{n-k}$. These parity packets are the last *n-k* packets in $\bar{y}$ (e.g. $p_i = y_{k+i}$ for $i \in \{1, \ldots, n-k\}$).

An example presented in [12] demonstrates how a Reed-Solomon erasure code is used in actual reliable multicast protocols. In this example, a shortened RS code was designed around 8 bit symbols and used an RS(255,k) code as its basis. First, *x* data symbols at the encoder and decoder are zero-filled so that a shortened RS codeword can be created. This zero-filling results in the transmission of *k-x* codewords over the link. By choosing *k*=235 and *x*=215, a shortened RS(40,20) code is formed that enables the generation of 20 parity packets per 20 data packets. These parity packets are transmitted as determined by the reliable multicast protocol. The following section discusses the different ways in which the parity packets are transmitted

2.2.1.3 Hybrid Error Control (HEC)

Even though FEC reduces the residual error rate, it cannot provide full reliability since there is no mechanism by which data can be retransmitted. Therefore, FEC needs to be augmented with an ARQ scheme so information can be retransmitted if network conditions deteriorate beyond a certain threshold. This threshold is normally determined as the maximum allowable number of packet errors occurring over the communication links or the total number of transmitted parity. ARQ mechanisms can reliably deliver data but require the retransmission of individually lost packets. FEC strengthens ARQ mechanisms because a single parity packet can correct different packet errors at different receivers. Therefore, the combination of these two schemes results in a more robust mechanism that can guarantee reliability.

These two schemes can be combined in two ways; layered and integrated. In the layered approach [17], the reliable multicast (RM) layer sits atop an FEC layer in the protocol stack. The RM layer generates k data packets (called a transmission group, TG) which are sent to the FEC layer. Then the FEC layer generates h parity packets using the k data packets. Both the data and parity packets (a total of $n = k + h$ packets – called an FEC block) are multicast to the receivers in the group. If fewer than k of these n packets are received, then the data cannot be reconstructed at the receiver. The receiver discards the received parity packets and requests the lost originals from the sender. These lost originals are transmitted as part of a new FEC block. For many situations [17], this scheme makes more efficient use of network resources than ARQ schemes since the numbers of repair requests and numbers of retransmissions are reduced.

The integrated approach fuses the RM and FEC layers into one protocol stack layer. Through this integration, parity is used more efficiently. For example, a parity packets (known as autoparity packets) may be sent in the initial transmission of k data packets. If the receivers lose more that a of the $k + a$ packets, then they must request new parity so that the k data packets can be reconstructed. Upon receiving parity requests, the sender multicasts parity packets until all parity packets have been used. When the parity has been used, packets requiring retransmission are placed in the next

TG. A performance comparison of the pure ARQ, layered FEC, and integrated FEC schemes was based upon the average number of transmissions needed to reliably send a packet to all receivers in a multicast group [17]. This comparison showed that both integrated and layered FEC outperform the ARQ scheme for a large number of receivers (over 100). For smaller numbers of receivers, the ARQ scheme outperformed the layered FEC scheme because there were fewer than *h* errors and thus bandwidth was unnecessarily wasted. When *a* equals zero, the integrated FEC scheme performed better than both other schemes over all group sizes.

By integrating FEC with ARQ mechanisms, the number of transmissions and therefore the bandwidth usage is reduced. This reduction occurs because the resource usage is shifted from the network to the source and the receivers in the form of parity encoding and/or decoding. Through the encoding and decoding of parity packets, the error-control feedback is reduced. Due to lower amounts of feedback and more efficient bandwidth usage, integrated FEC protocols have good scalability properties (i.e. – the protocol performs acceptably for up to 1 million receivers) [17].

There are other ways to implement hybrid FEC-ARQ schemes. Generally, most differences are with the methods used to deal with the depletion of fresh parity. The above scheme placed any packets requiring retransmission in the next FEC block when parity was depleted. However, for large number of receivers (much larger than the size of the TG), there is a high probability that all packets within the TG will need to be retransmitted. There are other schemes that propose the use of an explicit ARQ repair phase when the parity is depleted [6]. Regardless how depleted parity is handled, hybrid

FEC-ARQ schemes have been shown to yield better bandwidth performance than strictly ARQ schemes.

2.2.2 Error Control Architectures

In the preceding discussion, it was implicitly assumed that only the original source is allowed to generate retransmissions. Therefore, the above mechanisms fit within the CER (or global recovery) architecture. In DER (or local recovery) architectures, the responsibility of error recovery is distributed amongst members of the multicast group by allowing non-source nodes to aid in recovery. Distributed error recovery can further be divided into two sub-classifications; ungrouped DER and grouped DER. In ungrouped DER, any member of the global multicast group has the ability to perform retransmissions. Therefore, any particular node can potentially send retransmissions to the entire multicast group. Like ungrouped DER, grouped DER allows non-source nodes to retransmit data. However, in grouped DER, these retransmissions are performed within the local group (or local neighborhood). Intuitively, DER architectures reduce the amount of network-wide bandwidth consumed and contain errors to the locality in which they occurred. Since lost packets are recovered by local retransmissions, these architectures have the potential to provide significant performance gains in terms of end-to-end delay and higher system throughput. In the remainder of this section, several DER architectural examples are discussed.

2.2.2.1 Local Recovery Approaches

In this section, server-based local recovery and receiver-based local recovery are described. In the receiver-based approach, a receiver attempts to locally recover packets from the end hosts within its local neighborhood. In other words, any receiver that has correctly received a requested packet has the ability to send retransmissions during local recovery. An example of this type of approach is Scalable Reliable Multicast (SRM) with local recovery enhancements [3]. The SRM protocol proposes a general framework and requires additional specifications to be specialized for a particular application. This protocol allows any receiver that has the correct data to generate repairs. Such a concept increases scalability by reducing administrative feedback to the source. To suppress duplicate repair requests, receivers requiring data wait a random period of time prior to issuing their requests. Repairs are made following a similar process in which a random timer is set. For both the requests and the repairs, the timer is a function of the closeness between the receiver in need of the packet and the receiver transmitting the repair. Although the use of local repairs relieves the NAK implosion problem at the sender, there are no limitations upon traffic flows within the group. The absence of limitations could potentially lead to the inefficient use of network resources in some localities [15].

The server-based approach makes use of specially designated hosts called repair servers to perform local recovery. Only these repair servers can send retransmissions during the local recovery phase. An example of the server-based approach is the Reliable Multicast Transport Protocol (RMTP) [22]. This protocol increases scalability by creating a hierarchy that enables Designated Receivers to collect feedback messages and to provide any available repairs to nodes within its local domain. Using the global multicast tree, the sender sends every packet over the multicast tree to the entire multicast group. Rather than sending its status information directly to the source, receivers send this information to their corresponding DR over a local multicast tree. A DR does not consolidate the feedback, but rather sends its own status information to the source. [22]

The generic reliable multicast version of the receiver-based protocol presented in [10] performs ARQ error recovery in a DER grouped architecture. The source sends all transmissions over a multicast tree. When a receiver detects a loss, it performs the local retransmission phase. During this local retransmission phase, the receiver waits a random amount of time prior to sending its own NAK. In this time interval, if no other receiver's NAKs are received, then a NAK is multicast to its local neighborhood peers. When sending a NAK, the receiver sets a local retransmission timer. This timer determines the interval that the receiver waits for the request to be locally fulfilled. Upon receiving a local packet retransmission request that can be fulfilled, the receiver multicasts the packet to the entire neighborhood. However, prior to sending this packet, the receiver waits for a random amount of time and suppresses its own transmission if another "requested" packet is received. If no members in the local neighborhood respond prior to the expiration of the local retransmission timer, the receiver begins the global recovery phase. In this phase, the receiver transmits a global NAK over the entire multicast tree using a NAK suppression mechanism. If the requested packet is not received, then the receiver enters another local recovery phase. This process of alternating between local and global retransmissions repeats until the receiver has correctly received all desired packets. When the source receives a global NAK, it remulticasts the packet to the entire group. This process is followed for all packets multicast from the source to the multicast group.

The generic version of a sender-based protocol presented in [10] performs ARQ error recovery in a DER grouped architecture. This protocol uses repair servers to process the retransmission requests of receivers in the local neighborhood. Initially, the source sends a packet to all receivers and repair servers via a multicast tree. When a receiver detects a loss, it employs a NAK suppression mechanism to reduce the amount of feedback. However, rather than multicasting a NAK to the entire group, the receiver only multicasts a NAK to its repair server and to the receivers within its local neighborhood. A local neighborhood is defined as the set of receivers on the same subtree rooted at the nearest backbone router (see Figure 2.1). When a repair server receives a NAK for a particular packet from a member of its group, it determines the availability of the packet. If the repair server has the packet, then it is multicast to the local neighborhood. Otherwise, the repair server must retrieve the packet from the source or its upstream repair server. This retrieval is accomplished using the same NAK suppression mechanism that was used by the receivers to retrieve the packets from the repair server. When the source receives a NAK, it remulticasts the packet to all receivers and repair servers. This process repeats for each packet multicast from the source to the receivers.

Per the analysis completed in [10], the server-based approach yielded higher protocol throughput and lower bandwidth usage than the receiver-based one. However, these results required that the repair servers have processing power slightly higher than that of a receiver and several hundred kilobytes of buffer space available per multicast session.

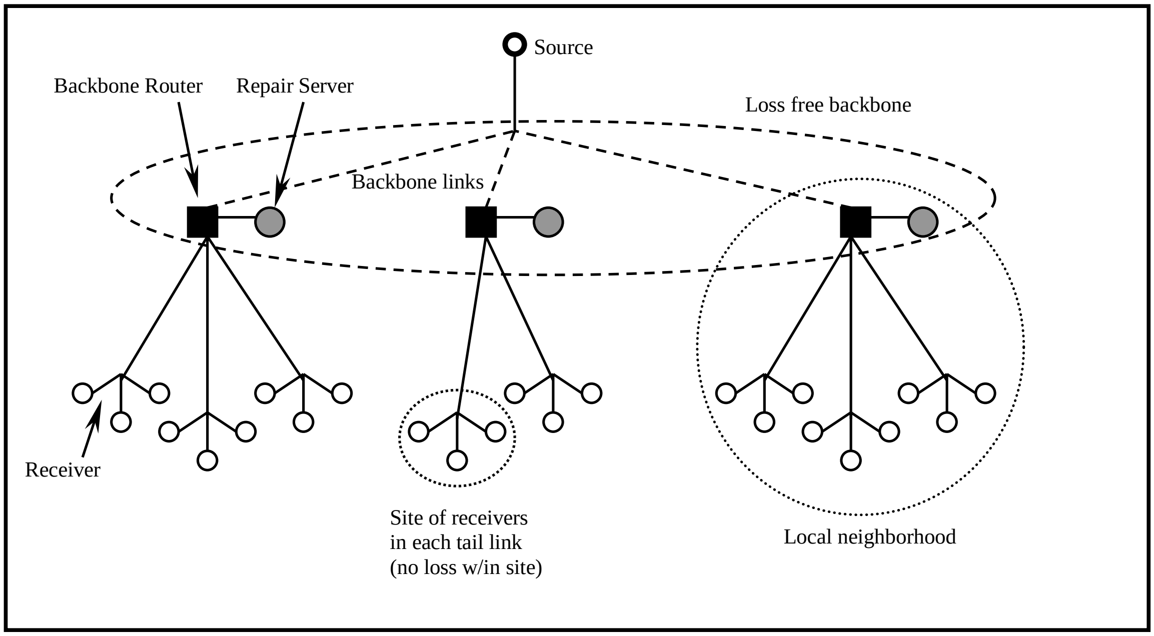

Figure 2.1: Local Recovery Network Model [5]

2.2.2.2 Local Recovery and Hybrid Error Control Approaches

Now that ARQ mechanisms have been studied with both CER and DER architectures, a discussion of hybrid error control within DER architectures is needed. As was previously discussed, the retransmission of parity has excellent scaling properties as a single parity packet can repair different losses at different receivers. In CER architectures, it was shown that using integrated hybrid error control the number of retransmissions was significantly reduced [17]. Nonnenmacher et al. [18] analyzed integrated hybrid error control within the DER architecture and compared performance with the hybrid error control within the CER architecture.

In the analysis of [17], one CER-based and two DER-based protocols were analyzed. The CER protocol, called C, is based upon the integrated hybrid error control model shown in Section 2.2.1.3. The two DER-based protocols conform to the DER grouped architecture. One of the DER group based protocols, D1, uses an ARQ mechanism. In this protocol, the source is considered the repair server for all internal DER nodes. These internal DER nodes are group leaders for all receivers that are their children. The first transmission is multicast to all receivers, but subsequent retransmissions are handled locally. This protocol basically works in a store-and-forward manner requiring that all data be received by all internal DER nodes on the first tree level. Then, the data will be forwarded in parallel from all DER nodes to their corresponding receivers. The other DER grouped based protocol, D2, is a variation of the D1 protocol. Rather than using ARQ as an error recovery mechanism, D2 uses integrated HEC. Protocol D2 transmits packets in the same manner used in D1. However, error recovery at both levels uses parity retransmissions rather than an ARQ mechanism.

A bandwidth analysis of these three protocols shows that D2 outperforms D1. This increased performance can be directly attributed to the use of parity packets to repair losses. If the transmission group size (i.e. – number of data packets), *k*, is large enough, then C also outperforms D1 due to the efficiency use of parity packets. For example, assume that the TG size is relatively large. Each of R receivers requires a different packet to correctly receive a TG. In protocol C only one packet would have to be transmitted by the source, whereas in D1 R packets need to be transmitted from internal DERs. Grouped DER protocols have better scalability than CER protocols due to their hierarchical structure which limits the scope of retransmissions. When increasing the block size (e.g. increasing *k*), the performance gain of the D2 protocol over the CER protocol decreases. Additionally, there are some weaknesses in using such a hierarchy for error recovery. In flat network architectures (such as with many satellite networks) there is no place for internal DER nodes because receivers tend to be directly connected to the satellite. Fortunately, the networks to be considered in this thesis are hybrid networks in which terrestrial networks extend from satellite access points and offer the potential to use server-based local recovery mechanisms.

Bandwidth analyses have shown that local recovery coupled with hybrid error control (protocol D2) performs more efficiently than both local recovery without hybrid error (protocol D1) control and HEC without local recovery (protocol C). Since HEC appears to be useful in reducing bandwidth usage, it follows that it is interesting to analyze situations in which differing levels of FEC encoding/decoding capabilities are placed at repair servers within the multicast tree. Protocols using this type of capability are known as active parity encoding services (APES) [25]. APES protocols send FEC-based repairs rather than retransmissions. APES protocols can be classified into three generic types:

1) The Store-Data-Build-Repairs (SDBR) Protocol: Once a repair server reliably obtains k source packets, it reproduces the k original data packets that are subsequently buffered. Whenever an additional repair is required, the repair server generates a new distinct repair via FEC encoding. Since these repairs are distinct, any receiver that needs additional repairs can use any combination of *k* original data packets and repairs obtained from the repair server.
2) The Build-Repairs-Store-Repairs (BRSR) Protocol: A repair server decides in advance upon a fixed number of repairs, *b,* per block to generate via FEC encoding. Once these b parity packets are buffered at the repair server, the remaining data packets are flushed from the cache. If the receivers within a given

repair server's repair domain need parity packets, then these packets are reliably transmitted to each of the receivers. By requiring reliable transmission of parity packets (via some mechanism such as ARQ), the repair servers do not have to retain the data so that new parity can be created.

3) The Get-Repairs-Store-Repairs (GRSR) Protocol: Rather than generating the repairs, this protocol requires that the repair servers request b parity packets from the sender. Once these b parity packets are received and buffered, this protocol behaves exactly like BRSR.

When analyzing the APES protocols, the general network model shown in Figure 2.1 was essentially used in [25]. Each of the repair servers was responsible for their respective repair domains (or local neighborhoods). In the analysis, the assumption that there was no loss between the source and the repair servers was made. The following two assumptions were also made: 1) a TG of size *k* data packets and 2) receivers lose any packet sent to it with probability *p*. Each repair domain was analyzed separately.

The bandwidth analysis was sub-divided into two parts, the bandwidth used between the source and repair servers, and the bandwidth used between the repair servers and the receivers. With the above assumption of error free transmission between source and repair servers, the difference in the bandwidth over these error-free links is due solely to the design of the protocols. In BRSR and GRSR, a repair server must retrieve needed repairs from the source if the number of errors experienced within its repair domain exceeds *b*. For sufficiently large values of *b*, this required bandwidth is negligible. BRSR and GRSR do not use substantially more bandwidth between the repair server and receivers than SDBR for reasonable packet loss rates (p=0.01, p=0.05). Since SDBR uses the minimal number of distinct repairs to provide reliability, it provides a lower bound on the expected bandwidth for BRSR and GRSR. However, for domain sizes and loss rates that one might expect in reality, the difference in bandwidth is negligible. The bandwidth used throughout the network in a CER hybrid error control approach is dominated by the bandwidth required by domains with high loss. Conversely, in networks with repair servers, this bandwidth consumption can be limited to the domains where it is required rather than penalizing the performance of the entire network.

3. Reliable Multicast via Satellite

3.1 Introduction

Many different reliable multicast protocols have been proposed and analyzed in the current literature. Examples of such protocols range from the previously mentioned SRM and RMTP to Reliable Adaptive Multicast Protocol (RAMP) and Multicast File Transfer Protocol (MFTP). Although the flexibility of SRM is a major advantage, it suffers from several flaws including its incompatibility with asymmetric network infrastructures and the negation of its scaling properties in satellite environments [14]. RMTP looks to increase scalability by allowing non-source nodes to participate in error recovery. Another protocol, RAMP was intended for use in military collaborative applications such as simulated war games. It attempts to reliably deliver multicast data while reducing latency [3]. However, this protocol was developed to operate over very high speed and low error networks such as optical circuit-switched networks operating at 800 Mbps. Finally, MFTP [16] was designed for the reliable non-real-time bulk transfer of data. Since latency is not a critical design constraint, this protocol sacrifices delay to gain extra scalability and universal operation over different network infrastructures including satellite and other asymmetric environments.

As seen from the above examples, reliable multicast protocols have been designed for specific applications in specific network environments. As reliable multicast applications begin to require low latency operation over hybrid networks, reliable multicast protocols need to be studied in such networks. A natural starting point for such studies is the consideration of the delay characteristics over satellite links. Since satellites are naturally a broadcast medium, multicast communications have the potential to greatly benefit from their wide-scale deployment. Satellite links suffer from relatively high raw bit error rates compared with terrestrial fiber links. However, with the use of encoding and decoding techniques, the satellite packet loss rates are comparable with single hop terrestrial packet loss rates. Unfortunately, satellites have higher delay characteristics than their corresponding terrestrial links. Therefore, with regards to reliable multicast applications, satellite communication provides an interesting set of technical complications in which latency becomes an important performance metric.

In the context of reliable multicast applications over satellite links, error recovery becomes a crucial issue. When using an ARQ scheme, a feedback channel is required for proper protocol function. However, due to the high latency over satellite links, such schemes achieve significantly lower throughput than when used on corresponding terrestrial networks. For this reason, as well as the relatively large amount of feedback bandwidth needed for multicast ARQ implementations, one realizes the importance of packet level FEC. As previously demonstrated in [17], hybrid protocols that allow feedback via ARQ mechanisms and utilize FEC to limit the number of feedback messages outperform those protocols based solely on ARQ or FEC mechanisms. For this reason, only hybrid error control protocols are studied in this thesis. Additionally, local recovery is studied as it could also play an important role in limiting the feedback to the source. The remainder of this section will be devoted to the detailed definition of the different scenarios and corresponding generic protocols that will be studied in this thesis. Also, general thesis-wide assumptions will be stated and explained.

3.2 Unconnected Clusters Scenario

In this scenario, a source sends multicast data over a satellite to R connectivity clusters. A connectivity cluster is a set of nodes that are virtually connected with each other. Within each of the connectivity clusters, there is a Privileged Receiver (PR) that has the ability to communicate with the satellite (see Figure 3.1). Since all other nodes within the cluster do not have this capability, these nodes are required to be connected (either directly or indirectly) to their corresponding PR.

When considering reliable multicast to a group consisting of both PRs and non-PRs, a hierarchical approach immediately presents itself as a viable option. There have been studies (e.g. [10],[18],[25]) that indirectly purport the use of hierarchical multicast as a way to reliably disseminate data. In these studies, a hierarchy is established only during the retransmission phase when qualified intermediate nodes retransmit packets. The hierarchy inherent in the unconnected cluster scenario appears in both the initial transmission phase and the retransmission phase. The satellite-capable nodes take an important role in the dissemination of data to the entire multicast group. As all packets destined for the connectivity cluster must pass through the PR, each PR has the opportunity to buffer these packets. If the PR correctly receives *k* packets (either data or parity packets), then it reconstructs the original *k* data packets. By providing the PRs with the additional ability to create parity packets, these parity packets combined with the original data packets can be used to locally satisfy retransmission and reduce the number of satellite link traversals. Therefore, the privileged receivers store the data packets and build repair (parity) packets that are subsequently used to fulfill retransmission requests. This scheme is similar to the APES scheme, SDBR proposed in [25].

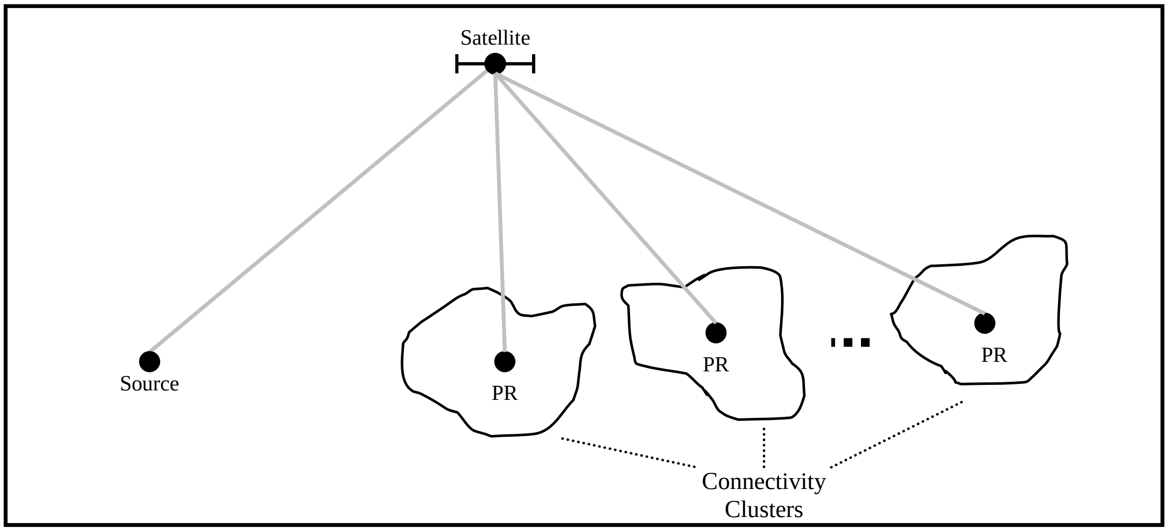

Figure 3.1: Unconnected Clusters Scenario Configuration

Although the PRs can forward any correctly received packets over the multicast tree as soon as they are received, they cannot generate parity unless they have the k original data packets. This requirement leads to the subdivision of the approach into two stages; 1) delivery from source to the PR's, and 2) delivery from PR's to nodes within connectivity clusters. Considering only the first stage reduces the problem to the reliable delivery of *k* data packets to *R* receivers. When considering the second stage, the problem becomes more difficult. The end result of this stage is the correct reception of *k* packets at every receiver within each connectivity cluster. However, the analysis changes based upon the actions taken during the first stage. In one possible case, the PRs do not forward packets until they correctly receive and decode the *k* data packets. Upon the successful reception of the *k* data packets, each PR assumes the responsibility as the multicast source for its corresponding connectivity region. In this case, the stages are distinct and the analysis of the first stage can be used in the analysis of the second stage.

Another possible course of action has the PRs immediately send correctly received original data packets to its connectivity cluster. This action allows all nodes to receive packets transmitted during the initial transmission round from the source. For scalability, parity requests from non-PR nodes do not propagate over the satellite link. Only the PRs participate in subsequent retransmission rounds with the source. Once they have correctly received enough packets to reconstruct the original data packets, the PRs generate parity repair packets for their corresponding non-PR nodes as needed. However, the analysis of this case does not lend itself to a straightforward sub-division, as the perceived packet loss probability at PR and non-PR nodes is different. The latency of the case where the stages are considered separately provides an upper bound for the case when the original data packets are immediately forwarded to the non-PR receivers. The former case in which there are two distinct stages is studied in this thesis.

Next, One must decide upon the reliable multicast protocol to be used in both stages. As previously mentioned, there are two ways to implement a hybrid reliable multicast scheme used to deliver *k* data packets to *R* receivers; layered FEC and integrated FEC. Nonnenmacher [17] studied these two implementations and found that integrated FEC outperformed layered FEC. The performance improvement results from the fact that in the layered FEC case *h* parity packets are always transmitted regardless of current network parameters (e.g. number of receivers, loss characteristics). However, in the integrated FEC scheme the number of parity packets transmitted depends upon feedback from the receivers. Using feedback ensures more efficient use of the network bandwidth by reducing the expected number of transmissions needed to reliably deliver an arbitrary data packet. In high performance terrestrial networks, the feedback delay may be negligible. However, defining protocol performance based solely on bandwidth usage neglects the delay incurred by using feedback. When feedback must traverse satellite links or other high latency links, delay no longer is negligible and becomes a critical performance metric.

The following generic integrated HEC protocol is similar to the one studied by Nonnenmacher and is used throughout this thesis.

- The sender sends a transmission group of k data packets and $a \leq h$ parity packets from the associated FEC block
- All packets within the TG can be recovered if there are fewer than a missing packets among the $k + a$ transmitted packets.
- During the initial transmission round, a receiver detecting more than a missing packets requests the number of parity packets required to complete the TG. In subsequent retransmission rounds, the receiver requests the number of packets required to complete the TG.
- The sender multicasts the maximum number of requested parity packets from all receivers until all parity packets associated with the TG have been used. At that time packets requiring retransmission are placed into a new transmission group.

Figure 3.2: Generic Integrated HEC protocol

In the immediately preceding generic protocol, the amount of parity sent during each retransmission round equaled the maximum number of packet errors from the previous round. These parity packets are subject to the same error probability as the data packets and therefore can be lost. Lost parity results in additional retransmission rounds for reliable delivery of data. As parity was sent in the initial round, extra parity can also be sent in subsequent retransmission rounds. This extra parity or “insurance” parity can be calculated using the maximal packet loss probability as measured during the initial transmission round [6].

The generic HEC protocol is studied under the infinite parity assumption as well as without this assumption. Combining the maximal packet loss probability scheme with these studies results in four basic variations; infinite parity without channel estimation considerations, infinite parity with channel estimation considerations, finite parity without channel estimation considerations, finite parity without channel estimation considerations. Additionally, simplistic adaptive mechanisms used for determining the amount of autoparity and “insurance” packets to be transmitted are studied under the infinite parity assumption.

3.3 Connected Clusters Scenario

This scenario differs from the previous scenario in that the connectivity clusters overlap (see Figure 3.3). Since these connectivity clusters overlap, there exists a terrestrial channel between the PRs in each cluster. This terrestrial connection allows for local recovery architectures to be used during the first stage of data dissemination. Since there is no simplistic way to “dynamically appoint” a repair server from the PRs (as they are all peer nodes), a local recovery scheme based upon the receiver-based, DER ungrouped architecture is assumed. In such a scheme, there exists a multicast tree connecting the privileged receivers. As mentioned in Section 2.2.2, the generic version of this protocol consists of two phases, the global retransmission phase and the local retransmission phase. The source sends the initial transmission to all PRs within a specified multicast group. When a PR detects a loss, it performs a local retransmission cycle in which it sets a local retransmission timer. This timer determines the interval that the receiver waits for the request to be locally fulfilled. If no members in the local neighborhood respond prior to the expiration of the local retransmission timer, the receiver begins a global recovery phase. In this phase, the receiver transmits a global NAK over the satellite to the source. If the requested packet is not received, then the receiver enters another local recovery cycle. This process is repeated until the entire TG is multicast from the source to the multicast group. Using the terrestrial network to obtain necessary retransmissions offers a potential reduction of the number of retransmissions traversing the high-latency satellite link.

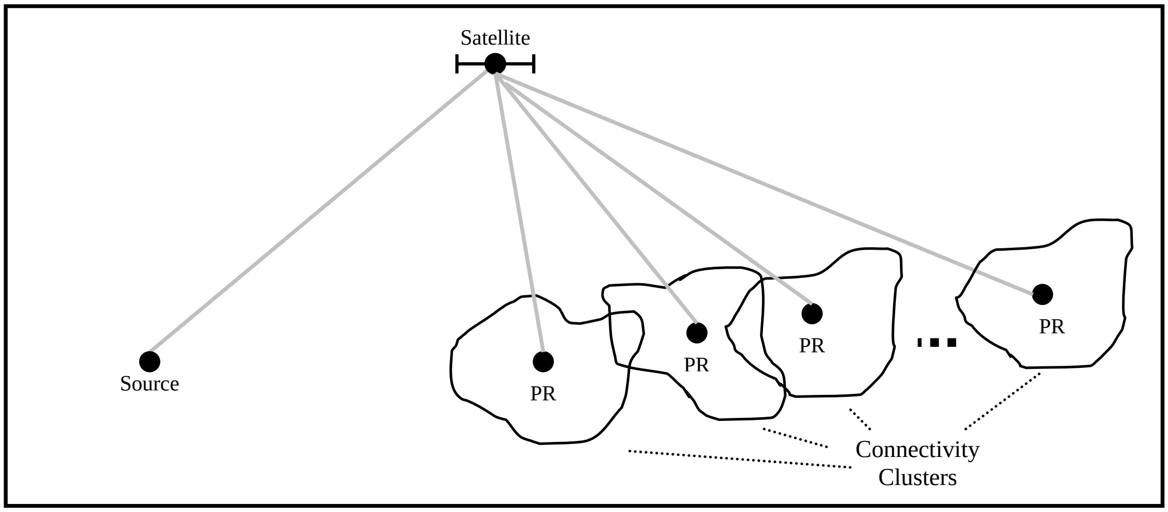

Figure 3.3: Connected Clusters Scenario Configuration

The basic protocol presented in the previous section (see Figure 3.2) is used to transmit the data from the source to the PRs. The infinite parity assumption was made for both the generation of both global and local parity packets. Therefore, only the infinite parity without channel estimation considerations and infinite parity with channel estimation considerations were combined with local recovery schemes. Only the original source is able to use autoparity and channel estimation techniques during satellite transmission (or retransmission) rounds. All local requests are fulfilled using distinct parity packets. Three local recovery schemes with different local versus satellite transmission round ratios (*LxS*) are studied. Each of these local recovery schemes are studied in conjunction with the simplistic adaptive mechanisms used in the unconnected scenario.

3.4 Assumptions

Prior to continuing, the assumptions concerning the packet error probability at receivers in the multicast group need to be considered. Throughout this thesis, packet errors (packet loses) are assumed to be both spatially and temporally independent at the receivers. In reality, the errors occurring at receivers depend upon many different factors, the most notable being the architecture of the multicast tree connecting the source to the receivers. During the first stage of the transmission in which the source transmits the TG to the PRs, the network has a star topology. However, in most other cases, multicast groups are connected via trees. When using tree structures, a loss within the tree will be shared by more than one receiver. Such a loss is known as a "shared" loss [17]. Multicast trees' shared losses are modeled well by a full binary tree (FBT) [20],[21].

In integrated FEC protocols where the sender transmits the maximum number of lost packets over the multicast tree (rather than multicasting packets singularly), the shared loss does not significantly increase the mean number of transmissions [17]. Independent losses uniformly scatter the losses amongst the receivers in the group; whereas shared losses have the potential to concentrate losses in particular areas of the tree. Regardless of the locality of the losses, the source still needs to transmit parity repairs over the entire tree. Therefore, when analyzing the protocol on a TG size granularity level, the location of the losses does not matter as much as the fact that the losses occurred somewhere within the multicast tree.

The above explanation only considers one aspect of spatial loss and does not deal with temporal loss. Burst losses are significant at the bit level over satellite links. Depending upon the strength of the bit-level coding and the amount of interleaving performed, burst errors may also appear at the packet level. Burst losses can be modeled using a two-state continuous-time Markov Chain [17]. When these types of errors occur, the timing of the retransmission influences the performance of the loss recovery. As interleaving improves FEC performance in the presence of burst errors, the same concept can be applied to the packet level. The operation of the integrated HEC results in a natural interleaving of packets. For example, the source sends 7 data packets and waits for NAK feedback indicating the need for retransmissions. During the feedback latency interval, the source can continue to send other packets (e.g. - packets from new TGs or parity packets from previous TGs). At the end of this interval, the source sends the maximum number of request parity packets. As long as there are transmission groups to be sent, the source continuously acts in this manner. In addition to the bit-level interleaving, this natural interleaving allows the source to spread the transmission of an FEC block over an interval longer than the burst-loss link thus transforming the burst loss into a more “random” loss.

4. Unconnected Clusters Scenario

4.1. Analysis

4.1.1 Infinite Parity Case without Channel Estimation

The generic integrated HEC protocol presented in Figure 3.2 is studied in this section. As most current analyses of such generic protocols do not account address delivery latency, this one focuses on the number of transmission rounds required to reliably send a TG to a group of receivers. By using the number of transmission rounds as the performance metric of interest, the analysis can be applied to different types of cost associated with multicast transmissions. Although, this analysis focuses on delay, one could easily extend the analysis to include a different type of cost; such as monetary cost or security risk.

The generic protocol is studied with the assumption that an infinite amount of parity is generated. Additionally, packet errors are assumed to be both temporally and spatially independent. The packet error probability is denoted as *p*. T denotes the round in which an arbitrary TG is successfully received at all R receivers. An example error distribution is given in Table 4.1.

Round #	Receiver 1	Receiver 2	Receiver 3	Receiver 4	Receiver 5	Max.
1	7	7	7	7	7	7
2	1	2	4	5	3	5
3	1	0	2	0	1	2
4	0	0	0	0	0	0

Table 4.1: Example Error Distribution (for k=7)

Round 1 is the initial transmission round in which the sender transmits *k+a* packets to the five receivers. For correct reception, each of the receivers must receive *k* of the *k+a* packets. During the first round, receiver 3 correctly receives *k-4* packets and therefore requires 4 parity packets to complete the TG. In the second round, the sender sends the maximum number of requested parity packets to all receivers. In the above example, the sender transmits 5 parity packets. The receivers inform the source of the number of additional parity packets needed to complete the TG. For example, receiver 3 only received 2 of the 5 transmitted parity packets and thus requires 2 additional parity packets. The process of parity request and retransmission continues until all receivers have *k* packets as shown in Round 3 of Table 4.1.

To analyze such a protocol, several random variables need to be defined. The random variable $f_j(i)$ denotes the number of outstanding packets at receiver j at the end of round i. If $f_j(i) = 0$, then receiver j has correctly received the TG during round i. Random variable, $Z(i)$ is the maximum number of required packets across all receivers for round i. The random vector $\bar{f}(i)$ contains the number of outstanding packets at each receiver at the end of round i.

$Z(i) = \max\{f_1(i), f_2(i), \ldots, f_R(i)\}$ $\qquad\qquad$ $\bar{f}(i) = [f_1(i), f_2(i), \ldots, f_R(i)]$

The probability that no retransmission rounds are needed is

$$P[T = 1] = \left[\sum_{j=0}^{a} \binom{k+a}{j} p^j (1-p)^{k+a-j}\right]^R. \tag{1}$$

To calculate the average number of retransmission rounds required to reliably deliver a TG to a set of receivers, a general probability mass function (pmf) for T must be found. Since the number of errors in the current round only depends upon the number of packets sent in the previous round, the pmf of T can be decomposed as shown in equation 2. The summation in equation 2 sums over all possible random vectors of errors from the previous round.

$$P\left[T=i\right]=\sum_{f(i-1)}P\left[T=i\mid \bar{f}(i-1)\right]\cdot P\left[\bar{f}(i-1)\right] \qquad \text{for } i\geq 2 \tag{2}$$

The conditional probability of T given the previous rounds' required packets (assuming that $\bar{f}(i-1)\neq 0$) is shown in equation 3,

$$P\left[T=i\,|\,\bar{f}(i-1)\right]=P\left[\bar{f}(i)=\bar{0}\,|\,\bar{f}(i-1)\right]=\prod_{\mathrm{s}=1}^{\mathrm{R}}P[f_s(\mathrm{i})=0\,|\,\bar{f}(i-1)]\,. \tag{3}$$

Equation 3 can be simplified into such a product form since given the previous round's errors, the mass function of the number of errors at each receiver are conditionally independent as shown in equation 4,

$$P\left[\bar{f}(i)\,|\,\bar{f}(i-1)\right]=\prod_{j=1}^{R}P\left[f_j(i)\,|\,\bar{f}(i-1)\right]. \tag{4}$$

With the additional observation that when $\bar{f}(i-1)=\bar{0}$ the TG was correctly received in the previous round, equation 3 is rewritten as shown below in equation 5,

$$P\left[T=i\,|\,\bar{f}(i-1)\right]=\begin{cases}\displaystyle\prod_{\mathrm{s}=1}^{\mathrm{R}}\sum_{\mathrm{k}=\mathrm{f_s}(\mathrm{i}\text{-}1)}^{\mathrm{Z(i\text{-}1)}}\binom{Z(i-1)}{k}p^{Z(i-1)-k}(1-p)^k & \text{if } \bar{f}(i-1)\neq\bar{0} \\ 0 & \text{if } \bar{f}(i-1)=\bar{0}\end{cases}\,. \tag{5}$$

With the conditional probability of T given the previous round's error distribution determined in equation 5, an expression for $P\left[\bar{f}(i-1)\right]$ needs to be found. This expression can be determined by summing the joint probability mass function over the i-2 previous rounds as shown below in equation 6,

$$P\left[\bar{f}(i-1)\right]=\sum_{\bar{f}(i-2)}\cdots\sum_{\bar{f}(1)}P\left[\bar{f}(i-1),\bar{f}(i-2),\ldots,\bar{f}(1)\right] \tag{6a}$$

$$P\left[\bar{f}(i-1)\right]=\sum_{\bar{f}(i-2)}\cdots\sum_{\bar{f}(1)} P\left[\bar{f}(i-1)\mid\bar{f}(i-2)\right]\cdots P\left[\bar{f}(2)\mid\bar{f}(1)\right]P\left[\bar{f}(1)\right]. \qquad (6b)$$

Since the errors in round i only depend upon the errors that occurred in round i-1 (and not in rounds i-2, …,2,1), equation 6a can be rewritten as shown in equation 6b. Using equation 4, one sees that $P\left[\bar{f}(i-1)\right]$ can be obtained using the receivers' conditional probability given the previous round's results. In general, these conditional probabilities can be written as (where $l > 0$):

$$P\left[f_j(i)=0\mid\bar{f}(i-1)\right]=\sum_{k=f_j(i-1)}^{Z(i-1)}\binom{Z(i-1)}{k}p^{Z(i-1)-k}(1-p)^k \qquad (7a)$$

$$P\left[f_j(i)=l\mid\bar{f}(i-1)\right]=\begin{cases}\binom{Z(i-1)}{f_j(i-1)-l}p^{Z(i-1)-(f_j(i-1)-l)}(1-p)^{f_j(i-1)-l} & \text{if } f_j(i-1)\geq l\\ 0 & \text{if } f_j(i-1)<l.\end{cases} \qquad (7b)$$

The last term in equation 6b, $P\left[\bar{f}(1)\right]$, is shown in equation 8. In this equation, n_i denotes the number of errors experienced by receiver *i* during the first round. This mass function can be decomposed in such a manner since packet errors were assumed to be independent.

$$P\left[\bar{f}(1)=\bar{n}\right]=P\left[f_1(1)=n_1,\ldots,f_R(1)=n_R\right]=\prod_{r=1}^{R}P\left[f_r(1)=n_r\right] \qquad (8)$$

The individual terms in the right hand side of equation 8 are solely based upon the number of packets transmitted *(k+a)* and the packet loss probability *p*. These terms are shown in equation 9,

$$P\left[f_r(1)=0\right]=\sum_{j=0}^{a}\binom{k+a}{j}p^j(1-p)^{k+a-j} \qquad (9a)$$

$$P\left[f_r(1)=j\right]=\binom{k+a}{j+a}p^{j+a}(1-p)^{k-j} \qquad \text{for k} \geq j > a\,. \quad (9b)$$

Equations 3-9 can be subsituted into equation 2 to find the pmf of T. The pmf of T can then be used to find an expression for the expected number of rounds needed to reliably deliver data to R receivers:

$$E[T] = \sum_{i=0}^{\infty} i \cdot P[T = i]. \qquad (10)$$

Using $E[T]$, the expected delay per transmission group, $E[D_{group}]$ can be written as

$$E[D_{group}] = d_S \cdot E[T]. \qquad (11)$$

In equation 11, d_s represents the delay associated with sending feedback over the satellite. More specifically, d_s equals twice the round trip time between the sender and each receiver.

Unfortunately, the above analysis does not lend itself to computational analysis. This is partly due to the fact that the number of probabilities per round equals (# data packets)^R. Even if only one such data structure were used for 7 data packets and 100 receivers, the memory requirement would be prohibitive. Additionally, the number of operations for such a large set of probabilities would be prohibitive as well. Therefore, a bound is needed so that the results can be analyzed.

An upper bound on $E[T]$ can be found by assuming each of the R receivers experiences the maximum number of errors that occurred in the entire group. This assumption means that each receiver must receive all parity packets transmitted in a particular round to be considered error free. Since the number of parity packets transmitted is the maximum over all receivers errors, each receiver must correctly receive a number of parity packets greater than or equal to their actual needs. However, this bound was so loose that the results were useless. Thus a tighter bound needed to be developed.

A tighter bound was obtained by viewing the situation from the perspective of an “aggregated” receiver. The total number of packets required for completed reception of the TG across all receivers is simply the addition of the number of outstanding packets at each receiver. Table 4.2 shows the summation of outstanding packets (Total column) of the example presented in Table 4.1.

Round	Receiver 1	Receiver 2	Receiver 3	Receiver 4	Receiver 5	Max.	Total
0	7	7	7	7	7	7	35
1	1	2	4	5	3	5	15
2	1	0	2	0	1	2	4
3	0	0	0	0	0	0	0

Table 4.2: Example Error Distribution with Total Column (for k=7)

By viewing the entire group of receivers as a black box, the complete reception of the TG in round i requires that the total number of outstanding packets from round i-1 enters the box. For example, 15 packets need to be correctly received in round 2 for reliable reception of the TG. In the generic protocol, each receiver has an opportunity to satisfy its packet needs in round i by receiving $f_j(i-1)$ parity packets from the $Z(i-1)$ transmitted from the sender. The aggregate receiver could correctly receive up to $R*Z(i-1)$ packets. Using the assumption that only the total number of outstanding packets from the previous round is sent to the aggregate receiver results in an upper bound since each packet will have to be correctly received. In other words, the sender sends $Z(i-1)$ in round i and each receiver’s outstanding packet count $f_r(i-1)$ must be less than or equal to $Z(i-1)$. In the generic protocol there is the chance that correct reception occurs in the presence of errors. To analyze the above situation, a new set of nonmenclature needs to be introduced. Let $S(i)$ denote the total number of outstanding packets in round i.

$$S(i) = \sum_{r=1}^{R} f_r(i) \tag{12}$$

As shown in Table 4.2, in the total column, $S(i)$ is the number of packets which the “aggregated” receiver must correctly receive during round i so that the transmission of the TG is complete. With the exception of the initial transmission round, the number of outstanding packets in round i depends only upon the number of outstanding packets in round i-1. Therefore, the probability mass function of $S(i)$ can be decomposed using conditional probabilities as shown in equation 13,

$$P[S(i) = m] = \sum_{s_{i-1}=0}^{s_{i-2}} \cdots \sum_{s_1=0}^{R*k} P[S(i) = m \mid S(i-1) = s_{i-1}] \cdots P[S(1) = s_1]. \tag{13}$$

In equation 13, *R*k* denotes the maximum possible number of outstanding packets that need to be correctly received at the aggregated receiver during the first round. In subsequent rounds, the maximum possible number of outstanding packets at the aggregated receiver depends upon the results from the previous rounds. For $i \geq 2$, the conditional probability mass functions in equation 13 are shown in equation 14,

$$P[S(i) = l \mid S(i-1) = x] = \begin{cases} \binom{x}{l} p^l (1-p)^{x-l} & \text{if } x \geq l \geq 0 \\ 0 & \text{if } x < l. \end{cases} \tag{14}$$

The last term in equation 13, $P[S(1)]$ is more difficult to compute since during the initial transmission round auto-parity may be transmitted. If auto-parity is transmitted $(a > 0)$, then it must be considered in the pmf of $S(1)$. Let $V_h(1)$ denote the total number of outstanding packets in round 1 if there are *h* receivers in the group.

$$V_h(1) = \sum_{r=1}^{h} f_r(1) \tag{15}$$

For an arbitrary number of receivers, $P[S(1)]$ can be determined using the following equations,

$$P[V_1(1) = j] = P[f_1(1) = j] \tag{16}$$

$$P[V_h(1) = j] = \sum_{i=0}^{k*(h-1)} P[f_h(1) = j - i \mid V_{h-1}(1) = i] \cdot P[V_{h-1}(1) = i]. \tag{17}$$

Since $i \leq j$, equation 17 can be rewritten as

$$P[V_h(1) = j] = \sum_{i=0}^{j} P[f_h(1) = j - i \mid V_{h-1}(1) = i] \cdot P[V_{h-1}(1) = i]. \tag{18}$$

Realizing that $P[S(1) = j] = P[V_R(1) = j]$, one can find the pmf of $S(1)$ through iterative calculations ranging from $h = 1,..., R$. Equation 9 is also needed in these calculations. Using equations 14 and 18, an upper bound on $E[T]$ can be found by evaluating the following equation.

$$E[T] = \sum_{i=1}^{\infty} i \cdot P[T = i] = \sum_{i=1}^{\infty} i \cdot \sum_{S(i-1)} P[T = i \mid S(i-1)] \cdot P[S(i-1)] \tag{19}$$

As shown in equation 20, $P[T = i \mid S(i-1)]$ has the same basic form as equation 14, but the case when $S(i-1) = 0$ must be handled separately.

$$P[T = i \mid S(i-1) = x] = \begin{cases} (1-p)^x & \text{if } x > 0 \\ 0 & \text{if } x = 0. \end{cases} \tag{20}$$

Using the “aggregated” receiver approximation, an upper bound for the previous analysis has been found. The variance of this mean can be found easily by calculating the second moment of T as shown in equation 21,

$$E[T^2] = \sum_{i=1}^{\infty} i^2 \cdot P[T = i]. \tag{21}$$

The variance of T is found using equation 22,

$$Var[T] = E[T^2] - E[T]^2. \tag{22}$$

Using this upper bound analysis, the expected number of transmission rounds was found for different cases by varying different parameters (namely the packet loss probability, the number of receivers, and the amount of autoparity). As can be seen in Figure 4.1, the use of autoparity significantly reduces the expected number of transmission rounds. When the packet loss probability equals 0.1 and the number of receivers equals 100 (see Figure 4.1-left), the use of three autoparity packets reduces the average number of retransmission rounds by approximately 1.7 (or about 47%) when compared with the no autoparity case. However, for 100 receivers with a packet loss probability of 0.2 (see Figure 4.1-right), the improvement is not as drastic. Using three autoparity packets reduces the expected number of retransmission rounds by approximately 1.4 (or 27%). Additionally, these graphs also show that as the number of receivers increase, the expected number of retransmission rounds grows slowly. This result agrees with the intuition that given a large number of receivers, the addition of another receiver does not result in a large change in the expected number of retransmission rounds. For example, given one receiver, the addition of another one increases the total number of receivers by 100% whereas, given eighty receiver, the addition of another results in an increase of approximately 1.25%. Therefore, the increase of expected number of transmission rounds in the former case is much larger than in the increase in the latter.

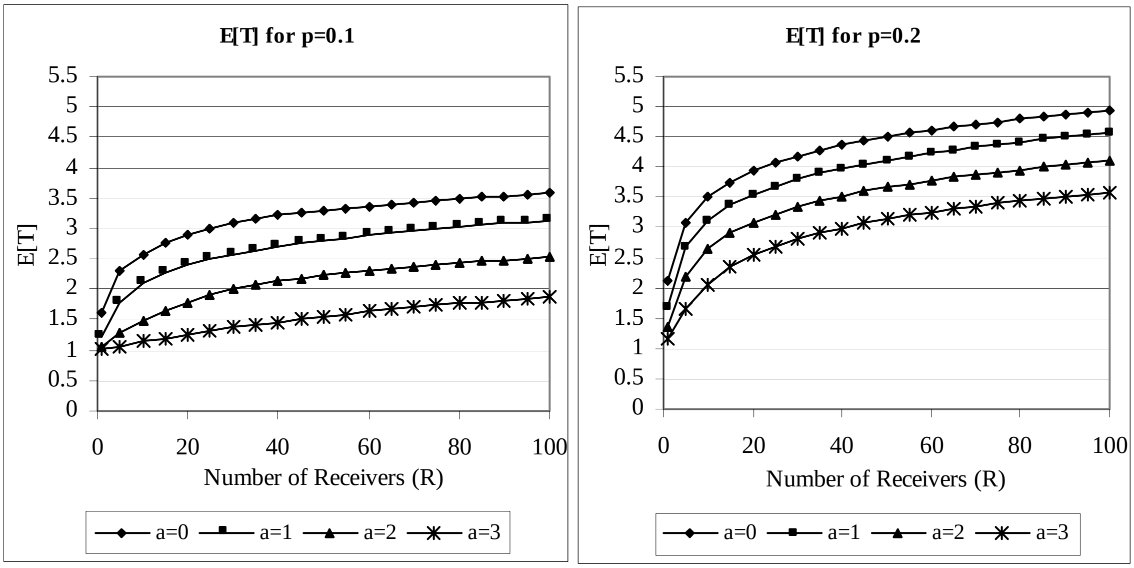

Figure 4.1 E[T] with packet loss probability of 0.1 & 0.2

In Figure 4.2, the expected number of transmission rounds is plotted versus the packet loss probability for autoparity values of 1 and 3. The phenomenon of the decreasing rate of change in $E[T]$ is observed by noticing the declining distance between the curves representing different numbers of receivers. By using more autoparity packets (see Figure 4.2-right), the expected number of transmissions decreases when compared with the case when fewer autparity parity (see Figure 4.2-left) packets are employed. Additionally, when a=3, $E[T]$ remains one until the probability of packet loss reaches approximately 0.03. When a=1, $E[T]$ begins to increase for values of *p* between 0.001 and 0.01. Across all packet loss probabilities the use of additional autoparity packets enables the data to be reliably delivered in fewer retransmission rounds.

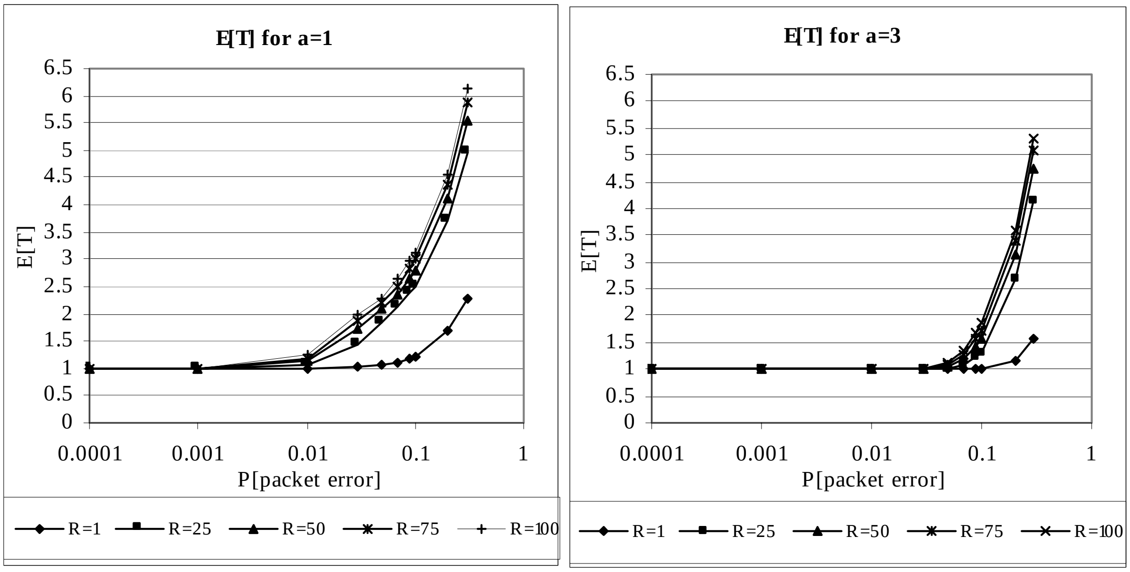

Figure 4.2 E[T] with a=1 & a=3

In Figure 4.3, the variance of T corresponding to the packet loss probabilities in Figure 4.1 are shown. From these two figures, one can see that the Var[T] appears to converge to a constant value. The above statement is made only as a particular observation and not as a generalization. The variance appears to converge to approximately 0.4 in Figure 4.3-left and 0.7 in Figure 4.3-right. The importance of this observation is that the variances are modest (0.4 on a mean of 3 is approximately 13%) and therefore any negotiations pertaining to quality of service (QoS) or pricing must account for this variability. The variance of T increases as the packet loss probability increases as shown in Figure 4.4. Therefore, larger packet loss probabilities result in more variable expected values.

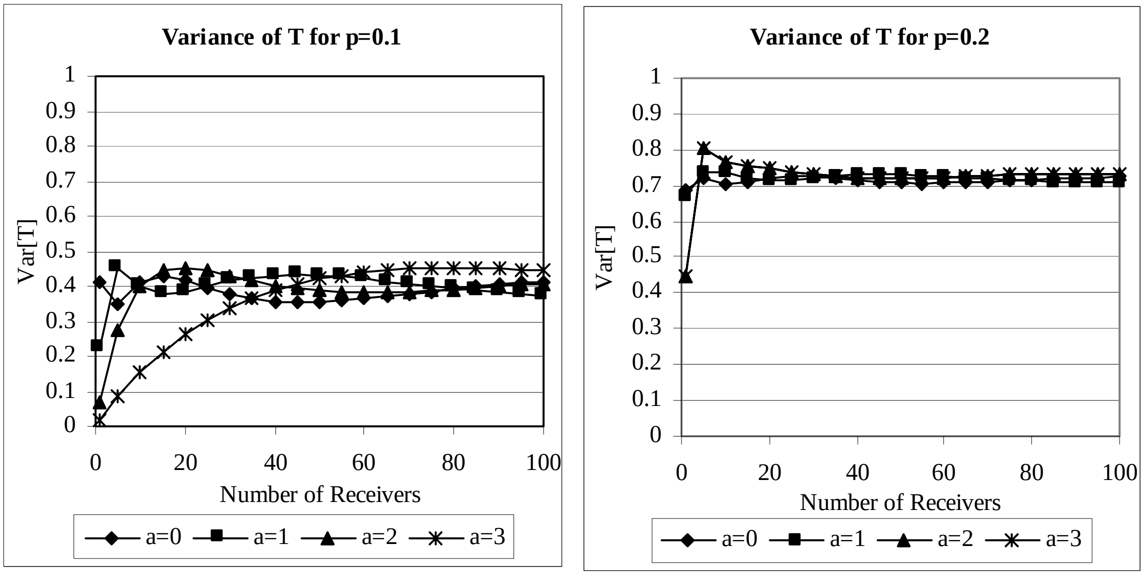

Figure 4.3: Variance of the Number of Transmission Round

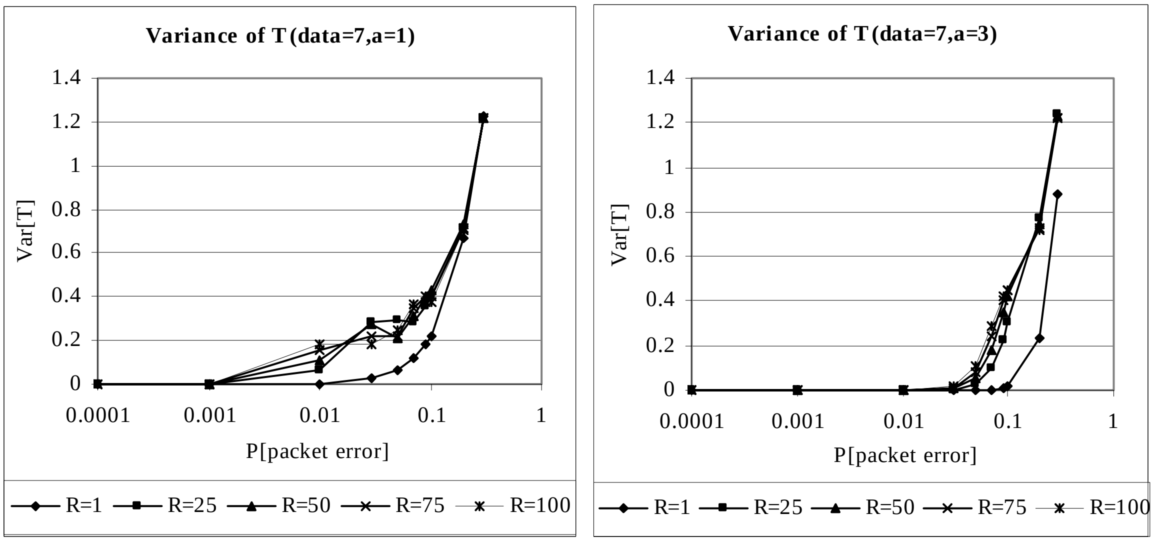

Figure 4.4: Var[T] versus packet loss probability

However, prior to suggesting the use of autoparity packets in actual implementations, one must first determine their associated cost (namely an increase in the number of transmitted packets required to reliably deliver the data). The expected number of transmissions (i.e. – the expected number of packets) to reliably deliver a TG needs to be found. The number of transmission rounds does not factor into the calculation of the number of packets since parity packets are sent only when they are needed. Therefore, the number of parity packets sent in response to parity requests is not coupled with the number of rounds needed to deliver the parity. The following analysis was completed by Nonnenmacher [17]. If L_r denotes the number of parity retransmissions required by a random receiver, then its distribution can be written as:

$$P[L_r = 0] = \sum_{j=0}^{a} \binom{k+a}{j} p^j (1-p)^{k+a-j} \tag{23a}$$

$$P[L_r = m] = \binom{k+a+m-1}{k-1} p^{m+a} (1-p)^k \qquad m = 1,2,\ldots \tag{23b}$$

Let L be the maximum number of parity retransmissions required by a group of R receivers.

$$P[L \le m] = (P[L_r \le m])^R = \left(\sum_{i=0}^{m} P[L_r = i] \right)^R \qquad m = 1,2,\ldots \tag{24}$$

Using equation 24, the expected value of the maximum number of parity retransmissions can be found using

$$E[L] = \sum_{m=0}^{\infty} (1 - P[L \le m]) . \tag{25}$$

With $E[L]$ calculated, the number of transmissions per TG, M, can be found using the following equation:

$$E[M] = E[L] + k + a . \tag{26}$$

The expected number of packets per TG (of size 7) for the cases when *a* equals one and *a* equals three are respectively shown in Figure 4.5. As seen from these figures, for low packet loss probabilities the use of autoparity results in more packets transmitted per TG. However at approximately *p*=0.1, the number of transmitted packets for *a*=1 and *a*=3 are the same for a large number of receivers. To better illustrate this point, the ratio between $E[M]$ for a=1 and a=0 as well as a=3 and a=0 in Figures 4.6. One can see that for larger number of receivers (e.g. 100), as the probability of error increases, both ratios tend toward one. The probability of error where the ratio becomes one increases as the amount of autoparity increases since errors need to occur more frequently so that the autoparity is actually used to correct packets rather than being wasted as unused overhead. However, if the receiver can obtain estimates of the packet error probability prior to the initial transmission of a TG, then the number of autoparity packets can be adjusted to reduce autoparity's bandwidth overhead to reasonable levels (the definition of reasonable is application dependent). As previously discussed the use of autoparity reduced the number of transmissions needed for reliable delivery. This result, coupled with the results shown in Figure 4.5 and Figure 4.6, leads one to conclude that the intelligent use of autoparity can help to lower the delay without drastically increasing bandwidth usage.

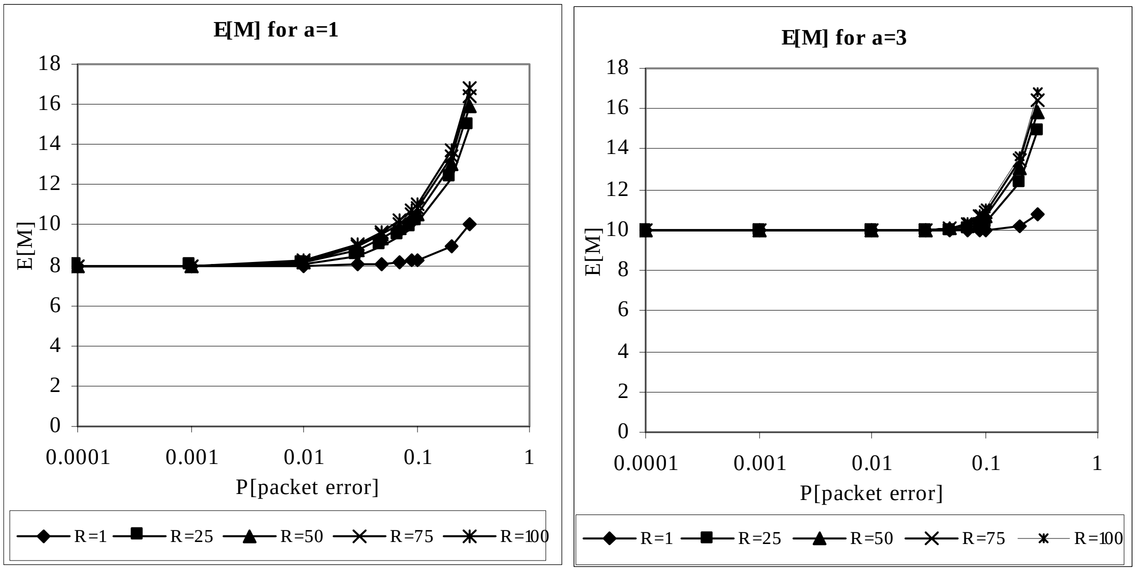

Figure 4.5: Expected Number of Packets Transmitted for a=1 & a=3

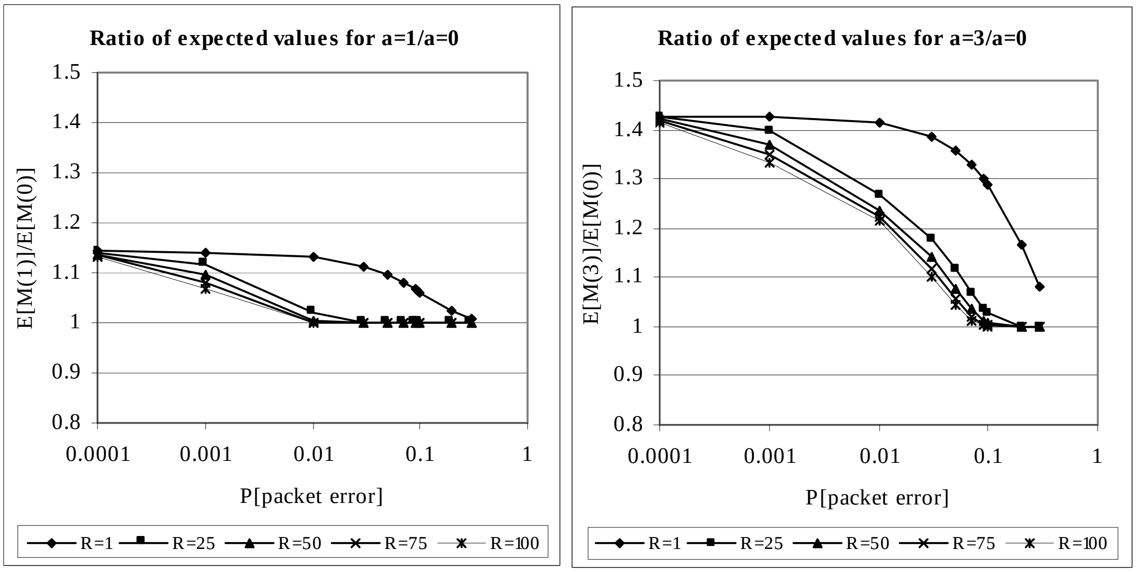

Figure 4.6 Expected value ratios (a=1/a=0) & (a=3/a=0)

4.1.2 Infinite Parity with Channel Estimation

In the previous scenario, the amount of parity sent during each transmission round equaled the maximum number of packet errors from the previous round. These parity packets are subject to the same error probability as the data packets and therefore can be lost. Lost parity results in additional retransmission rounds for reliable delivery of data. As parity was sent in the initial round, extra parity can also be sent in subsequent retransmission rounds. This extra parity or “insurance” parity can be calculated using the maximal packet loss probability[6]. By definition, this p′ remains constant over the entire set of retransmissions until available parity is depleted. When the parity is deleted, any outstanding packets are moved to the next TG and transmitted. The random variable *E* denotes the maximum number of lost packets in the initial transmission. This random variable can take values ranging from zero to $k+a$. The maximal packet loss probability, p′, equals

$$p' = \frac{\text{maximum \# of packets lost in initial transmission}}{\text{total \# of packets in initial transmission}} = \frac{E}{k+a} \tag{27}$$

This protocol functions as the one presented in section 4.1.1, however the number of parity packets transmitted during each subsequent retransmission round is modified so that the “insurance” packets are sent in addition to the maximum number of requested parity packets. For the source to properly determine the value for *E*, the minimum number of packets correctly received across the entire group is needed. Therefore, after the initial transmission, each receiver needs to send the number of packets (including data and parity) correctly received during the initial transmission. The random variable $g_j(i)$ denotes the number of correctly received packets at receiver j at the end of round i. The random vector $\overline{g}(i)$ contains the number of correctly received packets at each receiver at the end of round i. However, this additional information is only needed during the first round. After the first round, $\overline{g}(i)$ and $\overline{f}(i)$ yield the same information. Since $\overline{g}(1)$ contains more information than $\overline{f}(1)$, from any conditioning based upon $\overline{g}(1)$ one can completely determine $\overline{f}(1)$. Using $\overline{g}(1)$, *E* can be determined using equation 28,

$$E = k + a - \min\{g_1(1), g_2(1), \ldots, g_R(1)\}. \tag{28}$$

The random variable $N(i)$ is defined to be the number of parity packets transmitted in round i when channel estimation is employed (see equation 29),

$$N(i) = \left\lceil \left(\frac{1}{1-p'}\right) Z(i) \right\rceil. \tag{29}$$

The analysis for this case follows similar lines as the one presented in section 4.1.1. The probability that no retransmission rounds are needed for reliable multicast is given in equation 1. The primary difference between the analysis for the channel estimation case and the non-channel estimation case is the conditioning of the random variable T. In the channel estimation case, T not only depends upon the previous round's errors, but also upon the first round errors. Therefore, equation 2 must be modified so that the first round is appropriately handled as is shown in equation 30,

$$P[T=i] = \sum_{\bar{f}(i-1)} P\left[T=i \mid \bar{f}(i-1), \bar{g}(1)\right] \cdot P\left[\bar{f}(i-1), \bar{g}(1)\right]. \tag{30}$$

Equation 30 can be rewritten as:

$$P[T=i] = \sum_{\bar{f}(i-1)} P\left[T=i \mid \bar{f}(i-1), \bar{g}(1)\right] \cdot P\left[\bar{f}(i-1) \mid \bar{g}(1)\right] \cdot P[\bar{g}(1)]. \tag{31}$$

The conditional probability of T given the previous round and the first round results (assuming that $\bar{f}(i-1) \neq 0$) is shown in equation 32. The pmf of T is simplified into a product form since the packet error probability of the receivers is conditionally independent given the previous and first round results,

$$P\left[T=i \mid \bar{f}(i-1), \bar{g}(1)\right] = P\left[\bar{f}(i) = \bar{0} \mid \bar{f}(i-1), \bar{g}(1)\right] = \prod_{s=1}^{R} P[f_s(\mathrm{i}) = 0 \mid \bar{f}(i-1), \bar{g}(1)]. \tag{32}$$

Using a similar expansion to the one used in equations 3-5, one can write equation 32 as

$$P\left[T=i \mid \bar{f}(i-1), \bar{g}(1)\right] = \begin{cases} \prod\limits_{s=1}^{R} \sum\limits_{k=f_s(i-1)}^{N(i-1)} \binom{N(i-1)}{k} p^{N(i-1)-k} (1-p)^k & \text{if } \bar{f}(i-1) \neq \bar{0} \\ 0 & \text{if } \bar{f}(i-1) = \bar{0}. \end{cases} \tag{33}$$

With the conditional probability of T determined in equation 33, an expression for $P\left[\bar{f}(i-1) \mid \bar{g}(1)\right]$ must be found. This expression is summarized in equation 34,

$$P\left[\bar{f}(i-1) \mid \bar{g}(1)\right] = \sum_{\bar{f}(i-2)} \cdots \sum_{\bar{f}(2)} P\left[\bar{f}(i-1), \bar{f}(i-2), \ldots, \bar{f}(2) \mid \bar{g}(1)\right] \tag{34a}$$

$$P\left[\bar{f}(i-1) \mid \bar{g}(1)\right] = \sum_{\bar{f}(i-2)} \cdots \sum_{\bar{f}(2)} P\left[\bar{f}(i-1) \mid \bar{f}(i-2), \bar{g}(1)\right] \cdots P\left[\bar{f}(2) \mid \bar{g}(1)\right] . \tag{34b}$$

In general, the terms in the right-hand side of equation 34 can be decomposed using equation 35,

$$P\left[\bar{f}(i) \mid \bar{f}(i-1), \bar{g}(1)\right] = \prod_{j=1}^{R} P\left[f_j(i) \mid \bar{f}(i-1), \bar{g}(1)\right]. \tag{35}$$

The pmf of the term in the right-hand side of equation 35 is shown in equation 36. When i=2, the formulas in equations 35 and 36 respectively decompose to $P\left[\bar{f}(2) \mid \bar{g}(1)\right]$ and $P\left[f_j(2) \mid \bar{g}(1)\right]$. Since $\bar{g}(1)$ contains more information than $\bar{f}(1)$, these conditional pmfs can be found using equation 7.

$$P\left[f_j(i) = 0 \mid \bar{f}(i-1), \bar{g}(1)\right] = \sum_{k=f_j(i-1)}^{N(i-1)} \binom{N(i-1)}{k} p^{N(i-1)-k} (1-p)^{k} \tag{36a}$$

$$P\left[f_j(i) = l \mid \bar{f}(i-1), \bar{g}(1)\right] = \begin{cases} \dbinom{N(i-1)}{f_j(i-1)-l} p^{N(i-1)-\left(f_j(i-1)-l\right)} (1-p)^{f_j(i-1)-l} & \text{if } f_j(i-1) \geq l \\ 0 & \text{if } f_j(i-1) < l \end{cases} \tag{36b}$$

To complete the analysis, the last term in equation 31, $P[\bar{g}(1)]$, is shown in equation 37. In this equation, n_i denotes the number of errors experienced by receiver i during the first round. This mass function can be decomposed in such a manner since packet errors were assumed to be independent.

$$P[\bar{g}(1) = \bar{n}] = P[g_1(1) = n_1, \ldots, g_R(1) = n_R] = \prod_{r=1}^{R} P[g_r(1) = n_r] \tag{37}$$

The individual terms in the right hand side of equation 37 are solely based upon the number of packets transmitted *(k+a)* and the packet loss probability *p*. These terms are shown in equation 38,

$$P[g_r(1) = j] = \begin{pmatrix} k+a \\ j \end{pmatrix} p^j (1-p)^{k+a-j} \quad \text{where } k + a \geq j \geq 0 . \tag{38}$$

Using equations 30-38, the expected number of rounds required to reliably deliver data to a group of R receivers can be calculated using equation 10.

Now that the formula for $E[T]$ has been developed, the expected number of transmissions needed to reliably deliver a TG needs to be found. The analysis completed by Nonnenmacher [17] needs to be extended to account for the insurance parity packets. In this analysis, it becomes necessary to find the average number of required parity retransmissions given the number of transmission rounds required to reliably deliver the TG. Using this result, the average number of retransmissions can be calculated,

$$E[M] = \sum_{i=0}^{\infty} E[M \mid T = i] \cdot P[T = i]. \tag{39}$$

The probability that the number of retransmission rounds equals i, $P[T = i]$ is found using equation 31 from the preceding analysis. $E[M \mid T = i]$ is found using the following equation,

$$E[M \mid T = i] = \sum_{j=0}^{i} E[N(j)]. \tag{40}$$

The expression for $E[N(j)]$ is found in the following analysis,

$$E[N(j)] = \sum_{\bar{f}(i-1)} \sum_{\bar{g}(1)} E\left[N(j) \mid \bar{f}(i-1), \bar{g}(1)\right] \cdot P\left[\bar{f}(i-1) \mid \bar{g}(1)\right] \cdot P[\bar{g}(1)]. \tag{41}$$

Since expressions for $P\left[\bar{f}(i-1) \mid \bar{g}(1)\right]$ and for $P[\bar{g}(1)]$ have already been developed in equations 34 and 37 respectively, only the expression for $E\left[N(j) \mid \bar{f}(i-1), \bar{g}(1)\right]$ needs to be found:

$$E\left[N(j) \mid \bar{f}(i-1), \bar{g}(1)\right] = \sum_{m=0}^{\infty} P\left[N(j) > m \mid \bar{f}(i-1), \bar{g}(1)\right] \tag{42a}$$

$$= \sum_{m=0}^{\infty} \left(1 - P\left[N(j) \le m \mid \bar{f}(i-1), \bar{g}(1)\right]\right). \tag{42b}$$

Let $N_r(i)$ equal the number of parity packets (including insurance packets) sent for an arbitrary receiver r during round i if there are no other receivers in the group,

$$N_r(i) = \left\lceil \left(\frac{1}{1-p'}\right) f_r(i) \right\rceil. \tag{43}$$

Using the property of the maximum value of random variables, equation 42b can be rewritten as

$$E\left[N(j) \mid \bar{f}(i-1), \bar{g}(1)\right] = \sum_{m=0}^{\infty} \left(1 - \prod_{r=1}^{R} P\left[N_r(j) \le m \mid \bar{f}(i-1), \bar{g}(1)\right]\right). \tag{44}$$

Since $\lceil wx \rceil = y \Leftrightarrow x = \lfloor y / w \rfloor$ where x and y are integers and w is a real number greater than or equal to 1, the following equation holds

$$P\left[N_r(j) = l \mid \bar{f}(i-1), \bar{g}(1)\right] = P\left[f_r(j) = l' \mid \bar{f}(i-1), \bar{g}(1)\right] \qquad \text{where } l' = \lfloor (1 - p') \cdot l \rfloor. \tag{45}$$

Therefore, using equation 45 and equation 36, the right hand side of equation 44 can be determined and $E[N(j)]$ can be found.

As was the case in the non-channel estimation case, the above analysis does not lend itself to computational analysis. Therefore, a bound is needed so that the results can be analyzed. As previously discussed, the upper bound on $E[T]$ found by assuming each of the R receivers experiences the maximum number of errors that occurred in the entire group was useless. Unfortunately, viewing the situation from the perspective of an “aggregated” receiver does not result in an upper bound.

As in Section 4.1.1, $S(i)$ is the number of packets which the “aggregated” receiver must correctly receive during round i so that the transmission of the TG is complete (i.e. the total number of packets required across the entire multicast group). Using this “aggregated” receiver, the number of packets transmitted by the source during round i, $N_a(i)$ equals

$$N_a(i) = \left\lceil \left(\frac{1}{1-p'} \right) S(i) \right\rceil . \tag{46}$$

For this approximation to be considered as an upper bound for $E[T]$, $N_a(i)$ needs to be less than or equal to $N(i)$. This requirement is verified below:

$$S(i) \leq R \cdot Z(i) \ , \tag{47a}$$

$$S(i) \cdot \left(\frac{1}{1-p'} \right) \leq R \cdot Z(i) \cdot \left(\frac{1}{1-p'} \right), \tag{47b}$$

$$\left\lceil S(i) \cdot \left(\frac{1}{1-p'} \right) \right\rceil \leq \left\lceil R \cdot Z(i) \cdot \left(\frac{1}{1-p'} \right) \right\rceil, \tag{47c}$$

$$N_a(i) \leq N(i) . \tag{47d}$$

Unfortunately, the requirement does not verify that the “aggregated” receiver will result in an upper bound on $E[T]$. Considering the example first considered in Table 4.1, one can provide a case that shows the “aggregated” receiver method does not result in an upper bound. Consider the first retransmission round of Table 4.1 as shown below in Table 4.3. With the assumption that a=1, the number of packets actually transmitted to each receiver equals 14 for a total of 70 packets as seen by an aggregated receiver. Using the “aggregated” receiver, the total number of packets transmitted to the entire group equals 45.

Round	Receiver 1	Receiver 2	Receiver 3	Receiver 4	Receiver 5	Max.	Total
2	1	2	4	5	3	5	15

Table 4.3: Round 1 from Table 4.1

Since there are fewer parity packets transmitted in the “aggregated” receiver case, one may conclude that this does indeed result in a lower bound on the expected number of retransmission rounds. However, using the “aggregated” receiver approach certain cases arise in which the packet is received correctly even though this would not occur in the actual protocol. For example, there could be zero errors at four receivers, and 14 at the fifth (assuming the fifth actually needed 3 parity packets). Since the total number of errors is less than 45, the “aggregated” receiver would view this situation as successful transmission. In reality, the fifth receiver would not receive the required three parity packets. Therefore, this “aggregated” receiver can no longer be viewed as an upper bound for the number of transmission rounds, it can only be viewed as an approximation.

Analyzing the “aggregated” receiver case as an approximation can still provide useful insights into the problem. The probability mass function of $S(i)$ can be decomposed using conditional probabilities as shown in equation 48,

$$P[S(i)=m]=\sum_{s_{i-1}=0}^{R^*k}\cdots\sum_{s_1=0}^{R^*k}\sum_{e=0}^{k+a}P[S(i)=m\,|\,S(i-1)=s_{i-1},E=e]\cdots P[S(1)=s_1\,|\,E=e]P[E=e]. \quad (48)$$

For $i \geq 2$, the conditional probability mass functions in equation 45 are shown in equation 49,

$$P\left[S(i)=l \mid S(i-1)=x,E\right]=\begin{cases}\begin{pmatrix}N_a(i-1)\\ x-l\end{pmatrix}p^{N_a(i-1)-(x-l)}(1-p)^{x-l} & \text{if } 0\le x-l\le N_a(i-1)\\ 0 & \text{otherwise}\end{cases} \quad (49a)$$

$$P\left[S(i)=0 \mid S(i-1)=x,E\right]=\begin{cases}\displaystyle\sum_{k=x}^{N_a(i-1)}\begin{pmatrix}N_a(i-1)\\ k\end{pmatrix}p^{N_a(i-1)-k}(1-p)^{k} & \text{if } 0\le x-l\le N_a(i-1)\\ 0 & \text{otherwise.}\end{cases} \quad (49b)$$

The term $P\left[S(1)=s_1 \mid E=e\right]$ in equation 48 can be simplified to $P\left[S(1)=s_1\right]$ since $S(1)$ does not depend upon the channel estimation performed during the initial transmission. Therefore, the probability mass function for $S(1)$ is found following the same procedure outlined in equations 15-18 in Section 4.1.1. To complete the analysis, the last term of equation 48, $P\left[E=e\right]$, needs to be found. Using equation 38, the $P\left[E=e\right]$ can be found as shown in equation 47,

$$P\left[E\le e\right]=P\left[g_r(1)\le e\right]^R \quad (50a)$$

$$P\left[E=e\right]=P\left[g_r(1)\le e\right]^R-P\left[g_r(1)\le e-1\right]^R. \quad (50b)$$

Finally, an expression for determining the expected number of transmissions rounds required to reliably multicast a TG under the “aggregated” receiver approximation is shown in equation 51,

$$E[T]=\sum_{i=1}^{\infty}i\cdot P\left[T=i\right]=\sum_{i=1}^{\infty}i\cdot\sum_{S(i-1)}\sum_{E}P\left[T=i \mid S(i-1),E\right]\cdot P\left[S(i-1)\mid E\right]\cdot P\left[E\right]. \quad (51)$$

The $P\left[S(i-1)\mid \mathrm{E}\right]$ term can be easily found by modifying the equation 51 as follows

$$P\left[S(i)=m \mid E\right]=\sum_{s_{i-1}=0}^{R*k}\cdots\sum_{s_1=0}^{R*k}\sum_{e=0}^{k+a}P\left[S(i)=m \mid S(i-1)=s_{i-1},E\right]\cdots P\left[S(1)=s_1\right]. \quad (52)$$

$P[T = i \mid S(i-1), E]$ has the same basic form as equation 49a, but the case when $S(i-1) = 0$ must be handled separately.

$$P[T = i \mid S(i-1) = x, E] = \begin{cases} \binom{N_a(i-1)}{x} p^{N_a - x}(1-p)^x & \text{if } x > 0 \\ 0 & \text{if } x = 0 \end{cases} \qquad (53)$$

Using the "aggregated" receiver approximation, the expected number of transmission rounds was found for different cases by varying different parameters. As seen in Figure 4.7, the expected number of transmission rounds appears to be bounded by 2. By comparing channel estimation results shown in Figures 4.7 and 4.8 with their counterparts, one observes that the use of "insurance" packets appears to have reduced the number of transmission rounds in all cases. As can be expected, Figure 4.8 shows that using additional autoparity packets lowers the average number of transmission rounds for values of *p* between 0.001 and 0.01.

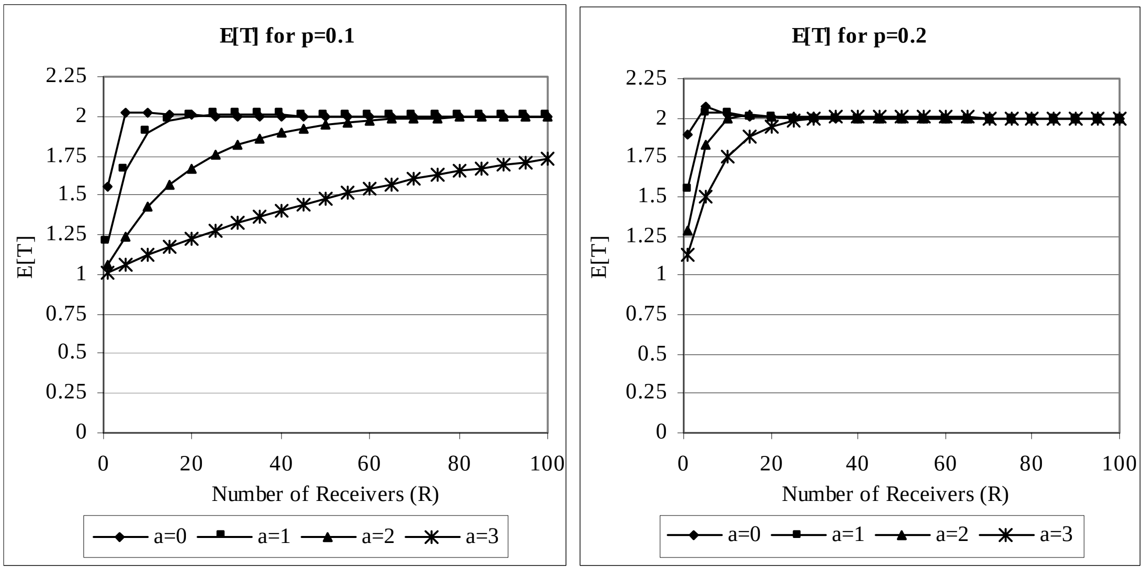

Figure 4.7: E[T] for channel estimation w/ packet loss probability of 0.1 & 0.2

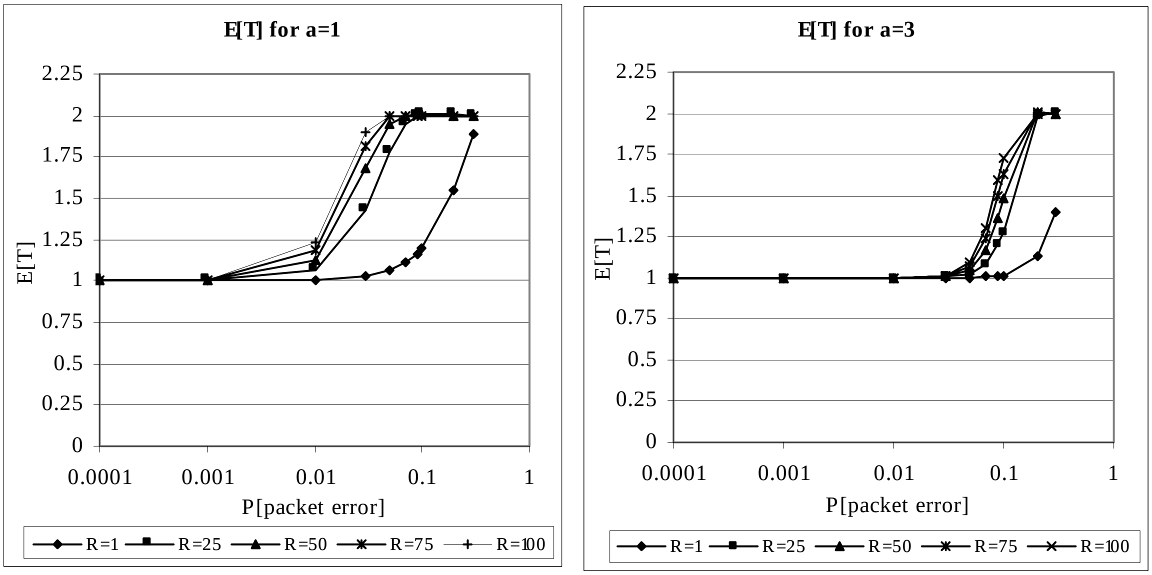

Figure 4.8: E[T] for channel estimation for a=1 & a=3

Figure 4.9 shows that the variance of T approaches zero for increasing numbers of receivers. This result implies that $E[T]$ should be observed most of the time for a large number of receivers.

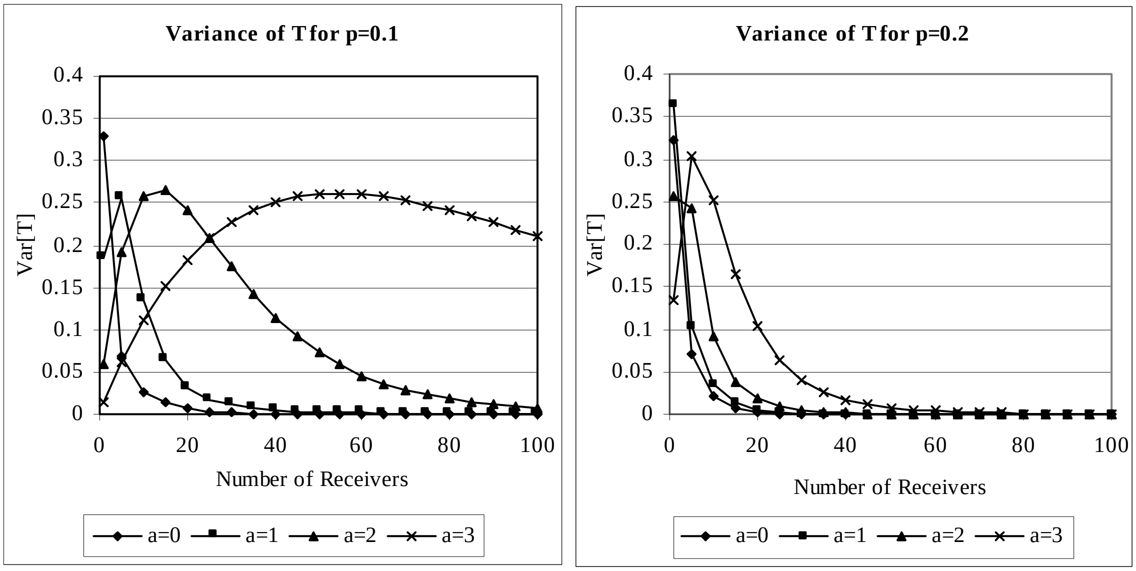

Figure 4.9: Var[T] for channel estimation w/ packet loss probability of 0.1 & 0.2

With the formula for $E[T]$ developed for the "aggregated" receiver approximation, the expected number of transmissions required to reliably deliver a TG under this approximation needs to be found. This analysis is essentially the same as the one presented in equations 39-45, but there are several necessary modifications. The expression for $E[N_a(j)]$ is found using the following equation,

$$E[N_a(j)] = \sum_{e=0}^{k+a} E[N_a(j) \mid E = e] \cdot P[E = e]. \tag{54}$$

Since the expression for $P[E = e]$ has been previously developed in equation 50, only the expression for $E[N_a(j) \mid E = e]$ needs to be found,

$$E[N_a(j) \mid E = e] = \sum_{m=0}^{\infty} m \cdot P[N_a(j) = m \mid E = e]. \tag{55}$$

Let $N_a(i)$ equal the number of parity packets (including insurance packets) sent during round i,

$$N_a(i) = \left\lceil \left(\frac{1}{1-p'}\right) S(i) \right\rceil. \tag{56}$$

As was shown for equation 45, equation 57 can be used in equation 55 to compute the conditional expectation,

$$P[N_a(j) = m \mid E = e] = P[S(j) = m' \mid E = e] \qquad \text{where m'=} \lfloor (1\text{-}\mathrm{p'}) \cdot m \rfloor. \tag{57}$$

Using equations 54-57, the expected number of transmissions required to reliably deliver a transmission group of k packets can be found. As shown in Figures 4.10, as the probability of error increases, then the number of packets increases. However, this approximation results in averages that are consistently smaller than those obtained in the non-channel estimation case. As was previously discussed, the actual number of packets transmitted in the approximation is (for most cases) less than the actual number transmitted. The nature of the approximation used to produce these graphs leads one to believe that these are lower bounds to the actual number of packet transmitted. The validity of this statement is shown later in the simulation results section.

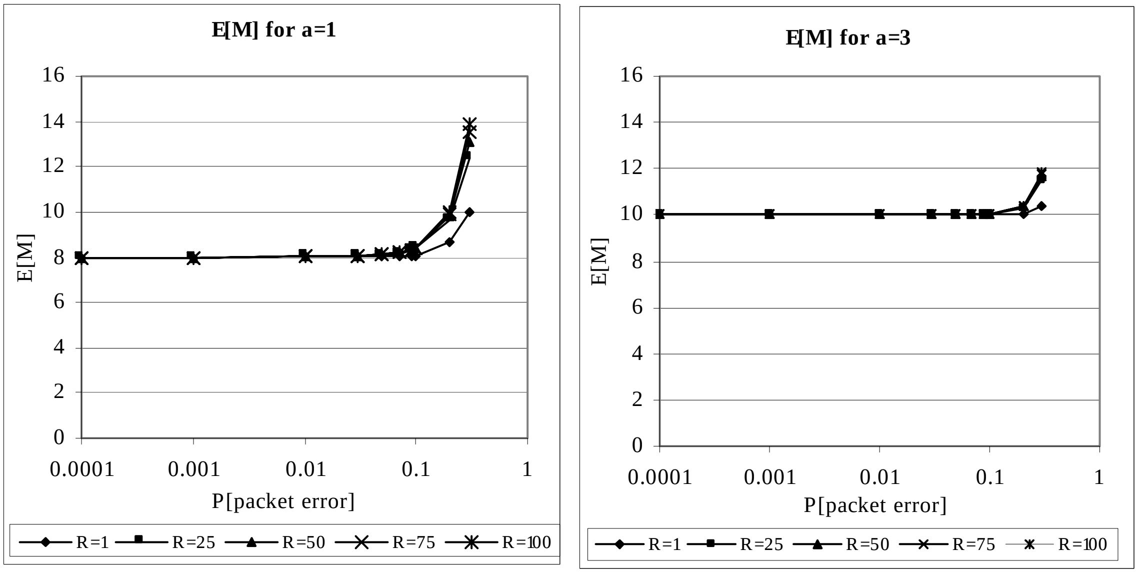

Figure 4.10: E[M] for a=1 & a=3: channel estimation

As shown in Figure 4.11, as the packet loss probability increases, the intrinsic overhead required for autoparity no longer becomes a cost but a benefit. However, since these graphs are based only on approximations, it is difficult to conclude that the use of autoparity actually reduces the number of transmissions for packet loss probabilities greater than 0.1. Further study of this phenomenon is needed prior to making any conclusive statements. However, one possible explanation is that by using more autoparity, the source is able to obtain a better estimate of the packet loss probability.

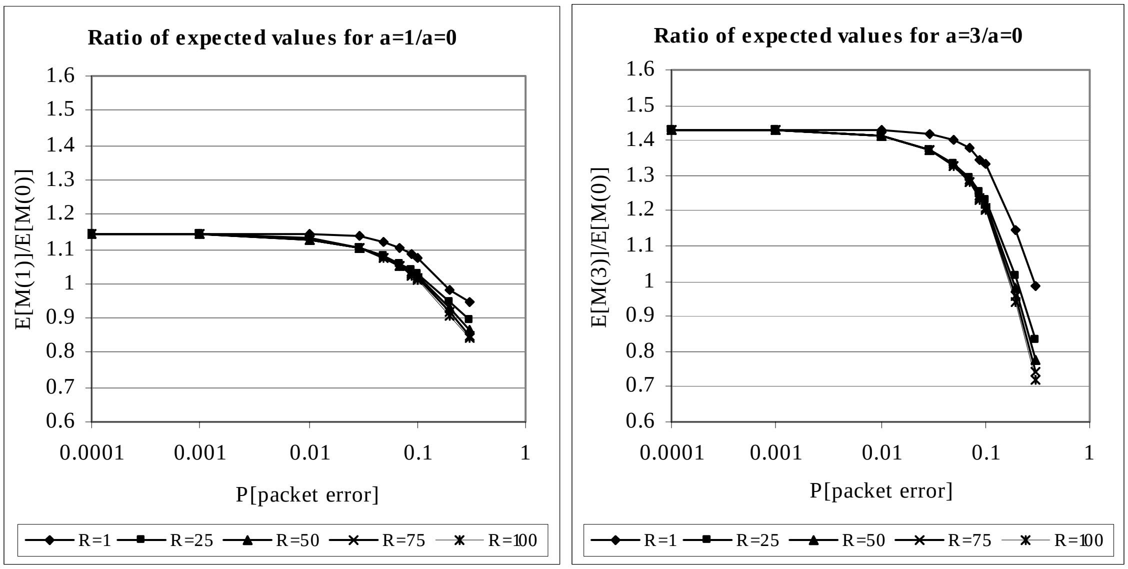

Figure 4.11: Expected value ratios (a=1/a=0) & (a=3/a=0): channel estimation

4.2. Simulation Results

In addition to the analytical work completed in the previous section, simulations were created so that results could be obtained in the cases where analytical results were not tractable and to serve for verification of the results that were obtained. Simulations were created for each of the four possible variations on the unconnected connectivity cluster scenario; infinite parity without channel estimation considerations, infinite parity with channel estimation considerations, finite parity without channel estimation considerations, finite parity without channel estimation considerations.

In the finite parity cases, the sender multicasts the maximum number of requested parity packets from all receivers until all parity packets associated with the TG have been used. Any packets requiring retransmission are placed into a new transmission group. When simulating the finite parity case with channel estimation, the number of “insurance” packets generated is calculated after the initial transmission round of each transmission group. The finite parity simulations are run for different amounts of generated parity packets (specifically, four and seven). For the remainder of this section, the simulation results are compared with the analytical ones. Additionally, the simulation results from the finite parity and infinite parity cases are compared.

4.2.1 Infinite Parity Case without Channel Estimation

The first comparison is made between the analytical results obtained in Section 4.1 and the results of the simulation. The expected number of transmission rounds as determined through the analysis were said to be upper bounds on the actual value, since the aggregated receiver would receive fewer packets than the number of packets actually sent. As can be seen by comparing the simluation results shown in Figure 4.12 and Figure 4.13 with Figure 4.1 and Figure 4.2, the analytical results actually upper bound the ones obtained via simulation. Although, the tightness of the bound has not been extensively studied, one can easily see by comparing these two sets of graphs that the lower the probability of error, the tighter the bound. Additionally, one notices that the upper bound improves when larger number of autoparity packets are used. In both instances (i.e. lower packet loss probability, and more autoparity packets), the number of outstanding packets is relatively small leading to a tighter bound. As the number of outstanding packets decreases, the difference between the actual and “aggregated” packets transmitted decreases.

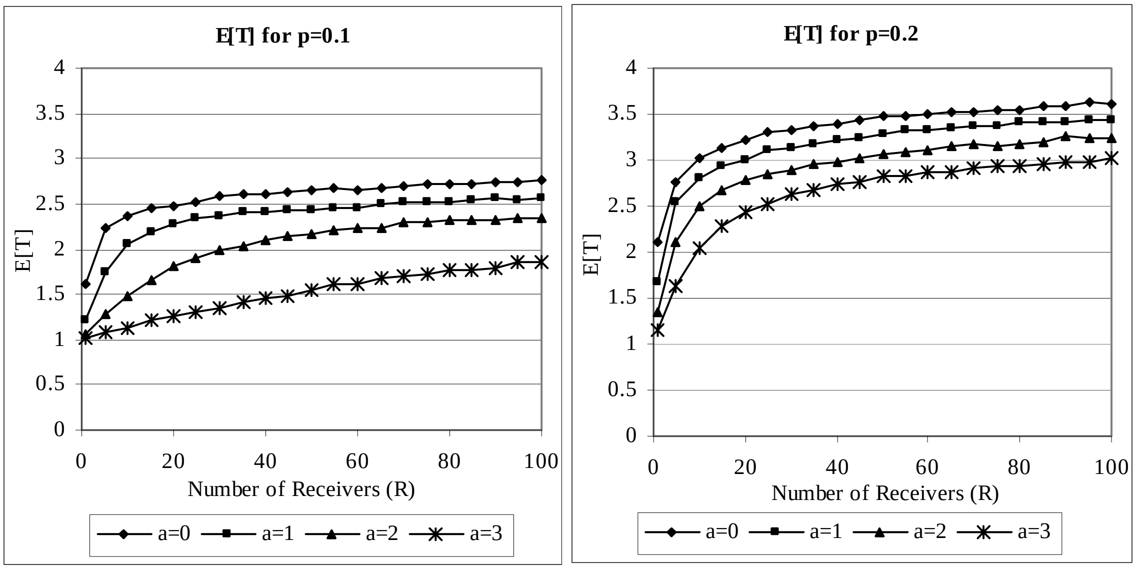

Figure 4.12: Simulation E[T] with packet loss probability of 0.1 & 0.2

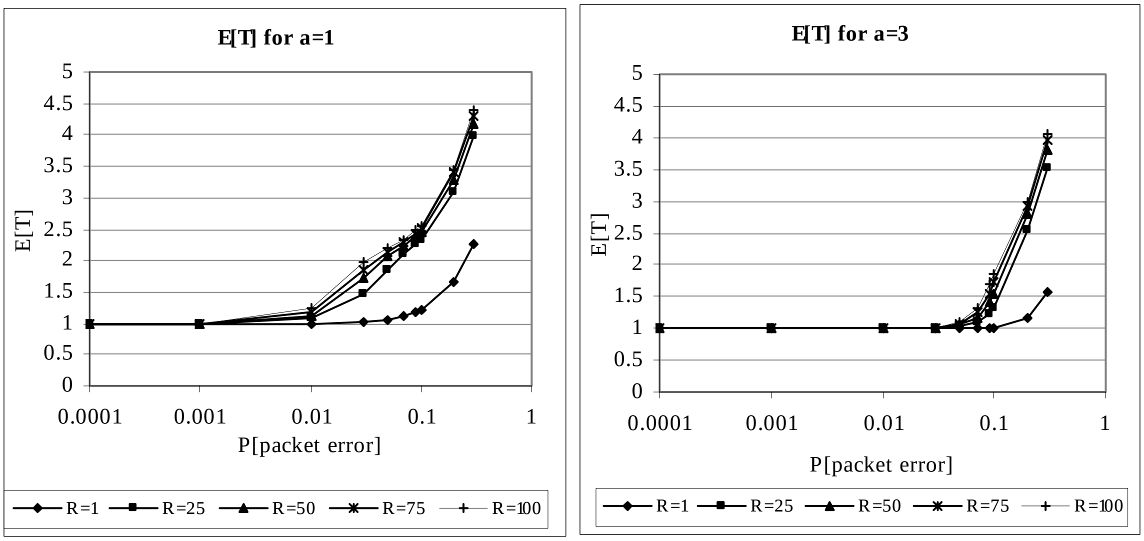

Figure 4.13: Simulation E[T] with a=1 & a=3

The expected value of the number of transmitted packets from the simulation and the analytical results are essentially the same as seen by comparing Figure 4.14 and 4.5. Therefore, the same statement may be made about their corresponding ratios.

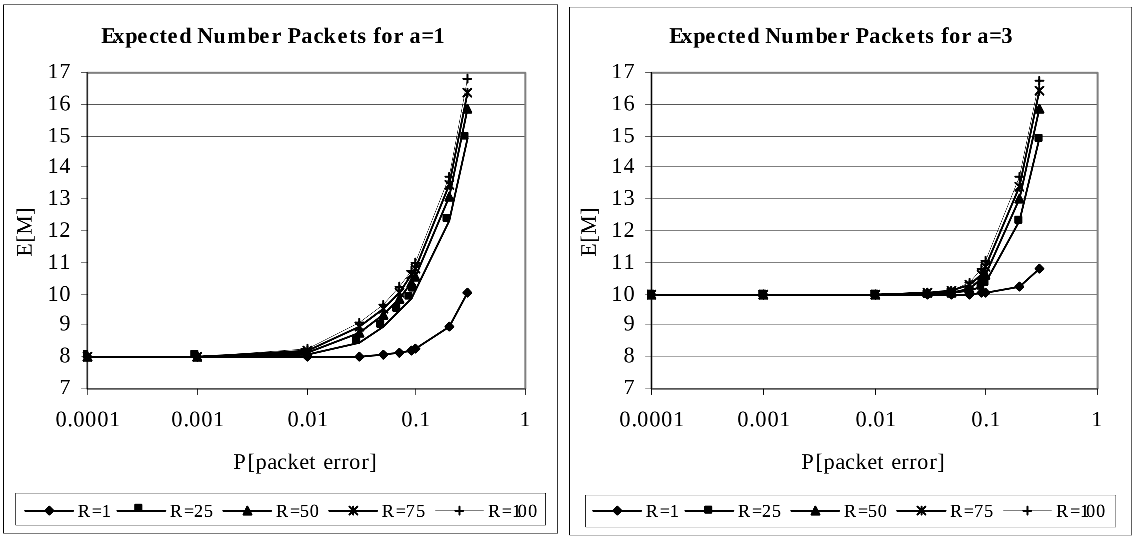

Figure 4.14: Expected Number of Packets for a=1 & a=3

4.2.2 Infinite Parity Case with Channel Estimation

From the approximation results presented in the previous section, one notices that using channel estimation (with infinite parity) can significantly reduce the number of transmission rounds. With an infinite amount of parity generated per TG, the delay can be reduced to two rounds using the channel estimation scheme. By comparing Figures 4.15 and 4.16 with their counterparts in section 4.1.2 (Figures 4.7 and 4.8), one observes that the expected numbers of transmission results obtained from the simulation agree with the ones obtained using the approximation. Additionally, when the amount of autoparity is increased, the simulations behave similarly to the approximations with regards to the expected number of retransmission rounds.

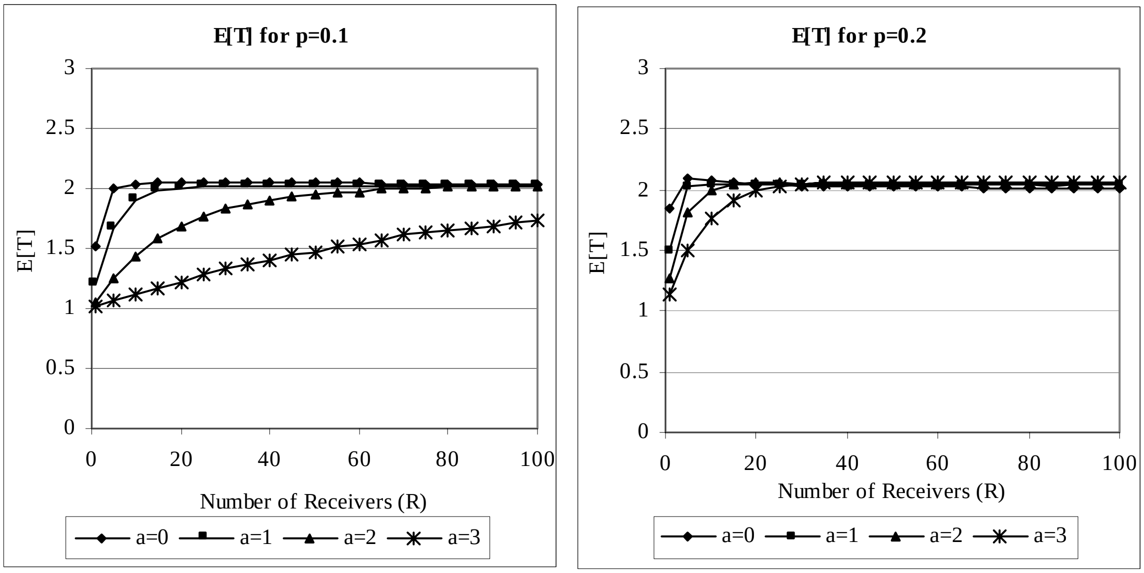

Figure 4.15: Expected Number of Transmission Rounds for p=0.1 & p=0.2

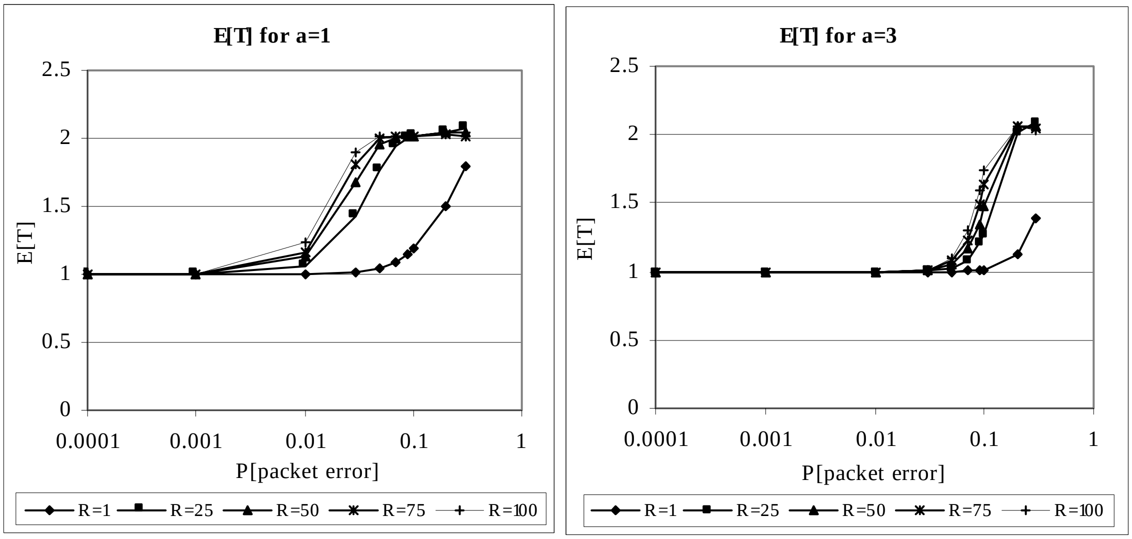

Figure 4.16: Expected Number of Transmission Rounds for a=1 & a=3

Regarding the expected number of packets required to reliably deliver the TG to a set of R receivers, the simulation results and those obtained via the approximation do not agree. By comparing Figure 4.17 with its approximation counterpart (Figure 4.10), one observes that the actual number of packets used is greater than the number obtained in the approximation. This result agrees with the intuition that the aggregated case results in fewer numbers of retransmitted packets. In fact, the number of actual packets used is approximately twice the number of packets sent to the "aggregated" receiver. This result shows that the approximation does not yield good results when considering the expected number of packets. However, this result suggests that there is a better way of performing channel estimation so that the scheme uses fewer packets while retaining the advantage of limiting $E[T]$. Another notable observation from Figure 4.17 is that for higher packet loss probability, the use of more autoparity packets actually reduces the number of packets needed for reliable delivery of a TG.

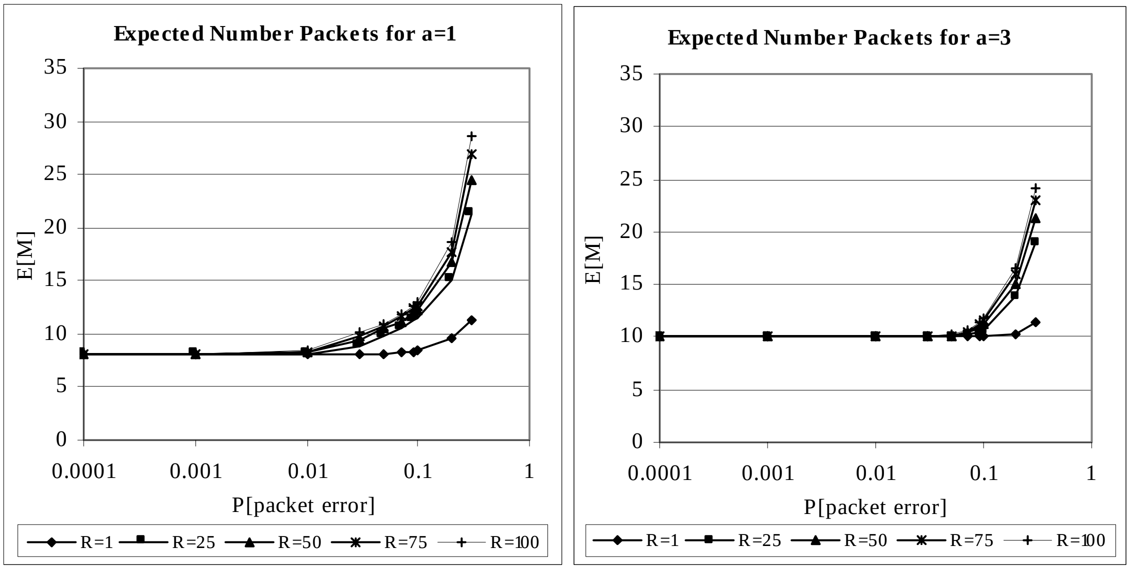

Figure 4.17: Expected Number of Packets for a=1 & a=3

4.2.3 Finite Parity Case without Channel Estimation

In previous analyses and simulations the assumption that an infinite amount of parity was generated per TG was made. In reality only a finite amount of parity can be generated per TG. The following section studies the effects of autoparity, packet loss probability, and number of receivers in the context of finite parity. Each set of simulations was run for two cases of generated parity, 4 packets and 7 packets.

4.2.3.1 Number of Parity Packets Generated = 4

As can be seen by comparing Figure 4.18 and Figure 4.12, using a finite number of parity packets negatively impacts the performance of the hybrid scheme when compared with the infinite parity case.

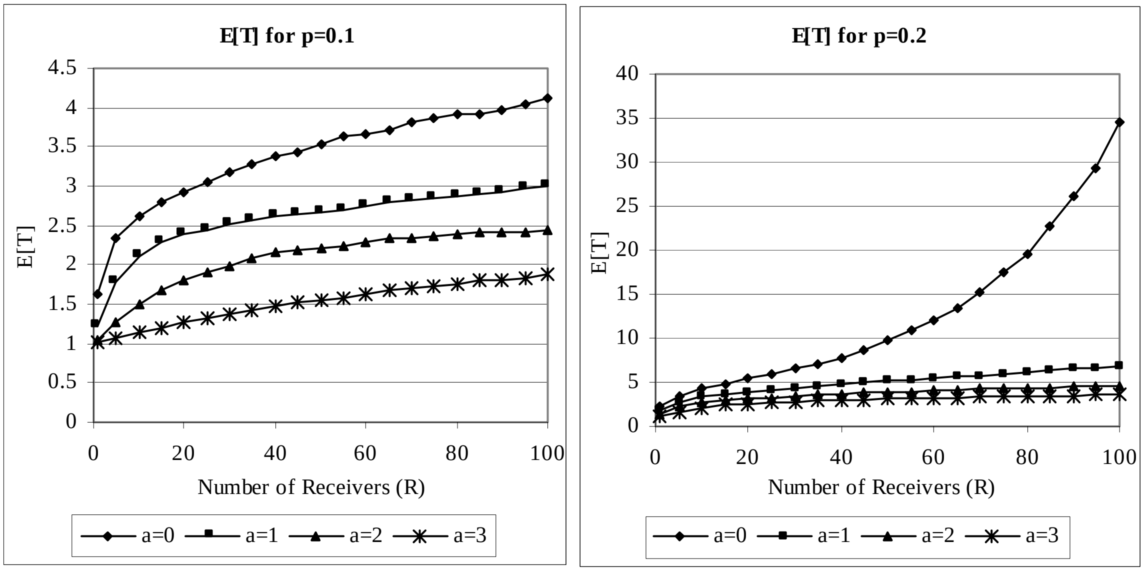

Figure 4.18: Expected Number of Transmission Rounds for p=0.1 & p=0.2

The ratio of the values plotted in Figure 4.18 and Figure 4.12 is shown in Figure 4.19. In the p=0.1 case, the use of substantial amounts of autoparity (e.g. a=2 and a=3) does not result in a significant increase compared to the infinite parity case. For higher levels of packet loss probability, the situation is much worse as shown in Figure 4.19-right. In this case, the usage of substantial autoparity does improve the overall transmission round performance of the protocol. Although, this improvement exists, the finite-4 parity case requires twice as many rounds as the infinite parity case. In Figure 4.19-right, one notices the drastically high ratio in the *a*=0 case. To avoid waiting for the 35 transmission rounds to complete in the *a*=0, *p*=0.2 case, it becomes necessary to use some amount (or be able to accurately predict the amount) of autoparity. One simple solution is to send all parity packets during the initial transmission round. This recommendation would effectively transform the integrated HEC protocol into its layered counterpart.

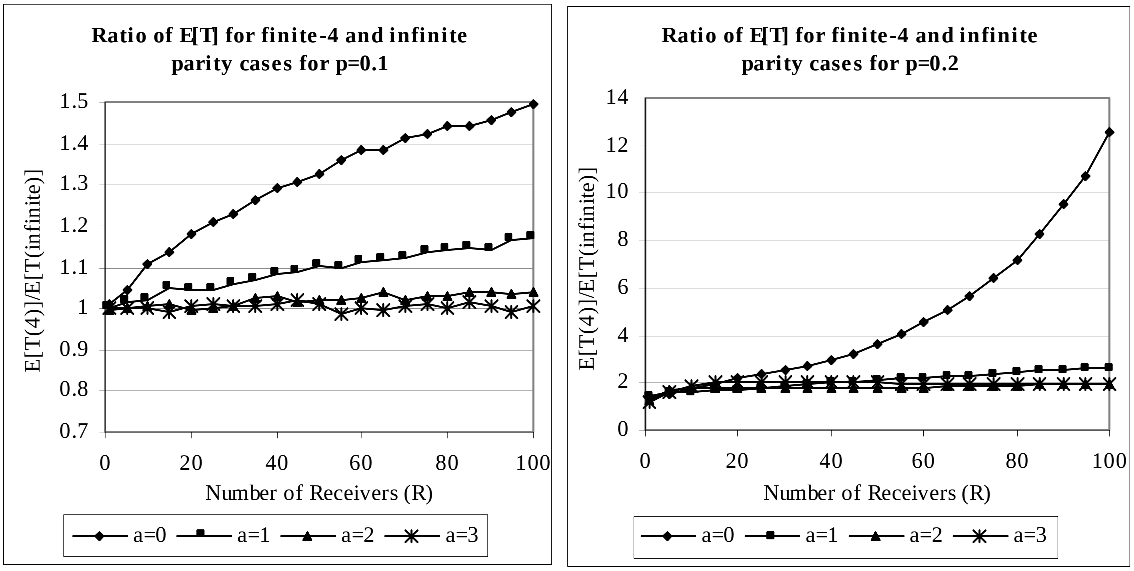

Figure 4.19: E[T] Ratio for finite-4 & infinite parity cases for p=0.1 & p=0.2

A more detailed quantification of the results between the infinite parity and the finite-4 parity cases is possible using the graphs presented in Figure 4.20. When comparing the left and right graphs of Figure 4.20, one must first be careful to note that there is one fewer data point in the right graph. One can posit that for smaller packet loss probabilities, the increase resulting from finite parity is negligible (i.e. – the ratios such as those presented in Figure 4.19 tend to one). These two graphs show that this hypothesis is correct as the finite parity and infinite parity cases yield similar results for values of p less than 0.07. However, for larger values of p, $E[T]$ in the finite case begins to increase compared to the corresponding infinite case values. The difference is largest for high packet loss probabilities and a large number of receivers (e.g. 100). For example, for 100 receivers, one autoparity packet, and p=0.2, the $E[T]$ in the finite case (6.89) basically doubled from the infinite case value of 3.45.

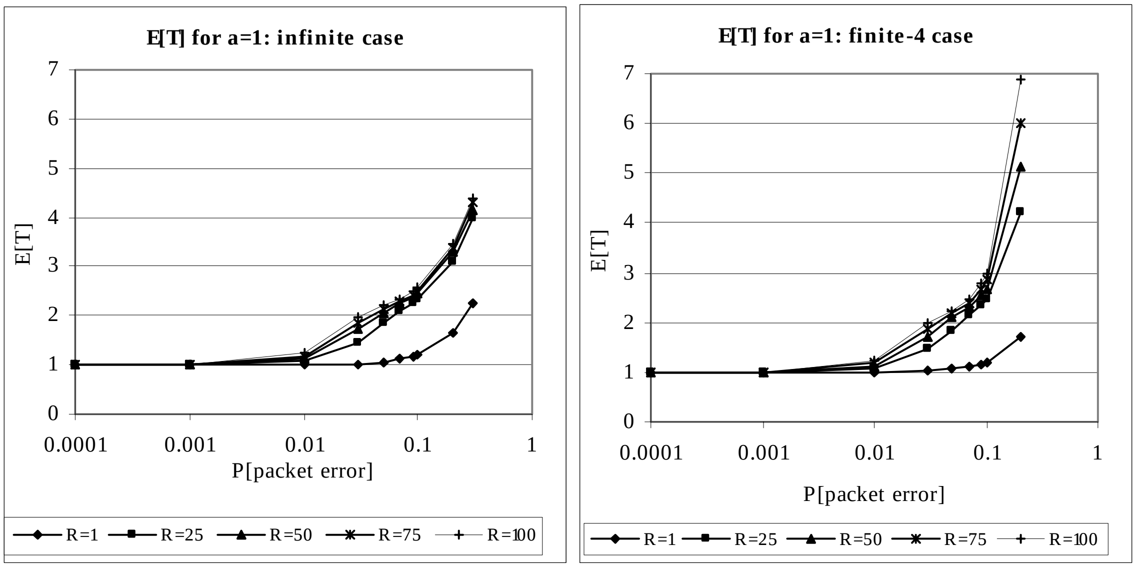

Figure 4.20: E[T] infinite & finite-4 cases for a=1

Since the number of transmission rounds increased, one also expects that the number of packets transmitted would increase. As seen from Figure 4.21, there is an increase in the number of packets required to reliably deliver the data to a set of receivers. Once again this increase is only noticeable for packet error probabilities larger than 0.1. For example, for 100 receivers, one autoparity packet, and p=0.2, the $E[M]$ in the finite case (28.2) is approximately 60% greater than the corresponding infinite case value of 17.8.

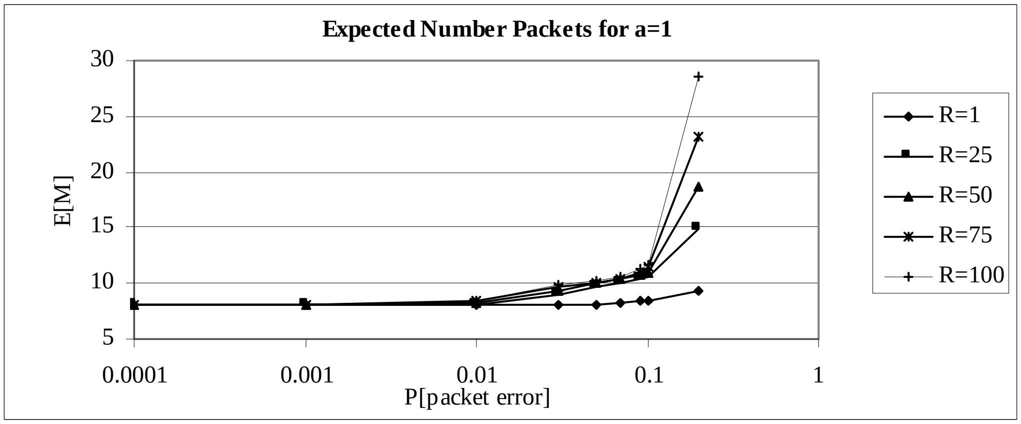

Figure 4.21: Expected Number of Packets: finite-4 case for a=1

The usage of autoparity also serves to decrease the number of transmitted packets for high packet loss probabilities. This result is shown in Figure 4.22. However, if one stresses bandwidth efficiency, then the use of additional autoparity packets may not be desirable for most packet loss probabilities. When considering the overall picture that includes both bandwidth and delay, one may decide that the cost of autoparity's extra overhead is worth the benefit of reduced delay.

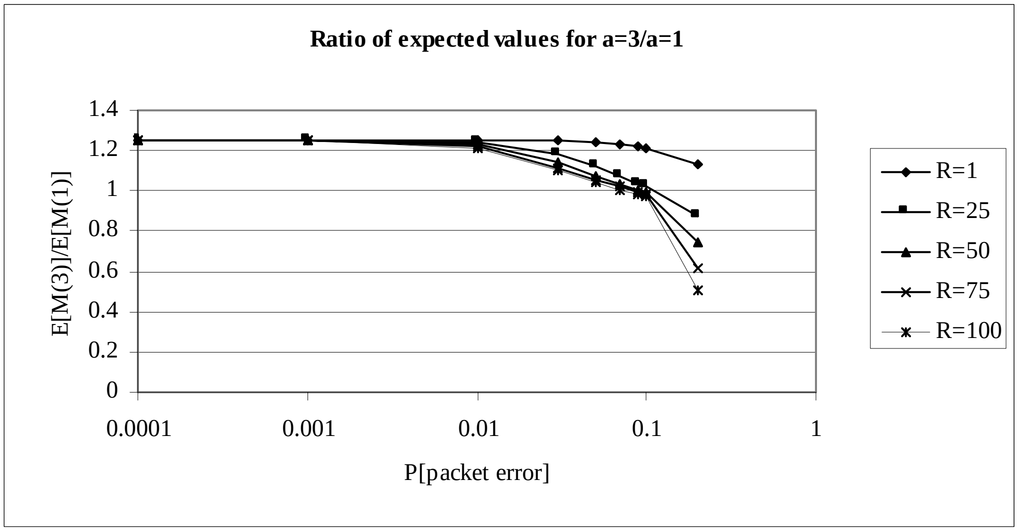

Figure 4.22: Ratio of expected values for a=3/a=1

4.2.3.2 Number of Parity Packets Generated = 7

For larger numbers of parity packets generated for each TG, the behavior of the protocol approaches the infinite parity case. The expected number of retransmission rounds is shown in Figure 4.23. As can be seen in Figure 4.24, the ratio between the expected number of retransmission rounds for the finite-7 parity case and the infinite parity case is much lower than the corresponding curves for the finite-4 parity case shown in Figure 4.19. For p=0.1, the ratio hovers around one for most of the data points shown. For p=0.2, the ratio is greater but still less than two for all receivers. With the additional generated parity packets, the source has a greater opportunity to meet parity request without generating new parity packets for another transmission group.

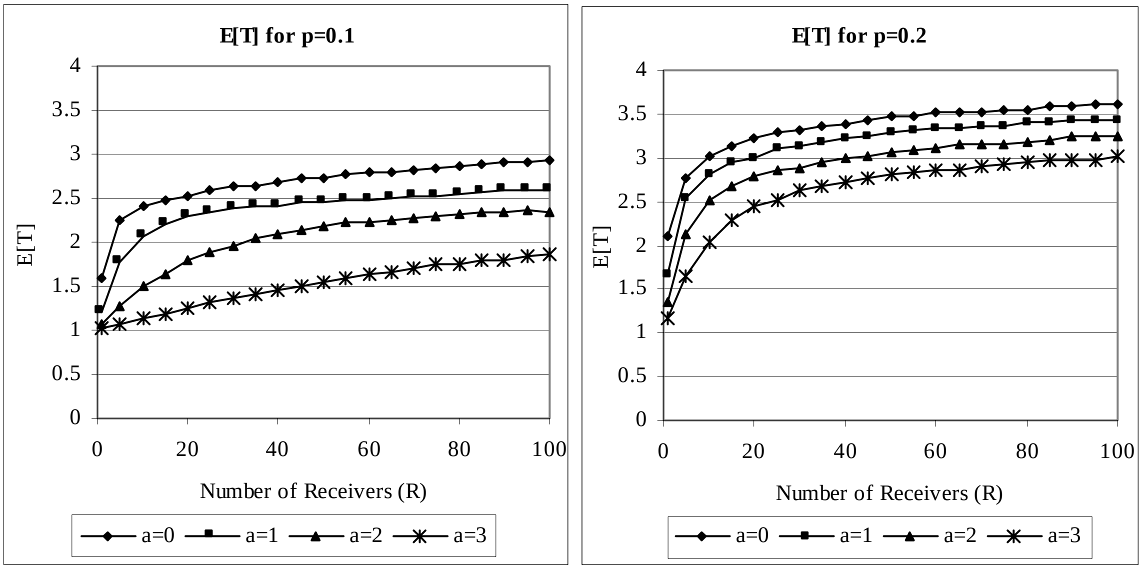

Figure 4.23: Expected Number of Transmission Rounds for p=0.1 & p=0.2

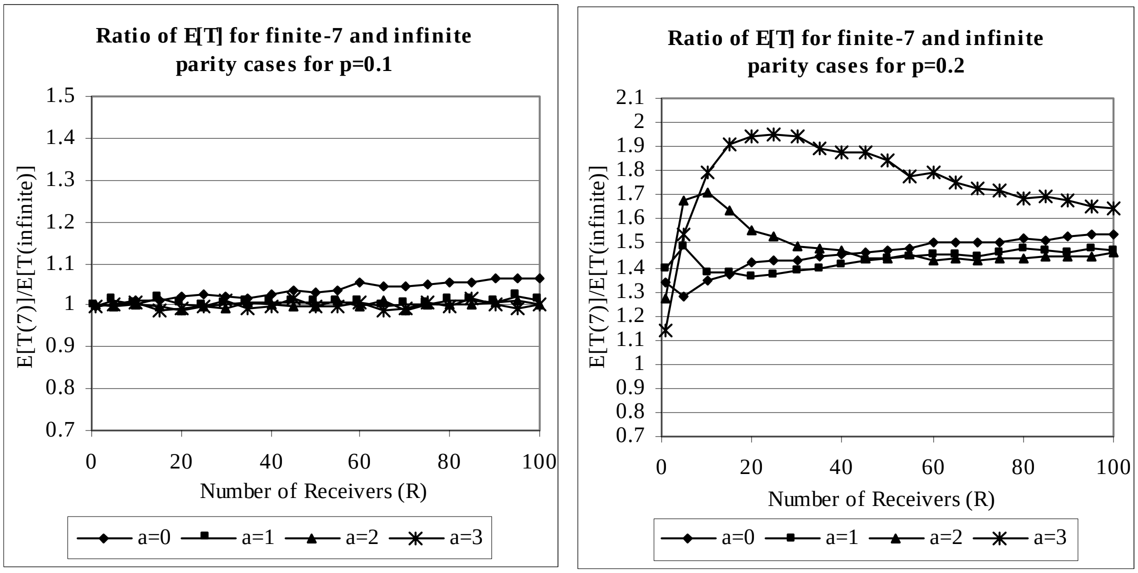

Figure 4.24: E[T] Ratio for finite-7 & infinite parity cases for p=0.1 & p=0.2

Additionally, the problem observed in Figures 4.18 and 4.19 no longer occurs when seven parity packets are generated per TG. Since the explosion in the number of transmission rounds and the corresponding increase in number of packets transmitted for the a=0 case does not occur, the ratio of expected values more closely resemble the theoretical infinite parity ratios shown in Figure 4.6.

4.2.4 Finite Parity Case with Channel Estimation

4.2.4.1 Number of Parity Packets Generated = 4

As seen in Figure 4.25 presented below, the used of channel estimation when only 4 parity packets are generated does not significantly improve the performance of the protocol over the non-channel estimate case shown in Figure 4.18. Since channel estimation requires the proactive transmission of parity during retransmission rounds, it normally requires the generation of a substantial number of parity packets. For example, when using three autoparity packets only one packet remains available for use in subsequent retransmission rounds.

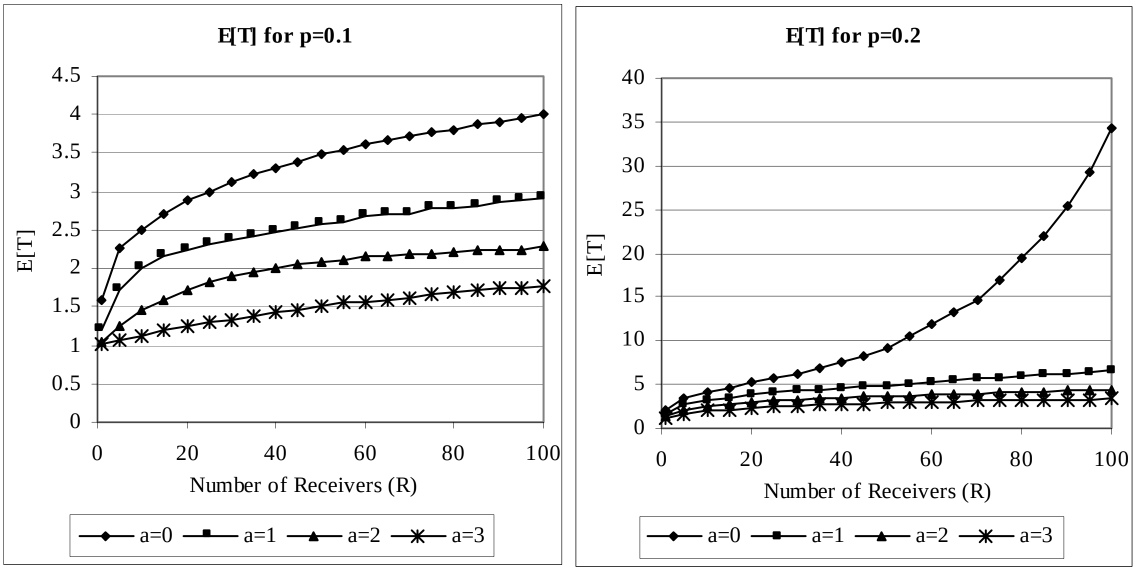

Figure 4.25: Expected Number of Transmission Rounds for p=0.1 & p=0.2

4.2.4.2 Number of Parity Packets Generated = 7

When larger amounts of parity packets are generated, channel estimation has a more pronounced effect. Since there are more parity packets per transmission group, the source can use channel estimation to its advantage and proactively transmit parity packets and keep enough in reserve for subsequent retransmission rounds. As seen in Figure 4.26 the use of channel estimation and one autoparity packet performs in a similar manner to the infinite parity case by delivering the TG in a maximum of approximately two rounds (see Figure 4.16). Additionally, Figure 4.27 shows the ratio of the expected values between the *a*=0 case and the *a*=3 case. As seen in this figure, the overhead associated with the *a*=3 case becomes negligible as the probability of error increases.

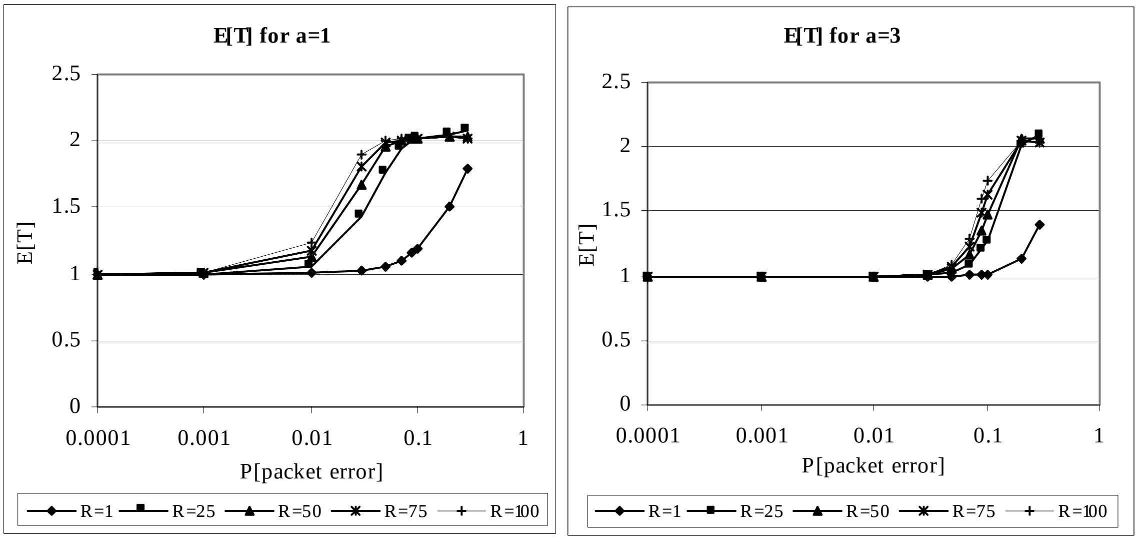

Figure 4.26: E[T] for a=1 & a=3

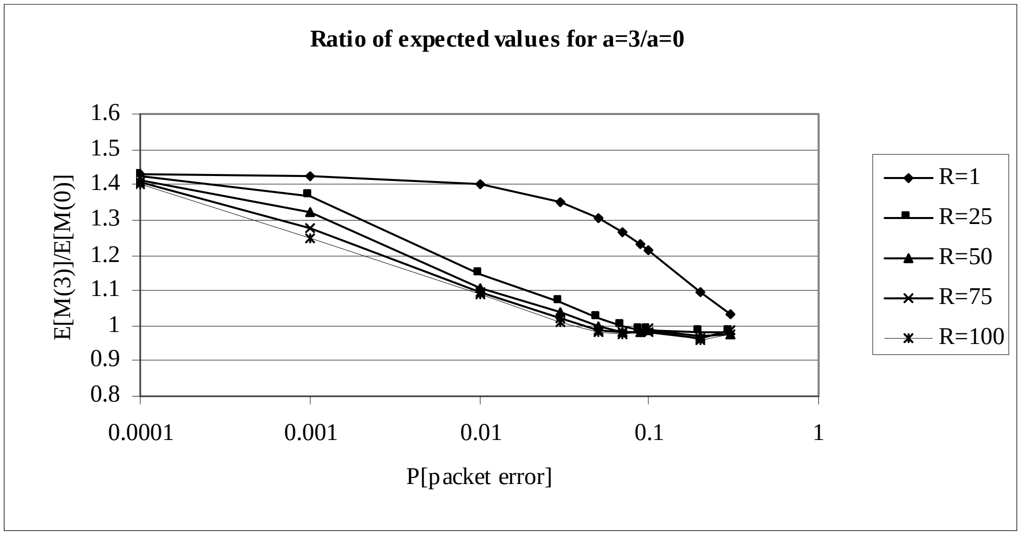

Figure 4.27: Ratio of expected values for a=3/a=0

4.3 Potential Adaptive Schemes

In Sections 4.1 and 4.2, the unconnected scenario was extensively studied and the results were presented. These results indicated that the use of autoparity reduces the number of transmissions needed for reliable delivery. This reduction coupled with the result that under certain network conditions, autoparity's overhead does not drastically increase the bandwidth usage suggests the intelligent use of autoparity in reliable multicast protocols. In the previous studies, the amount of autoparity was parameterized and remained constant during individual simulation runs. However, one can hypothesize that the use of adaptive, sender-based schemes able to react to dynamic network conditions will increase the efficiency (in terms of both bandwidth and delay) of reliable multicast protocols.

4.3.1 Dynamic Autoparity

One possible scheme dynamically changes the amount of autoparity transmitted per TG based upon past observations of the packet loss probability. For the remainder of this discussion, two assumptions are made. First, a sufficient amount of interleaving and enough bit-level encoding/decoding is assumed so that the bursty nature of bit errors does not result in bursty packet errors. Secondly, the occurrence of packet errors is assumed to be independent events occurring with a relatively constant frequency. With the above assumptions, the determination of the number of transmitted autoparity packets could, for example, be made using a moving average mechanism. Using a moving average mechanism requires that the last *x* values of the observed packet loss probability are stored by the source. These stored values, $p(i)$, are used to determine the amount of autoparity sent for the current transmission group. The actual values for $p(i)$ are determined based upon the total number of packets lost at the receivers during the initial transmission round of TG *i* and the total number of potentially correct packets that can be received by the entire group (e.g. $(k+a)*R$). These values are determined as shown in equation 58.

$$p(i) = \frac{\text{total \# of packets lost at receivers during initial round of TG i}}{\text{total \# of potentially correct packets for initial round of TG i}} \qquad (58)$$

In the example presented in Table 4.1, the value for $p(i)$ equals $15/35$, or approximately 0.43. An estimate of the packet loss probability, $\hat{p}$, is obtained using the values for $p(i)$ in the following equation.

$$\hat{p} = \sum_{i=1}^{x} \alpha_i p(i) \tag{59}$$

Since packet errors were assumed to occur independently, the values of all weights, α_i, should equal $1/x$. The accuracy of this estimate depends upon the actual characteristics of the packet errors and "correctness" of the assumptions.

Using these *x* observations, the source obtains an estimate of the packet loss probability. With this estimate, the expected number of errors that will occur at each receiver can be determined. Since these errors have a binomial probability mass function, the expected number of errors equals the number of transmitted packets per TG (*k*) times $\hat{p}$ as shown in equation 60,

$$E[\#\text{ of errors at a particular receiver}] = k \cdot \hat{p}\,. \tag{60}$$

This expected value can then be used by the source to determine the amount of transmitted autoparity. This determination could be made upon a complex set of rules and cost functions involving both delay and bandwidth, or could be made upon a simple rule such as the first integer greater than the calculated expected value. Using this simple rule, a simulation was created to gain a basic understanding of this dynamic autoparity scheme.

The simulation was run for two transmission group sizes (k=7, and k=20). The results from the simulations for k=7, k=20 are shown in Figures 4.28 and 4.29, respectively. Comparing Figure 4.28 with the simulations results for k=7 shown in

Figure 4.13 and 4.14, one notices that the dynamic autoparity scheme offers some performance improvements over the static autoparity schemes. However, depending on the amount of static parity used, this scheme may not perform as well as its static autoparity counterpart. Assuming the estimates obtain by the moving average mechanism are relatively close to the actual probabilities of packet loss, the E[T] performance of the k=7, dynamic autoparity case mirrored the E[T] performance for k=7, a=1 for $p \le 0.1$. For higher probabilities of error, the performance was more similar to k=7, a=3 case. These performance comparisons agree with the actual amount of autoparity generated by the above scheme; $\lceil 7*0.1 \rceil = 1$ and $\lceil 7*0.3 \rceil = 3$.

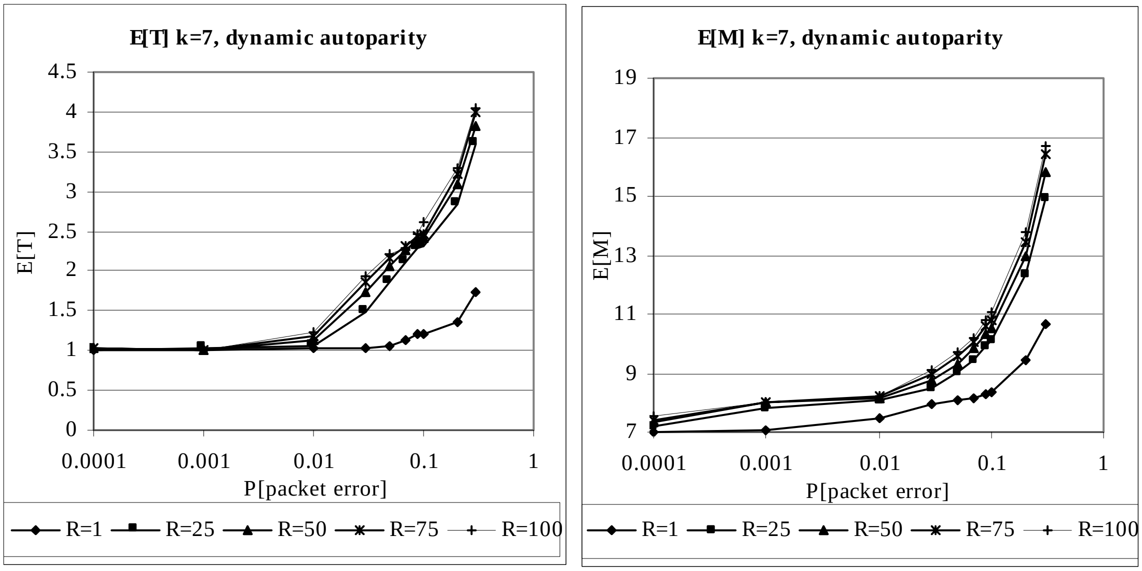

Figure 4.28: E[T] & E[M] for k=7, dynamic autoparity

The performance of the dynamic autoparity case does not outperform the a=3 case over the entire range of packet loss probabilities, it offers improved bandwidth efficiency at lower packet loss probabilities without drastically increasing the expected number of retransmission rounds. Though, this solution does not offer the best delay performance, it may yield the best overall solution if both bandwidth efficiency and delay are equally important.

Figure 4.29 shows that the previously described shortened RS(40,20) code, on average, generates enough parity packets to fulfill the repair requests. The same statement cannot be made for a similar shortened RS(14,7) code used in the k=7 case. Although not studied in this thesis, there appears to be a relationship between number of required parity packets and the size of the TG. This relationship needs to be further studied so that the increased encoding/decoding times are properly compared with the increased delay and bandwidth usage of finite parity cases.

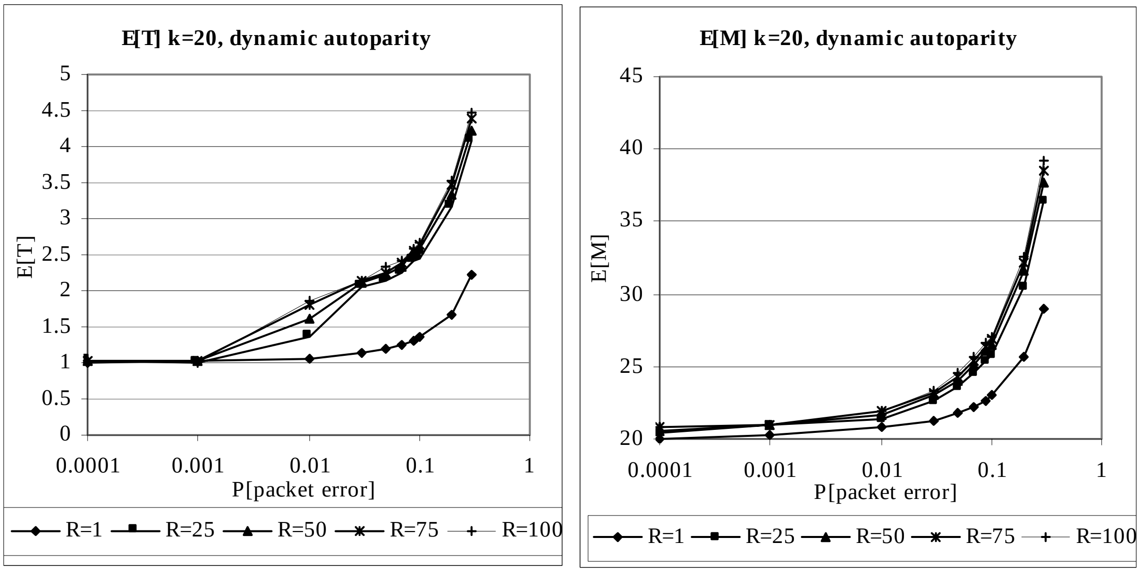

Figure 4.29: E[T] & E[M] for k=20, dynamic autoparity

4.3.2 Moving Average Channel Estimation Technique

Additionally, the channel estimation scheme previously described (see Section 4.1.2) can be modified so that the number of "insurance" packets transmitted is calculated in a manner than results in the more efficient usage of bandwidth. In the original scheme, the maximal packet loss probability was used to determine the number of "insurance" packets transmitted. However, the results obtained in Section 4.1.2 & Section 4.2.2 suggest that there may be a more efficient way to determine the number of additional packets transmitted during the retransmission rounds.

One possible improvement on the maximal packet loss probability scheme (defined in Section 4.1.2) is very similar to the aforementioned moving average scheme. As feedback is gathered immediately following the initial retransmission round, it can be used in the calculation of the estimated packet loss probability. The inclusion of this feedback from the current TG was not possible in the estimation of autoparity because the autoparity estimate is made prior to the initial transmission. Although, the particular observations used are different in the two cases, the values of the observations, $p(i)$, are determine as shown in previous equation 58. A composite estimate using the last x values is then computed using equation 59. With this estimate of the packet loss probability, the number of "insurance" packets sent in subsequent retransmission rounds is generated using equation 61. In this equation, j corresponds to the transmission round for a particular TG,

$$N(j) = \left\lceil \left(\frac{1}{1-\hat{p}} \right) Z(j) \right\rceil . \tag{61}$$

If both the dynamic autoparity and the moving average channel estimation techniques are used together, then some modification to the aforementioned channel estimation technique may be warranted. When two such schemes are simultaneously used, the source needs to make two estimates within a relatively short period of time. Depending upon the processing restrictions at the source, the second estimate calculated to determine the number of "insurance" packets transmitted during the retransmission rounds might not be necessary. In its place, the first estimate used during the dynamic autoparity calculation is employed to determine the number of “insurance” packets to be transmitted. Although, this first estimate does not use the most current feedback, it may still provide a reasonable estimate of the packet loss probability.

This moving average technique (without the most recent estimate) was studied using simulations. These simulations were run for transmission group sizes of 7 and 20. As shown in Figure 4.30, this scheme keeps the expected number of transmissions less than 2 up to p=0.1. Over this particular range of probabilities, the moving average scheme functions as well as the constant a=1 case. For higher packet loss probabilities, the moving average scheme does not perform as well as the previously described schemes using the maximal probability of error. For these higher packet loss probability, the moving average technique does respectively reduce the expected number of transmitted packets by approximately 30% and 40% for the a=1 and a=3 cases. This reduction occurs with at the cost of a 25% increase of the expected number of transmission rounds.

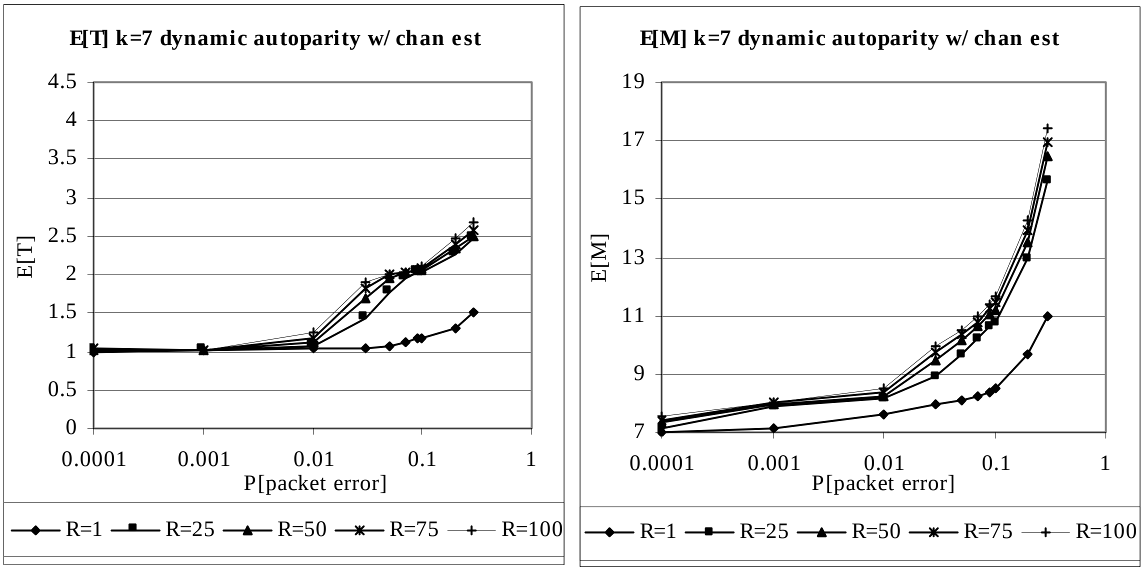

Figure 4.30: E[T] & E[M] for k=7, dynamic autoparity w/ channel estimation

The most important observation can be made between the dynamic autoparity case and the dynamic autoparity case with channel estimation. By comparing Figures 4.30 and 4.31 with Figures 4.28 and 4.29, one notices the expected number of retransmission rounds decreases by approximately 33 % in both the k=7 and k=20 cases. This performance gain of the additional use of channel estimation is realized with a negligible increase in the number of transmitted packets.

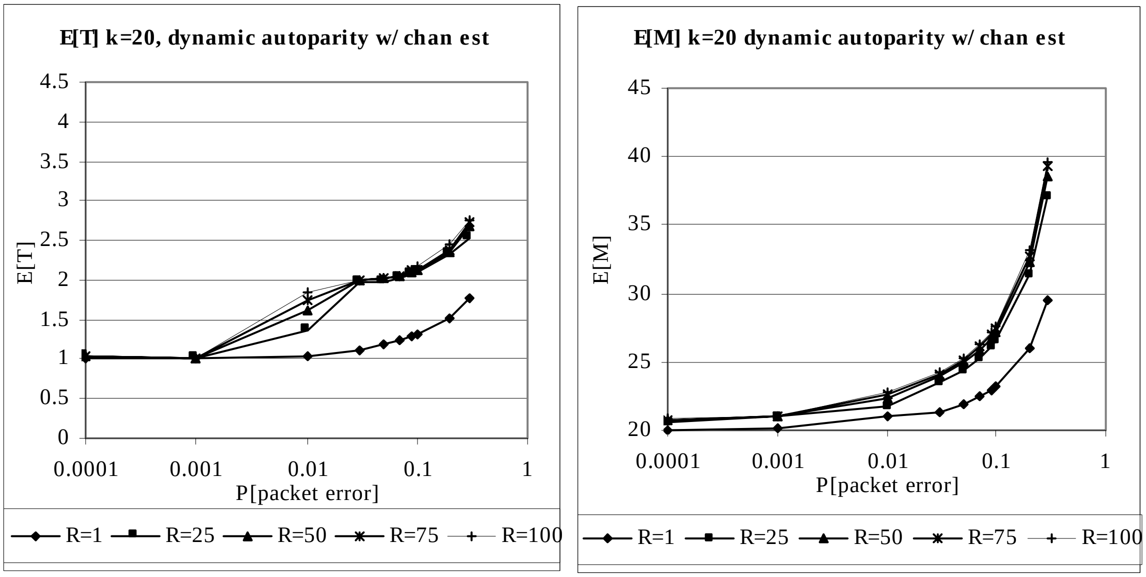

Figure 4.31: E[T] & E[M] for k=20, dynamic autoparity w/ channel estimation

5. Connected Cluster Scenario

5.1 Introduction to Simulations

In the connected cluster scenario, several simulations were created so that the benefits of local recovery could be quantified. As mentioned in Chapter 3, the same basic protocol from the unconnected scenario is used to transmit TGs from the source to the PRs. A PR that has not correctly received the TG alternates between local and satellite retransmission rounds as determined by the *LxS* ratio. When considering the connected scenario, the use of parity becomes more complicated. The complication arises from the fact that more information is needed to send the correct local repair packets. Due to the added complexity of this situation, more complex feedback messages and more complex processing at the PRs is required.

One possible option is that local nodes that have received the entire TG create their own parity packets to be sent over the local recovery multicast tree. These nodes would send these "new" parity packets when fulfilling a local repair request. Unfortunately, these "new" parity packets could potentially be useless as repair packets. Such a situation arises when, for example, a receiver informs the local group that it needs three packets to complete a transmission group of seven packets. However, of this receiver's four packets, three are source-generated parity packets and one is an original data packet. The mixing of the three locally received, "new" parity packets and the three source-generated parity packets may not enable the receiver to correctly decode its packets to obtain the original TG. Another potential solution is that correctly received global parity packets are sent during local retransmission rounds. However, globally created parity packets have lessened repair capabilities during the local recovery cycle. This lessening results from the possibility that some receivers may have already received the parity packet during the previous global transmission rounds. Therefore, during the local recovery phase, global parity is assumed to have the same repair capabilities as the original data packets. Due to these complications, receivers sending repair requests should transmit bit vectors that inform its neighbors of the packets it has correct received and those it currently needs to complete the TG. Using this information, receivers able to fulfill the request send the appropriate repair packets.

These parity problems can be overcome, if the assumption that the global and local parity is identically generated and there is a sufficiently large amount of this parity to meet all retransmission requests is made. Under such an assumption, local receivers can send fresh parity (i.e. – parity that has not yet been used by the source) and inform the source of the number of locally transmitted parity packets. Then the source can send the next available fresh parity packets. Although, the above solution can be easily studied, it fails to offer valuable insight into the subtleties and the important parameters of local recovery schemes

More realistic solutions can be categorized based upon their use of local and globally generated packets during the local recovery phase. In one solution, only data packets (those that are correctly received or successfully reconstructed) are transmitted during the local recovery phase. However, transmitting these data packets (either by unicasting or multicasting) could potentially result in local network congestion. Another method that was previously discussed allows only locally generated parity packets to be locally multicast during the local recovery phase. Such a solution requires the successful reception of the TG on which these local parity packets can be constructed. For local neighborhoods containing a larger number of receivers, the probability that at least one receiver correctly receives all data packets is relatively high. In this potential solution, the receivers need to differentiate between locally and globally received parity packets. The PRs requiring packets would have to request parity packets from their local neighbors based upon the number of correctly received data packets (not including global parity packets). The TG could be reconstructed if either the number of data packets and local recovery packets equals the TG size or the number of data packets and global recovery packets equals the TG size. A third possible solution allows for both original data and locally generated parity to be transmitted during the local phase. Although, the third solution is quite feasible, only the second option when locally generated parity is transmitted is studied in these simulations.

Any receiver's repair request only reaches a subset of the multicast group consisting of the PRs. The size of this subset depends upon the local retransmission timer. To avoid flooding the local network with repairs, the receivers use a repair suppression mechanism. Additionally, the multicast tree containing the PRs contributes to the local packet loss probability. Since there are a large number of potential topologies, the local packet loss probability needs to be abstracted to a higher level. The results presented in [20] show that full binary trees (FBT) are good generic models for the loss characteristics of real multicast trees. The simulations assume that the PRs are connected via a full binary tree and that local repair requests are error free. Depending upon the local recovery timer and the delay of each local hop, the simulation determines the nodes within each local neighborhood. The depth of the local multicast tree (or a subtree within the FBT) also affects the local packet loss probability. The local packet loss probability at each node depends upon the location of both the sender and receiver within the subtree. For example, a sender-receiver pair that are siblings (e.g. – the retransmission has to travel over two hops) has a lower packet loss probability than the sender-receiver pair that are cousins (e.g. – the retransmission has to travel over four hops). The simulation determines a local packet loss probability based only upon the depth of the subtree. It does not determine the packet loss probability at each receiver depending upon the PR sending the local repairs. Additionally, since local parities are used a repairs, the local packet loss probability is assumed to be independent amongst receivers.

In Chapter 4, it was shown that if enough parity is generated, the protocol functions relatively close to the infinite parity case. Therefore, the simulations assume that both the source and the PRs generate enough parity to fulfill their corresponding retransmission requests. Without this assumption, the local retransmission rounds become more complicated and require more state information to be stored by the PRs. Additionally, the feedback messages between the PRs have to contain more information. The protocol would also have to independently address the cases when the global parity packets and the local parity packets are exhausted. To gain a simple understanding and the potential benefits obtained by using local transmitted repair packets among PRs, these issues were not studied in the following simulations. The simulations studies can be divided into four different classes; 1) *1x1* local recovery without channel estimation over the satellite, 2) *1x1* recovery with channel estimation over the satellite, 3) *LxS* local recovery without channel estimation, and 4) *LxS* local recovery with adaptive channel estimation schemes (shown in section 4.3). In all cases, any channel estimation mechanism is only used over the satellite link and not within the local neighborhoods.

5.2 Infinite Parity without Channel Estimation: LxS ratio = 1x1

By comparing Figure 5.1, one observes the reduction in the number of satellite transmission rounds in this local recovery case compared with the corresponding non-local case. The three graphs (Figure 5.1-left, Figure 5.1-right, and Figure 5.2-left) show that the depth of the subtree plays an important role in reducing the expected number of satellite transmission rounds. In the worst case (i.e. p=0.3 and R=128), local recovery decreases the number of satellite retransmission rounds by 22% and 44% for subtree depths of 2 and 5, respectively. To gain a better understanding of the improvement over the entire range of packet loss probabilities, the R=128 data from each of the three simulation runs (no local, local(2) = local recovery with a subtree depth equal to two, and local(5) = local recovery with a subtree depth equal to five) is plotted in Figure 5.2-right. As can be seen from this graph, both local recovery schemes function better than the non-local recovery schemes regardless of the packet loss probability.

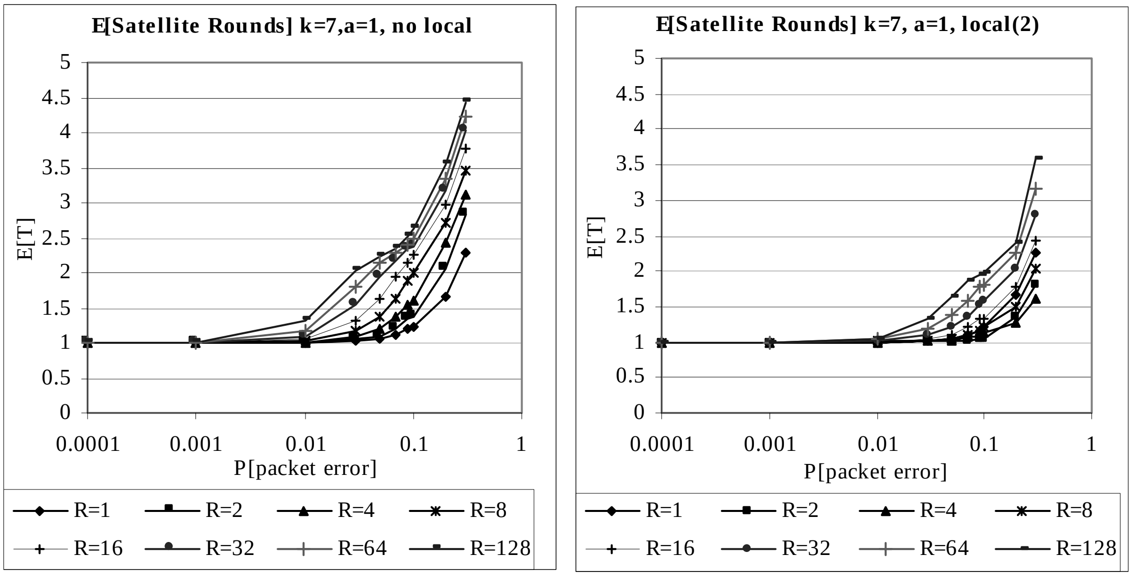

Figure 5.1: Satellite E[T] for k=7, a=1 for local(2) & no local

One important observation that can be drawn from Figure 5.2-right is that there is a packet loss probability threshold in which the local(5) scheme begins to outperform the local(2) scheme. Although, this threshold is approximately p=0.15, the local(5) and the local(2) cases performance is very similar for the packet loss probabilities immediately preceding the threshold. The performance crossover can be attributed to the fact that for larger packet loss probabilities, there is a higher chance that none of the 4 PRs in the local(2) subtree correctly received the entire TG. However, for the local(5) case, the probability of none of the 32 PRs receiving the entire TG is substantially lower. Therefore, local groups size combines with the satellite link’s packet loss probability to affect the performance of different local recovery schemes.

To gain a better understanding of the behavior of the local recovery schemes, the simulation was also run for different TG sizes (k=20, and k=100). Figure 5.3 shows that for k=20, the expected number of satellite rounds also decreases albeit not as much as in the k=7 case. Figure 5.3 also shows that the decrease does not preserve the relationship between different values of R. For example in Figure 5.3-right, the R=8 case outperforms the R=1 case.

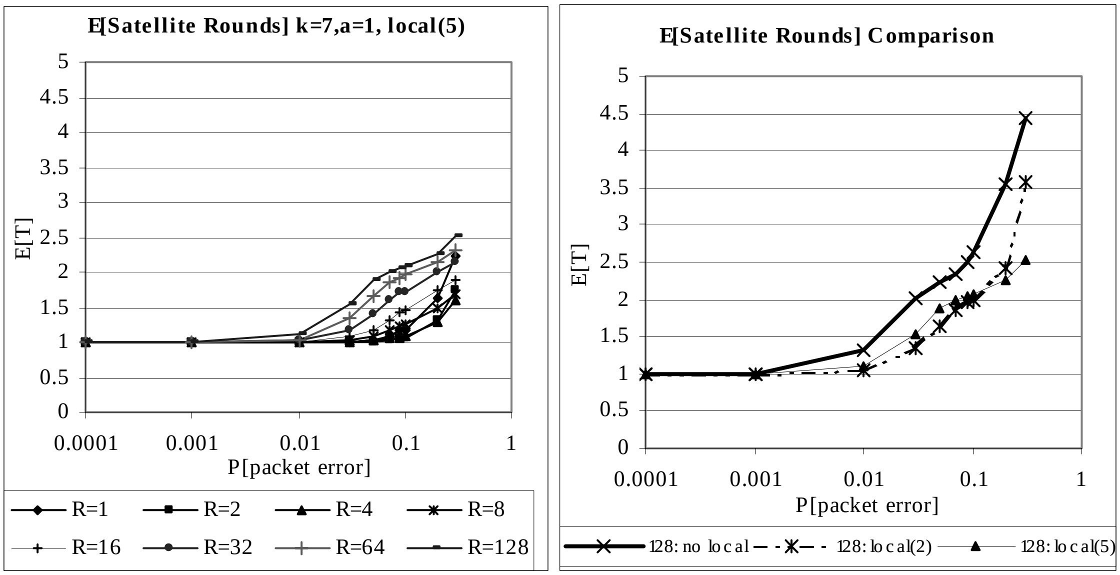

Figure 5.2: Satellite E[T] for k=7, a=1 local(2)&Satellite E[T] Comparison

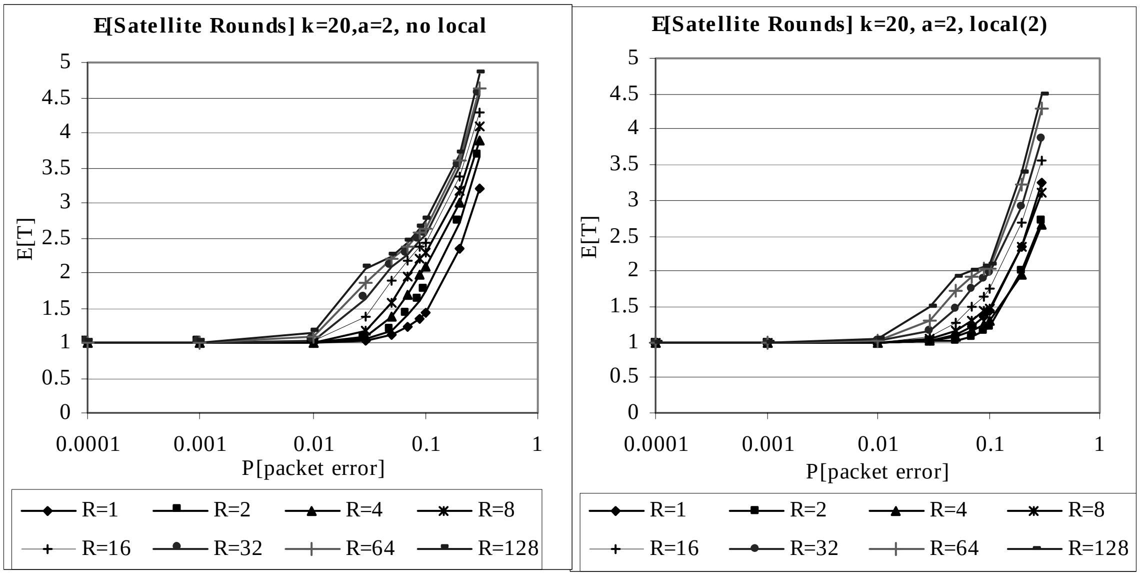

Figure 5.3: Satellite E[T] for k=20,a=2 for no local & local(2)

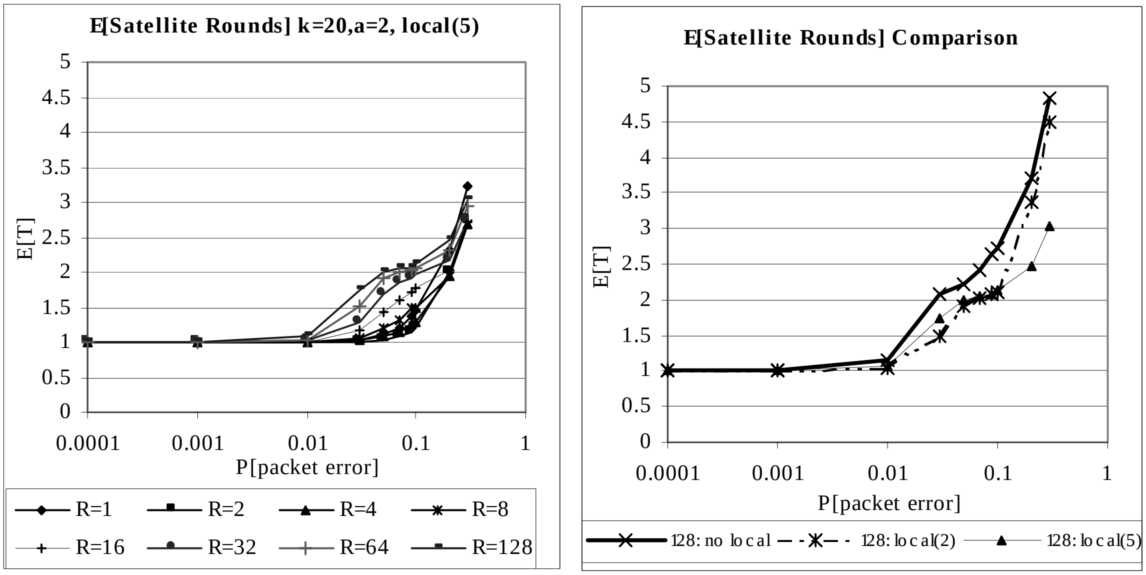

Figure 5.4: Satellite E[T] for k=20, a=2 local(5) & Satellite E[T] Comparison

This same phenomenon is also observed in Figure 5.4-left. This graph demonstrates that for higher values of packet loss probability ($p \geq 0.2$) all values of R outperform the R=1 case. By observing Figure 5.4-right, one notices a packet loss probability threshold that occurs at slightly lower packet loss probability (*p*=0.9 although not clear from the figure) than in the *k*=7 case. This graph also shows that the performance improvements gained in the *k*=20 case are smaller than those improvements shown for the *k*=7 case.

Figures 5.5 and 5.6 show the results for the case when *k*=100 and *a*=10. Ten autoparity packets were used so that in both the *k*=20 and *k*=100 cases, autoparity was 10% of the TG size. The use of ten parity packets serves to keep the expected number of satellite transmission rounds at one until approximately *p*=0.03 as compared to approximately *p*=0.01 in the k=20 case. The effect of using local recovery is apparent by comparing the two graphs in Figure 5.5. In the packet loss probability range between 0.03 and 0.1, the local(2) scheme enabled the protocol to reliably deliver the TG in under approximately 2.2 global retransmission rounds. This is an improvement of approximately 25 % over the non-local recovery case. For higher packet loss probability, the improvement diminishes as the number of packet losses over the satellite link is too great from the local(2) scheme to handle. In Figure 5.6-left, one notices the same type of behavior. However, for high *p* values and large values of R, the protocol actually outperforms the cases when R is smaller. By observing Figure 5.5-right and Figure 5.6 left, it appears that the best performance in high error environments occurs when the size of the TG is equal to the size of a local neighborhood (e.g. for R=4 in the local(2) case and for R=32 in the local(5) case). In such a situation, all PRs are contained within one local neighborhood. Finally, as shown in Figure 5.6-right, the packet loss probability threshold is lower than both of the preceding cases (*p*=0.7 although somewhat unclear from these figures).

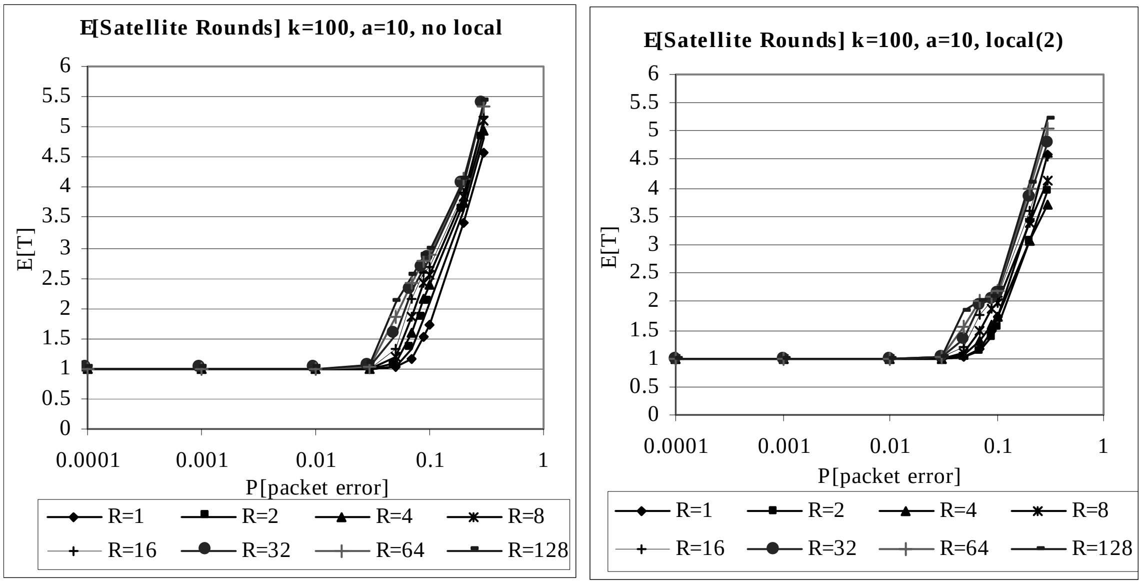

Figure 5.5: Satellite E[T] for k=100, a=10 for no local & local(2)

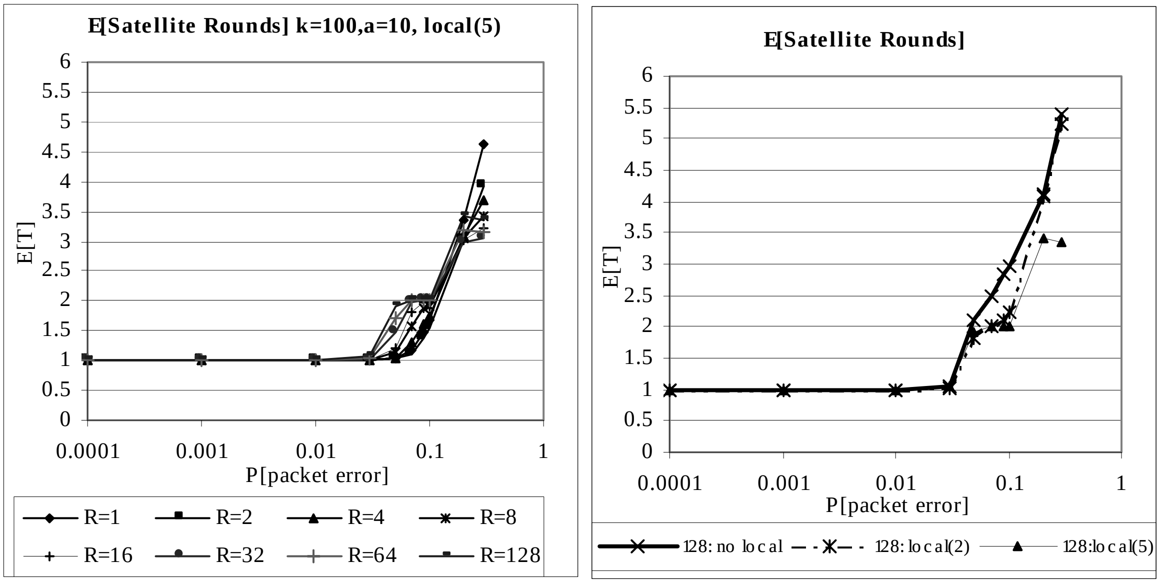

Figure 5.6: Satellite E[T] for k=20, a=2 local(5) & Satellite E[T] Comparison

Using the preceding figures, one sees that local recovery schemes reduce expected number of satellite retransmission rounds. For the transmission group sizes studied, the use of larger subtrees improves the overall performance of the protocol. The reduction in satellite transmission rounds is generally greater for larger TG sizes than for smaller TG sizes. For smaller groups, there are fewer potential receivers to correctly receive the entire TG. For larger groups, there are more local subtrees and therefore a greater chance that additionally satellite rounds are required. The best overall performance occurs for the TG with size equal to one local subtree.

This performance improvements come at the cost of local transmission rounds and therefore places a larger load on the local networks. Figure 5.7 presents the number of local transmission rounds and number of locally transmitted parity packets for the *k*=20, *a*=2, local(5) case. The number of parity packets transmitted is smaller than the size of the TG. As previously discussed in section 2.2.1.2, 20 parity packets can be generated from 20 using a shortened RS(40,20) code. Therefore, if the PRs use this code, then they can generate enough parity packets to meet the local demand. Although, the results are not graphed here, smaller local subtrees require fewer local parity packets.

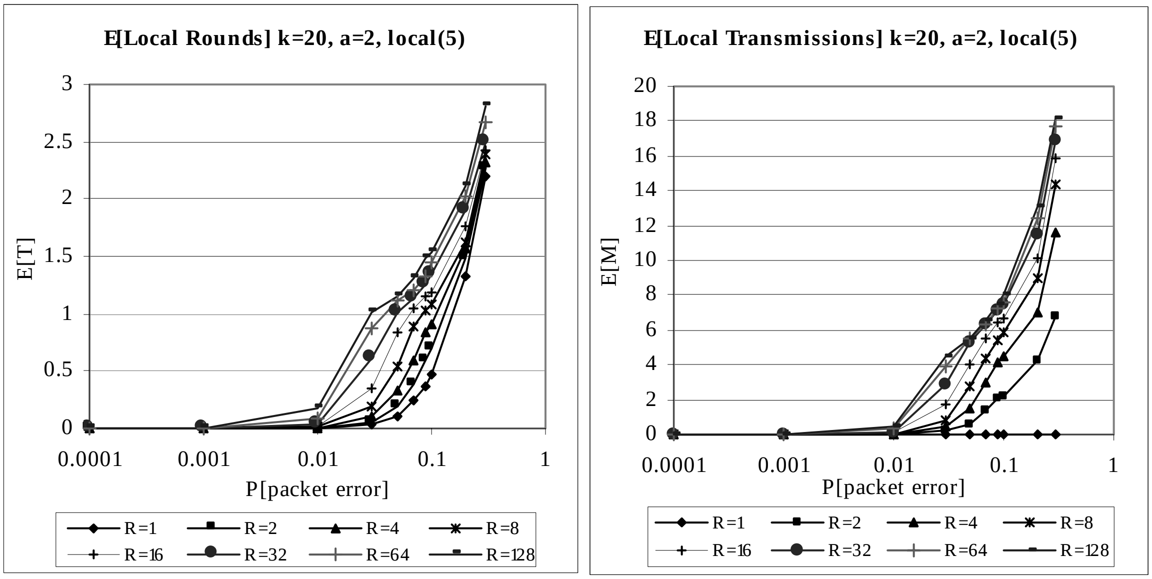

Figure 5.7: Local E[T] & Local E[M] for k=20, a=2 local(5)

5.3 Infinite Parity Case with Channel Estimation: LxS ratio = 1x1

When using the channel estimation scheme without local recovery schemes, there is a reduction in the number of satellite retransmission rounds. However, as shown in Section 4.2.2, the channel estimation scheme held the number of retransmissions to approximately two. Therefore, one can hypothesize that the performance increase seen by adding a local recovery scheme to a protocol that already employs channel estimation is insignificant. Figures 5.8 and 5.9 show that using the local recovery scheme, the expected number of transmission rounds holds at approximately two for higher packet loss probabilities. Figure 5.9-right shows that for a smaller number of receivers and high packet loss probabilities, the expected number of retransmission rounds is larger than two. This perturbation occurs since the source is only able to obtain packet loss probabilities from a few receivers. For a smaller number of observations, the accuracy of the maximal probability estimate decreases. This affect is more pronounced for larger TGs.

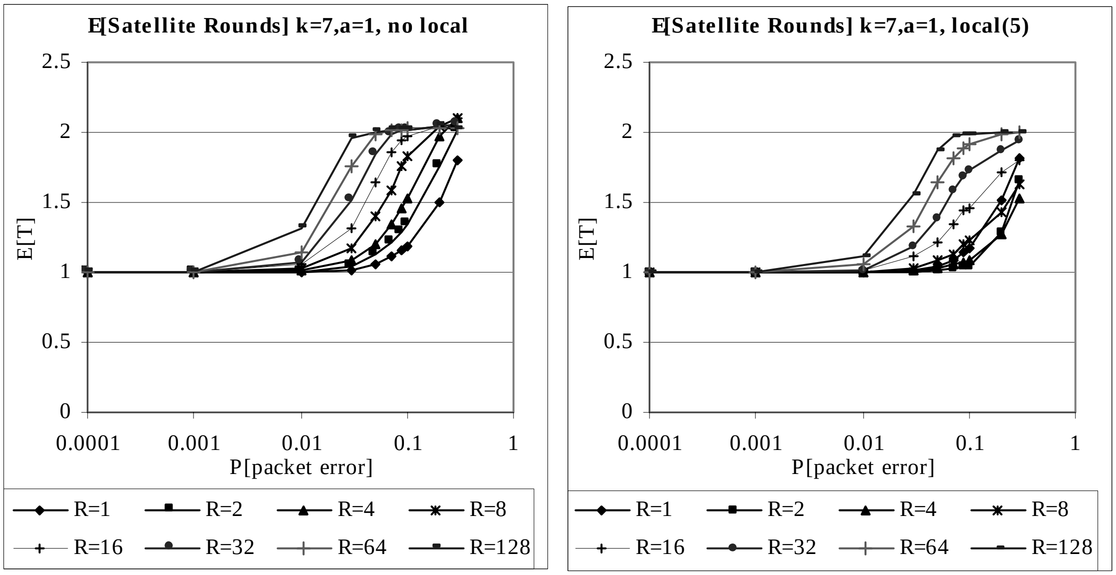

Figure 5.8: E[Satellite Rounds] for k=7, a=1

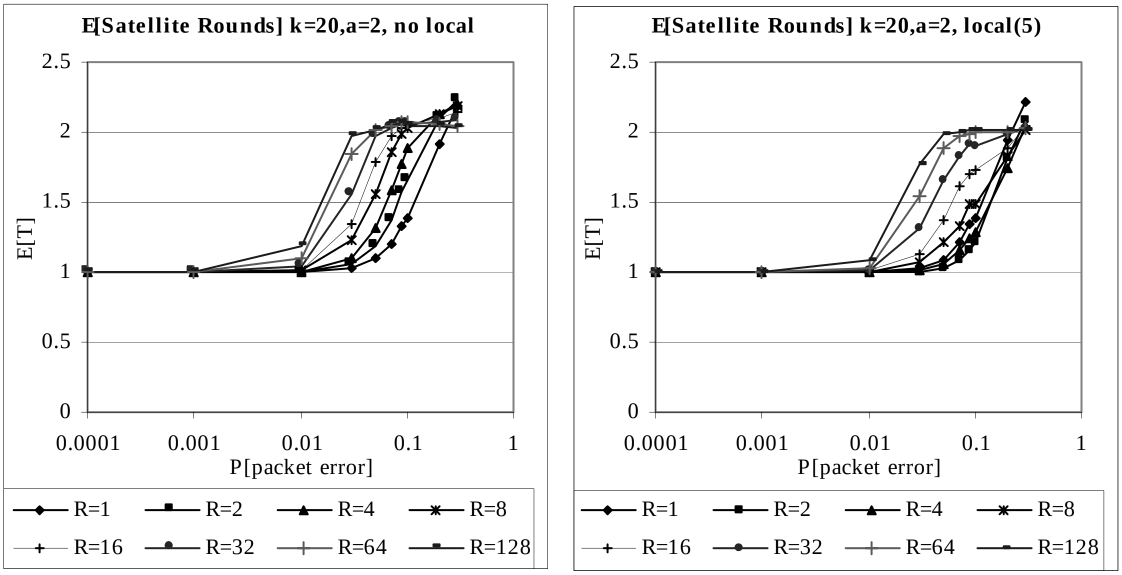

Figure 5.9: E[Satellite Rounds] for k=20, a=2

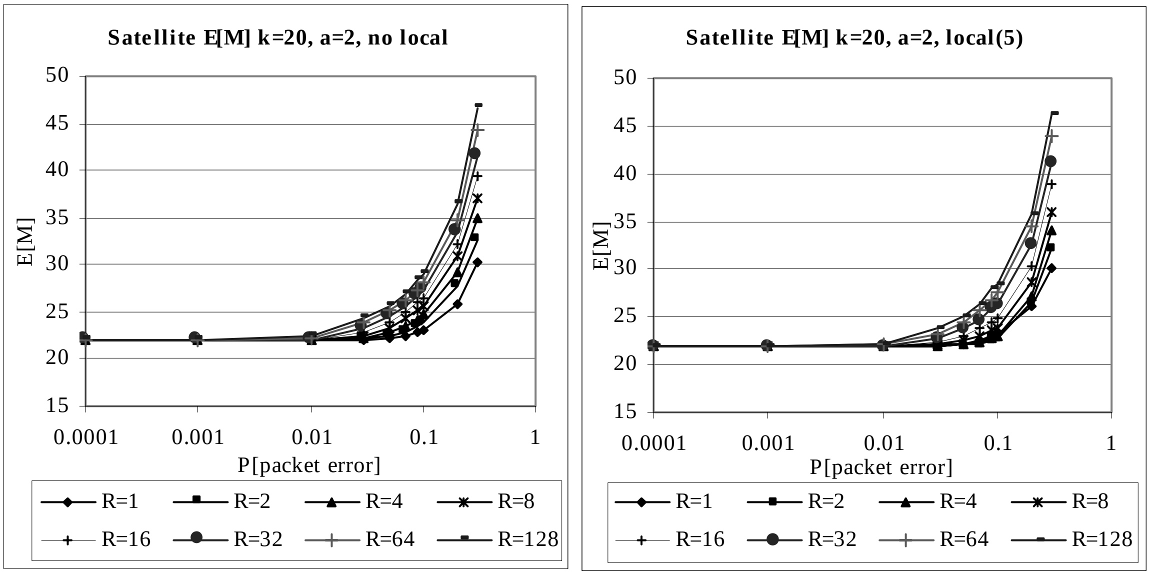

Figure 5.10: Satellite E[M] k=20,a=2 for no local & local(5)

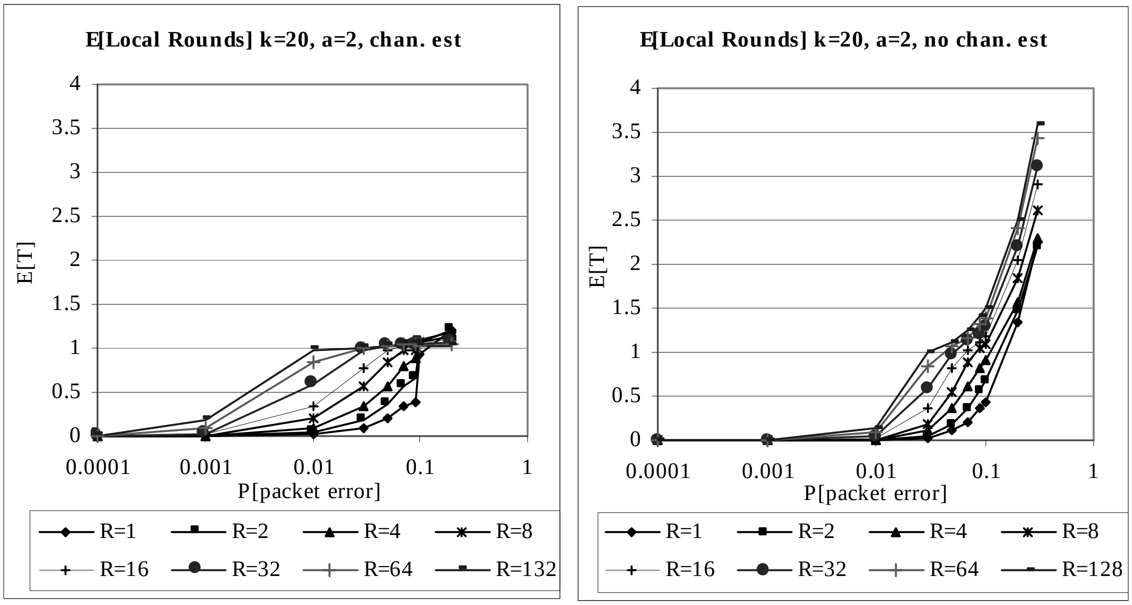

Figure 5.11: Expected Number Local Transmission Rounds (a=1)

The comparison of the two graphs in Figure 5.10 reveals that that there is little difference in the mean number of packets transmitted over the satellite when using channel estimation. As seen in Figure 5.11, the use of channel estimation results in fewer local retransmission rounds thus reducing the local network load. This reduction of local network load places a higher bandwidth requirement on the satellite links. For these results, the channel estimation appears to mask any large advantages gained through local recovery. Any increased local performance may not be worth the additional satellite bandwidth usage.

5.4 Alternate LxS Local Recovery Schemes

In the previous simulations, the generic protocol functioned by alternating between one satellite and one local retransmission round. However, this scheme does not react to changing satellite and terrestrial network situations. This failure to react could potentially result in both more satellite transmission rounds thereby wasting satellite bandwidth and unwanted delay. One solution to this problem is to dynamically change the number of attempted local rounds, *L*, per the number of satellite rounds, *S* (referred to as *LxS*) so that both satellite and terrestrial resources are efficiently used.

The first consideration in such schemes is the network entity responsible for determining the ratio between the number of local rounds to the number of satellite rounds. To guarantee that each receiver performs the correct number of retransmission rounds, the ratio decision is made by the source based upon the feedback from the receivers. Each receiver sends the source the required number of packets before the local recovery phase and the required number of packets after the local recovery phase. Based upon the feedback from the entire group, the source makes a decision on the number of local recovery cycles to be performed immediately following next satellite transmission. The source informs the receivers of this decision via a field in the header of the next set of transmitted packets belong to the TG.

5.4.1 Intra-TG Dynamic LxS Schemes

In one possible scheme, the *LxS* ratio is dynamically changed prior to a TG completely being received at all receivers. The source initially sends the TG over the satellite link to the group of PRs. The PRs spend one round trying to locally recover any missing packets. If this local retransmission round fails, then the PRs send the appropriate feedback to the source. The source makes its decision based upon the maximum number of outstanding packets before local recovery, M_{BL}, and the maximum number of outstanding packets after local recovery, M_{AL}. One potential set of rules is shown in Figure 5.12. Depending upon the feedback from the previous round, the process of *L* local transmission rounds followed by *S* satellite transmission rounds repeats until all receivers have correctly receiver the TG.

If $M_{AL} = M_{BL}$, then the ratio remains 1x1

If $M_{BL} > M_{AL} \geq \left\lceil \frac{1}{2} M_{BL} \right\rceil$, then the ratio becomes 2x1

If $\left\lceil \frac{1}{2} M_{BL} \right\rceil < M_{AL}$, then the ratio becomes 3x1.

Figure 5.12: Set of Source Rules for determining the *LxS* ratio

The main problem with the aforementioned scheme is that it requires two global transmission rounds and one local retransmission round before the protocol adapts to the network conditions. Therefore, the only difference between the adaptive *LxS* protocol and the *1x1* protocols occurs after the fourth overall round (i.e. – both schemes have the same four transmission pattern of satellite, local, satellite, local). To determine the effectiveness of such a proposed change, one needs only look at the Figure 5.13. This figure demonstrates that for both local(2) and local(5) cases with packet loss probabilities less than or equal to 0.1, the originally proposed 1x1 scheme approximately finishes in at most four rounds on average. Except for the highest packet loss probability, the adaptive *LxS* scheme would perform no better than the *1x1* scheme. The aforementioned scheme adapts too slowly to the situation to significantly reduce the expected number of satellite transmission rounds required for reliable delivery.

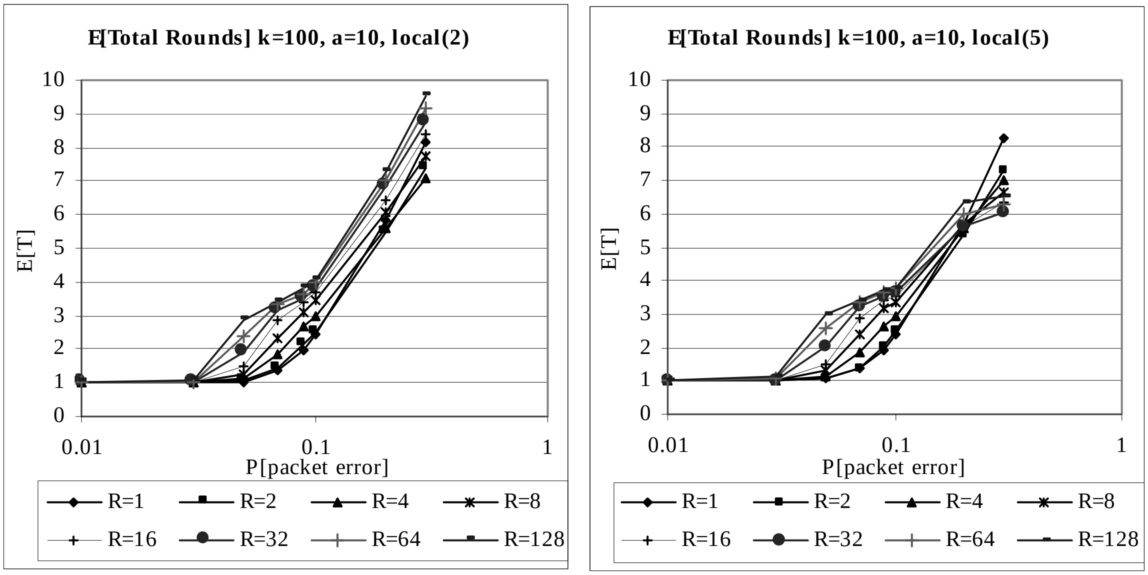

Figure 5.13: E[Total Rounds] k=100, a=10 for local(2) and local(5)

5.4.2 Inter-TG Dynamic LxS Schemes

In another possible adaptation, the source modifies the *LxS* ratio prior to the initial satellite transmission round. Such a modification allows the adaptation to affect protocol performance prior to the fourth overall round of a TG and remains constant for the remainder of the TG. The modification is studied for two different *LxS* ratios; *2x1* and *3x1*. As intra-TG feedback does not significantly alter protocol performance prior the completion of 2 *LxS* recovery cycles (e.g. 2 global rounds and 4 local rounds for the *2x1* case and 2 global rounds and 6 local rounds for the *3x1* case), this ratio is assumed constant for the duration of each particular transmission group.

In Figure 5.14, the simulation results for the *1x1* and the *2x1* schemes are compared. There are two main observations to be made from Figure 5.14. First, one immediately notices a reduction in the expected number of satellite rounds required for large number of receivers (e.g. $R \geq 16$) Second, one notices that for smaller numbers of receivers, there is little or no improvement.

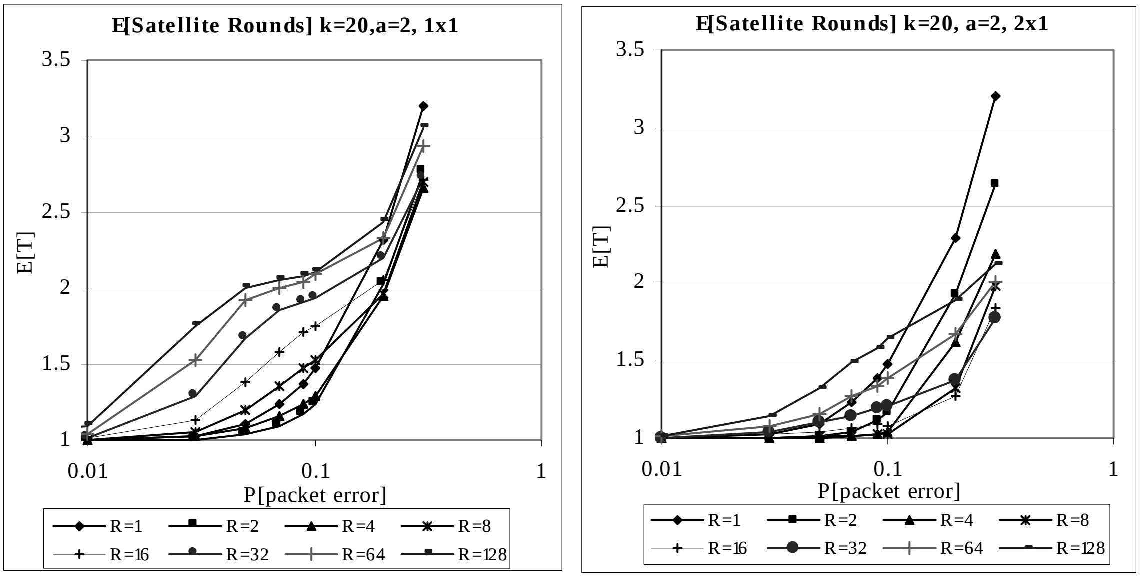

Figure 5.14: E[Satellite Rounds] k=20, a=2, local(5) for 1x1 & 2x1

The cost associated with the reduction in satellite retransmission rounds is the increased usage of the local network. By comparing Figures 5.15-left with 5.7-left, one notices a general increase in the number of local transmission rounds. Depending upon the cost of using the local network, the minimal improvement in the satellite performance of small multicast groups (e.g. $R \leq 16$) may not offset the corresponding cost incurred through increased local retransmission rounds. The graphs shown in Figure 5.14-right and Figure 5.15-left verify the previous statement that intra-TG modification of the *LxS* ratio is unlikely to significantly improve the performance of the protocol. This insignificance results from the fact that, except in the highest packet loss probability scenario (p=0.3), the expected number of satellite transmission rounds is less than 2 and the expected number of local retransmission rounds is less than 4. The graph in Figure 5.15-right shows that, on average, the previously described shortened RS(40,20) code generates enough parity packets to fulfill the local repair requests.

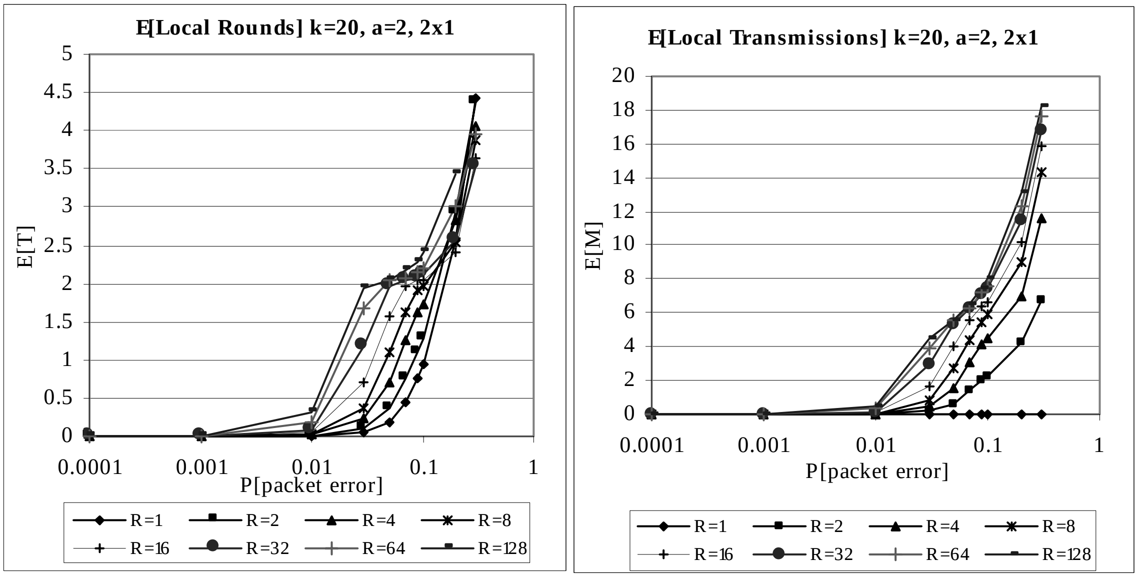

Figure 5.15: E[Local Rounds] & E[Total Rounds] k=20, a=2, local(5), 2x1

Figure 5.16 shows that by increasing the *LxS* ratio to *3x1*, the performance improvement for values of R is greater than in the those observed when the ratio was *2x1*. On average, the protocol only requires approximately one satellite retransmission for $R \geq 16$ up to a packet loss probability of 0.1. For similar parameter ranges, the expected number of local retransmission is approximately equal to 3. This result suggests that for most packet loss probabilities, the protocol reliably transmits a TG in one complete *LxS* cycle.

The above results show that for large number of receivers, the use of *LxS* ratios (greater than *1x1*) improves the protocol's performance over the satellite link. By setting this ratio for each TG based upon current network characteristics (such as satellite packet loss probability, size of multicast group, and size of local recovery subtrees), the source has the capability to efficiently use satellite resources to deliver the TG.

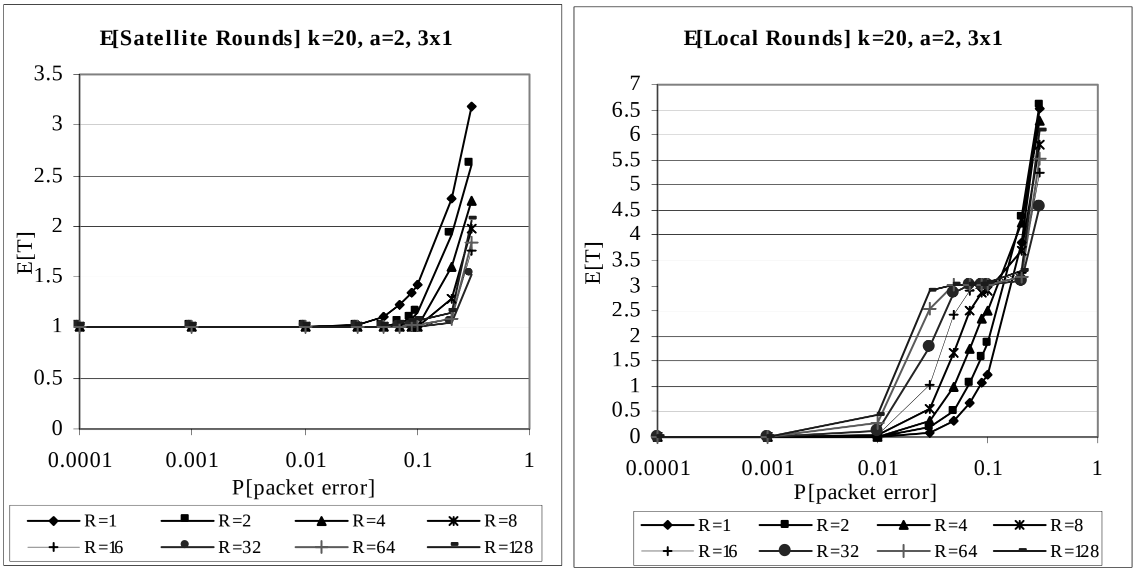

Figure 5.16: E[Satellite Rounds] & E[Local Rounds] k=20, a=2, local(5), 3x1

5.5 Inter-TG Dynamic LxS schemes with Adaptive Channel Estimation

The last set of simulations combined the inter-TG *LxS* scheme with the moving average technique mentioned in Section 4.3. In this combination, the source employs the moving average technique to estimate the packet loss probability over the satellite link. The source uses this estimate to determine the number of autoparity packets used during the initial transmission round and the number of insurance packets used in subsequent satellite retransmission rounds. As was the case in the previous study of inter_TG *LxS* scheme, all parity requests are fulfilled using locally generated parity packets. It is assumed that the source obtains critical information needed to make the appropriate *LxS* ratio determination. For the remainder of this section, this scheme is referred to as the combination scheme.

In the simulations created for the study of this combination, the local subtree depth was assumed to equal five. The simulation results are shown in Figures 5.17-5.19. The graphs in Figure 5.17-left, 5.17-right, and 5.18-left show that as the *LxS* ratio increases, the expected number of satellite transmissions decrease for larger values of R. Even in the *1x1* combination, the moving average technique combined with local recovery performs holds the expected number of satellite transmission round less than approximately two. For small R values, this scheme outperforms the protocols that combine static autoparity, maximal channel estimation, and *1x1* local recovery (see Figure 5.9 right).

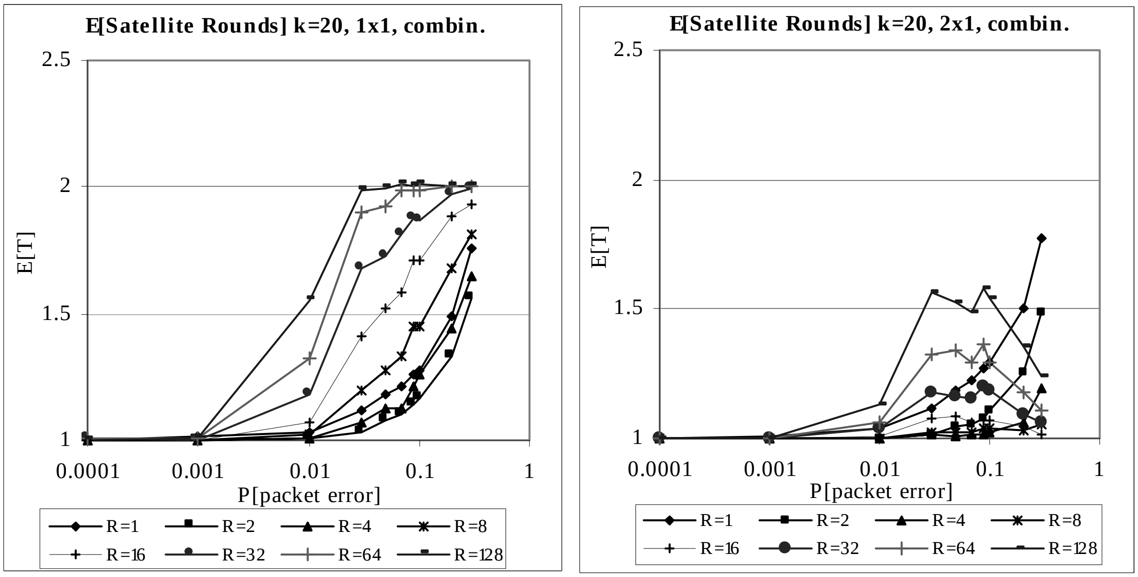

Figure 5.17: E[Satellite Rounds] k=20, combination, a=2, 1x1 and 2x1

The results in Figure 5.18 show that by using a sufficiently high *LxS* enables the protocol to decrease the number of satellite transmission rounds to approximately less

1.15 for multicast groups larger than 4 receivers.   For large groups, this is a significant improvement over the preceding Chapter 5 and Chapter 4 schemes. Additionally, the number of satellite and local transmissions required for this favorable transmission round perform does not substantially exceed the number of transmission rounds used in other cases (see Figure 5.19). By using relatively simple techniques, the combination protocols achieves better bandwidth and delay performance metrics than any of the previous protocols studied in this these.

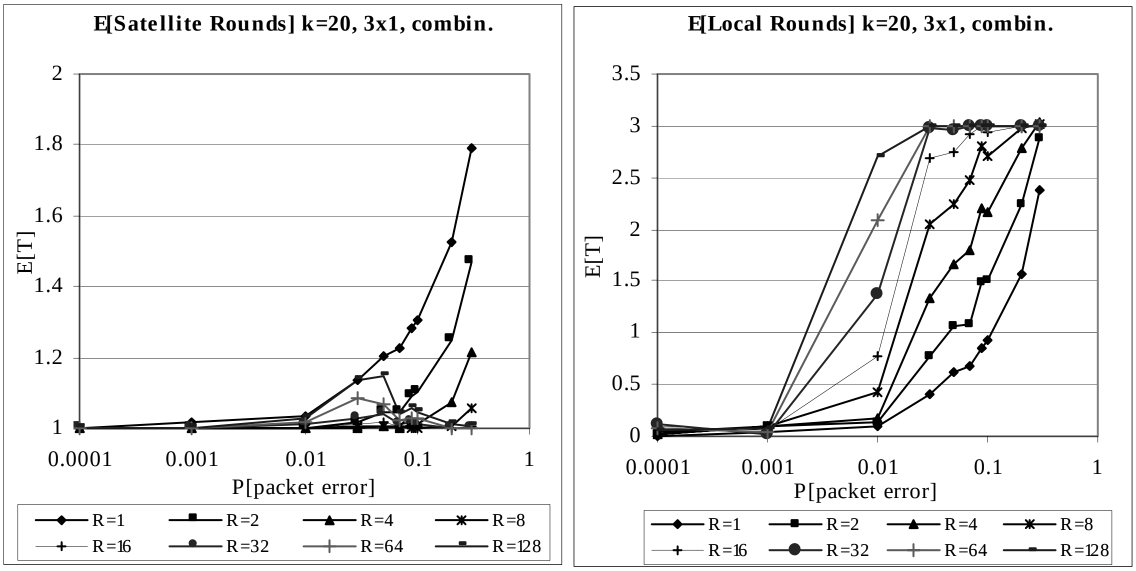

Figure 5.18: E[Satellite Rounds] & E[Local Rounds] k=20, combination, 3x1

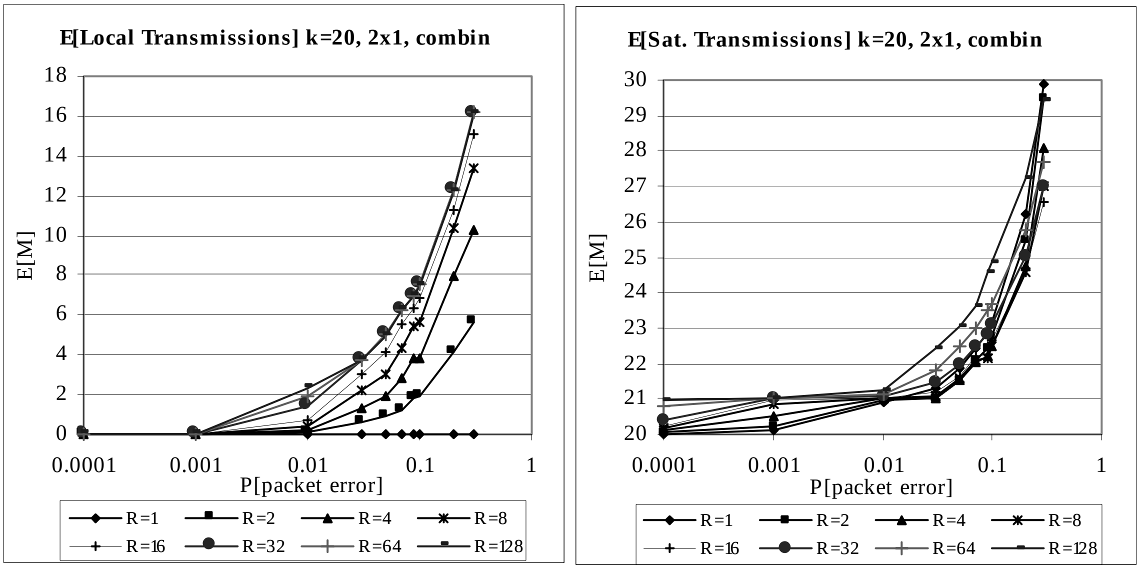

Figure 5.19: Local E[M] & Satellite E[M] k=20, 2x1, combination

6. Conclusions and Future Work

Generic protocols were studied in a variety of different scenarios and with a variety of different adaptations. In the unconnected scenario, the use of autoparity was examined and found to reduce the number of transmission rounds required to reliably deliver a transmission group. The reduction comes at the cost of additional bandwidth usage for low packet loss probabilities. However, as the packet loss probability increases, the intrinsic overhead cost associated with autoparity is reduced. Such a reduction suggests that intelligent use of autoparity can help to lower the delay without drastically increasing bandwidth usage. Secondly, the maximal packet loss probability, channel estimation technique [6] was investigated using different levels of autoparity. It was found that the number of required retransmission rounds was reduced to approximately two. However, a substantial increase in the number of packets was observed when using this scheme. Under the maximal channel estimation scheme, the use of more autoparity packets actually reduces the number of required packets for high probabilities of error.

The above results were obtained under the infinite parity assumption. This assumption was removed and the effects of different amounts of finite parity are documented. Using a finite number of parity packets negatively impacts the performance of the hybrid scheme when compared with the infinite parity case. The degree of this impact depends upon the actual number of parity packets generated. Additionally, the use of channel estimation in the finite parity case is adversely affected. For small values of generated parity, the channel estimation technique offers no substantial improvement. However, if the number of generated parity packets equals the number of data packets, then channel estimation has a more pronounced effect. The increased number of parity packets provides the source with enough packets for both autoparity and subsequent retransmissions.

Before studying local recovery schemes, simplistic adaptive mechanisms where the parity provided during each transmission is adjusted based upon observed packet loss statistics were investigated. The performance of the dynamic autoparity technique does not outperform the a=3 case over the entire range of packet loss probabilities. However, this scheme is more bandwidth efficient at lower packet loss probabilities and does not drastically increase the expected number of retransmission rounds. The moving average channel estimation scheme was also analyzed. This scheme does not perform as well as the maximal channel estimation technique at higher packet loss probabilities. When comparing the dynamic autoparity case protocols that use and do not use the moving average channel estimation technique, one notices that the former's performance enhancement is realized with a negligible increase in the expected number of transmitted packets.

To find other methods to limit the expected number of satellite rounds, local network usage was examined and found to reduce both the expected number of satellite transmissions and the expected number of satellite transmission rounds. The first local recovery mechanism to be examined alternated between one global and one local transmission round. This scheme was studied in the absence of channel estimation techniques and in their presence. Without channel estimation, the local recovery schemes used fewer satellite rounds and transmitted fewer satellite packets than their non-local recovery counterparts. This reduction in satellite load was transferred to the local network. In the presence of the maximal channel estimation technique, the *1x1* local recovery scheme did not offer substantial gains over the non-local recovery case. However, in the presence of dynamic autoparity and the moving average channel estimation scheme, *LxS* local recovery schemes offer noticeable performance gains over their non-local counterparts. In terms of both bandwidth and delay, these *LxS* local recovery schemes coupled with simple dynamic parity schemes offer the lowest average number of retransmission rounds using the fewest number of packets.

Although several different schemes were studied in this thesis, further efforts should be applied to the following areas. More realistic local packet loss probabilities that include correlated losses need to be incorporated into study. The preceding study assumed that all receivers have the same processing power. If they do not have similar capabilities, then the schemes suggested in this thesis need to be adapted to account for this additional design constraint. The local recovery scheme in which both local parity packets and original data packets are transmitted as repairs can be studied. Perhaps, the most promising area of further research lies in obtaining and studying satellite packet loss statistics and creating advanced estimation schemes.

BIBLIOGRAPHY

[1] M. Allman et al., “Ongoing TCP Research Related to Satellites.” draft-ieft-tcpsat-res-issues-09.txt.

[2] K. Almeroth, Y. Zhang. “Using Satellite Links as Delivery Paths in the Multicast Backbone (MBone)”. Proceedings WOSBIS 1998. p. 47-53. October, 1998.

[3] R. Braudes, S. Zabele. “ Requirements for Multicast Protocols” RFC 1458, May 1993.

[4] C. Diot, W. Dabbous, J. Crowcroft. “Multipoint communication: A survey of protocols, functions and mechanisms”. IEEE Journal on Selected Areas in Communications, vol.15, no.3, p. 277-90. April 1997.

[5] S. Floyd, V. Jacobson, S. McCanne, C. Liu, L. Zhang. “A reliable multicast framework for light-weight sessions and application level framing”. Comput. Commun. Rev. (USA), Computer Communication Review, vol.25, no.4, p. 342-56, September 1995

[6] D. Gossink, J. Macker. “Reliable Multicast and Integrated Parity Retransmission with Channel Estimation Considerations”. IEEE Globecom, 1998.

[7] M. Hofmann, “A generic concept for large-scale multicast”, Proceedings 1996 International Zurich Seminar on Digital Communications, IZS 96. p. 95-106. February 1996.

[8] C. Huitema. “The case for packet level FEC”, Proceedings IFIP 5th International Workshop on Protocols for High Speed Networks (PfHSN’96), Sophia Antipolis, France, pp. 110-120, October. 1996.

[9] M. Jung, J. Nonnenmacher, E. Biersack. “Reliable Multicast via Satellite: Uni-directional versus Bi-directional Communication”. Proceedings of KiVS 1999.

[10] S. Kasera, J. Kurose, D. Towsley. “A Comparison of Server-Based and Receiver-Based Local Recovery Approaches for Scalable Reliable Multicast”. Proceedings IEEE INFOCOM 1998, San Francisco, CA, USA March 1998.

[11] S. Lin and D.J. Costello. *Error Correcting Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice Hall, 1983

[12] J. Macker. "Reliable multicast transport and integrated erasure-based forward error correction". MILCOM 97 Proceedings. vol. 2, p. 973-7, November 1997.

[13] J. Macker, Klinker, M. Corson. "Reliable multicast data delivery for military networking", MILCOM 96 Proceedings. vol. 2, p. 399-403, October 1996.

[14] A.J. McAuley, "Reliable broadband communication using a burst erasure correcting code". Computer Communication Review, vol.20, no.4, p. 297-306, Philadelphia, PA. September 1990 pp. 287-306.

[15] K. Miller. *Multicast Networking and Applications.* Reading, Massachusetts, Addison Wesley, 1999

[16] K. Miller, K. Robertson, A. Tweedly, M. White. "StarBurst Multicast File Transfer Protocol (MFTP) Specification". Internet Draft, Work in Progress, draft-miller-mftp-spec03.txt, April 1998.

[17] J. Nonnenmacher, E. Biersack, and D. Towsley. "Parity-based loss recovery for reliable multicast transmission". IEEE/ACM Transactions on Networking, vol.6, no.4, p. 349-61, August 1998.

[18] J. Nonnenmacher, M. Lacher, M. Jung, E. Biersack, G. Carle. "How bad is reliable multicast without local recovery?". Proceedings. IEEE INFOCOM '98, vol. 3, p. 972-9, 1998.

[19] J. Nonnenmacher and E. W. Biersack. "Scalable Feedback for Large Groups". IEEE Transactions on Networking, vol. 7, no.3, p. 375-386. June 1999.

[20] J. Nonnenmacher and E. W. Biersack. "The impact of routing on multicast error recovery," Computer Communication Review, vol.21, no.10, p. 867-79, July 1998.

[21] J. Nonnenmacher and E. W. Biersack. "Performance modelling of reliable multicast transmission", Proceedings IEEE INFOCOM '97, vol. 2, p. 471-9, April 1997

[22] S. Paul, K.K. Sabnani, J.C. Lin, and S. Bhattacharya, "Reliable multicast transport protocol (RMTP)". IEEE Journal on Selected Areas in Communications, vol.15, no.3, p. 407-21, April 1997.

[23] Papadopoulos C., Parulkar G., Varghese. "An error control scheme for large-scale multicast applications". Proceedings. IEEE INFOCOM '98, vol 3, p. 1188-96, 1998.

[24] L. Rizzo. "Effective erasure codes for reliable computer communication protocols". Computer Communication Review, vol.27, no.2, p. 24-36, April 1997.

[25] D Rubenstein, S. Kasera, D. Towsley, and Jim Kurose. “Improving Reliable Multicast Using Active Parity Encoding Services (APES)”. Proceedings IEEE INFOCOM 1999 vol. 3, p. 1248-1255, March 1999.

[26] D. Towsley and J. Kurose. “A comparison of sender-initiated and receiver-initiated reliable multicast protocols”. IEEE Journal on Selected Areas in Communications, vol.15, no.3, p. 398-406, April 1997.

[27] M. Yamamoto, J. Kurose , D. Towsley and H. Ikeda. “A delay analysis of sender-initiated and receiver-initiated reliable multicast protocols”. Proceedings IEEE INFOCOM '97, p. 480-8, vol.2, April 1997.